Technische Universität München
Fakultät für Informatik

Diplomarbeit

Klassifikation von Informationsquellen im
Bereich der Computerforensik
bei Linux

Severine Hammer

Aufgabensteller: Prof. Dr. Hegering
Betreuer: Dr. Helmut Reiser
Abgabedatum: 14. Mai 2004

Ich versichere, dass ich diese Diplomarbeit selbstständig verfasst und nur die angegebenen Quellen und Hilfsmittel verwendet habe.

München, den 14. Mai 2004

Unterschrift der Kandidatin

Zusammenfassung

In dieser Diplomarbeit wird ein Kriterienkatalog für Informationsquellen bei Linux entworfen, der den Computerforensikern eine Hilfestellung bieten soll zu einem bestimmten Vorfall die Informationsquellen nach bestimmten Kriterien auszuwählen, die beispielsweise die kürzeste Lebensdauer haben, am Ergiebigsten sind etc., um den Vorfall auf einem zu untersuchenden System zu bestätigen. Informationsquellen sind dabei Daten bzw. Spuren, die ein Benutzer nach Ausführung eines Vorganges oder mehreren Vorgänge auf dem Rechner hinterläßt. Ein Vorfall besteht dabei aus mehreren Vorgängen.

Die Arbeit enthält eine Top Down Sicht, die zu einem Vorgang alle relevanten Informationsquellen angibt und eine Bottom Up Sicht, die umgekehrt anhand von ermittelten Informationsquellen eines Systems klärt, was auf dem System vorgefallen ist. Um einen konkreten Vorfall zu klären muss die Top Down Sicht mehrere Male angewendet werden, da sie nur einzelne Vorgänge enthält, aber ein Vorfall aus mehreren Vorgängen besteht. Der Forensiker hat bei der Bottom Up Analyse keinen konkreten Verdacht, möchte aber wissen was in einer bestimmten Zeitspanne auf einem System passiert ist. Auch hier muss der Forensiker die Bottom Up Sicht mehrere Male anwenden.

Nach Anwendung der Top Down Sicht auf ein zu untersuchendes System erhält der Forensiker eine der folgenden Aussagen als Ergebnis:

1. *Der Vorfall hat nicht stattgefunden*
2. *Der Vorfall hat mit einer Wahrscheinlichkeit von ... % stattgefunden*

Nach Anwendung der Bottom Up Sicht auf ein zu untersuchendes System bekommt der Forensiker eine der drei Aussagen von der Bottom Sicht zurück.

1. *Vorgang X hat stattgefunden*
2. *Möglicherweise war es Vorgang X*
3. *Mehrere Vorgänge kommen in Frage: Vorgang 1 bis m*

Unteranderem wird in der Arbeit die Vorgehensweise beschrieben, wie die Vorgänge ausgesucht, die Informationsquellen ermittelt, die Bewertungskriterien für den Kriterienkatalog festgelegt und zu letzt wie die verschiendenen Informationsquellen bewertet wurden. Es ist nicht möglich alle Vorgänge zu finden, aber zumindest die ausschlaggebenden d.h. die am häufigsten von Benutzern ausgeführt werden und die dazugehörigen Informationsquellen.

Zuletzt wird in der Arbeit darauf eingegangen in wie fern die beiden Sichten automatisiert werden können, d.h. ein Programm wertet das zu untersuchende System aus und gibt dem Forensiker, die oben genannten Aussagen zurück.

Inhaltsverzeichnis

Kapitel 1

Einführung

In den folgenden Abschnitten werden die beiden Begriffe *Computerforensik* und *Intrusionforensik* erklärt, das Thema dieser Arbeit vorgestellt, der Aufbau dieser Arbeit grob geschildert und in dem Abschnitt State of the Art kurz beschrieben wie computerforensische Analysen in der Praxis durchgeführt werden.

1.1 Was ist Computerforensik

Der Forensik beschreibt in der Gerichtsmedizin Untersuchungen, die Aufschlüsse über Mordfälle geben. Auch die Computerforensik kommt in Anschluss an kriminelle Aktivitäten zum Einsatz. Seit Ende der neunziger Jahre hat die Bedeutung forensischer Analysen durch die steigende Anzahl von externen Angriffen auf die Netze von Organisationen und Unternehmen zu genommen.

Definition: Die *Computerforensik* beschäftigt sich mit dem Aufspüren, sicherstellen, analysieren, dokumentieren von Beweismitteln im Bereich der Computerkriminalität.
Der Forensiker versucht auf einem Rechner, der für ein Verbrechen verwendet wurde, Beweismittel sicherzustellen.

Im Gegensatz zur Computerforensik versucht die Intrusionforensik zu klären, ob in ein System eingebrochen wurde.

Definition: Die *Intrusionforensik* versucht zu klären, ob auf einem Rechner ein Einbruch stattgefunden hat oder nicht. Falls ein Einbruch stattgefunden hat, ist zu klären wie der Angreifer vorgegangen ist.

1.2 Aufgabenstellung

Verschiedene Vorgänge, die von Benutzern ausgeführt werden, hinterlassen auf dem System Informationen. Mit Hilfe dieser Informationen versuchen die Forensiker Vorgänge zu rekonstruieren, die Benutzer ausgeführt haben. Im Rahmen der Diplomarbeit sind die verschiedenen Informationsquellen zu den Vorgängen zu finden. Die Informationsquellen sind auf dem Rechner alle Daten (Spuren), die ein Vorgang auf dem System hinterläßt. Die Informationsquellen müssen klassifiziert und nach bestimmten Kriterien bewertet werden. Diese Klassifikation, die für das Betriebssystem Linux erstellt wird, soll dem Forensiker dabei helfen heraus zu finden, welcher Benutzer zu welcher Zeit eine bestimmten Vorgang ausgeführt hat. Unter Umständen kann nicht geklärt werden, welcher Vorgang stattgefunden hat, da die wichtigsten Informationsquellen, die den Vorgang bestimmen, nicht mehr auf dem zu untersuchenden System vorhanden sind. Die wichtigesten Kriterien werden im Rahmen der Diplomarbeit festgelegt und jede Informationsquelle im Laufe der Diplomarbeit mit diesen Kriterien bewertet.

1.3 Aufbau dieser Arbeit

Im folgenden wird ein kurzer Überblick über die einzelnen Kapitel gegeben.

Kapitel 2 beschreibt die computerforensische Analyse, beginnend mit der Sicherstellung der Daten auf einem laufenden bzw. heruntergefahrenen System und die Analyse des zu untersuchenden Systems, um zu zeigen, an welcher Stelle der Analyse die Klassifikation zum Einsatz kommt.

Das Kapitel 3 beschreibt, wie bei der Erstellung der Klassifikation vorgegangen wurde.

Kapitel 4 wird der Aufbau, sowie die Verwendung und zuletzt die Implementierung der Top Down Sicht und Bottom Up Sicht näher beschrieben. Für die Bottom Up Sicht wurden zwei Darstellungsmöglichkeiten gewählt, Informationsquellen und Vorgänge in einem Graphen und Einteilung in Informationsquellen in Verzeichnisse und Unterverzeichnisse.

Kapitel 5 zeigt wie die Darstellungsmöglichkeit der Bottom Up Sicht als Graphen implementiert werden kann, um dem Forensiker das Suchen der nächsten Informationsuellen zu erleichtern.

Kapitel 6 entält die Top Down Sicht, die zu jedem Vorgang die relevanten Informationsquellen nennt.

Kapitel 7 enthält das Gegenstück zur Top Down Sicht, die Bottom Up Sicht. Dabei wird die Darstellungsmöglichkeit Einteilung in Verzeichnisse und Unterverzeichnisse genannt.

Kapitel 8 fasst diese Arbeit nochmals zusammen und gibt kurz die Vorteile und Nachteile der Darstellungsmöglichkeiten und Implementierungen der Top Down und Bottom Up Sicht an,

Kapitel 9 enthält Erlärungen zu einigen Begriffen, die bei den Informationsquellen verwendet werden.

1.4 State of the Art

Computerforensische Analysen werden in der Praxis anhand von forensischer Fachliteratur und Tools durchgeführt. In den nächsten beiden Abschnitten werden kurz forensische Tools und Literatur speziell für Unix/ Linux beschrieben.

1.4.1 Forensische Tools:

Für die forensische Analyse eines UNIX Systems wird beispielsweise der *"The Coroner's Toolkit"*, kurz TCT eingesetzt. TCT ist eine Sammlung von Programmen, die sowohl die Sicherung und die Analyse der flüchtigen Daten als auch die Analyse der Dateisysteme ermöglichen. Eine Auswahl von Werkzeugen ist:

- **grave–robber:**

  Der Befehl `grave{robber` (Grabräuber) versucht die Daten entsprechend ihrer Flüchtigkeit zu sammeln. Graver-robber sammelt bei Standardeinstellungen Daten über laufende Prozsse und den Zustand von Netzwerkverbindungen, Hardwarekonfigurationen und durchsucht anschließend das Dateisystem nach kritischen Dateien. Bei der Ausführung kann mit Optionen angegeben werden, welche Informationen der graver-robber sammeln soll. Zum Beispiel werden mit der Option `-E` alle Daten gesammelt oder mit `.f` wird das Dateisystem nicht analysiert.

- **mactime:**

  Mit den M(modify)A(access)C(change)–Zeitstempel einer Datei kann eine chronologische Zeittafel der Ereignisse auf einem Rechner generiert werden.

Forensische Tools erleichtern die Analyse und beschleunigen das Auffinden von relevanten Daten auf einem kompromittierten Rechner.

Im Landeskriminalamt in München wird für eine computerforensische Analyse sehr häufig das Programm *EnCase* verwendet. EnCase ist ein forensisches Datenwiederherstellungs- und Analyseprogramm. Auf einfache Art und Weise kann EnCase ein Computersystem untersuchen, Beweismaterial finden und sichern. Da EnCase ein non-invasive Programm ist, werden die ursprünglichen Datenbestände nicht verändert. Unabhängig von der Speicherkapazität einer Festplatte kann die dort gefundene Information komprimiert und auf z.B. einer CD-Rom oder einem Zip- Laufwerk gespeichert werden. Dies ermöglicht die Mitnahme von Beweismaterial bei einer Ermittlung vor Ort. Es ist sogar dank EnCase möglich Daten, die gelöscht oder verschlüsselt oder verborgen sind, schnell und einfach zu lokalisieren.

Unteranderem beglaubigt EnCase und legalisiert alle Kopien des originalen Beweismaterials, um die Integrität des Beweismaterials und den Erfolg beim Echtheitstests zu garantieren.

1.4.2 Forensische Fachliteratur:

Das Buch *"Comuter Forensics: Incident Response Essentials"* von Warren G. Kruse und Jay G. Heiser [4] zeigt wie ein UNIX Host nach wichtigen Informationsquellen hin untersucht wird. Dieses Buch gibt die ausschlaggebenden Informationsquellen unabhängig von bestimmten Vorgängen, Informationen über den Benutzer und was er am System gemacht hat, an.

Das Buch *"Incident Response: Investigating Computer Crime"* von Kevin Mandia [3] beschreibt wie ein UNIX Host nach einem Einbruch untersucht wird. Dabei werden folgende Schritte durchlaufen:

- Alle Log- Dateien überprüfen und durchsuchen
- Suche nach bestimmten Schlüsselwörtern
- Alle relevanten Dateien untersuchen
- Identifikation von unautorisierten Benutzer- / Gruppenaccounts
- Identifikation von verdächtigen Prozessen
- Überprüfen von unautorisierten Zugriffspunkten
- Analyse von trusted relationships

Im Abschnitt 2.2.3. wird die Vorgehensweise bei der computerforensischen Analyse genauer erklärt.

Kapitel 2

Die Computerforensische Analyse

2.1 Aufgabe der Analyse

Die forensische Analyse ist für das dokumentierte und nachvollziehbares Auffinden von Daten verantwortlich. Dabei müssen be-/ entlastende Beweise gefunden werden. Wobei die Computerforensiker klären wer was wann und warum auf dem Computersystem getan hat, versuchen die Intrusionforensiker herauszufinden, ob in das System eingebrochen wurde. Falls eingebrochen wurde versuchen sie zu klären was die Ursache für den Einbruch war. Die Ursache kann beispielsweise eine Sicherheitslücke sein. Außerdem klären sie auf, wie bei dem Einbruch vorgegangen wurde. Die Sicherstellung der Daten erfolgt durch physikalische Analysen, logische Analysen und Datenintegritätsanalysen. Die physikalische Analyse wird vorgenommen falls die Festplatte beschädigt ist. Dabei wird die Festplatte sequentiell nach Sektoren untersucht, die möglicherweise lesbare Datenblöcke enthalten. Die Forensiker erhalten in dem Sinne Nullen und Einsen, die dann in der logischen Analyse zu Daten, Programme etc. mit Hilfe spezieller Programme rekonstruiert werden. Die forensische Analyse garantiert Sicherheit bei einem Gerichtverfahren, dadurch dass die Vorgehensweise nachvollziehbar ist und die Feststellungen, die dem Auftraggeber präsentiert werden, exakt dem Zustand entsprechen, in dem sie sich vor der forensischen Untersuchung befanden. Dabei werden die Daten von dem zu untersuchenden System vor der Analyse mit einem Hashwert (z.B. md5) versehen und dann mit dem Hashwert des Images, das von der Festplatte des Systems gemacht wird, verglichen. Die Werte müssen gleich sein, um Manipulationsvorwürfe abzuwehren. Es wird während der Analyse jeder Schritt genau dokumentiert.

2.2 Ablauf der Analyse

Der Ablauf der forensischen Analyse richtet sich nach dem Zustand des zu untersuchenden Systems. Folgende Zustände sind möglich:

- Das System ist noch in Betrieb, d.h. online.
- Das System ist ausgeschaltet, d.h. offline

Im laufenden Betrieb wird noch unterschieden, ob das System noch am Netz hängt oder nicht.

In den nächsten beiden Abschnitten wird der jeweilige Ablauf erläutert.

2.2.1 Sicherstellung der Daten auf einem laufenden System (Online):

Zu Beginn werden alle relevanten Daten gesichert. Die Daten werden dabei in der Reihenfolge ihrer Vergänglichkeit (Flüchtigkeit) gesichert. Zuerst wird der Hauptspeicher, dann der aktuelle Zustand des Netzwerkes, laufende Prozesse, Daten auf Festplatten, Daten auf Disketten, CD-RW, Streamer etc. gesichert. Peripheriegeräte werden unteranderem auch untersucht. Von der Festplatte wird bei einer forensischen Analyse ein Bitstream Image gesichert. Ein Bitstream Image ist eine bitgenaue 1:1 Kopie des zu untersuchenden Systems. Das Bitstream Image wird verwendet, da auf diesem noch Spuren von gelöschten Dateien zu finden sind und es ist dann auch möglich, vorhandene Reste von bereits überschriebenen Dateien zu retten. Bei einem normalen Datei- Backup ist dies nicht möglich. Das Bitstream Image wird beispielsweise mit dem Open- Source- Tool *dd*, das Bestandteil jeder Unix- Distribution ist, erzeugt. Die Tabelle 2.1 zeigt die Linux Kommandos, die verwendet werden um die einzelnen Beweismittel zu sichern. In der Tabelle wird mit Quelle das zu untersuchende System (Netcat Client) und mit Ziel (Netcat Server) das System, auf denen die Daten gesichert werden, bezeichnet. Mit dem Kommado `nc` für netcat werden die Daten von dem zu untersuchenden System auf einen anderen Rechner übertragen. Dabei kann sowohl das TCP als UDP Protokoll mit beliebigen Ports genutzt werden. In seiner Anwendung als Server ist Netcat in der Lage, sich auf einen beliebigen Port zu binden und dort Netzwerkverbindungen entgegen zu nehmen. Vorteil von Netcat ist die originale Datenübertragung der Daten auf den Server.

Sofern das zu untersuchende System moch am Netzwerk angeschlossen ist, kann der Forensiker sogar Benutzersitzungen mitverfolgen.

1. Hauptspeicher

Quelle:
`dd if=/dev/mem | nc -w 2 [Ziel IP] 6666`
Ziel:
`nc -l -p 6666 > mem.img`

2. Prozesse

Über ps:
`ps enf -Aelf -cols 1000`
Über /proc:
`ls -IR /proc/[0-9]*`

3. Netzwerkzustand

Konfiguration der Netzwerkkarte:
`ifconfig -a; netstat -iea; ip addr show`
Routingtabellen:
`route -n; netstat -n; ip route show table main; ip rule show`
Paketfilterregeln:
`iptables -L -vn -line-numbers`

4. Festplatte

Quelle:
`dd if =/dev/hda1 | nc -w 2 [Ziel IP] 6666`
Ziel:
`nc -l -p 6666 > hda1.img | nc -l -p 6666 | gzip >> hda1.img.gz`

Tabelle 2.1: Datensammlung

Im folgenden werden die einzelnen Kommandos näher erklärt.

Zu 1: Die Quelle ist das zu untersuchende System von dem der Hauptspeicher `/dev/mem` mit netcat auf einen anderen Rechner, der mit der Ziel IP Adresse identifiziert wird, übertragen wird. Die Option `-w` gibt die Zeit an, die Netcat auf weitere Daten wartet. Der andere Rechner nimmt dann auf dem Port 6666 die Daten entgegen und speichert den Hauptspeicher als Datei `mem.img` ab.

Zu 2: Alle Prozesse, die mit ein Terminal (tty) gebunden sind, werden baumartig angezeigt. Mit `ls -IR /proc/[0-9]*` werden alle aktiven Prozesse mit deren Unterordnern rekuriv angezeigt.

Zu 3: Das Kommando `ifconfig -a` gibt detaillierte Informationen über alle aktiven und inaktiven Netzwerkschnittstellen an. `Netstat -iea` zeigt alle Netzwerkverbindungen mit zusätzlichen Details an. Alle Firewallregeln mit Nummer werden mit dem Kommando `iptables -L -vn -line-numbers` angegeben. Die IP- Routing Tabelle mit numerischen Hostnamen wird mit `route -n` angezeigt. Die Hauptroutingtabelle wird mit `ip route show table main` ausgegeben.

Zu 4: Die Quelle ist das zu untersuchende System von dem die Partition hda1 der Festplatte mit netcat auf einen anderen Rechner, der mit der Ziel IP Adresse identifiziert wird, übertragen wird. Die Option `-w` gibt die Zeit an, die Netcat auf weitere Daten wartet. Der andere Rechner nimmt dann auf dem Port 6666 die Daten entgegen und komprimiert sie mit `gzip` .

2.2.2 Sicherstellung der Daten auf einem ausgeschalteten System (Offline)

Bei einem ausgeschalteten System wird nur ein Bitstream Image von der Festplatte erstellt. Ein anderes System, auf dem das Bitstream Image untersucht werden soll, wird hochgefahren und das Bitstream Image wird als Dateisystem gemountet.

2.2.3 Vorgehensweise bei der Analyse

Bei der Analyse selbst werden zuerst allgemeine Informationen über das zu untersuchende System eingeholt. Dieser Abschnitt soll zeigen, an welcher Stelle der Analyse, die Klassifikation zur Anwendung kommt.

1. **Allgemeine Informationen zum System:**

   - *Welches Betriebssystem wurde verwendet und welche Version?*
     In der virtullen Datei `/proc/version` steht diese Information.
   - *Welche Art von CPU?*
     Mit dem Kommando `cat /proc/cpuinfo` werden Informationen zur CPU ausgegeben.
   - *Speichernutzung und Speicherbelegung?*
     Mit dem Kommando `free` kann die Speichernutzung des System angezeigt werden. Informationen zur Speicherbelegung auf der Festplatte liefert das Kommando `df`.
   - *RAM Auslastung und Swap Auslastung?*
     Informationen über die Arbeitsspeicherauslastung befinden sich in der Datei `/proc/meminfo` und für die Swapauslastung in der Datei `/proc/swaps`.
   - *Wie lange läuft das System schon?*
     Das Kommando `uptime` gibt die aktuelle Zeit, wie lange dann System schon läuft und wieviele Benutzer gerade eingeloggt sind.
   - *Welche Hardware befindet sich auf dem System?*
     Mit dem Kommando `lspci -vv` können alle PCI Geräte angezeigt werden. Genauere Informationen über die gesamte Hardware des Systems bekommt man mit dem Kommando `hwinfo`.
   - *Wie wurde die Festplatte partioniert?*
     Mit dem Kommando `fdisk -p` werden alle Partitionen und der jeweilige Dateisystemtyp angezeigt.
   - *Welche Dateisysteme sind gemountet?*
     Mit dem Kommando `mount` können alle derzeit auf dem System gemounteten Dateisystem angezeigt werden.
   - *Welche Dienste bietet das System?*
     In der Konfigurationsdatei `/etc/inetd.conf` sind die Dienste angegeben, die das System anbietet.

2. **Passwort und Shadow Dateien:**

   Welche Benutzer sind auf dem System angelegt? Mit `cat /etc/passwd` werden die Benutzer, die am System einen Account besitzen angezeigt.

3. **Log - Dateien:**

   Unter Verwendung einer bestimmten Shell aus der Konfigurationsdatei `/etc/shells` werden alle Kommandos, die über die Kommandozeile aufgerufen werden, in der History Datei im Home Verzeichnis des Benutzers gespeichert. Folgende Shells mit den History Dateien existieren: `sh_shell` mit `.sh_history`, `csh_shell` mit `.history`, `ksh_ shell` mit `.sh_history`, bash mit `.bash_history`, zsh mit `.history` und tcsh mit `.history`.

   Die Konfigurationsdatei `/etc/syslog.conf` gibt Auskunft darüber, was protokolliert wird. Die wichtigste Log- Datei ist `/var/log/messages`.

4. **SUID Root Dateien und SGID Root Dateien:**

   Die SUID und SGID Root Dateien werden mit dem Kommando find ermittelt:
   `find bin/` $-perm - 004000 - o - perm - 002000$ `-type f -ls`

5. **Versteckte Verzeichnisse und /dev:**

   Das `/dev` Geräteverzeichnis wird untersucht, da ein Angreifer möglicherweise dort seinen Root- Kit verbirgt. Mit folgender Kommandozeile `find -not -type c -not -type b -ls` werden alle nicht Block oder Character Einträge in `/dev` ausgegeben.

6. **MAC Time Analyse:**

   Mit dem MAC Zeitstempeln kann nachvollzogen werden, wann welcher Zugriff auf auf eine Datei erfolgte.

   - M(modify): Der letzte Schreibzugriff auf eine Datei wird hier abgespeichert.
   - A(Access): Der letzte lesende Zugriff auf die Datei wird hier abgespeichert.
   - C(change): Die letzte Änderung der Dateieigenschaften (Rechte, Besitzer, etc.) wird hier abgespeichert.

   Grave–robber speichert in der Datei *body* die MAC Times sämtlicher Dateien. Das mactime Kommando liest diese Datei und bringt die Einträge in chronologische Reihenfolge und erlaubt dann die Ausgabe eines bestimmten Zeitraumes.

7. **Wiederherstellen von gelöschten Dateien:**

   Um auf dem zu untersuchenden System gelöschte Dateien wiederherzustellen bietet TCT die zwei kleinen Programme *ils* und *icat*. Mit ils werden die Inodes der gelöschten Dateien angezeigt. Mit dem Kommando `icat <partition><inode number>><filename>` kann der Inhalt der gelöschten Datei wieder hergestellt werden. Mit dem Kommando `file <filename>` kann der Dateityp der gelöschten Datei bestimmt werden.

8. **Anwendung der Klassifikation:**

   Auf dem zu untersuchenden System muss mindestens SuSE Linux 8.1. installiert sein, um die Klassifikation verwenden zu können, da bestimmte Konfigurationsdateien andere Namen haben oder garnicht mehr verwendet werden, als in der vorherigen Version. Beispielsweise werden die START Variablen, die in der Version 7.3. zum Starten von Diensten erforderlich waren, in der Version 8.1. nunmehr durch die entsprechenden Links in den Runlevel- Verzeichnissen, die zum Starten vorhanden sein müssen, ersetzt. Außerdem wurden einige Änderungen im Zusammenhang mit dem File System Hierarchy

Standard gemacht, d.h. beispielsweise die Beispiel- Umgebung für HTTPD (Apache) unter httpd angelegt (früher war es `/usr/local/httpd`). Mit der Klassifikation kann geklärt werden, ob ein bestimmter Vorgang, der ein normaler Benutzer, der Superuser oder möglicherweise ein Angreifer ausgeführt hat, tatsächlich stattgefunden hat.

2.3 Sinn und Zweck der Klassifikation

Die Klassifikation dient als Hilfe bei der Top Down und Bottom Up Analyse. Bei der *Top Down Analyse* geht der Forensiker von einem bestimmten Vorfall aus und sucht aus der Top Down Sicht die Informationsquellen heraus. Die Informationsquellen werden dann auf der Kopie des kompromittierten Systems heran gezogen, um dann herauszufinden, ob der Vorfall wirklich stattgefunden hat. Falls der Vorfall stattgefunden hat, ist zu klären wer wann und warum für diesen Vorfall verantwortlich war.

Bei der *Bottom Up Analyse* haben die Forensiker keinen bestimmten Verdacht sondern suchen nach Spuren für einen Einbruch, d.h. sie gehen von bestimmten (verdächtigen) Informationsquellen aus und versuchen dann mit der Klassifikation die Informationsquellen einem bestimmten Vorgang zu zuordnen. Wobei es auch möglich ist, dass verschiedene Vorgänge gleiche Informationsquellen mit der gleichen Information bzw. Ergiebigkeit haben können. Eine exakte Zuordnung der Informationsquellen zu einem Vorgang ist nicht immer möglich. Unter anderem kann die Klassifikation dazu verwendet werden Analyse Tools zu entwickeln, die die forensische Analyse automatisieren und erleichtern. Nachdem eine forensische Analyse auf einem kompromittierten System durchgeführt und die Ursache eines Einbruches heraus gefunden wurde kann das Computersysteme besser abgesichert werden, um die Wahrscheinlichkeit eines erneuten Einbruches zu verringern. Dabei werden Sicherheitslücken geschlossen, wie Patches für bestimmte Programme aufgespielt.

2.4 Bemerkungen zum Betriebssystem Linux

In diesem Abschnitt wird kurz erkärt, welche Vorteile Linux sowohl als forensische Plattform und als Betriebssystem auf dem zu untersuchenden System hat.

Forensische Platform: Mit forensischer Platform wird das System bezeichnet, das die Daten der Festplatte des zu untersuchenden Systems, untersucht.

- SuSE Linux ist dem Filesystem Hierarchy Standard (FHS) verpflichtet. Dabei handelt es sich um ein gemeinsam mit anderen Institutionen erarbeitetes Dokument, in dem die Namen und Speicherstellen vieler Dateien und Verzeichnisse festgelegt sind. Aus diesem Grund ist es erforderlich, Dateien oder Verzeichnisse an die richtigen Plätze zu verschieben, wie sie die FHS festlegt. Den vollständigen Standard findet man unter: http://www.pathname.com/fhs.
- Bei Linux kann man auf die Hardware über Dateien zugreifen, d.h. im Verzeichnis /proc/dev ist die Hardware verzeichnet.
- Linux unterstützt sehr viele Dateisysteme (40+). Auch andere Dateisysteme, wie FAT32 oder NTFS können auf der forensischen Platform gemountet und untersucht werden.
- Dateien (Images) können als loopback devices gemountet werden. Ein Loopback-Device ist ein Pseudo-Device, das nach außen als “Gerät” erscheint.

Es entspricht aber keiner bestimmten Hardware, leitet aber Input/Output an ein beliebiges “echtes” Gerät, Datei, etc. weiter.

- Ein System, das noch läuft (Live System) kann sicher und durch minimal invasive Analyse ohne Verwendung von Hard- oder Software writeblocker untersucht werden. Ein Hardware writeblocker verhindert das Schreiben eines Laufwerkes, während auf das Laufwerk beispielsweise zu gegriffen wird oder untersucht wird.

- Linux unterstützt das Überwachen (Monitoring) und Logging von Prozessen und Kommandos. Forensiker können bei vorhandener History Liste Aktionen des Benutzers nachvollziehen.

- Bei Linux besteht die Möglichkeit Standard Output nach Input (command chaining) umzuleiten. Dieser Punkt spielt eine Rolle bei der Analyse des Systems, die Standardausgaben in Dateien zu speichern und möglicherweise als Beweismittel verwenden zu können.

- Der Source Code und die meisten Werkzeuge von Linux sind für jedermann zugängig und können eingesehen werden.

- Die Standard Linux Installation bietet schon sehr viele für die forensische Zwecke nutzbare Werkzeuge (wie z.B. `file`, `grep`, `strings`, `hexedit`, `ldd`, `script`). Mit dem Kommando `script` kann bei der Analyse die Terminal Sitzung mit geschrieben werden. Diese wird in einer Datei gespeichert. Die Datei ist dann für das nachvollziehbare Auffinden der Daten verantwortlich. Das Kommando `file` wird verwendet, um den Dateityp von unbekannten Dateien zu bestimmen. Hierzu verwendet er die Magic Mime Types. Dabei wird meist der Anfang der Datei betrachtet und nach charakteristischen Zeichenfolgen gesucht. Mit dem Kommando `strings` können bestimmte Zeichenfolgen in einer Datei gesucht werden. Mit dem Kommando `grep` kann beispielsweise der Aufenthaltsort von Dateien im Dateisystem gefunden werden.

Kapitel 3

Vorgehensweise bei der Erstellung

Bei der Erstellung der Klassifikation wird zuerst Top Down vorgegangen, d.h. es wird zuerst von den Vorgängen ausgegangen, um dann die dazugehörigen Informationsquellen zu ermitteln. Dabei wird in mehreren Schritten vorgegangen. Die Vorgänge wurden in Gruppen eingeteilt, um so die wichtigsten Benutzerhandlungen abzudecken. Im nächsten Schritt wurden die Vorgänge ausgeführt, um so die hinterlassenen Informationsquellen zu finden. Durch das Auffinden verschiedener Informationsquellen (beispielsweise Konfigurationsdateien, Logdateien, Dateiattribute) konnten gleichzeitig die Bewertungskriterien festgelegt werden, aufgrund von Unterschieden, wie Lebensdauer, Ergiebigkeit etc.. Zuletzt wurden fehlende Informationsquellen bei den einzelnen Vorgängen durch die Bottom Up Vorgehensweise nachgetragen. Bei der Bottom Up Vorgehensweise wird von den Informationsquellen ausgegangen und versucht durch die Sammlung von Informationsquellen auf den Vorgang zu schließen. In den nachfolgenden Abschnitten wird erläutert, wie in jedem Schritt genau vorgegangen wurde.

3.1 Begriffserklärungen

- Es gibt die drei Benutzertypen:
  - *root:* Der Superuser, der in der Regel root heisst, hat volle Zugriffsberechtigungen für das gesamte System. Der root Benutzer kann auf alle Dateien des Systems zugreifen und ist gleichzeitig in der Regel der einzige Benutzer, der bestimmte Programme ausführen kann (z.B. httpd).
  - *Normaler Benutzer:* Normale Benutzer sind User, die sich am System anmelden dürfen. Normale Benutzer verfügen in der Regel über ein Stammverzeichnis, das sogenannte Home Verzeichnis. Für normale Benutzer ist der Zugriff auf Dateien und Verzeichnisse des Systems in der Regel eingeschränkt. Daraus resultiert, dass sie nicht allzu viele Funktionen auf der Systemebene ausführen können.
  - *Angreifer:* Ein Angreifer ist eine nicht berechtigter Nutzer, der unter einem fremden berechtigten Account das System nutzt. Ziel eines Angreifers ist es in der Regel Root Zugriff zu erlangen.
- *Vorgang:* Ein Vorgang setzt sich aus mehreren Aktionen zusammen, die vom Benutzer ausgeführt werden oder nur aus einer einzelnen Aktion. Die einzelnen Aktionen sind atomar, d.h. können nicht weiter zerlegt werden. Wenn ein

Vorgang betrachtet wird, wird davon ausgegangen, dass alle Softwarepakete und Dienste auf dem System installiert sind, um den Vorgang ausführen zu können.

- *Vorfall:* Ein Vorfall setzt sich aus mehreren Vorgängen zusammen und ist vor der Anwendung der Top Down Analyse bekannt.

- *Informationsquellen:* Informationsquellen sind Spuren in Form von Daten, die ein Benutzer nach Ausführung seines Vorganges, auf dem System hinterlässt.

3.2 Einteilung der Vorgänge

Zuerst müssen alle relevanten Vorgänge festgelegt werden. Um die relevanten Vorgänge zu finden, werden die Funktionsweise und Aufgaben eines abtrakten Betriebsystemes betrachtet. Abbildung 3.1. zeigt das Schichtenmodell eines abtrakten Betriebssystems.

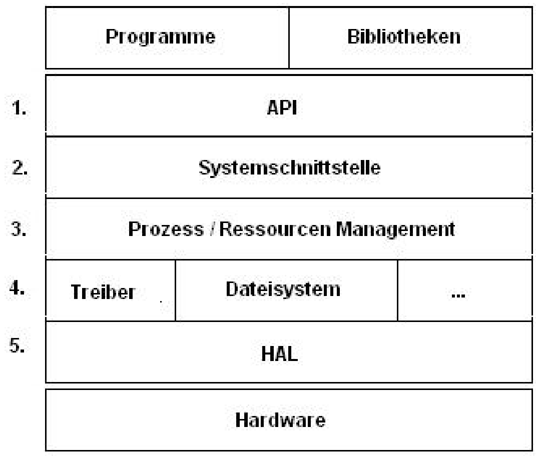


Abbildung 3.1: Modell eines abstrakten Betriebssystems

Das Schichtenmodell wird dann auf SuSE Linux angewendet und es ergibt sich folgende Einteilung in Klassen.

- **Benutzerverwaltung:** Loginverwaltung, Nutzerverwaltung, Resourcenverwaltung

- **Dateiverwaltung:** Dateien, Verzeichnisse, Dateisysteme

- **Prozessverwaltung:** Prozesse

- **Automatisierung von Vorgängen:** at, cron

- **Netzwerk:** SSH, Telnet, DNS, NIS, NFS, FTP

- **Softwareverwaltung:** RPM Pakete, Source Pakete, SuSE Source Pakete

Jede einzelne Klasse wird zusätzlich noch unterteilt, um die einzelnen Vorgänge besser zuordnen zu können. Die Loginverwaltung und Nutzerverwaltung wurde insofern gewählt, da Angreifer bevor sie beispielsweise ein System ausspionieren oder es zum Angriff auf ein anderes System verwenden können, sich ersteinmal Zutritt zum System verschaffen müssen.

Die Klasse Automatisierung mit den beiden Programmen **at** und **cron** werden gewählt, da sie sehr häufig von Benutzern verwendet werden. Die Unterteilung der Klasse Netzwerk erfolgt aufgrund von Angriffen auf verschiedene Netzdienste, wie Domainname Service, Network File System etc. und bekannte Sicherheitsprobleme der Vergangenheit. Von [13] findet man eine Liste mit den 10 verwundbarsten Internetdiensten. Dazu gehören DNS, SSH und sendmail. Die anderen Dienste wie Telnet, NIS, NFS und FTP werden gewählt, da sie sehr häufig von Einzelpersonen und Firmen gleichermassen verwendet werden.

Die Softwareverwaltung wird in die verschiedenen Paketarten eingeteilt, die auf einem SuSE Linux System installiert werden können. Diese Unterteilung ist erfolgt, da die Installation, Deinstallation oder Updates der Pakete auf unterschiedliche Weise erfolgen kann. SuSE Source Pakete sind Pakete, die bei der ausgelieferten SuSE Linux Distribution dabei sind und können dann je nach Bedarf vom Benutzer installiert werden. Die Source Pakete sind in dem Sinne gezippte Tar Archive und können beispielsweise im Internet für jede Linux Distribution frei herruntergeladen werden. Im Gegensatz zu Source Paketen und SuSE Source Paketen kann die Originalität bei RPM Paketen aus unbekannter Quelle durch ein **checksig** durch überprüfung der MD5-Prüfsumme sicher gestellt werden.

In den einzelnen Unterteilungen werden dann die relevanten bzw. die am häufigsten ausgeführten Vorgänge aufgespürt. Das Systemadministrationsbuch [1] hilft dabei die relevanten Vorgänge, die nur der Superuser ausführen darf, zu bestimmen. Die anderen Vorgänge, die von normalen Benutzern ausgeführt werden dürfen, werden anhand des Linux Benutzerhandbuches [2] und [7] ausgewählt. Eine Tabelle mit allen untersuchten und dokumentierten Vorgängen befindet sich in der Tabelle 6.1..

3.3 Auffinden der Informationsquellen zu den Vorgänge

Um zu jedem Vorgang die relevanten Informationsquellen aufzuspüren wurden folgende drei Hilfsmittel verwendet: Literatur, Man Pages von einzelnen Kommandos, einen Dateiintegritätschecker und spezielle Tools. Die Vorgänge werden auf einem Pentium III 500 MHz mit der SuSE Linux Distribution 8.1 durchgeführt (siehe Abschnitt 3.3.2). Die Informationsquellen zu jedem Vorgang werden durch Kombination der verschiedenen Hilfsmittel ermittelt. Die Kombination der drei verschiedenen Hilfsmittel dient auch zur Überprüfung, dass keine relevanten Informationsquellen bei den Vorgängen vergessen werden. Die ermittelten Informationsquellen müssen noch offline ausgwertet werden, d.h. es muss überprüpft werden, ob die Informationsquelle bzw. beim Herunterfahren nicht modifiziert werden.

3.3.1 Man Pages

Linux verfügt über ein ausgereiftes Hilfesystem, die Manual Pages. Zu fast jedem Kommando findet sich dort eine ausführliche Beschreibung. Die Manual Page wird mit dem Kommando `man <section> <command>` aufgerufen. Die Man Page für das ausgewählte Kommando enthält eine kurze Beschreibung, wofür das Kommando verwendet wird, die Aufrufsyntax, eine detaillierte Beschreibung der Wirkungsweise aller möglichen Optionen und die vom Kommando benötigten/ modifizierten Dateien. Die benötigten und modifizierten Dateien werden als Informationsquellen herangezogen. Es wird dann nur noch überprüft, was sich an ihnen geändert hat.

3.3.2 Literatur

Das Systemadministrationshandbuch [1] hilft dabei die wichtigsten Dateien für die einzelnen Vorgänge zu finden.

3.3.3 Ausführen der Vorgänge

Bei der Ausführung der einzelnen Vorgänge wird dabei unterschieden, ob der Vorgang nur vom Superuser oder vom normalen Benutzer und Superuser ausgeführt werden dürfen. Der Vorgang wird kann mit maximal drei Methoden ausgeführt.

1. **manuell:** (z.B. editieren der vorgangsbezogenen Konfigurationsdateien)
2. mit Hilfe von **Kommandos** (z.B. privilegierte Kommandos)
3. mit Hilfe einer **graphischen Oberfläche** (z.B. Yast2)

Nicht alle Vorgänge können mit allen drei Methoden ausgeführt werden.

Um die Vorgänge im Bereich Netzwerk ausführen zu können, wurden Client und Server für jeden Netzdienst auf dem Rechner installiert.

3.3.4 Einsatz von speziellen Tools

Es werden der Dateiintegritätschecker `tripwire`, das Kommando `strace`, das Kommando `diff` und das Programm `Ethereal` eingesetzt. In den nachfolgenden Abschnitten wird kurz erklärt wie die Tools eingesetzt werden.

3.3.4.1 Datei- Integritätschecker

Damit potentielle Informationsquellen bei der Untersuchung des Vorganges nicht vergessen werden, wird der Datei- Integritätschecker *Tripwire* eingesetzt. Tripwire dient dabei zur Sicherstellung von Datenintegrität überwacht die Zuverlässigkeit kritischer Systemdateien und -verzeichnisse, indem sie alle daran durchgeführten Änderungen feststellt. Mögliche Änderungen treten durch Ausführen des Vorganges auf. Die Änderungen soll tripwire erkennen.

Im nächsten Abschnitt wird der Ablauf erklärt wie tripwire Änderungen von Dateien erkennt. Die Abbildung 3.2. zeigt den groben Ablauf bei der Verwendung von tripwire.

1. **Installieren von Tripwire und benutzerdefiniertes Einstellen der Konfigurationsdatei:**

   In die Konfigurationsdatei `/etc/tw.config` von Tripwire werden die Pfade der Verzeichnisse der Systemdateien eingetragen, deren Integrität überprüft werden soll, d.h. wo Änderungen erwartet werden. Das gesamte Filesystem wird nicht bei jedem Vorgang angegeben, da dies einige Stunden dauern würde, um die Datenbank anzulegen. Die meisten Änderungen finden im /proc und /etc Verzeichnis statt.

2. **Initialisieren der Triwire Datenbank:**

   Bei der Initialisierung der Datenbank erstellt Tripwire eine Sammlung von Dateisystemobjekten, die auf den Regeln in der Konfigurationsdatei beruhen. Diese Datenbank dient als Basis für Integritätsprüfungen. Die Datenbank wird mit dem Kommando `/usr/sbin/tripwire -initialize` erstellt. Die Datenbank `tw.linux_db` wird im Verzeichnis `/etc/database` erstellt.

3. **Ausführung des Vorganges:**

   Der bestimmte Vorgang wird mit Methode 1 ausgeführt.

4. **Ausführung einer Integritätsprüfung:**

   Die Tripwire-Integritätsprüfung wird dann mit dem Kommando `/usr/sbin/tripwire -interactive` durchgeführt. Bei der Integritätsprüfung vergleicht tripwire den aktuellen Stand der Dateisystem-Objekte mit den in der Datenbank gespeicherten Eigenschaften. Tripwire überprüft dabei die Prüfsummen (md5) über den Inhalt der Dateien in der Datenbank und überprüft die Grösse der Dateien, die Modifikationszeit und die letzte Statusänderung. Eventuelle Differenzen werden in `/etc/databases/tw.db_linux` gespeichert und es wird zusätzlich eine verschlüsselte Kopie des Berichtes erstellt.

5. **Analyse der Tripwire Berichtdatei:**

   Die Berichtdatei wird mit `/usr/sbin/tripwire -d /etc/databases/tw.db_linux` angezeigt. Der Bericht enthält die Dateien, die durch den Vorgang modifiziert wurden. Für die modifizierten Dateien werden die Dateiattribute wie geänderte Inodenummer, Grösse, Modifikationszeit und Zeit der letzten Statusänderung vor Ausführung des Vorganges (expected) und die aktuellen Dateiattribute (observed) angezeigt.

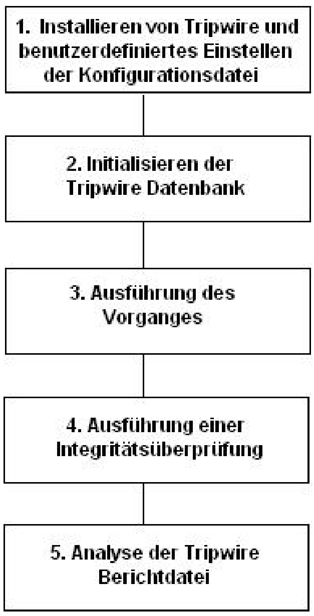


Abbildung 3.2: Ablaufdiagramm von tripwire

3.3.4.2 strace

Mit dem Kommando strace werden nur die Vorgänge untersucht, die mit dem Aufruf von Kommandos ausgeführt werden. Der Aufruf von `strace 2>&1 |grep open` zeigt alle Dateien an, die durch den Aufruf des Kommandos des Vorganges geöffnet werden. Mit `strace 2>&1 <command >|grep read` werden alle Dateien angezeigt, die durch den Vorgang gelesen werden und zuletzt werden mit `strace 2>&1`

|`grep write` Dateien angezeigt, die durch den Vorgang modifiziert werden. Das Kommando wird bereits vor der Ausführung des Kommandos verwendet, um die Informationsquellen zum Kommando zu bestimmen.

3.3.4.3 diff

Mit dem Kommando `diff <file1.txt> <file2.txt>` werden zwei Dateien Zeile für Zeile mit einander verglichen. Diese Methode wird angewendet, wenn schon eine potentielle Informationsquelle, d.h. eine Datei, die durch den Vorgang modifiziert wird, bekannt ist. Um das `diff` Kommando anzuwenden, muss die potentielle Informationsquelle, bevor der Vorgang ausgeführt wird, gesichert werden. Nachdem der Vorgang ausgeführt wurde wird die gesicherte Informationsquelle mit der aktuellen Informationsquelle mittels dem `diff` Kommando miteinander verglichen. Unterscheiden sich die Dateien, wird auf die Standardausgabe zuerst für die erste Datei die Zeilen angegeben beginnend mit >, in denen sie sich von der zweiten Datei unterscheidet. Die Zeilen werden dann von Zeilen der zweiten Datei gefolgt, beginnend mit <in denen sie sich von der ersten Datei unterscheidet.

Im folgenden wird anhand des Vorganges *Benutzer bindet Dateisystem ein* die Verwendung von den verschiedenen Tools beschrieben.

1. **Verwendung von Tripwire:**
   - In der Konfigurationsdatei `/etc/tw.config` von Tripwire muss auf jeden Fall das Verzeichnis /etc zum Durchsuchen angeben werden.
   - Die Tripwire Datenbank wird mit dem Kommando `/usr/sbin/tripwire -initialize` erstellt.
   - Der Vorgang wird mit dem Kommando `mount /dev/hda1` ausgeführt und das Dateisystem der Partition hda1 wird eingehängt.
   - Die Tripwire-Integritätsprüfung wird dann mit dem Kommando `/usr/sbin/tripwire -interactive` durchgeführt.
   - Die Berichtdatei wird mit `/usr/sbin/tripwire -d /etc/databases/tw.db|linux` angezeigt. Die Berichtdatei sieht dann folgendermaßen aus.

changed: -rw-r–r– root	264 Feb 28 12:20:38 2004	/etc/mtab
Attributes /etc/mtab	Observed (what it is)	Expected (what it should be)
st_ino	68229	100647
st_size	264	326
st_mtime	Sat Feb 28 12:20:38 2004	Fri Feb 27 15:02:46 2004
stctime	Sat Feb 28 12:20:38 2004	Fri Feb 27 15:02:46 2004
md5 (sig1)	3hbkVwB3gDquBq7:X6n12O	2CE1BFpopGaRZoQ2.Gb0OQ
snefru (sig2)	1aeTN0sdJNfLSUeu4xac3a	01KvXB4o5gwWFWLPwA35Pe

   Die Berichtdatei zeigt, dass die Datei `/etc/mtab` durch den Vorgang geändert wurde. Alle Attribute der Datei wurden geändert. Dadurch, dass sich die Grösse der Datei grösser geworden ist, kann man schließen, dass etwas hinzugefügt wurde.

2. **Verwendung von diff:**

   Um nun die Änderung in der Datei /etc/mtab festzustellen, d.h. welche

Informationen hinzugefügt wurden, wird der Vorgang nochmals ausgeführt, aber vorher eine Kopie von `/etc/mtab` gemacht. Nach der Ausführung des Vorganges kann dann die Kopie und die aktuelle Datei `/etc/mtab` mit dem Kommando `diff /etc/mtab /etc/mtab.copy` miteinander verglichen werden und so die Änderung festgestellt werden.

3.3.4.4 Ethereal

Da bei der Untersuchung von Vorgängen im Bereich Netzwerk auf dem Rechner sich Client und Server befinden, ist es schwierig die aufgebauten Verbindungen mit dem Kommando `netstat` voneinander zu unterscheiden. Deshalb wurde der Packet-Sniffer Ethereal verwendet, um den Datenverkehr auf einer Netzwerkschnittstelle mitzuschneiden und die Verbindungen von Client zum Server und umgekehrt voneinander unterscheiden zu können.

3.4 Ermittelte Informationsquellen

3.4.1 Einteilung der Informationsquellen

Nachdem die Informationsquellen für jeden Vorgang ermittelt wurden, werden diese in statisch lesbare und statisch modifizierbare Informationsquellen eingeteilt. Statische Informationsquellen werden durch den Vorgang, den der Benutzer initiiert, gelesen oder modifiziert. Dynamische Informationsquellen, die wärend eines Vorganges durch das System modifiziert werden, unabhängig von Aktionen des Benutzers, werden nicht betrachtet.

1. **Lesbare Informationsquellen:**
   Der Vorgang der die Informationsquellen verändert greift nur lesend auf die Informationsquelle zu, d.h. der Inhalt der Datei wird nicht verändert, sondern nur das Dateiattribut “letzter Zugriff” der Datei.

   - System- und Konfigurationsdateien werden beim Starten eines Prozesses/ Dämons durch den Benutzer gelesen.

2. **Modifizierbare Informationsquellen:**
   Die Informationsquelle wird durch einen Vorgang geändert. Dabei wird der Inhalt der Informationsquelle geändert. Die Informationsquelle kann direkt oder indirekt geändert werden. Eine direkte Änderung findet beispielsweise statt, wenn der Benutzer selbst etwas in eine Datei schreibt. Verwendet er ein Kommando wird die Datei indirekt geändert.

   Beispiel 1:

   - Die Passwortdatei /etc/passwd und die Shadowdatei werden vom Benutzer root editiert, wenn dieser einen neuen Benutzer anlegen möchte. Durch Editieren der Konfigurationsdatei wird diese direkt geändert.

   Mit Hilfe eines Kommandos, Programms oder der graphischen Oberfläche YAST2 bzw. Konqueror wird eine Informationsquelle indirekt geändert.

   Beispiel 2:

   - Mit dem Kommando `useradd` werden die beiden Dateien indirekt geändert.

- Mit der graphischen Oberfläche YAST2 (Kontrollzentrum): Sicherheit und Benutzer: Benutzer bearbeiten und anlegen werden die beiden Dateien indirekt geändert.

3. **Ausführbare Informationsquellen:**
   Die Informationsquelle ist ein Programm oder Script.
   - Rootkits, laufende Prozesse, Scripte von Benutzern etc.
4. **Neu erstellte Informationsquellen:**
   Die Informationsquelle wir durch einen Vorgang des Benutzers neu erstellt, d.h. sie ist nach dem Vorgang nicht mehr existent.
   - Dateien und Verzeichnisse, etc.
5. **Entfernte Informationsquellen:**
   Die Informationsquelle wir durch einen Vorgang des Benutzers entfernt, d.h. sie ist nach dem Vorgang nicht mehr existent.
   - Dateien, Verzeichnisse, etc.

Die Informationsquellen werden durch Vorgänge des Benutzers gelesen, geändert, erstellt, entfernt oder ausgeführt.

3.4.2 Liste von gefundenen Informationsquellen

Zu 90% sind die gefundenen relevanten Informationsquellen Dateien. Der Rest sind Datei –, Verzeichnisattribute, Environment Variablen, TCP und UDP- Verbindungen und verschiedene Softwarepakete. Tabelle 3.1 zeigt Überblick von den ermittelten Informationsquellen zu den dokumentierten Vorgängen.

Typ	**Untertypen**	**Beispiele**
Datei	Binärdatei Textdatei NIS Map	/var/log/utmp Konfigurationsdatei /etc/passwd <passwd>.byname
Softwarepaket	RPM Paket Source Paket SuSE Source Paket	<package name>.rpm Tar Archiv <package name>.spm
Datei –/Verzeichnisattribute	letzter Zugriff letzter schreibender Zugriff letzte Statusänderung etc.	
Environment Variablen	benutzerspezifisch programmspezifisch	HOME, PATH SSH_CLIENT
Netzwerkverbindungen	TCP UDP	eingehende ausgehende eingehend und ausgehend

Tabelle 3.1: Ermittelte Informationsquellen

3.4.3 Benennung der Informationsquellen

Die Informationsquellen werden unabhängig vom Vorgang benannt, d.h. dass bei mehreren Vorgängen, die gleiche Benennung der Informationsquelle vorkommt, aber haben unterschiedliche Eigenschaften, d.h. haben unterschiedliche Bewertungen bezogen auf die Bewertungskriterien. Bei der Benennung wird unterschieden, ob die Informationsquelle statisch lesbar, statisch modifizierbar, ausführbar ist oder neu erstellt wurde.

1. **Statisch lesbare Informationsquellen:**
   Eine statisch lesbare Informationsquelle, falls es eine Datei ist, wird nach ihrem Aufenthaltsort im Dateisystem benannt, d.h. der vollständige Pfad zur Datei im Dateisystem wird angegeben. Es wird zusätzlich noch angegeben, ob es eine Konfigurationsdatei, Log- Datei, Gerätedatei, virtuelle Datei etc. ist. Für die anderen Informationsquellen wird der Typ bzw. der Untertyp der Informationsquelle angegeben. Konfigurationsdateien, die beim Vorgang gelesen werden, bestimmen den Ablauf und das Verhalten des Vorganges. Sie werden deshalb als Informationsquelle des Vorganges berücksichtigt. Der Vorgang “Benutzer legt neuen Benutzer an” beispielsweise hat folgende statisch lesbare Informationsquellen:

   - Konfigurationsdatei `/etc/login.defs`
     Diese Konfigurationsdatei enthält Parameter, die die z.B. festlegen in welchem Bereich die UserID und GroupID für den neuen Bnutzer liegen dürfen.
   - Konfigurationsdatei `/etc/shells`
     Diese Konfigurationsdatei enthält die verfügbaren shells und spielt dann eine besondere Rolle, wenn der Benutzer, die Shell ändern will. Er kann nur eine neue Shell wählen, die in der Konfigurationsdatei spezifiziert ist.

2. **Statisch modifizierbare Informationsquelle:**
   Statisch modifizierbare Informationsquellen werden nach ihrem Aufenthaltsort im Dateisystem benannt, d.h. der vollständige Pfad zur Datei im Dateisystem wird angegeben und es wird zusätzlich angegeben, was sich in der Datei geändert hat. Bei Textdateien kann es sich um einen Eintrag, ein Feld oder mehrere Felder handeln. Bei Binärdatei handelt es sich dabei um Änderung eines Strukturelementes. Desweiteren wird unterschieden, ob es eine Konfigurationsdatei, Log- Datei, Gerätedatei, virtuelle Datei etc. ist.

   Eintrag: Ein Eintrag wird zur Textdatei als Zeile hinzugefügt.

   Beispiel 1: In die Log- Datei `/var/log/messages` wird beim Einloggen des Benutzers ein Eintrag hinzugefügt.

   Feld/ Felder: Ein Feld /Felder wird/ werden in der Textdatei abgeändert.

   Beispiel 2: In der Passwortdatei `/etc/passwd` wird beim Ändern der Login-Shell durch das Kommando chsh das letzte Feld Shell geändert und auf den neuen Wert gesetzt.

   Strukturelement: Dateien, die ein spezielles Format haben, beispielsweise binär haben eine Struktur als Aufbau. Die

   Beispiel 3: In der Log- Datei `/var/log/utmp` wird beim Einloggen das Strukturelement ut–pid auf die Process ID des Login Prozesses gesetzt.

   Der Vorgang “Benutzer legt neuen Benutzer” beispielsweise hat folgende statisch modifizierbare Informationsquellen:

   - Eintrag in der Passwortdatei `/etc/passwd`

- Eintrag in der Shadowdatei `/etc/shadow`
- Eintrag in der Log –Datei`/var/log/messages`
- Home–Verzeichnis `/$HOME/` <`user`>
- Eintrag in der History Liste des Benutzers

3. **Ausführbare Informationsquelle:**
Ausführbare Informationsquellen werden auch nach ihrem Aufenthaltsort im Dateisystem benannt.

4. **Neu erstellte Informationsquelle:**
Neu erstellte Informationsquelle kann eine Datei oder ein Verzeichnis sein. Diese werden auch nach ihrem Aufenthaltsort im Dateisystem benannt.

5. **Entfernte Informationsquelle:**
Diese Informationsquelle ist nach Ausführung des Vorganges nicht mehr vorhanden. Sie wird trotzdem nach dem Aufenthaltsort im Dateisystem vor dem Löschen benannt.

6. **Verwendete Abkürzungen:**
Aus Platzgründen werden folgende Abkürzungen, bezogen auf Dateiattribute von Verzeichnissen verwendet:

   Source Directory: S.S.
   Destination Directory: D.D.
   Parent Directory: P.D.

3.5 Festlegung der Bewertungskriterien:

In diesem Abschnitt wird erläutert wie die Bewertungskriterien festgelegt wurden. Konkrete Kriterien konnten in der Literatur nicht gefunden werden. Durch Vergleichen der einzelnen Informationsquellen untereinander wurden die Kriterien **Beschreibung der Quelle** und **Modifikationsmöglichkeit** ermittelt. Die Modifikationsmöglichkeit hat sich dabei ergeben, dass die verschiedenen Informationsquellen unterschiedliche Zugriffsrechte besitzen.

Die Kriterien **Auswertungsaufwand**, **Möglichkeit der Fehlinterpretation** und **Fälschungsmöglichkeit** wurden durch Betrachtung von Vorfällen im Landeskriminalamt München festgelegt.

Das Kriterium **Auswertung** wurde durch die Tatsache, dass zu untersuchende System nach einem Vorfall noch laufen oder im heruntergefahrenen Zustand sichergestellt werden, festgelegt. Dabei kommt die Online und Offline Auswertung ins Spiel.

Das Kriterium **Existenz** hat sich dadurch ergeben, dass beispielsweise ein Vorfall auf einem System erst zu einem späteren Zeitpunkt erkannt wird und dann manche Informationsquellen, die von anderen Vorgängen auch geändert werden, dann nicht mehr vorhanden sind. Mit Existenz werden die Informationsquellen angegeben, die unbedingt auf dem System nach der Analyse vorhanden sein mussen, um den Vorfall bestimmen zu können.

Das letzte Kriterium **Ergiebigkeit** einer Informationsquelle ist eines der wichtigsten Kriterien, ohne das der Forensiker nichts über Informationsquelle aussagen kann bzw. etwas daraus folgern kann.

Die Kriterien werden in allgemeine Informationen zur Informationsquelle und Bewertungskriterien eingeteilt.

3.5.1 Allgemeine Kriterien

Die allgemeine Kriterien werden nicht bewertet. Sie dienen dazu mehr Informationen über die Informationsquelle zu geben.

- **Beschreibung der Quelle:**
  - *Typ der Informationsquelle:* Binärdatei, Textdatei, Verzeichnis, Script, Dateiattribut, etc.)
  - von wem unter welchen Bedingungen wird diese gelesen, modifiziert oder neu erstellt. Statisch lesbare Informationsquellen, das sind grössten Teils Dateien, die entweder schon existent sind, da sie zum Basissystem gehören oder sie sind nur vorhanden, falls ein Parameter bezüglich des Dateinamens in einer Konfigurationsdatei angegeben ist.

    Beispiel 1: Die Passwortdatei `/etc/passwd` ist schon existent, da sie zum Basissystem gehört.

    Beispiel 2: Die Datei `/etc/nologin` ist nur vorhanden, falls ein bestimmter Parameter in der Konfigurationsdatei `/etc/login.defs` gesetzt ist.
    Statisch modifizierbare Informationsquellen, falls sie Dateien sind, sind schon vorhanden und es wird nur ein Eintrag (Zeile), Einträge (Zeilen), Feld (Wert) oder Felder (Werte) geändert. Bei Datei –/ Verzeichnis –werden neu gesetzt.

- **Verweis auf andere Vorgänge:**
  Ein dokumentierter Vorgang kann wiederum aus mehreren einzelnen dokumentierten Vorgängen bestehen. Eine Einzelaktion ist zum Beispiel die Ausführung eines Kommandos und das Drücken eines Buttons. Damit die Informationsquellen und die dazugehörigen Bewertungen nicht wiederholt werden, werden bei den Vorgängen auf den Einzelvorgang verwiesen. Anhand des Vorganges *Benutzer legt neuen Benutzer an* wird dies kurz veranschaulicht. Der Benutzer fügt zuerst eine Zeile für den neuen Benutzer in der Passwortdatei `/etc/passwd` hinzu. Diese Aktion ist atomar und verweist dann auf den Vorgang *Benutzer ändert Datei*. Dann wird mit dem Kommando `pwconv` ein dazugehöriger Shadoweintrag in `/etc/shadow` hinzugefügt. Mit `mkdir` das Home- Verzeichnis erstellt. Diese Aktion verweist auf den Vorgang `Benutzer erstellt Verzeichnis`. Zuletzt werden die Zugriffrechte mit `chown` des Home- Verzeichnisses und das Passwort vergeben. Diese beiden Aktionen verweisen dann auf die Vorgänge *Benutzer ändert Zugriffsrechte* und *Benutzer ändert Passwort*.

- **Existenz:**
  Muss die Informationsquelle zusammen mit anderen Informationsquellen unbedingt vorhanden sein, um auf den Vorgang, den der Benutzer ausgeführt hat, eindeutig schliessen zu können. Statisch lesbare Informationsquellen müssen beispielsweise vorhanden sein, da sie den Aufenthaltsort von anderen Dateien, die auch wichtige Informationsquellen sind, durch Parameter angeben oder beim Anruf des Kommandos mit bestimmten Parametern, zu lässige Parameterwerte angeben. Der Forensiker muss bei den statischen Informationsquellen darauf achten, dass sie nicht modifiziert wurden, d.h. nur das Dateiattribut "letzter Zugriff" muss einen Zeitpunkt innerhalb der Zeitspanne des Vorfalls haben. Bei einer modifizierten statisch lesbaren Datei könnten dann falsche

Informationen auftauchen, die der Angreifer eingebracht hat, um seine Spuren zu verwischen. Bei dem Kriterium **Existenz** werden die Informationsquellen per Nummer angegeben. Dieses Kriterium spielt eine sehr grosse Rolle, wenn die Ausführung des Vorganges von einem möglichen Angreifer vertuscht werden soll. Je nach dem wie gut die Systemkenntnisse des Angreifers sind, kann er die relevanten Informationsquellen löschen und so seine Spuren verwischen. Dies wird auch unter dem Begriff Verschleierung des ausgeführten Vorganges durch den Angreifer zusammengefasst. Bei diesem Kriterium wird zusätzlich noch unterschieden, ob die Informationsquelle bei einer Online bzw. Offline Auswertung unbedingt vorhanden sein muss. Wie schon in einem früheren Abschnitt erklärt wurde, können die Vorgänge mit maximal drei verschiedenen Methoden durch den Benutzer ausgeführt werden. Deshalb kann es Vorkommen, dass die eine Methode eine Informationsquelle hinterläßt, aber die andere nicht. Damit der Forensiker noch sagen kann, wie der Vorgang genau ausgeführt worden ist, wird deshalb bei dem Kriterium Existenz verschiedene Kombinationen von Informationsquellen angegeben.

- **Nicht- Existenz:**

  Bei manchen Vorgängen, wie z.B. Benutzer löscht Benutzeraccount, werden Informationsquellen gelöscht bzw. entfernt. Dieses Kriterium gibt an, welche Informationsquellen nicht mehr vorhanden sein müssen, um den Vorgang eindeutig bestimmen zu können.

3.5.2 Bewertungskriterien:

Diese Kriterien werden mit Skalen bzw. Standardwerten belegt, um die Informationsquellen mit den Kriterien zu bewerten. Jedes Kriterium wird in den nachfolgenden Abschnitten beschrieben und die Bedeutung der Standardwerte zu den einzelnen Kriterien erläutert.

- **Ergiebigkeit** *(gut/mittel/schlecht):*

  - Welche Informationen enthält die Informationsquelle, bezogen auf Dateien ist hier der Inhalt, d.h. die Zeile oder das Feld der Datei interessant. Bei statisch lesbaren Dateien sind die Parameterwerte und der Zeitpunkt (letzter Zugriff), wann die Datei gelesen wurde wichtig. Der Zeitpunkt des letzten Zugriffs ist gleichzeitig, der Zeitpunkt, zu dem der Vorgang gestartet wurde.

  - *gut:* Die Informationsquelle gibt Aufschluss darüber wer den Vorgang ausgeführt hat und gibt Informationen über den Vorgang.

    Beispiel 1:
    Die Log- Datei `/var/log/messages` gibt beispielsweise beim Vorgang: *Benutzer legt neuen Benutzer an* , wann sich der Benutzer den Vorgang ausgeführt hat, den Rechnername des Hosts, die Pid vom `useradd` Programm, den Namen des neuen Benutzers, die UID des neuen Benutzers, die GID des neuen Benutzers, das Home- Directory für den neuen Benutzers und die Login Shell des neuen Benutzers an.

    Der Eintrag in der `/var/log/messages` alleine klärt schon, dass der Vorgang von root ausgeführt wurde.

  - *mittel:* Die Informationsquelle gibt Aufschluss darüber, wie der Vorgang ausgeführt wurde und mit welchen Parametern. Mit Parametern ist beispielsweise gemeint, das das Verzeichnis, das durch den Vorgang: Benutzer ertstellt Verzeichnis neu erstellt wurde, dann auch tatsächlich auf dem System existent ist. Ist ein Eintrag in der History Liste vorhanden, der das Kommando `mkdir <directory name>` enthält, kann der Forensiker den Verzeichnisnamen ermitteln und das Verzeichnis auf dem zu untersuchenden System suchen.

Beispiel2:
Die Existenz von Einträgen in der History Liste in bezug auf den Vorgang gibt an, dass der Vorgang nur mit Kommandos ausgeführt wurde.

  - *schlecht:* Die Informationsquelle klärt nur, wo sich weitere Informationsquellen im Dateisystem befinden, wenn sie nicht zum Basissystem gehören oder enthält Parameter, die das Kommando bzw. Programm braucht um den Vorgang erfolgreich auszuführen. Dies sind statisch lesbare Informationsquellen. Bei den statisch lesbaren Informationsquellen ist die Ergiebigkeit ist der "letzte Zugriff"auf die Informationsquellen von Bedeutung. Sie sagt aus, wann der jeweilige Vorgang innerhalb der Zeitspanne genau stattgefunden hat. Bei allen statisch lesbaren Informationsquellen wird der Inhalt der Informationsquelle angeben und erklärt.

- **Modifikationsmöglichkeit** (*root/ normaler Benutzer/ System):*

  Wer kann bzw. darf die Informationsquelle modifizieren bzw. darf sie lesen. Zu diesem Kriterium werden die Zugriffsrechte der Informationsquelle betrachtet.

  - *root:* Der Super user mit der Kennung root, hat volle Zugriffsberechtigungen auf das gesamte System.
  - *normaler Benutzer:* Für den normalen Benutzer ist der Zugriff auf Dateien und Verzeichnisse des Systems in der Regel eingeschränkt. Manchen Dateien kann er nicht lesen, beispielsweise die Shadow- Datei `/etc/passwd`.
  - *System:* Das Prozessdateisystem kann nur vom System geändert werden. Root hat nur Leserecht und kann die Zugriffsrechte mit dem Kommando chmod nicht ändern.

- **Auswertung** *(online/offline/online und offline/netzangebunden):*

  Ist die Informationsquelle bzw. die Information bei der jeweiligen Auswertung überhaupt noch vorhanden.

  - *online:* Das zu untersuchende System ist heruntergefahren und die Festplatte wurde sichergestellt. Hier bei wird davon ausgegangen, dass das System ordnungsgemäß heruntergefahren wurde.
  - *offline:* Das zu untersuchende System läuft noch und möglicherweise sind noch Benutzer eingeloggt.
  - *online und offline:* Die Informationsquelle ist online und offline verfügbar.
  - *netzangebunden:* Das zu untersuchende System ist online und ist an das Netz gebunden.

- **Auswertungsaufwand** (*niedrig/hoch):*

  Wie hoch ist der Aufwand an die Informationsquelle zu kommen und deren Informationen zu ermitteln. Dieses Kriterium bezieht sich auf das Kriterium **Auswertung**.

- *niedrig:* Der Auswertungsaufwand ist niedrig, falls die Information der Informationsquelle (Eintrag) durch Vergleich der aktuellen Informationsquelle mit einer älteren Version der Informationsquelle, die beispielsweise durch ein Backup zur Verfügung steht, ermittelt werden kann. Dies bezieht sich nur auf Dateien. Back Ups von Systemdateien werden beispielsweise mit `logrotate` oder dem `cron` Dämon vewaltet. Bei den restlichen Informationsquellen kann mit einem einfachen Kommando die Information angezeigt werden.
- *hoch:* Der Auswertungsaufwand ist hoch, falls für Dateien kein Backup zur Verfügung steht, um so die Information zu ermitteln. Um die Information zu ermitteln müssen die Informationen aus anderen gefundenen Informationsquellen herangezogen werden und möglicherweise deren Informationen kombiniert werden.

Bei den meisten ermittelten Informationsquellen können die Informationen (Ergiebigkeit) mit dem `cat` Kommando oder mit einem Texteditor ermittelt werden. Dateien, die keine Textdateien sind werden, werden mit speziellen Kommandos ausgewertet. Das heisst die Informationen der Informationsquelle werden mit dem Kommando ausgelesen und auf die Standardausgabe weitergegeben. Für manche Informationsquellen, die ein besonderes Format haben, müssen mehrere Kommandos verwendet werden, um an alle Informationen zu gelangen.

- **Lebensdauer** *(kurz/lang):*

  Wie lange kann man auf die Informationsquelle bzw. Information zu greifen, bevor sie beispielsweise gelöscht, geändert oder überschrieben wird. Bei der Lebensdauer der Informationsquelle kann keine absolute Zeit angeben werden.

  - *kurz:* Die Informationsquelle wird durch einen anderen Vorgang geändert. Dies bezieht sich beispielsweise auf Dateiattribute, die bei sehr vielen Vorgängen neu gesetzt werden.
  - *lang:* Dateien können eine bestimmte Dateigrösse erreichen. Es kann vorkommen, dass ein Eintrag nicht in die Datei geschrieben wird, da die maximale Grösse überschritten wird. Diese Bewertung wird nur bei Dateien angewendet, bei denen immer nur Einträge hinzugefügt werden (z.B. /var/log/messages). Wurde das System ausgeschaltet und die Informatsionsquelle ist dann noch verfügbar, ist die ihre Lebensdauer lang.

- **Möglichkeit der Fehlinterpretation** *(Möglich/Unmöglich):*

  Ist es möglich die Informationen aus der Informationsquelle falsch zu deuten.

  - *Möglich:* Es ist möglich die Informationen aus der Quelle falsch zu deuten, da die Informationsquelle beispielsweise von einem anderen Vorgang auch geändert wird.
  - *Unmöglich:* Die Informationsquelle wird von keinem anderen Vorgang geändert, deshalb ist die Fehlinterpretation der Informationen aus der Quelle nicht möglich.

- **Fälschungsmöglichkeit** *(leicht/mittel/schwer):*

  - Ist es möglich die Informationen aus der Informationsquelle zu fälschen.
  - Wie das gegebenenfalls möglich.

  - *leicht:* Wenige oder fast keine Vorkenntnisse sind nötig, um die Informationsquelle zu fälschen. Ein normaler Benutzer kann die Informationsquelle fälschen.
  - *mittel*: Mit root Rechten kann die Informationsquelle gefälscht werden.
  - *schwer:* Vorkenntnisse eines erfahreren Programmierers im Bereich IT- Sicherheit und Systemprogrammierung sind nötig.

- Wie wird die Informationsquelle für einen Angriff verwendet.

3.6 Bewertung der Informationsquellen bzgl. der Kriterien:

Der letzte Schritt ist nun jede einzelne Informationsquelle aller dokumentierten Vorgänge nach den festgelegten Kriterien zu bewerten. Bei der Bewertung der Informationsquellen ist aufgefallen, dass viele Informationsquellen die gleichen Bewertungen bei den Bewertungskriterien besitzen und dass sie auch mehrere Bewertungen bei einem Kriterium, wie beispielsweise bei dem Kriterium **Auswertungsaufwand** den Wert niedrig und hoch, haben können, aber nur unter bestimmten Bedingungen. Um Wiederholungen zu vermeiden, die die Lesbarkeit der Top Down Sicht dieser Arbeit erschwert, muss eine Darstellungsform gefunden werden, die die Lesbarkeit erleichtert.

3.6.1 Wichtige Informationsquellen:

Dieser Abschnitt nennt die Informationsquellen, die sehr häufig nach der Ausführung der Vorgänge aufgetaucht sind. Auf die Beschreibung der einzelnen Informationsquellen wird später verwiesen, um Wiederholungen zu vermeiden.

- **History Liste des Benutzers**

Beschreibung der Quelle:
Die History- Liste ist eine Textdatei und wird durch den Vorgang um einen Eintrag (Einträge) erweitert, wenn der Vorgang durch ein Kommando oder manuell ausgeführt wird. Die History Liste entsteht beim Starten der Login- Shell des Benutzers. Die History Liste wird die aus der dazugehörigen History Datei erstellt. Folgende Login- Shells und die dazugehörigen Default History Dateien im Home Verzeichnis des Benutzers sind möglich:

1. sh shell: `.sh_history`
2. csh shell: `history`
3. ksh shell: `.sh_history`
4. bash: `.bash_history`
5. zsh: `.history`
6. tcsh: `.history`

Die Umgebungsvariable $HISTFILE spezifiziert die History Datei.

- **Log- Datei /var/log/messages:**

Beschreibung der Quelle:
Die Log- Datei ist eine Textdatei und wird durch den Vorgang um einen Eintrag (Einträge) erweitert, wenn der Vorgang durch ein privilegiertes Kommando oder mit der graphischen Oberfläche ausgeführt wird. Der Eintrag entsteht nur wenn die Log- Datei in der Konfigurationsdatei `/etc/syslog.conf` des syslogd Dämons spezifiziert wird.

- **Log- Datei /var/log/vsftpd.log**

Beschreibung der Quelle:
Die Log- Datei ist eine Texdatei und wird durch den Vorgang um einen Eintrag (Einträge) erweitert, wenn der Vorgang durch ein Kommando oder mit der graphischen Oberfläche ausgeführt wird. Diese Konfigurationsdatei protokolliert Aktivitäten, die mit FTP zu tun haben, z.B. das Einloggen des Benutzers via FTP, den Upload und Download einer Datei durch den Benutzer. Der Eintrag entsteht nur wenn die Log- Datei in der Konfigurationsdatei `/etc/vsftpd.conf` durch den Parameter. `xferlog_file` spezifizert wird und die Parameter `xferlog_enable`, `xferlog_std_format` und `log_ftp_protocol` auf "yes" gesetzt sind. In der Log-Datei werden nur Einträge geloggt, wenn der Vorgang mit dem ftp Kommando ausgeführt wird.

- **Log- Datei /var/log/lastlog**

Beschreibung der Quelle:
Die Log- Datei ist eine Binärdatei und wird durch den Vorgang wird die betreffende Zeile des Benutzer abgeändert, wenn der Vorgang durch ein Kommando oder mit der graphischen Oberfläche ausgeführt wird. Damit das letzte Einloggen protokolliert wird, muss in der Konfigurationsdatei `/etc/login.defs` die Parameter LASTLOG_ENAB auf "yes" gesetzt sein.

- **Log- Datei /var/log/utmp:**

Beschreibung der Quelle:
Die Log- Datei ist eine Binärdatei, in der alle eingeloggten Benutzer eingetragen sind und was sie gerade tun. Durch den Vorgang wird die Log- Datei um einen Eintrag erweitert, wenn Vorgang manuell oder durch ein Kommando ausgeführt wird. Die Log- Datei ist auf dem System vorhanden, wenn protokolliert werden soll.

- **Log- Datei /var/log/wtmp**

Beschreibung der Quelle:
Die Log- Datei ist eine Binärdatei und wird durch den Vorgang um einen Eintrag erweitert, wenn der Vorgang durch ein Kommando oder mit der graphischen Oberfläche ausgeführt wird. Die Log- Datei ist auf dem System vorhanden, wenn protokolliert werden soll.

Kapitel 4

Aufbau der Klassifikation

Wichtig ist nun eine gute Darstellung zu finden, die die Informationsquellen, die zu jedem Vorgang ermittelt wurden und dazu gehörige Bewertungskriterien, in Beziehung zu setzen. Wie schon im ersten Kapitel erwähnt, soll die Klassifikation für eine Top Down Analyse, d.h. der Forensiker geht von einem bestimmten Vorgang aus und ermittelt dann die dazugehörigen Informationsquellen aus der Top Down Sicht, verwendet werden. Von der anderen Richtung her, d.h. von den gefundenen Informationsquellen auf dem System auf einen bestimmten Vorgang zu schliessen, wird die Bottom Up Sicht eingeführt.

In der Praxis wird häufiger die Top Down Analyse verwendet. Im LKA München werden Systeme untersucht, bei denen der Zeitpunkt des Vorfalls bzw. des ausgeführten Vorganges (Vorgänge) bekannt ist und es muss geklärt werden, ob der Vorfall stattgefunden hat.

In den zwei folgenden Abschnitten werden die beiden Sichten beschrieben.

4.1 Aufbau der Top Down Sicht:

In der Top Down Sicht wird jeder Vorgang kurz beschrieben und die verschiedenen Methoden angegeben, wie der Vorgang vom Benutzer ausgeführt werden kann. Maximal drei verschiedene Methoden zur Ausführung des Vorganges sind möglich (siehe Abschnitt 3.3.3). Vorgänge, die von root und vom normalen Benutzer ausgeführt werden können und dabei unterschiedliche Informationsquellen ändern, werden getrennt voneinander dokumentiert. Bei jeder einzelnen Methode wird auch angegeben, wie die Kommandos bzw. welcher Botton der graphischen Oberfläche zur Ausführung des Vorganges gedrückt werden muss. Danach folgen die Informationsquellen, die bezüglich des Vorganges ermittelt wurden.

Die Informationsquellen werden durchnummeriert. Bevor die erste Informationsquelle des Vorganges genannt wird, wird das Kriterium Existenz angegeben, das alle Informationsquellen nennt, die unbedingt vorhanden sein müssen, um die Aussage "Der Vorgang hat stattgefunden" treffen zu können. Das Kriterium **Nicht- Existenz** wird nur für Vorgängen verwendet, bei denen Informationsquellen entfernt werden.

Für jede Informationsquelle werden dann das allgemeine Kriterium **Beschreibung der Quelle** und das Bewertungskriterium **Ergiebigkeit** angegeben. Es stellt sich nun die Frage, warum in der Top Down Sicht nur zwei Kriterien angegeben werden. Der Grund dafür ist, dass viele Informationsquellen die gleichen Eigenschaften und so die gleichen Bewertungen haben.

Um sich nicht zu wiederholen und eine gute Lesbarkeit des Dokumentes zu gewährleisten, wird hinter jeder dokumentierten Informationsquelle ein Verweis auf die Bottom Up Sicht, durch eine Seitenzahl, gemacht. Die Bottom Up Sicht gliedert dabei Informationsquellen mit gleichen Eigenschaften und Kriterien. Die Gliederung wird im Aufbau der Bottom Up Sicht im Abschnitt 4.1.3.2 erläutert. In der Bottom Up Sicht befinden sich dann die restlichen Kriterien der Informationsquelle.

4.1.1 Aussagen der Top Down Sicht

Die Top Down Sicht soll dem Forensiker helfen die Informationsquellen schnell aufzufinden, um den Vorfall möglichst schnell zu klären. D.h. nach der Anwendung der Top Down Sicht kann entweder die Aussage "*Der Vorfall hat nicht stattgefunden*" oder "*Der Vorfall hat mit einer Wahrscheinlichkeit von ... % stattgefunden*". Diese Aussage wird noch folgendermaßen unterteilt:

Wortlaut der Aussage	**Wahrscheinlichkeit**
mit indifferenter Wahrscheinlichkeit	**ca. 50 %**
mit überwiegender Wahrscheinlichkeit	**ca. 75 %**
mit hoher Wahrscheinlichkeit	**ca. 90 %**
mit sehr hoher Wahrscheinlichkeit	**ca. 95 %**
mit außerst hoher Wahrscheinlichkeit	**ca. 99 %**
mit an Sicherheit grenzender Wahrscheinlichkeit	**ca. 99.99 %**

Die Wahrscheinlichkeit repräsentiert dabei den Grad der Unsicherheit. Die Bewertung eines Vorfalls mit einer Wahrscheinlichkeit ist die Basis des probabilistischen Schließens. Da ein Vorfall aus mehreren Vorgängen zusammengesetzt ist, muss für jeden einzelnen Vorgang bestimmt werden mit welcher Wahrscheinlichkeit (siehe Abschnitt 4.1.2.) dieser stattgefunden hat. Sind alle Wahrscheinlichkeiten bestimmt, wird die Wahrscheinlichkeit des Stattfindens des Vorfalls durch Mltiplikation der Einzelwahrscheinlichkeiten errechnet. Die Endwahrscheinlichkeit wird dann mit den Werten aus obiger Tabelle verglichen. Ist sie grösser als 50 %, hat der Vorgang mit dieser Wahrscheinlichkeit stattgefunden und die Aussage entsprechend der Wahrscheinlichkeit aus der obigen Tabelle ausgewählt. Ansonsten wird die Aussage getroffen, dass der Vorfall nicht stattgefunden hat.

4.1.2 Verwendung der Top Down Sicht

Dem Forensiker ist der Zeitpunkt bzw. Zeitspanne bekannt, in der der Vorfall (Vorgang) stattgefunden hat. Der Zeitpunkt ist wichtig um Informationsquellen, die Dateien sind, als statisch lesbare, statisch modifizierbare, neu erstellte und ausführbare Dateien zu erkennen. Eine gelesene Datei wird erkannt, wenn die Zeit des "letzten Zugriffs" eine Zeit in der Zeispanne enthält. Das kann der Forensiker mit dem Kommando `stat` `<filename>` herrausfinden. Eine modifizierte Datei wird dadurch erkannt, dass bei ihr die Zeit des "letzten Zugriffs", des "letzten schreibenden Zugriffes" und die "letzte Statusänderung" eine Zeit in der Zeitspanne besitzt. Dabei ist zu unterscheiden, ob der Benutzer sie direkt modifiziert hat, d.h. mit einem nichtgraphischen Texteditor (z.B. vi) editiert hat, dann haben die MAC Zeiten alle den gleichen Zeitpunkt innerhalb der Zeitspanne. Findet die Modifizierung mit einem Kommando statt, also indirekt, sind alle MAC Zeiten gleich. Beispielsweise ändert `mount oder umount` die Datei `/etc/mtab` ab, alle MAC Zeiten sind gleich. Bei den graphischen Texteditoren kwrite, pico sind die drei MAC Zeiten nicht unbedingt die gleichen. Bei dem Texteditor `pico` stimmt die Zeit der "letzten Statusänderung" und des "letzten schreibenden Zugriffs" überein. Beide liegen in der Zeitspanne. Die Zeit des "letzten Zugriffs" hat einen früheren Zeitpunkt als die anderen beiden Zeitstempel. Daraus kann man ableiten, wie lange an der Datei

gearbeitet wurde bis die Datei gespeichert wurde.
Bei neu erstellten Dateien bzw. Verzeichnisse haben die MAC Zeiten alle den gleichen Zeitpunkt innerhalb der Zeitspanne. Wie man nun erkennen kann, kann man neu erstellte Verzeichnisse und Dateien und modifizierte Dateien nur voneinander unterscheiden, falls die modifiziertenn Dateien vom Benutzer mit einem graphischen Texteditor, wie pico, editiert wurden oder ein Kopie der Datei als Backup vor dem Vorfall zur Verfügung steht, um das Kommando `diff <file> <file>.copy` anzuwenden können. Eine modifizierte Datei wird dann dadurch erkannt, wenn das Kommando `diff` eine Ausgabe zurück liefert. Bei ausführbaren Dateien ist die Zeit des "letzten Zugriffes" auf einen Zeitpunkt in der Zeitspanne gesetzt.

Die nachfolgende Tabelle fasst den geschilderten Sachverhalt nochmals zusammen:

neu erstellt	**ausführbar**
m: innerhalb Zeitspanne a: innerhalb Zeitspanne c: innerhalb Zeitspanne wobei m = a = c	m: innerhalb Zeitspanne a: nicht in Zeitspanne c: nicht in Zeitspanne

Tabelle 4.1: Zeitstempel von neu erstellten und ausführbaren Dateien

	modifizierbar
graphischer Editor (pico)	m: innerhalb Zeitspanne a: innerhalb Zeitspanne c: innerhalb Zeitspanne wobei m = c != a
nicht graphischer Editor (vi)	m: innerhalb Zeitspanne a: innerhalb Zeitspanne c: innerhalb Zeitspanne wobei m = a = c
Kommando	m: innerhalb Zeitspanne a: innerhalb Zeitspanne c: innerhalb der Zeitspanne wobei m = a = c

Tabelle 4.2: Zeitstempel von modifizierten Dateien

4.1.2.1 Suchen und Überprüfen der Informationsquellen auf dem zu untersuchenden System

Der Forensiker sucht in der Tabelle (siehe Tabelle 6.1.) in der dazugehörigen Kategorien die Vorgänge aus, die den Vorfall bestimmen und wird dann auf die Seiten verwiesen, auf der die Vorgänge mit den relevanten Informationsquellen dokumentiert sind. Der Forensiker muss dann auf dem zu untersuchenden System für jeden einzelnen Vorgang überprüfen, ob die genannten Informationsquellen mit der entsprechenden Ergiebigkeit auf dem System vorhanden sind. Sind alle die im Kriterium Existenz erwähnten Informationsquellen vorhanden, hat der Vorgang stattgefunden. Ansonsten muss der Forensiker die Wahrscheinlichkeit für das Stattfinden des Vorganges bestimmen. Die Wahrscheinlichkeit ergibt sich aus der Anzahl der gefundenen Informationsquellen geteilt durch die Anzahl der Informationsquellen, die unbedingt existent sein müssen. Welche Informationsquellen nun existent bzw.

nicht existent (z.B. Löschvorgang) sein müssen, gibt das Kriterium **Existenz** bzw. **Nicht- Existenz** an. Bei den Informationsquellen, die existent sein müssen, wird unterschieden, ob bei der Analyse des Systems, es online oder offline ist. Dementsprechend muss der Forensiker die Informationsquellen und deren Ergiebigkeit auf dem zu untersuchenden System bestätigen. Am Ende werden alle Wahrscheinlichkeiten multipliziert, um auf die Endwahrscheinlichkeit für den Vorfall zu kommen. Zuletzt wird die Endwahrscheinlichkeit mit den Werten in der Tabelle verglichen und die Aussage aus der Tabelle getroffen.

Der nachfolgende Pseudocode fast den Ablauf der Top Down Analyse zusammen:

Pseudocode

```
declare anzInf as integer //Anzahl gefundener Informationsquellen auf
//dem System
declare gesamtanzInf as integer //Anzahl der existenten Informationsquellen
declare ws as float //Wahrscheinlichkeit fuer stattgefundenen Vorgang
declare schwellenwert as float

for Vorgang 1 to Vorgang n by 1
            suche Informationsquellen + Ergiebigkeit auf dem System
            ws = anzInf / gesamtanzInf //WS fuer Vorgang i
            ws = ws * ws //WS fuer Vorfall
end

if (ws <0.5)
    print (''Vorfall hat nicht stattgefunden'')
else if (ws >= 0.5 && ws <= 0.75)
         print (''Vorfall hat überwiegender Wahrscheinlichkeit
         stattgefunden'')
else if (ws >0.75 && ws <= 0.9)
         print (''Vorfall hat mit hoher Wahrscheinlichkeit
         stattgefunden'')
else if (ws >0.9 && ws <= 0.95)
         print (''Vorfall hat mit sehr hoher Wahrscheinlichkeit
         stattgefunden'')
else if (ws >0.95 && ws <= 0.99)
         print (''Vorfall hat mit äußerst hoher Wahrscheinlichkeit
         stattgefunden'')
else
         print (''Vorfall hat mit an Sicherheit grenzender Wahrschein-
         lichkeit stattgefunden'')
```

4.1.3 Implementierung der Top Down Sicht

Die Top Down Sicht kann in ein Programm verwandelt werden, dass die Analyse automatisiert. Der Forensiker lässt auf einem Rechner, der das Dateisystem der gesicherten Festplatte enthält das Programm laufen. Der Forensiker muss dabei den Vorfall, aufgegliedert in die einzelnen Vorgänge, eingeben und eine Zeitspanne in der der Vorfall angeblich stattgefunden hat. Das Programm gibt dann eine der beiden Aussage zurück, die in Abschnitt 4.1.1. erwähnt werden.

4.1.3.1 Speicherung der Vorgänge

Jeder Vorgang wird mit seinen Informationsquellen und der entsprechenden Ergiebigkeit in einer seperaten Datei abgespeichert. In jeder Datei befindet sich der Name der Informationsquelle und die drei Zeitstempel, falls es sich um Dateien bzw. Verzeichnisse handelt. Jede Datei ist dann ein ausführbares Perl Script und wird vom Hauptprogramm aus aufgerufen. Das Perl Script sucht die Informationsquelle auf dem zu untersuchenden System und vergleicht, falls die Informationsquelle vorhanden ist, dessen Ergiebigkeit mit der Ergiebigkeit der dokumentierten Informationsquelle. Für jeden Vorgang schaut das Perl Script anders aus. Das liegt an den verschiedenen Informationsquellen und den Parametern (Options), die beispielsweise beim Aufruf eines Kommandos angegeben werden. Die Zeitstempel in jedem Perl Script müssen jenachdem, ob die Informationsquelle, neu erstellt, modifiziert oder ausgeführt wurde, gesetzt werden. Die werden durch den Zeitpunkt oder Zeitspanne, die dem Forensiker bekannt ist, nach Tabelle 4.1. und 4.2. gesetzt.

Pseudocode

```
//Perl Script Vorgang i
declare anzInf as integer //Anzahl gefundener Informationsquellen
//auf dem System
declare gesamtanzInf as integer //Anzahl der Informationsquellen die auf dem
//System vorhanden sein muessen
declare erg as bool //Ergiebigkeit gleich
//Falls die Informationsquelle eine Datei ist werden im Perl Script die drei
//Variablen mac für die MAC Zeiten auf den Zeitpunkt des Vorfalles gesetzt
... //Ist der Aufenthaltsort der Datei variabel oder wird durch eine be-
//stimmten Parameter in einer Konfigurationsdateien festgelegt, muss die Kon-
//figurationsdatei gelesen werden
... //Um an bestimmte Aufrufparameter des Kommandos für einen Vorgang zu-
//kommen, wird beispielweise die History Liste ausgelesen
...

for Informationsquelle 1 to Informationsquelle n
    suche Informationsquelle auf dem zu untersuchenden System
    if (gefunden) und (Ergiebigkeit)
        anzInf = anzInf + 1
end
    return anzInf/ gesamtanzInf
```

4.1.3.2 Hauptprogramm

Das Hauptprogramm nimmt die Zeitspanne, die einzelnen Vorgänge, die den Vorfall zusammen setzen entgegen. Das Hauptprogramm ruft dann für jeden Vorgang sein Perl Script auf, die dann die Informationsquellen und deren Ergiebigkeit mit den Informationsquellen auf dem zu untersuchenden System vergleichen. Dabei wird dann noch jeweils die Wahrscheinlichkeit berechnet, mit der der einzelne Vorgang stattgefunden hat. Das Hauptprogramm gibt dann zuletzt die Aussage, die zu der berechneten Wahrscheinlichkeit aus der Tabelle entspricht zurück.

Pseudocode

```
declare ws as float //Wahrscheinlichkeit fuer stattgefundenen Vorgang
declare schwellenwert as float
Input (Zeitspanne)
Input (Vorgaenge 1 bis n)
for Vorgang 1 to Vorgang n
do  rufe Perl Script fuer jeden Vorgang i auf
    ws = ws * Output (Perl Script Vorgang i)
end
    if (ws < 0.5)
        print (``Vorfall hat nicht stattgefunden'')
    else if (ws >= 0.5 && ws <= 0.75)
             print (``Vorfall hat überwiegender Wahrscheinlichkeit
             stattgefunden'')
    else if (ws > 0.75 && ws <= 0.9)
             print (``Vorfall hat mit hoher Wahrscheinlichkeit
             stattgefunden'')
    else if (ws > 0.9 && ws <= 0.95)
             print (``Vorfall hat mit sehr hoher Wahrscheinlichkeit
             stattgefunden'')
    else if (ws > 0.95 && ws <= 0.99)
             print (``Vorfall hat mit äußerst hoher Wahrscheinlichkeit
             stattgefunden'')
    else
             print (``Vorfall hat mit an Sicherheit grenzender Wahrschein-
             lichkeit stattgefunden'')
```

4.2 Aufbau der Bottom Up Sicht

Für die Bottom Up Sicht werden zwei Darstellungsmöglichkeiten gewählt:

1. **Einteilung in Verzeichnisse**
2. **Darstellung als Graph**

4.2.1 Einteilung in Verzeichnisse:

Die Informationsquellen, überwiegend Dateien werden aufgrund ihres Aufenthaltsortes im Dateisystem, d.h. der vollständige Pfad zur Datei im Dateisystem, eingeordnet. Die Dateien werden dabei in statisch lesbare, statisch modifizierbare, ausführbare und neu erstellte Informationsquellen eingeteilt. Viele Dateien kommen in mehreren Tabellen für statisch modifizierbar und statisch lesbar vor. Es gibt zwei Gründe dafür: Die Bottom Up Sicht wird aus der Top Down Sicht von den Informationsquellen her erstellt. Ein und die selbe Informationsquelle kann von einem Vorgang nur gelesen werden und von einem anderen Vorgang wird sie modifiziert. Die restlichen Informationsquellen, die gleiche Eigenschaften und Kriterien haben, werden folgendermaßsen zusammengefasst:

- **Verzeichnisse und Unterverzeichnisse**
  Dort befinden sich die Dateien, deren Aufenthaltsort fest vorgegeben ist. Dazu gehören beispielsweise die Konfigurationsdateien im `/etc` Verzeichnis. Die meisten Dateien aus diesem Verzeichnis gehören zum Basissystem.

- **Environment Variablen**:
  Bei den Environment Variablen wird zwischen benutzerspezifischen und programmspezifischen unterschieden.

- **Datei −/ Verzeichnisattribute**
  Die Verzeichnisattribute werden zu den Dateiattributen zusammenfasst, da Verzeichnisse im engeren Sinne auch Dateien sind.

- **Sonstige Dateien: / <path>/ <filename>variabel**
  Unter sonstige Dateien werden Dateien zusammengefasst, deren Aufenthaltsort im Dateisystem variabel ist, d.h. der Benutzer sei es root oder ein normaler Benutzer kann den Aufenthaltsort selbst festlegen. Dazu zählen zum Beispiel reguläre Dateien.

- **Sonstige Verzeichnisse: / <path>/ <filename>variabel**
  Zu den sonstigen Verzeichnissen zählen Verzeichnisse, die in keine der oben genannten Verzeichnisse und Unterverzeichnisse passen.

Für die die zusammengefassten Informationsquellen werden die restlichen Kriterien, wie **Entstehungsbedingungen**, **Modifikationsmöglichkeit**, **Auswertungsaufwand**, **Auswertung**, **Lebensdauer**, **Möglichkeit der Fehlinterpretation** und **Fälschungsmöglichkeit**, angewendet. Das Kriterium **Verweis auf andere Vorgänge** wird bei den Informationsquellen nur erwähnt, falls die Informationsquelle durch mehrere Methode modifiziert wird. Die verschiedenen Methode wurden schon in vorhergehenden Abschnitten erwähnt.

4.2.2 Darstellung als Graph:

Ein Baum, der eine Wurzel besitzt, die eine Informationsquelle ist, kann zur Darstellung der Bottom Up Sicht nicht verwendet werden, da der Einstiegspunkt d.h. die erste gefundene Informationsquelle nicht immer die gleiche ist. Deshalb wird als Darstellung ein Graph verwendet, der auf jeder Ebene Knoten (vertices) besitzt, die die Informationsquellen darstellen. Zwischen den Ebenen befinden sich Kanten, die die Ergiebigkeit der Informationsquelle darstellen sollen. Diese sind beschriftet. Für den Graphen werden nur die Informationsquellen verwendet, die bei der Bottom Up Sicht bei dem Kriterium **Existenz** genannt werden. Es werden dadurch einige Informationsquellen nicht betrachtet, die für den Forensiker keine relevante Ergiebigkeiten haben. Jeder Vorgang und die dazugehörigen Informationsquellen werden als einzelne Teilgraphen dargestellt, wobei dann der Gesamtgraph, der dann alle Vorgänge mit den Informationsquellen enthält, aus diesen aufgebaut wird.

4.2.2.1 Aufbau des Graphen

Auf Höhe 0 des Gesamtraphen befinden sich alle Informationsquellen, die insgesamt gefunden worden sind, aber auch dann nur diese die bei jedem Vorgang existent sein müssen. Die Höhe h - 1 des Graphen ist durch die größte Anzahl von Informationsquellen eines Vorganges beschränkt. Auf jeder Höhe des Graphen befinden sich die gleiche Anzahl von Knoten. Die Knoten im Gesamtgraphen werden folgendermaßen nummeriert:

- Durch Tabellierung einer bijektiven Abbildung F: V $\longrightarrow$ n werden die Informationsquellen von allen Vorgängen auf die natürlichen Zahlen abgebildet.
- Die Knoten auf der Höhe 0 werden dort aufsteigend dargestellt
- Jede weitere Ebene i wird jeder nummerierte Knoten um i - 1 nach links permutiert.

Der Gesamtgraph in Abbildung 4.1. hat eine Höhe von 2 und gibt es insgesamt vier verschiedene Informationsquellen. Am Anfang besteht der Gesamtgraph nur aus Knoten.

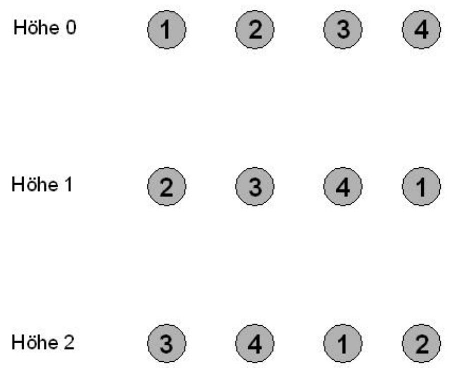


Abbildung 4.1: Leerer Graph

Ein einzelner Vorgang wird folgendermaßen dargestellt:

- Aus der Tabelle (siehe Tabellierung der Informationsquellen vom obigen Abschnitt) werden für den Vorgang die Zahlen für die jeweiligen Informationsquelle bestimmt.

- Die Knoten auf der Höhe 0 werden dort aufsteigend dargestellt
- Jede weitere Ebene i wird jeder nummerierte Knoten um i - 1 nach links permutiert.

Die nachfolgende Abbildung zeigt die Graphen einzelner Vorgänge.

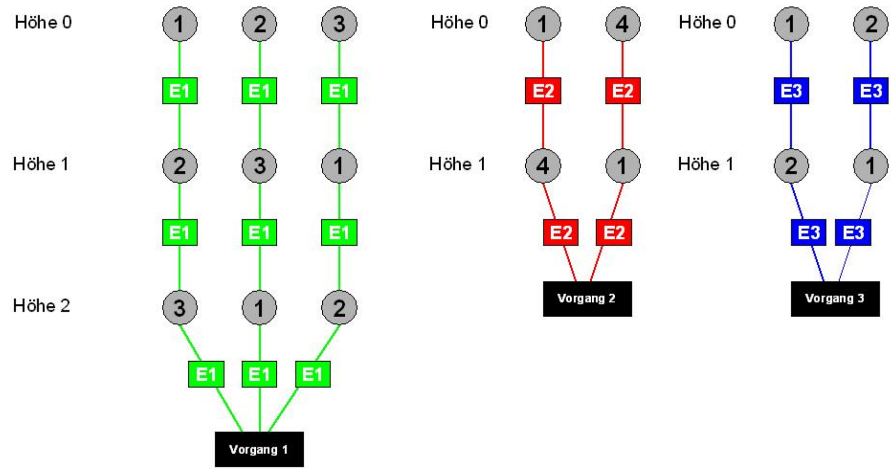


Abbildung 4.2: Graphen einzelner Vorgänge

Um zu dem fertigen Graphen aller Vorgänge in Abbildung zu kommen müssen die Kanten, die in in jedem einzelnen Graphen vorkommen in den Gesamtgraphen übertragen werden.

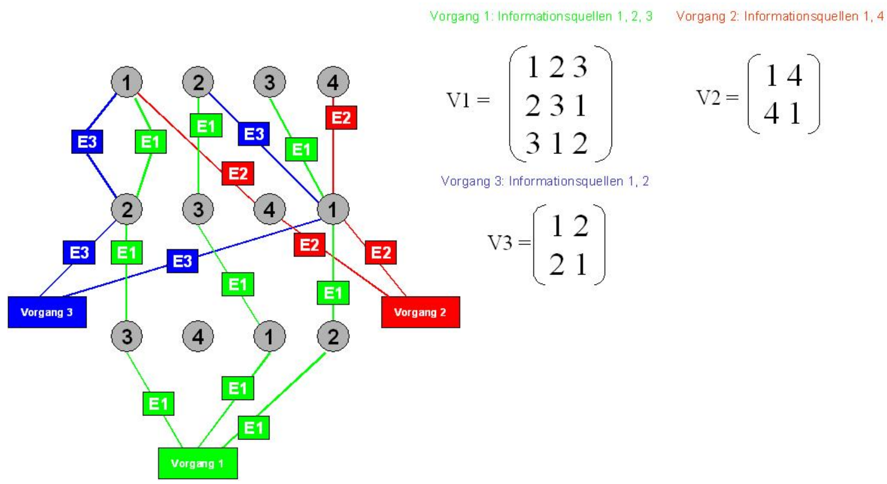


Abbildung 4.3: Graph aller Vorgänge

Auf jeder Ebene befinden sich jeweils die vier Informationsquellen. Die maximale Höhe h - 1 des Graphen ist zwei, da Vorgang 1 die größte Anzahl von Informationsquellen besitzt, nämlich 3. Die Knoten werden über Kanten so miteinander verbunden, dass sich auf dem entstehenden Pfad, der im Knoten Vorgang endet, nur die Informationsquellen befinden, die zum Vorgang gehören. Aus der Abbildung 4.2. wird ersichtlich, dass im Vorgang 1 drei Pfade enden. Grund dafür ist die Darstellung eines einzelnen Vorganges als Graph. Durch die Permutation der Knoten auf jeder Ebene entstehen beispielsweise für den Vorgang 1 folgende Pfade $1 \rightarrow 2 \rightarrow 3$, $2 \rightarrow 3 \rightarrow 1$ und $3 \rightarrow 2 \rightarrow 1$ (siehe Abbildung 4.2.). Dies wird durch eine Matrix dargestellt. In Abbildung 4.2. ist die jeweilige Matrix des Vorganges dargestellt. Die Permutation, der Informationsquellen hat den Vorteil, dass keine Kanten in der Matrix gespeichert werden müssen. Ein Pfad aus der Matrix wird spaltenweise von der ersten Informationsquelle der Spalte zur nächsten darunteren liegenden Informationsquelle etc. erstellt. Aus einem Knoten führen mehrere Kanten herraus, da Vorgänge auch die gleiche (n) Informationsquelle (n) haben können. Die herrausführende Kante wird beschriftet. Zur Beschriftung wird der Buchstabe E für Ergiebigkeit und eine natürliche Zahl gewählt. Die natürliche Zahl entspricht der Nummer des Vorganges, zu der die Informationsquelle mit ihrer Ergiebigkeit gehört (vorgangsbezogen). Mit Ergiebigkeit sind nicht die Werte gut, mittel und schlecht gemeint (siehe das Kriterium **Ergiebigkeit** Abschnitt 3.5.). Sondern z.B. bei Konfigurationsdateien, der neu hinzugekommene Eintrag nach der Ausführung des Vorganges und desen enthaltenen Informationen. Ist die Ergiebigkeit der gleichen Informationsquelle unterschiedlicher Vorgänge gleich, kann diese nicht sofort eindeutig einem Vorgang zugeordnet werden. Sind wiederum die Ergiebigkeiten der gleichen Informationsquelle mehrerer Vorgänge unterschiedlich, kann diese eindeutig einem Vorgang zugeordnet werden. Die Ergiebigkeiten einer Informationsquelle werden in der Tabelle 4.3. festgehalten. Die erste Informationsquelle `/etc/passwd` beispielsweise hat zwei Kanten, nämlich E1 und E3. Das bedeutet, dass die Informationsquelle mit der gleichen Ergiebigkeit von den Vorgängen 1 und 3 geändert wird.

Nr.	**Informationsquelle**	**Ergiebigkeit**	**Kante**
1	Eintrag in /etc/passwd	uid, gid, home, gecos, shell	E1, E3
2	Feld in /etc/group	Mitgliederliste Gruppenverwalter	E1 E3
3	Eintrag in /var/log/messages	<time> <host> useradd[<pid>]: newuser: name=<newuser>, uid=<new uid>, gid=<gid>, home=<home directory for newuser>, shell=<shell> from /etc/shells>	E2

Tabelle 4.3: Graphentabelle

4.2.2.2 Aussagen der Bottom Up Sicht

Die Bottom Up Sicht soll dem Forensiker anhand von gefundenen Informationsquellen helfen, den stattgefundenen Vorgang zu bestimmen und auf weitere Informationsquellen verweisen, die zu vermuteten Vorgäng gehören. Diese Sicht trifft entweder die Aussage *"Der Vorgang X hat stattgefunden"*, *"Möglicherweise war es Vorgang X* oder *Es kommen Vorgang 1..m in Frage"*. Mehrere Vorgänge kommen kommen nur in Frage, wenn sie die gleiche Anzahl von Informationsquellen mit der gleichen Ergiebigkeit haben.

4.2.3 Verwendung der Bottom Up Sicht:

Der Forensiker hat eine Informationsquelle auf dem System gesichert. Ist die Informationsquelle eine Datei muss der Forensiker feststellen, ob sie statisch lesbar, statisch modifizierbar oder ausführbar ist. Die weitere Vorgehensweise mit dem Umgang von Dateien wurde in Abschnitt 4.1.2. bei der Verwendung der Top Down Sicht geschildert.

Es ist meistens nicht möglich eine statisch gelesene oder modifizierte Datei genau einem Vorgang zuzuordnen. Es werden weitere Informationsquellen gebraucht, um den Vorgang eindeutig bestimmen zu können. Bei anderen Informationsquellen wie Dateiattribute es auch nicht möglich den zugehörigen Vorgang eindeutig zu bestimmen. Um an weitere Informationsquellen zu gelangen, kann der Forenisker entweder die Bottom Up Sicht in Form eines Graphen oder die Bottom Up Sicht, die die Informationsquellen nach Aufenthaltsort im Dateisystem anordnet, wählen. In den beiden folgenden Abschnitten werden die Abläufe der beiden Top Down Sichten geschildert.

1. Verwendung der Bottom Up Sicht Einteilung in Verzeichnisse (siehe Abschnitt 7.1.)

Ablauf

Der Forensiker muss die erste Informationsquelle in der Einteilung in Verzeichnisse gemäß der Unterteiluung in Abschnitt 4.2.1. suchen. Bei Dateien wie bereits oben schon erwähnt, muss der Forensiker feststellen, ob sie ausgeführt, modifiziert, neu erstellt oder nur gelesen wurden. Die Verzeichnisse sind nochmals in statisch modifizierbare bzw. neu erstellte, statisch lesbare und ausführbare Dateien eingeteilt. Der Forensiker muss dann die richtige Unterteilung wählen, um auf die richtigen Seiten verwiesen zu werden. Neben der Informationsquelle stehen Verweise auf verschiedene Vorgänge über eine Seitenzahl. Der Forensiker begibt sich somit wieder in die Top Down Sicht. Die erste Informationsquelle ist dafür verantwortlich, ob nur ein Vorgang oder gegebenenfalls mehrere Vorgänge in Frage kommen. Gibt es die Informationsquellen mit ihrer dazugehörigen Ergiebigkeit nur bei einem Vorgang, muss der Forensiker nur die bei diesem Vorgang dokumentierten Informationsquellen auf dem System suchen. Der Forensiker bricht dann die Suche ab, wenn er keine Informationsquelle mit der passenden Ergiebigkeit findet oder alle Informationsquellen überprüft hat. Dies entspricht dem Schritt 3.(a). Im anderen Fall gibt es mehrere Vorgänge, die die gleiche Informationsquelle mit der gleichen Ergiebigkeit haben. Diese Vorgänge werden im Laufe der Analyse nur verwendet. Für jeden dieser Vorgänge untersucht der Forensiker die noch nicht überprüften Informationsquellen und deren Ergiebigkeit auf dem System. Kommt nun nur noch Vorgang mit der Ergiebigkeit in Frage, bleibt der Forensiker bei diesem Vorgang (zurück zu Schritt 3.(a)). Die Analyse endet dann auch so wie oben, wenn nur ein Vorgang in Frage kommt. Kommen aber wieder mehrere Vorgänge unter den schon in Frage kommenden Vorgängen mit der gleichen Ergiebigkeit vor, muss der Forensiker die noch nicht überprüften Informationsquellen und deren Ergiebigkeit auf dem System überprüfen (Schritt 3.(b)). Dies kann so lange weitergehen bis nur noch ein Vorgang übrig bleibt (Schritt 3.(a)), keine Übereinstimmung mit der Ergiebigkeit stattfindet bzw. die Informationsquelle ist garnicht vorhanden oder alle Informationsquellen positiv überprüft wurden. Dann kommen alle diese Vorgänge, die als letztes positiv überprüft wurden in Frage (Schritt 3.(b)C). Die Unterscheidung von einem und mehr als einem Vorgang ist nötig, wenn mehrere in Frage kommende Vorgänge die gleiche Anzahl von Informationsquellen und die gleiche Ergiebigkeit haben und die nächste Informationsquelle nur bei einem Vorgang vorkommt. Die anderen Vorgänge kommen dann nicht mehr in Frage.

Die Vorgehensweise wird im nachfolgenden in einzelne Schritte unterteilt:

1. Suche den Namen der ersten sichergestellten Informationsquelle aus der Bottom Up Sicht Abschnitt 7.1. heraus
2. Nimm Verweise, die dort angegeben sind und blättere zu den angegeben Seiten, die einzelnen Vorgängen entsprechen
3. Vergleiche für alle angebenen Vorgänge die Ergiebigkeiten der sichergestellten Informationsquelle
   (a) Falls es nur einen Vorgang gibt (d.h. es gibt nur einen Vorgang mit dieser Informationsquelle und mit dieser Ergiebigkeit)
      - Bleibe bei diesem Vorgang und überprüfe alle Informationquellen mit der dazugehörigen Ergiebigkeit
      - Erhöhe bei positiver Überprüfung die Anzahl der gefundenen Informationsquellen um eins
      - Gehe dann zu 4., falls keine Informationsquelle mit der passenden Ergiebigkeit mehr gefunden wird oder alle Informationsquellen schon überprüft wurden

   (b) Falls es mehr als einen Vorgang gibt (Das Überprüfen von Informationsquellen beschränkt sich in späteren Schritten nur auf diese Vorgänge)
      - Mache folgendes für alle in Frage kommenden Vorgänge i
        – Erhöhe die Anzahl der gefundenen Informationsquellen des Vorganges i um eins
        – Gehe noch nicht untersuchte Informationsquellen des Vorganges i durch

      i. Falls Informationsquelle gefunden
         - Überprüfe nur in den oben in Frage kommenden Vorgängen die Ergiebigkeit der Informationsquelle

         A. Kommen mehr als ein Vorgang von diesen Vorgänge in Frage
            – Gehe nach 3. (b)

         B. Kommt nur ein Vorgang in Frage
            – Gehe nach 3. (a)

         C. Falls Ergiebigkeit nicht übereinstimmt
            – Merke alle Vorgänge die in 3. (b) noch untersucht wurden
            – Gehe zu 4.

      ii. Falls Informationsquelle nicht gefunden
         - Merke alle Vorgänge die in 3. (b) noch untersucht wurden
         - Gehe zu 4.

4. Aussagen
   (a) Falls nur ein Vorgang X, bei dem die Anzahl der gefundenen Informationsquellen gleich der maximalen Anzahl von übereinstimmenden Informationsquellen des Vorganges X aus der Top Down Sicht ist, existiert
   Vorgang X hat stattgefunden

(b) `Falls nur ein Vorgang X, bei dem die Anzahl der gefundenen Informationsquellen kleiner als die maximale Anzahl von übereinstimmenden Informationsquellen des Vorganges X ist, existiert`
Vorgang X hat möglicherweise stattgefunden

(c) `Mehr als einen gemerkten Vorgang`
Vorgänge 1 .. m kommen in Frage

2. Verwendung der Bottom Up Sicht Darstellung als Graph

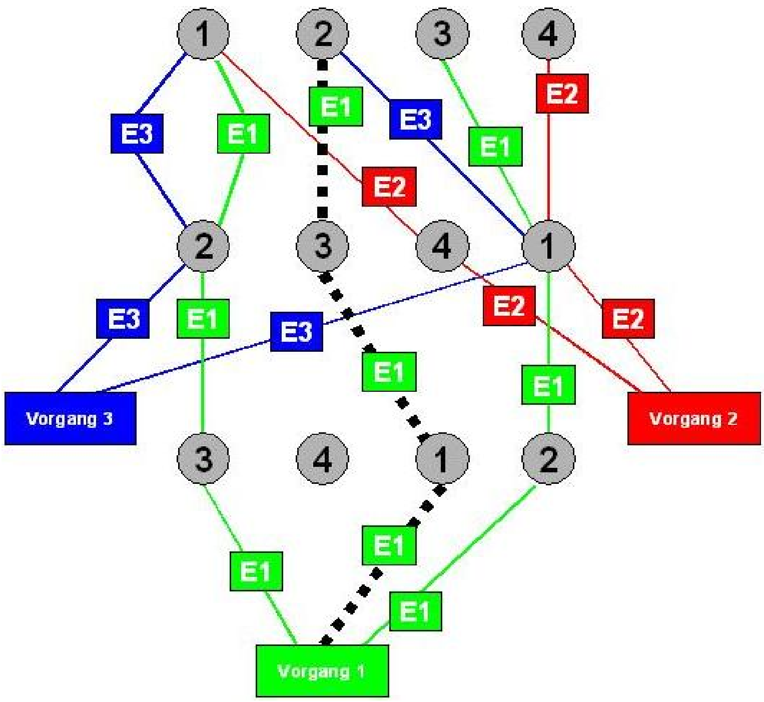
Ablauf

Die Vorgehensweise ist die gleiche wie bei Verwendung der Bottom Up Sicht *Einteilung in Verzeichnisse*, wobei der Forensiker den Graphen und die Tabelle 4.3. verwendet. Der Forensiker arbeitet den Graphen von oben nach unten ab. Er startet bei Höhe 0 im Graphen mit der ersten sichergestellten Informationsquelle. Die Vorgänge, die bei der Bottom Up Sicht in Frage kommen, sind hier die Kanten, die vorgangsbezogen beschriftet sind. Die Unterscheidung von genau einem Vorgang und mehreren in Frage kommenden Vorgänge, die die gleiche Informationsquelle mit der dazugehörigen Ergiebigkeit haben, wird hier auf verschiedene Kanten, die zwei Informationsquellen verbinden.

Nach Anwendung des Graphen kann eine Aussage der drei folgenden Aussagen getroffen werden:

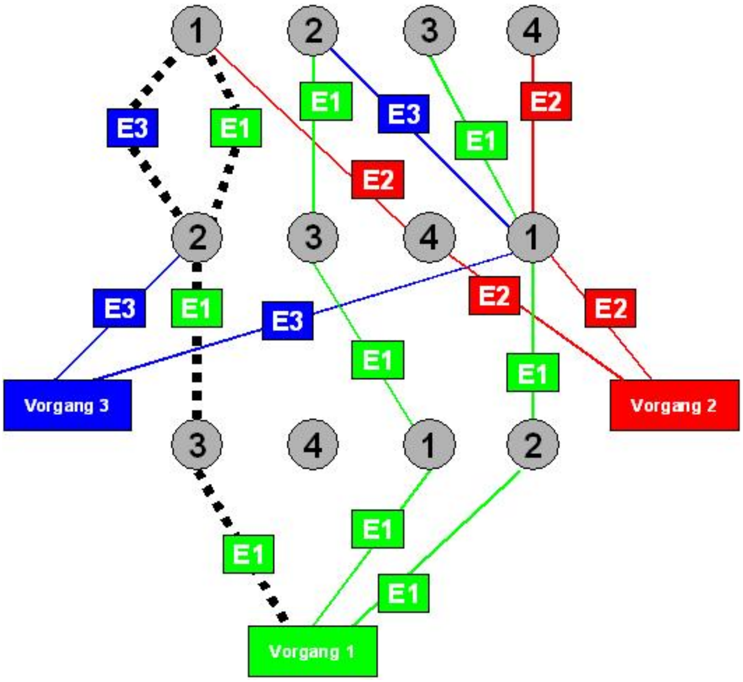
1. **Vorgang X hat stattgefunden:**
   Es existiert genau ein Pfad mit der gleichen Kantenbeschriftung von einem Knoten auf Höhe 0 des Graphen zu einem Blatt. Das Blatt stellt den Vorgang X dar.

   Beispiel 1:
   Die Informationsquellen 2, 3, und 1 wurden auf dem System mit den



   entsprechenden Ergiebigkeiten gefunden. Der Pfad, der durch die schwarze gestrichelte Linie dargestellt wird, endet in dem Blatt von Vorgang 1. Die Aussage "Der Vorgang 1 hat stattgefunden" wird getroffen.

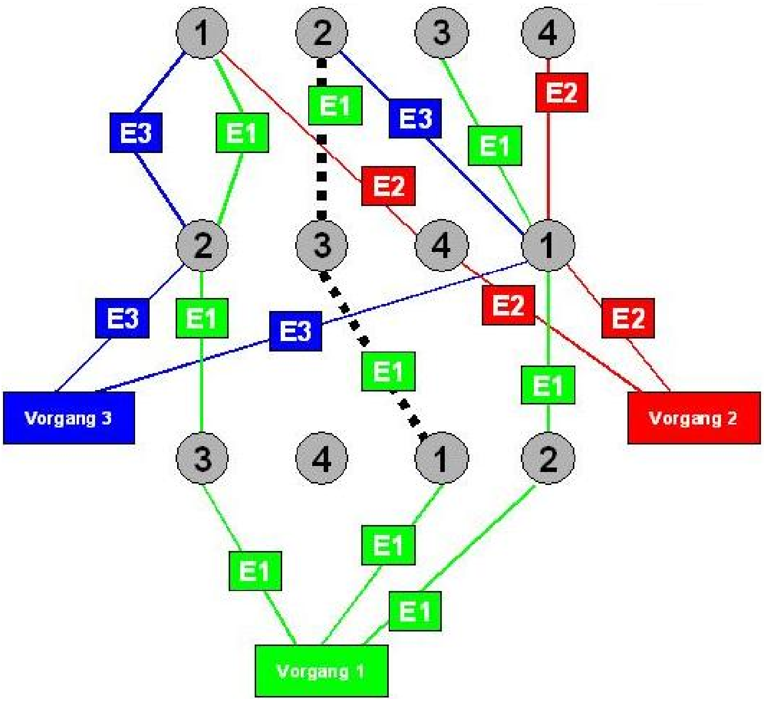
Beispiel 2:
Es ist auch möglich, dass anfangs mehrere Vorgänge in Frage kommen, diese



aber dann zurück gewiesen werden, da eine weitere Informationsquelle und die dazugehörige Ergiebigkeit nur genau zu einem Vorgang zu geordnet werden kann. Hier in Beispiel 2 haben die Vorgänge 1 und 3 die Informationsquelle 1 mit der dazugehörigen Ergiebigkeit gemeinsam. Jedoch findet der Forensiker, die beiden Informationsquellen 2 und 3 mit den passenden Ergiebigkeiten auf dem System. Vorgang 3 kommt dann nicht mehr in Frage.

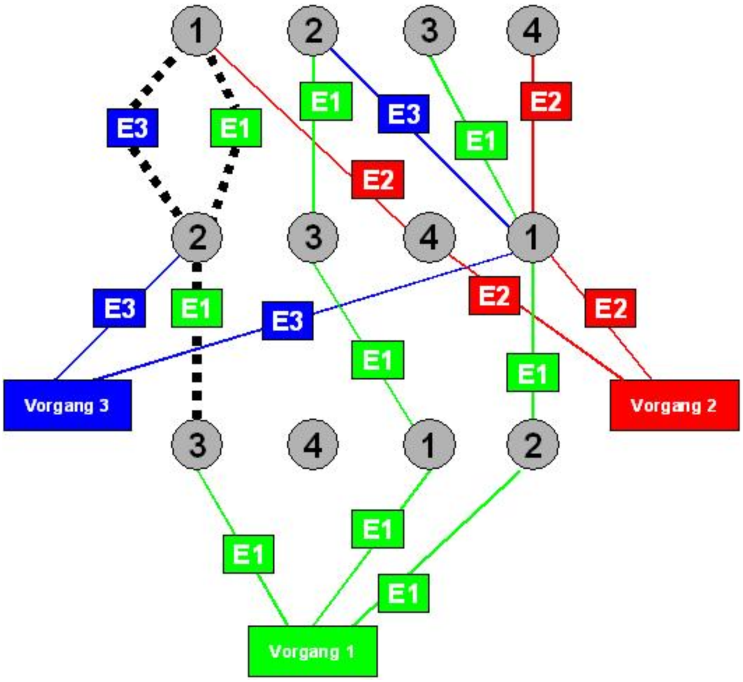
2. Möglicherweise war es Vorgang X:
Es existiert nur ein Pfad (gleiche Kantenbeschriftung). Der Pfad endet in einem internen Knoten. Eingehende Kante der letzten positiv überprüften Informationsquelle mit bestimmter Kantenbeschriftung ordnet den Vorgang X zu. Je weiter unten im Graphen der Pfad endet, desto grösser ist die Wahrscheinlichkeit, dass der Vorgang stattgefunden hat.

Beispiel 3:
Die Informationsquellen 2, 3 wurden auf dem System mit den entspre-



chenden Ergiebigkeiten gefunden. Die letzte Informationsquelle 1 wurde auf dem System nicht gefunden. Der Pfad durch die schwarze gestrichlete Linie dargestellt wird, endet nicht in dem Blatt von Vorgang 1. Die Aussage “Der Vorgang 1 hat möglicherweise stattgefunden” wird getroffen.

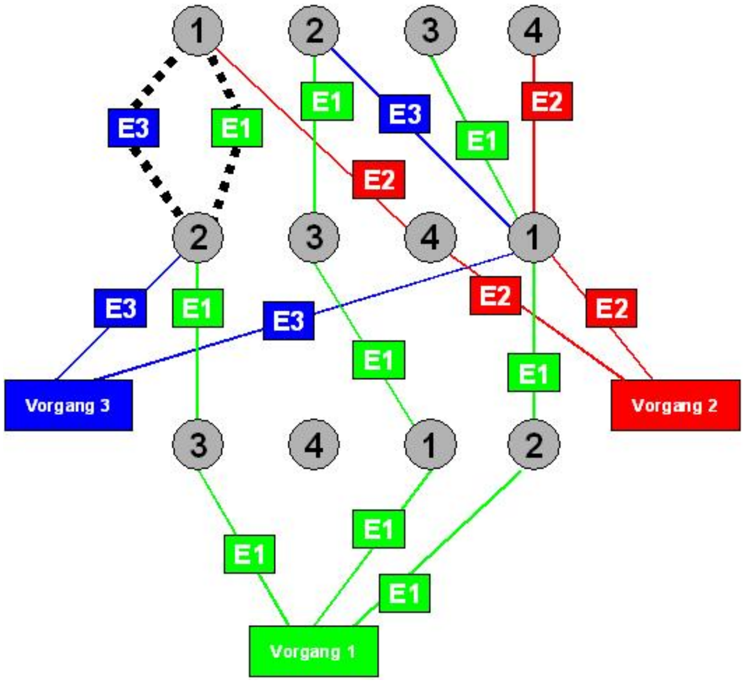
Beispiel 4:
Es ist auch möglich, dass anfangs mehrere Vorgänge in Frage kommen, diese



aber dann zurüch gewiesen werden, da eine weitere Informationsquelle und die dazugehörige Ergiebigkeit genau zu einem Vorgang zu georndet werden kann. Hier in Beispiel 4 haben die Vorgänge 1 und 3 die Informationsquelle 1 mit der dazugehörigen Informationsquelle gemeinsam. Jedoch findet der Forensiker, nur die Informationsquellen 2 auf dem System. Vorgang 1 kommt dann nicht mehr in Frage.

3. Mehrere Vorgänge kommen in Frage:
Es existieren mehrere Pfade mit Kantenbeschriftungen, vom selben Knoten der Höhe 0 des Graphen und enden im gleichen Blatt oder im gleichen internen Knoten. Die Kantenbeschriftung ist dabei unterschiedlich und weißt somit auf die in Frage kommenden Vorgänge hin.

Beispiel 5:
Die Informationsquelle 1 mit der dazugehörigen Ergiebigkeit stimmt bei



den Vorgängen 1 und 3 überein. Die Informationsquelle 2 auf der nächsten darunterliegenden Ebene stimmt mit Ergiebigkeit bei beiden nicht überein oder die Informationsquelle wird auf dem System garnicht gefunden.

Kapitel 5

Implementierung des Graphen

In diesem Kapitel wird beschrieben welche Datenstruktur für den Graphen verwendet wird und es wird ein Pseudocode angegeben, wie die Bottom Up Analyse mit Hilfe des Graphen automatisiert werden kann.

5.1 Interne Darstellung des Graphen

5.1.1 Defintion:

Der gesamte Graph wird intern nicht als gesamter Graph sondern als einzelne Teilgraphen dargestellt. Die Teilgraphen sind die einzelnen Vorgänge mit den dazugehörigen Informationsquellen als Knoten. Zur internen Darstellung der Teilgraphen wird ein dreidimensionaler Array verwendet. Die erste Dimension stellt die Nummer des Vorganges bzw. Nummer des Teilgraphen dar, die zweite und dritte Dimension die Informationsquellen des einzelnen Vorganges. Es werden zwei Dimensionen für die Informationsquellen verwendet, um die möglichen Pfade (Permutationen der Informationsquellen) als Graph darzustellen. Kanten werden nicht direkt gespeichert. Durch die Speicherung der einzelnen Informationsquellen, d.h. dem Namen nach, in der Matrix, wird durch das spaltenweise Lesen der Matrix von der ersten Zeile zur letzten Zeile, die verschiedenen Pfade angegeben. Graphen können im Rechner auch durch Adjazenzmatrizen und Adjazenzlisten dargestellt werden. Eine Adjazenzmatrix ist eine n x n Matrix A = a[i][j], für $1 \leq$ i, j $\geq$ n wobei n die Anzahl der Knoten ist. Dabei ist bei einem ungerichteten Graphen G = (V, E) genau dann a[i][j] = 1, wenn i und j adjazent sind, d.h. {i, j} $\in$ E. Um den Graphen mit n Knoten und m Kanten zu speichern bräuchte man immer quadratischen Platz. Würde man den gesamten Graphen als Adjazenmatrix speichern, würde diese sehr groß werden. Deshalb wird sie für die Darstellung des Vorgangsgraphen nicht verwendet. Eine andere Möglichkeit wäre den Graphen als Adjazenzliste zu speichern. Hierbei werden die zu einem Knoten v adjazenten Knoten in einer linearen Liste gespeichert. Entweder werden in diesen Listen die Namen der adjazenten Knoten (d.h. die Zahlen aus [0 : n-1]) gespeichert oder, meist besser, nur der Verweis auf den Knoten selbst. Bei den Ajazenzlisten braucht man nur linearen Speicherplatz. Operationen, wie z.B. (i, j) $\in$ E ?, braucht O(n), da im schlimmsten Fall die gesamte Adjazenzliste des Knotens i durchlaufen werden muss. Deshalb kommen Adjazenzlisten hier auch nicht in Frage.

5.1.2 Datenstruktur für den Graphen

Die Datenstruktur für den Graphen könnte beispielsweise folgendermaßen aussehen:

```
struct knoten
{
       int anzinfo; //Anzahl gefundener Informationsquellen
       char nameInfo[100]; //Name
       char nameVorgang [100]; //Name des Vorganges
       char ergiebigkeit[100]; //Ergiebigkeit der Informationsquelle
       int maxinfo; //Gesamtanzahl von Informationsquellen des einzelnen
       //Vorganges
       int typ; //statisch lesbar oder modifizierbar
} Graph [m][n][n];

int anzVorgang; // Anzahl der Vorgänge
int maxAnzInfo; // maximale Anzahl von Informationsquellen
```

Die Ergiebigkeit jeder Informationsquelle wird als String dargestellt, wobei nicht die Werte *gut, mittel, schlecht* wie sie bei den Kriterien festgelegt sind stehen. Beispielsweise steht bei den statisch modifizierten Konfigurationsdateien der Eintrag mit den dazugehörigen Informationen, der nach Ausführung des Vorganges neu hinzugekommen ist, als Ergiebigkeit.

Da es vorkommen kann, dass mehrere Vorgänge die gleiche Informationsquelle mit der gleichen Ergiebigkeit haben, wird eine Liste gebraucht, die die jeweiligen Vorgänge speichert und an welcher Position im Graphen (Spalte), sich die Informationsquelle befindet.

Die Datenstruktur für die Liste sieht folgendermaßen aus:

```
struct Liste
{
       int nummervorgang; //Nummer des Vorganges
       int xposition; //Spalte in der Matrix mit bestimmter Informationsquelle
} L[anzVorgang];
```

5.1.3 Speichern der Vorgänge und der dazugehörigen Informationsquellen

Jeder Vorgang muss einzeln gespeichert werden. Für alle Strukturattribut werden für die einzelnen Informationsquellen, als Permutation um jeweils ein Element pro Zeile eingetragen. Die restlichen Elemente werden auf Null gesetzt, da nicht jeder Vorgang die gleiche Anzahl an Informationsquellen hat. Die nachfolgende zweidimensionale Matrix zeigt einen Vorgang mit drei Informationsquellen. Die anderen Strukturattribute werden dementsprechend eingetragen. Jede Zeile enthält die gleichen Informationen, nur dass sie an anderen Positionen innerhalb der Zeile auftreten.
Folgende Matrix zeigt für Vorgang 1 die Ergiebigkeit jeder einzelnen Informationsquelle 1 - 4.

Graph[1][n][n].ergiebigkeit =

$$\begin{pmatrix} <uid>, <gid> & <Mitglieder> & <username> & E4 & 0 \\ <Mitglieder> & <username> & E4 & <uid>, <gid> & 0 \\ <username> & E4 & <uid>, <gid> & <Mitglieder> & 0 \\ E4 & <uid>, <gid> & <Mitglieder> & <username> & 0 \\ 0 & 0 & 0 & 0 & 0 \end{pmatrix}$$

Für die Ergiebigkeit E4 der /var/log/messages würde folgendes <time> <host> useradd[<pid>]: newuser: name=<newuser>, uid=<new uid>, gid=<gid>, home=<home directory for newuser>, shell=<shell from /etc/shells>. Diese Zeile wäre zur Darstellung zu groß, um sie in die Matrix zu schreiben. Die einzelnen Informationen, die in den beiden Klammern < > stehen bedeuten, dass sie variabel, sind das heißt sie müssen beim Start der Analyse zu erst belegt werden. Die /var/log/messages hat auf dem zu untersuchenden System, den entsprechenden Eintrag und die uid, gid kann dann als Ergiebikeit von Informationsquelle 1 belegt werden. Bei den anderen Ergiebigkeiten erfolgt dies gegebenenfalls auch aus anderen Informationsquellen, die zum Vorgang gehören.

Folgende Matrix zeigt für Vorgang 1 die einzelnen Namen der Informationsquellen 1 - 4. Die Passwortdatei /etc/passwd kann bei verschiedenen Vorgängen gelesen, modifiziert oder ausgeführt werden. Um diesen Sachverhalt zu unterscheiden wird die Variable typ vom Typ integer eingeführt. Wobei der Wert 1 bedeutet, dass die Informationsquelle gelesen wird, Wert 2 bedeutet modifiziert und 3 steht für ausführbar. Ist die Informationsquelle beispielsweise ein Dateiattribut ist der Wert der Variablen typ gleich 0.

Graph[1][n][n].nameInfo =

$$\begin{pmatrix} /etc/passwd & /etc/group & /etc/shadow & /var/log/messages & 0 \\ /etc/group & /etc/shadow & /var/log/messages & /etc/passwd & 0 \\ /etc/shadow & /var/log/messages & /etc/passwd & /etc/group & 0 \\ /var/log/messages & /etc/passwd & /etc/group & /etc/shadow & 0 \\ 0 & 0 & 0 & 0 & 0 \end{pmatrix}$$

5.1.4 Suchen und überprüfen der Informationsquellen auf dem zu untersuchenden System

In Abschnitt 4.2.3. wurde bereits beschrieben, wie der Forensiker den Graphen und die dazugehörige Tabelle 4.3. bei der Analyse verwenden muss. Die Vorgehensweise des Programmes ist genau die gleiche, wie beim Forensiker. Das Programm gibt die gleiche Aussagen, wie in Abschnitt 4.2.3. erwähnt zurück.

Der nachfolgende Pseudocode in C Notation veranschaulicht das Suchen und Überprüfen der Informationsquellen durch das Programm. Das Programm selbst besteht aus einem Hauptprogramm, das insgesamt vier Funktionen aufruft. Die Funktionen selbst rufen sich rekursiv auf.

Die folgenden Variablen werden global verwendet.

```
int anzVorgangGleicheErg; // Anzahl der Vorände, die die gleiche
//Informationsquellen mit der gleichen Ergiebigkeit haben
bool vorhanden; //Ist die Informationsquelle auf dem zu untersuchenden
//System vorhanden
char nameInfo; //Name der Informationsquelle auf dem zu untersuchenden
//System
char ergiebigkeit[100]; //Ergiebigkeit der Informationsquelle
//auf dem zu untersuchenden System
```

Die Funktion *berechneListeEinmal* sucht beim Start zuerst, die Vorgänge herraus, die die erste Informationsquelle mit der gleichen Ergiebigkeit haben. Bei der weiteren Analyse wird die Liste immer aus der Liste der Vorgänge berechnet, so dass nur noch mit den Vorgängen gearbeitet wird, die die am Anfang der Analyse Informationsquellen mit der Ergiebigkeit in Frage kamen. Dabei ist wieder die Unterscheidung von einem und mehreren in Frage kommenden Vorgänge zu treffen. Die Variable Höhe entspricht die Höhe im Graphen. Die Funktion wird insgesamt nur einmal aufgerufen.

```
berechneListeEinmal (hoehe)
{
int s = 0;
for (int z = 1; z <= anzVorgang; z++)
     for (int x = 1; x <= maxanzInfo ; x++)
          if (Graph[z][x][hoehe].nameInfo == nameInfo) //erste Zeile jeder
          //2D Matrix
          {
          L[++s].nummervorgang = z;
          L[s].position = x; //Position x der übereinstimmenden
          //Informationsquelle
          }
     anzVorgangGleichErg = s; //s gibt die Anzahl der neuen Einträge
     //in der Liste an
}
```

Die Funktion *berechneListe* geht die Liste L bei der weiteren Analyse durch und wählt diese Vorgänge aus, die nochmals eine andere Informationsquelle mit der gleichen Ergiebigkeit besitzen. Die Liste hat dabei Einträge bis zu anzVorgang-GleicheErg. Die Liste wird mit neunen Werten überschrieben.

```
berechneListe (int hoehe)
{
int s = 0; //für die Liste zum Speichern
for (int z = 1; z <= anzVorgangGleicheErg; z++)
     {
     for (int x = 1; x <= maxanzInfo; x++)
          nummer = L[z].nummervorgang;
          if (Graph[nummer][x][hoehe].nameInfo == nameInfo &&
          Graph[nummer][x][hoehe].ergiebigkeit == ergiebigkeit)
          {
              L[++s].nummervorgang = nummer; //Nummer des Vorgangs
              L[s].position = x; //Position x (Spalte)der übereinstimmenden
              //Informationsquelle
          }
     } anzVorgangGleichErg = s; //s gibt die Anzahl der neuen Einträge
//in der Liste an
}
```

Die Funktion *ueberpruefeVorgang* sucht die Informationsquellen, die zum Vorgang gehören und falls sie vorhanden sind, wird die dazugehörige Ergiebigkeit verglichen. Werden alle Informationsquelle mit der passenden Ergiebigkeit auf dem

System gefunden, gibt die Funktion die Aussage “Der Vorgang X hat stattgefunden”. Werden nicht alle Informationsquellen auf dem System gefunden, gibt das Programm die Aussage “Vorgang X hat möglicherweise stattgefunden” zurück. Dabei wird in der while Schleife in den else Teil übergegangen.

```
ueberpruefeVorgang (int hoehe, int nummervorgang, int position)
{
           Graph[nummervorgang][0][0].anzinfo ++;
           hoehe++;

           //Nächste Informationsquelle
           while (Graph[nummervorgang][position][hoehe].nummerInfo != 0)
           //Gibt es noch weitere Infos?
           {
                  vorhanden = false;
                  Suche Graph[nummervorgang][position][hoehe].nameInfo auf System

                  if (vorhanden && Graph[nummervorgang][position][hoehe].ergiebikeit == ergiebigkeit)
                  //Informationsquelle vorhanden mit der passenden Ergiebigkeit
                  {
                      Graph[nummervorgang][0][0].anzInfo ++;
                      hoehe++;
                  }
           }
                  else
                      print ‘‘Der Vorgang Graph[nummervorgang][0][0].nameVorgang hat
                      mölicherweise stattgefunden’’
}
           print‘‘Der Vorgang Graph[nummervorgang][0][0].nameVorgang hat stattgefunden’’
           exit (0);
}
```

Die Funktion *ueberpruefemehereVorgaenge* überprüft weiter, ob die in Frage kommenden Vorgänge noch eine Informationsquelle mit der passenden Ergiebigkeit gemeinsam haben. Um dies zu überprüfen wird die Funktion *berechneListe* aufgerufen. Dabei wird wieder unterschieden, ob nur noch ein Vorgang aus den in Frage kommenden Vorgänge die Informationsquelle mit der entsprechenden Ergiebigkeit besitzt oder mehrere Vorgänge. Ist nur noch ein Vorgang übrig wird er wie in der Funktion *ueberpruefeVorgang* weiter analysiert. Ansonsten ruft sich die Funktion selbst auf und endet falls kein Vorgang die Informationsquelle mit der passenden Ergiebigkeit besitzt. Die zuletzt berechnete Liste enthält die in Frage kommenden Vorgänge. Diese Vorgänge gibt die Funktion dann zurück.

```
ueberpruefemehrereVorgaenge (int anzVorgangGleichErg)
{
for (z = 1; z <= anzVorgangGleichErg; z++) //für jeden in Frage kommenden Vorgang
{
     nummer = L[z].nummerVorgang;
     Graph[nummer][0][0].anzinfo++;
}
     hoehe ++;

     berechneListe (hoehe);
     if (anzVorgangGleichErg == 0) //die Ergiebigkeit der Informationsquelle passt nicht
     {
         //Liste enthalt, die in Frage kommenden Vorgänge
         for (int z = 1; z <= anzVorgangGleichErg; z++)
              print ‘‘Vorgang Graph[z][0][0].nameVorgang kommt in Frage‘‘
     }
     else if (anzVorgangGleichErg == 1) //es kommt nur ein Vorgang in Frage
              ueberpruefeVorgang (hoehe, L[1].nummervorgang, L[1].position);

     else if (// es kommen nur noch die Vorgänge unter den voher in Frage
//kommenden Vorgänge in Frage
              ueberpruefeMehrereVorgaenge (anzVorgangGleichErg); }
```

Der Forensiker gibt den Namen der ersten sichergestellten Informationsquelle und dessen Ergiebigkeit in das Programm ein. Die Eingabe wird mit dem Wort `Input` angedeutet.

```
Hauptprogramm

nameInfo = Input (Name der ersten Informationsquelle);
ergiebigkeit = Input (Ergiebigkeit der ersten Informationsquelle);

berechneListeEinmal (1); die Zahl 1 bedeutet, dass die Analyse auf Höhe 0 des Graphens
//beginnt

if (anzVorgangGleichErg == 1) //es kommt nur ein Vorgang in Frage
    ueberpruefeVorgang (1, L[1].nummervorgang, L[1].position);

else // es kommen mehrere Vorgänge in Frage
    ueberpruefeMehrereVorgaenge (anzVorgangGleichErg);
}
```

Kapitel 6

Top Down Sicht

Der Forensiker sucht in der nachfolgenden Tabelle die Vorgänge herraus, die den Vorfall bestimmen. Der Forensiker wird dann auf die jeweilige Seitenzahl verwiesen, an der der Vorgang mit seinen Informationsquellen dokumentiert ist.

Kategorie	**Vorgang**	**Seite**
Loginverwaltung	Benutzer loggt sich ein	54
Resourcenverwaltung	Benutzer richtet Festplattenquoten ein	60
Nutzerverwaltung	Benutzer legt einen neuen Benutzer an	63
	Benutzer legt neue Gruppe an	66
	Benutzer ändert Benutzereintrag	69
	Benutzer ändert Gruppeneintrag	73
	Benutzer ändert Passwort	76
	Bnutzer ändert Gruppenpasswort	79
	Benutzer löscht Benutzeraccount	82
	Benutzer löscht Gruppenaccount	83
	Benutzer fügt Gruppenmitglied hinzu	84
	Benutzer entfernt Gruppenmitglied(er)	84
	Benutzer nimmt einen Gruppenwechsel vor	84
	Benutzer ändert Gruppenverwalter	84
Dateiververwaltung	Benutzer öffnet Datei zum Lesen	85
	Benutzer erstellt Datei	86
	Benutzer ändert Dateiinhalt	88
	Benutzer kopiert Datei	90
	Benutzer ändert Zugriffsrechte der Datei	93
	Benutzer löscht Datei	94
	Benutzer benennt Datei um oder verschiebt Datei	96
	Benutzer erstellt neues Verzeichnis	97
	Benutzer kopiert Verzeichnis	98
	Benutzer benennt Verzeichnis um oder verschiebt Verzeichnis	98
	Benutzer ändert Zugriffsrechte des Verzeichnisses	99
	Benutzer löscht Verzeichnis	99

Kategorie	Vorgang	Seite
Partitionsverwaltung	Benutzer legt neue Partition an	100
	Benutzer erstellt Dateisystem	101
	Benutzer bindet Dateisystem ein	103
	Benutzer löst Dateisystem	106
Prozessverwaltung	Benutzer startet Prozess	108
	Benutzer beendet Prozess	112
Automatisierung von Vorgängen	Benutzer startet Prozess zu bestimmter Zeit	113
	Benutzer lässt Jobs zu bestimmter Zeit ausführen	114
Secure Shell	Benutzer meldet sich an	174
	Benutzer kopiert Datei remote	187
File Transfer Protocol	Benutzer loggt sich via FTP ein	117
	Benutzer macht Download	124
	Benutzer macht Upload	127
Network File System	Benutzer exportiert Dateisystem	142
	Benutzer importiert Dateisystem	144
Network Information Service	Benutzer erzeugt Map	158
	Benutzer frägt Informationen über NIS ab	163
	Benutzer ändert Passwort in der NIS Datenbank	168
Domainname Service	Benutzer macht DNS Anfrage	151
	Benutzer macht Zone Transfer	
Telnet	Benutzer loggt sich remote beim Host ein	133
Softwareverwaltung	Benutzer installiert RPM Paket	190
	Benutzer stellt RPM Anfrage	192
	Benutzer verifiziert RPM Paket	193
	Benutzer deinstalliert RPM Paket	195
	Benutzer installiert Source Paket	196
	Benutzer deinstalliert Source Paket	197
	Benutzer installiert Source Paket von SuSE Linux	197
	Benutzer erzeugt RPM Paket	198

Tabelle 6.1: Tabelle für das Auffinden der Vorgänge in der Top Down Sicht

6.1 Benutzerverwaltung

Zur Benutzerverwaltung zählen Vorgänge, die die Benutzer –bzw. die Gruppenaccounts verwalten und das Anmelden und Abmelden am System regeln und die Resourcen des einzelnen Benutzers verwalten.

6.1.1 Vorgang: Benutzer loggt sich ein

Der Benutzer kann sich folgendermaßen am System anmelden:

1. Direkt über ein Konsolen-Login
2. Über eine grafische Anmeldung (`xdm, kdm, gdm`)
3. Remote über das Netzwerk. Das remote login wird im Abschnitt Netzwerk behandelt.

Existenz:
Online Auswertung: Methode 1. und 2.: (1), (5), (6), (12), (15), (16), (17)
*Offline Auswertung:*Methode 1. und 2.: (1), (5), (6), (12), (15)

(1) Konfigurationsdatei /etc/login.defs (Seite 205)

Beschreibung der Quelle:
Die Konfigurationsdatei ist eine Textdatei und wird vom login Programm gelesen und ausgewertet. Dabei wird das Dateiattribut "letzter Zugriff" der Konfigurationsdatei auf die aktuelle Systemzeit und Datum gesetzt.

Ergiebigkeit:
Das Loginverhalten der Shadow Suite wird durch die Konfigurationsdatei /etc/login.defs festgelegt.
Schlecht. Die Konfigurationsdatei enthält folgende Parameter:

- DEFAULT_HOME: Die Werte "true" bzw. "false" legen fest, ob ein Benutzer sich anmelden darf, falls sein Home-Verzeichnis nicht verfügbar ist (bspw. wenn es per NFS gemountet wird, die Verbindung zum NFS-Server aber nicht hergestellt werden kann).
- ENV_PATH: Initiale Belegung der Parametern PATH
- ENV_ROOTPATH: Initiale Belegung von PATH für Root
- FAILLOG_ENAB: "yes" werden fehlgeschlagende Anmeldeversuche in der Datei `/var/log/faillog` protokolliert
- FAIL_ DELAY: Nach einem fehlgeschlagenen Login-Versuch wird die anegegebene Zeitspanne (in Sekunden) gewartet, bevor ein erneutes Login-Prompt erscheint.
- LOG_UNKFAIL_ENAB: "yes" werden unbekannte Benutzernamen bei fehlgeschlagenen Anmeldeversuchen mit protokolliert.
- LASTLOG_ENAB: "yes" nach erfolgreichem Anmelden werden Informationen zum Zeitpunkt der letzten Anmeldung und ggf. zu zwischenzeitlichen fehlgeschlagenen Anmeldeversuchen ausgegeben.

- LOGIN_RETRIES: Bei fehlgeschlagenem Anmeldeversuch lässt das Kommando login die angegebene Anzahl erneuter Versuche zu, bis es sich selbst beendet. Bei lokalen Login-Konsolen startet i.d.R. ein getty anschließend erneut den Anmeldevorgang; bei einer Anmeldung übers Netz wird jedoch meist die Verbindung getrennt.
- LOGIN_TIMEOUT: Anzahl Sekunden, die login auf die Eingabe eines Passworts wartet. Nach Ablauf der Zeitspanne gilt der Versuch als gescheitert.
- MOTD_FILE: Vollständiger Pfad zu einer Datei mit der "Nachricht des Tages" (/etc/motd). Mehrere Dateien können - per Doppelpunkt getrennt - angegeben werden. Existieren sie, wird ihr Inhalt nach dem erfolgreichen Anmelden angezeigt.
- TTYPERM: Rechte, mit denen ein Login-Terminal (Terminalgerätedatei `/dev/pts`) versehen wird.
- TTYTYPE_FILE: Pfad zu einer Datei mit den Terminal-Spezifikationen Diese Datei enthält Zeilen, die einem Terminal einen konkreten Typ ("Terminalemulation") zuordnen. So sind lokale Login-Konsolen (tty1..tty6) meist vom Typ "linux"; Pseudoterminals (ttypX) zur Anmeldung übers Netz verwenden häufig die "vt100"-Emulation.

Das Dateiattribut "letzter Zugriff" gibt den Zeitpunkt an, an dem der Vorgang: *Benutzer loggt sich ein*, gestartet wurde.

---

(2) Konfigurationsdatei /etc/securetty (Seite 205)

Beschreibung der Quelle:
Die Konfigurationsdatei ist eine Textdatei und wird vom login Programm gelesen und ausgewertet. Dabei wird das Dateiattribut "letzter Zugriff" der Konfigurationsdatei auf die aktuelle Systemzeit und Datum gesetzt.

Ergiebigkeit:
Schlecht. Die Konfigurationsdatei `/etc/securetty` enthält vertrauenswürdige Terminals über die sich root einloggen kann. Das Dateiattribut "letzter Zugriff" gibt den Zeitpunkt an, an dem der Vorgang: *Benutzer loggt sich ein*, gestartet wurde.

---

(3) Konfigurationsdatei /etc/nologin (Seite 205)

Beschreibung der Quelle:
Die Konfigurationsdatei ist eine Textdatei und wird nur vom login Programm gelesen. Wenn diese Datei existiert, ist jedes "normale" Einloggen im System unmöglich. Die Konfigurationsdatei `/etc/nologin` ist nur vorhanden, wenn die Parameter NOLOGIN_FILE in der Konfigurationsdatei `/etc/login.defs` auf diese Konfigurationsdatei gesetzt ist. Dabei wird das Dateiattribut "letzter Zugriff" der Konfigurationsdatei auf die aktuelle Systemzeit und Datum gesetzt.

Ergiebigkeit:
Schlecht. Die Konfigurationsdatei `/etc/nologin` enthält Benutzernamen, die sich nicht einloggen dürfen. Das Dateiattribut "letzter Zugriff" gibt den Zeitpunkt an, an dem der Vorgang: *Benutzer loggt sich ein*, gestartet wurde.

---

(4) Konfigurationsdatei /etc/shells (Seite 205)

Beschreibung der Quelle:
Die Konfigurationsdatei ist eine Textdatei und wird vom login Programm gelesen und ausgewertet. Dabei wird das Dateiattribut “letzter Zugriff” der Konfigurationsdatei auf die aktuelle Systemzeit und Datum gesetzt.

Ergiebigkeit:
Schlecht. Die Konfigurationsdatei enthält Shells, die ein Benutzer als default Shell verwenden darf. Das Dateiattribut“letzter Zugriff” gibt den Zeitpunkt an, an dem der Vorgang: *Benutzer loggt sich ein*, gestartet wurde.

(5) Konfigurationsdatei /etc/passwd (Seite 205)

Beschreibung der Quelle:
Die Passwortdatei ist eine Textdatei und wird vom login Programm gelesen und ausgewertet. Dabei wird das Dateiattribut “letzter Zugriff” der Konfigurationsdatei auf die aktuelle Systemzeit und Datum gesetzt.

Ergiebigkeit:
Schlecht. Aus der Passwortdatei kann man die Loginnamen aller Benutzer des Systems, Passwort (falls keine `/etc/shadow` existiert) die UserID und GroupID aller Benutzer, Info, Namen des Home Directories und die verwendete Shell bekommen.
Die Passwortdatei hat folgendes Format:

```
Username:Password:UID:GID:Info:Home:Shell
```

Das Dateiattribut “letzter Zugriff” gibt den Zeitpunkt an, an dem der Vorgang: *Benutzer loggt sich ein*, gestartet wurde.

(6) Konfigurationsdatei /etc/shadow (Seite 205)

Beschreibung der Quelle:
Die Shadowdatei ist eine Textdatei und wird vom login Programm gelesen und ausgewertet. Die Datei ist nur vorhanden, falls in der Passwortdatei `/etc/passwd` an 2. Stelle ein “x” steht. Dabei wird das Dateiattribut “letzter Zugriff” der Konfigurationsdatei auf die aktuelle Systemzeit und Datum gesetzt.

Ergiebigkeit:
Mittel. Aus der Shadow Datei kann man folgenden Informationen beziehen:

- Username: wie in /etc/passwd
- Passwort: Verschlüsseltes Passwort
- DOC: Day of last change, Tage ab dem 1.1.1970, an dem das Passwort zuletzt geändert wurde
- MinD: Minimale Anzahl Tage, die das Passwort gültig ist
- MaxD: Maximale Anzahl Tage, die das Passwort gültig ist

- Warn :Anzahl der Tage vor Ablauf der Lebensdauer des Passwortes, ab der vor dem Verfall zu warnen ist
- Exp: Expire, wieviele Sekunden das Passwort trotz Ablauf der MaxD gilt
- Dis: Bis zu diesem Tag (gezählt ab 1.1.1970) ist dieser Account gesperrt
- Res: Reserve, Feld wird derzeit nicht ausgewertet

Die Shadow Datei hat folgendes Format:

```
Username:Password:DOC:MinD:MaxD:Warn:Exp:Dis:Res
```

Das Dateiattribut “letzter Zugriff” gibt den Zeitpunkt an, an dem der Vorgang: *Benutzer loggt sich ein*, gestartet wurde.

(7) Konfigurationsdatei /etc/ttytype (Seite 205)

Beschreibung der Quelle:
Die Konfigurationsdatei ist eine Textdatei und wird vom *login* Programm gelesen und ausgewertet. Die Konfigurationsdatei `/etc/ttytype` ist nur vorhanden, wenn die Parameter TTYTYPE_FILE in der Konfigurationsdatei `/etc/login.defs` auf diese Konfigurationsdatei gesetzt ist. Dabei wird das Dateiattribut “letzter Zugriff” der Konfigurationsdatei auf die aktuelle Systemzeit und Datum gesetzt.

Ergiebigkeit:
Schlecht. Die Konfigurationsdatei enthält eine Liste von Terminaltypen. Das Dateiattribut “letzter Zugriff” gibt den Zeitpunkt an, an dem der Vorgang: *Benutzer loggt sich ein*, gestartet wurde.

(8) Konfigurationsdatei /etc/usertty (Seite 205)

Beschreibung der Quelle:
Die Konfigurationsdatei ist eine Textdatei wird vom login Programm gelesen und ausgewertet. Dabei wird das Dateiattribut “letzter Zugriff” der Konfigurationsdatei auf die aktuelle Systemzeit und Datum gesetzt.

Ergiebigkeit:
Schlecht. Die Konfigurationsdatei enthält Terminals, von denen sich der Benutzer aus einloggen dürfen. Das Dateiattribut ”letzter Zugriff” gibt den Zeitpunkt an, an dem der Vorgang: *Benutzer loggt sich ein*, gestartet wurde.

(9) Konfigurationsdatei /etc/profile (Seite 205)

Beschreibung der Quelle:
Die Konfigurationsdatei ist eine Textdatei und wird von den Loginshells aller Benutzer gelesen und als Shellscript ausgeführt. Hier werden Grundeinstellungen der Shellumgebung für alle User vorgenommen. Wenn sie nicht vor dem Überschreiben geschützt werden (siehe beim Shellkommando typeset), können alle Einstellungen von einer benutzereigenen Initialisierungsdatei im Home- Verzeichnis `.profile` (bei Bourne- und Korn- Shell, `.cshrc` und `.login` bei C- Shell). Dabei wird das Dateiattribut “letzter Zugriff“ der Konfigurationsdatei auf die aktuelle Systemzeit und Datum gesetzt.

Ergiebigkeit:
Schlecht. In der Konfigurationsdatei befinden sich die Standardeinträge für die Umgebungsvariablen aller Benutzer. Das Dateiattribut "letzter Zugriff" gibt den Zeitpunkt an, an dem der Vorgang: *Benutzer loggt sich ein*, gestartet wurde.

(10) Konfigurationsdatei /dev/pts/<number>(Seite 206)

Beschreibung der Quelle:
Der Benutzer wird als neuer Eigentümer und Gruppeneigentümer für die Terminalgerätedatei eingetragen und die Zugriffsrechte werden gesetzt. Die Nummer hängt davon ab, ob der Benutzer bei letzten Abmeldung seine Sitzung gespeichert hat und Konsolen geöffnet waren. Wurde die Sitzung nicht gespeichert, ist die Konsolennummer 0. Die Zugriffsrechte werden bezüglich der Rechte, die in der Parametern TTYPERM in der Konfigurationsdatei `/etc/login.defs` angegeben sind, gesetzt.

Ergiebigkeit:
Gut. Name der Benutzers, der sich eingeloggt hat.

(11) Konfigurationsdatei /etc/motd (Seite 205)

Beschreibung der Quelle:
Die Konfigurationsdatei ist eine Textdatei und wird vom login Programm gelesen und der Inhalt wird auf die Konsole des Benutzers ausgegeben. Die Konfigurationsdatei `/etc/motd` ist nur vorhanden, wenn die Parameter MOTD_FILE in der Konfigurationsdatei `/etc/login.defs` auf diese Konfigurationsdatei gesetzt ist. Dabei wird das Dateiattribut "letzter Zugriff" der Konfigurationsdatei auf die aktuelle Systemzeit und Datum gesetzt.

Ergiebigkeit:
Schlecht. Der Inhalt dieser Datei enthält die "Message of the day". Das Dateiattribut "letzter Zugriff" gibt den Zeitpunkt an, an dem der Vorgang: *Benutzer loggt sich ein*, gestartet wurde.

(12) Umgebungsdatei des Benutzers (Seite 220)

Beschreibung der Quelle:
Mit der Datei `.profile` (bei Bourne- und Kornshell) oder `.cshrc` und `.login` (bei C- Shell), die im Home- Verzeichnis des Benutzers liegt, kann der Benutzer sein eigenes Umfeld gestalten. Wenn sich der Benutzer einloggt liest das Betriebsystem erst die `/etc/profile` und dann sucht er im Home- Verzeichnis des Benutzers nach der Umgebungsdatei des Benutzers. Dabei wird das Dateiattribut "letzter Zugriff" der Umgebungsdatei auf die aktuelle Systemzeit und Datum gesetzt. Die Umgebungsdatei ist je nach Wahl der Shell vorhanden.

Ergiebigkeit:
Schlecht. In der Umgebungsdatei befinden sich die vom Benutzer festgelegten Environment Variablen und deren Werte. Das Dateiattribut "letzter Zugriff" gibt den Zeitpunkt an, an dem der Vorgang: *Benutzer loggt sich ein*, gestartet wurde.

(13) UserID (uid) / Group- ID (gid) (Seite 230)

Beschreibung der Quelle:
Das login Programm setzt die User- ID‘s (reale, effektive, saved Set-User-ID) und Group- ID‘s der (ausführbare) Dateien, Verzeichnisse, die dem Benutzer gehören.

Ergiebigkeit:
Gut. Welche (ausführbare) Dateien und Verzeichnisse dem Benutzer gehören.

(14) Environment Variablen des Benutzers (Seite 229)

Beschreibung der Quelle:
Die Environment Variablen HOME, SHELL, USER/LOGNAME, PATH, und TERM u.a. werden vom login Programm gesetzt.

Ergiebigkeit:
Mittel. Umgebung des Benutzers:

- HOME: Heimatverzeichnis des Benutzers
- SHELL: Voreingestellte Benutzershell
- USER/LOGNAME: Benutzernamen
- PATH: Aktueller Suchpfad des Benutzers
- TERM: Name des Terminals

(15) Eintrag in Log- Datei /var/log/lastlog (Seite 210)

Beschreibung der Quelle:
vgl. Seite 27

Ergiebigkeit:
Gut. Der Eintrag in der Log- Datei `/var/log/lastlog` enthält den Benutzernamen, den Port :0, und die Zeit des letzten erfolgreichen Einloggens. Der Port:0 bedeutet, dass sich der Benutzer direkt am Rechner und nicht über das Netz eingeloggt hat.

(16) Eintrag in Log- Datei /var/log/wtmp (Seite 210)

Beschreibung der Quelle:
vgl. Seite 27

Ergiebigkeit:
Gut. Der Eintrag in der Log- Datei enthält den Benutzernamen, den Terminal, die Loginzeit und von wo aus sich der Benutzer eingeloggt hat.

(17) Eintrag in Log- Datei /var/log/utmp (Seite 210)

Beschreibung der Quelle:
vgl. Seite 27

Ergiebigkeit:
Gut. Der Eintrag enthält folgende Informationen über den Benutzer:

- Ut_type: Typ des Logins (Login_Prozess)
- Ut_pid: Prozess- ID des Login Prozesses
- Ut_line: Gerätename des Terminals (pts)
- Ut_id: ID des Init Prozesses oder Name des Terminals
- Ut_user: Name des Benutzers
- Ut_host: Konsole
- Ut_session: Session ID
- Ut_addr_v6: IP Adresse des remote Host

6.1.2 Vorgang: Root richtet Festplattenquoten ein

Jedem Benutzer wird ein bestimmtes und beschränktes Kontingent an Plattenplatz zur Verfügung gestellt. Alle Dateisysteme außer ReiserFS unterstützen Quotas. Quotas können nur durch root erstellt werden.

1. In der Konfigurationsdatei `/etc/fstab` muss für das jeweilige Partition, die Quotas unterstützen sollen, `usrquota` und/oder `grpquota` hinzugefügt werden.
2. Die jeweilige Partition wird mit dem Kommando `mount <mount point>` oder `<device>` gemountet.
3. Die Quotas für einzelne Benutzer in der Datei `/<mount point>/aquota.user` und/oder `/<mount point>/aquota.group` für eine Gruppe werden mit dem Kommando `edquota -u <user>` festgelegt.
4. Die Quotas werden dann mit dem Kommando `quotaon /<mount point>` aktiviert.

Existenz:
Online Auswertung: (1), (2), (3), (4), (5)
Offline Auswertung: (1), (2), (5)

(1) Feld in Konfigurationsdatei /etc/fstab (Seite 206)

Beschreibung der Quelle:
Die Konfigurationsdatei ist eine Textdatei und root fügt `usrquota` und/ oder `grpquota` mit Hilfe eines Texteditors hinzu.

Ergiebigkeit:
Mittel. Die Konfigurationsdatei enthält folgendende Informationen über Dateisystem das die Quotas unterstützen soll: Name des Blockdevices oder entferntes Dateisystem, den Mountpoint des Dateisystems, Typ des Dateisystems, Optionen, mögliche Sicherung des Dateisystems.

- Device: Blockdevice oder entferntes Dateisystem, welches die Quotas unterstützen soll
- Mountpunkt: Verzeichniseintrag, in dem das Dateisystem mit den Quotas erscheinen soll
- Type: Typ des Dateisystems, das die Quotas unterstützen soll.
- Options: Optionen
- Dump: Ob das Dateisystem mit den Quotas vom Kommando dump zu sichern ist, nur beim Dateisystemtyp ext2 bewertet.
- Check: Ob das Dateisystem mit den Quotas vor dem Mounten zu überprüfen ist.

Der Eintrag hat folgendes Format:

```
<Device> <Mountpunkt> <Type> <Options>,usrquota,grpquota <Dump>
<Check>
```

(2) Quotadateien /<mountpoint>/aquota.{user, group} (Seite 233)

Beschreibung der Quelle:
Die beiden Dateien quota.user und/ oder quota.group mit den Rechten 600 für die Ressourcenverteilung im Mountpunkt der Partition sind Textdateien und werden durch den Aufruf des Kommandos `edquota - u <user>` um einen Eintrag erweitert.

Ergiebigkeit:
Mittel. Die beiden Dateien enthalten für jeden Benutzer eine Zeile mit folgenden Informationen:

1. Den Gesamtspeicherplatz (blocks in use) für einen bestimmten Benutzer bzw. Gruppe, die Grenzwerte soft limit und hard limit.
2. Mit dem Grenzwert soft limit wird die maximale Festplattenspeichermenge definiert, die ein Benutzer laut Quota auf dem System belegen darf. Beim Übertreten des Limits für einen bestimmten Zeitraum ("grace period", siehe 4.) wird das Soft-Limit in das Hard-Limit (siehe 3.) umgewandelt.

3. Mit dem Grenzwert hard limit wird die physische Grenze definiert, die vom Benutzer nicht überschritten werden darf. Wird das Hard-Limit übertreten, ist es nicht mehr möglich, weitere Dateien zu speichern.

4. Die "grace period" bezeichnet den Zeitraum, nach dem das überschrittenes Soft- Limit in das Hard- Limit umgewandelt wird.

---

(3) Eintrag in Konfigurationsdatei /etc/mtab (Seite 206)

Beschreibung der Quelle:
Die Konfigurationsdatei ist eine Textdatei und wird durch den Aufruf von `mount <mount point>` oder `<device>` um einen Eintrag erweitert.

Ergiebigkeit:
Mittel. Die Konfigurationsdatei enthält das gemountete Dateisystem, das die Quotas unterstützt. Der Eintrag enthält folgenden Informationen über das Dateisystem:

- Device: Blockdevice
- Mountpoint: Verzeichniseintrag, in dem das Dateisystem, das die Quotas unterstützt, erscheinen soll
- Type: Typ des Dateisystems
- Options: Optionen für den Mountvorgang
- Dump: Gibt an, ob das Dateisystem vom Kommando dump zu sichern ist. Dieser Eintrag wird derzeit nur beim Dateisystemtyp ext2 bewertet.
- Check: Gibt an, ob das Dateisystem vor dem Mounten zu überprüfen ist. Beim Root-Dateisystem sollte hier eine "1" stehen und bei allen anderen entweder eine "0" (keine Prüfung) oder eine "2". Dateisysteme mit gleicher Nummer werden parallel überprüft, das Root- Dateisystem sollte immer allein und als erstes getestet werden.

Der Eintrag hat folgendes Format:

```
<Device> <Mountpunkt> <Type> <Options>, usrquota, grpquota <Dump>
<Check>
```

---

(4) Eintrag in virtueller Datei /proc/mounts (Seite 225)

Beschreibung der Quelle:
Die virtuelle Datei ist eine Textdatei und wird durch durch den Aufruf von `mount` um einen Eintrag erweitert.

Ergiebigkeit:
Mittel. Die virtuelle Datei `/proc/mounts` ist sehr ähnlich zum Inhalt von der Konfigurationsdatei `/etc/mtab` (vgl. Ergiebigkeit der Informationsquelle (4)), nur dass sie aktueller sein kann.

---

(5) Einträge in History Liste des Benutzers (Seite 221)

Beschreibung der Quelle:
vgl. Seite 26

Ergiebigkeit:
Mittel. Die History Liste enthält folgende Kommandos:

```
vi /etc/fstab

cd /<mount point>

touch aquota.user und/ oder touch aquota.group

chmod 600 aquota.user

chmod 600 aquota.group

edquota -u <user>

quotaon <mount point>
```

6.1.3 Vorgang: Root legt einen neuen Benutzer an

Der Benutzer root kann auf drei verschiedene Wege einen neuen Benutzer anlegen:

1. Mit Nutzerkommandos:
   - Der Benutzer editiert die Systemdateien `/etc/passwd` und fügt eine Zeile für den neuen Benutzer hinzu.
   - Mit dem Kommando `pwconv` wird der entsprechende Eintrag in der Shadowdatei `/etc/shadow` erstellt.
   - Das Home- Verzeichnis des neuen Benutzers wird mit dem Kommando mkdir erstellt und dann wird der neue Benutzer Eigentümer des Verzeichnis mit `chown`.
   - Der Benutzer gibt dem neuen Benutzer mit dem Kommando passwd ein Passwort (vgl. Seite 76 Vorgang: *Benutzer ändert Passwort*)
2. Mit Hilfe privilegierter Kommandos:
   - `useradd <options> , mkdir <home directory>, chown` (vgl. Seite 99 Vorgang: *Benutzer ändert Zugriffsrechte des Verzeichnisses*) und passwd (vgl. Seite 99 Vorgang: *Benutzer ändert Passwort*)
3. Durch Verwendung der graphischen Oberfläche YAST2 → Sicherheit und Benutzer → Benutzer bearbeiten und anlegen (vgl. Seite 99 Vorgang: *Benutzer ändert Passwort*)

Existenz:
Online Auswertung und Offline Auswertung:
Methode 1: (1), (2), (3), (4), (6)
Methode 2: (1), (2), (3), (4), (5), (6)
Methode 3: (1), (2), (3), (4), (5)

(1) Konfigurationsdatei /etc/login.defs (Seite 205)

Beschreibung der Quelle:
Die Konfigurationsdatei ist eine Textdatei und wird durch Methode 1. 2. und 3. gelesen und ausgewertet. Dabei wird das Dateiattribut "letzter Zugriff" der Konfigurationsdatei auf die aktuelle Systemzeit und Datum gesetzt.

Ergiebigkeit:
Schlecht. Das Loginverhalten der Shadow Suite wird durch die Konfigurationsdatei `/etc/login.defs` festgelegt. Die Konfigurationsdatei enthält folgende Parameter:

- PASS_MIN_DAYS, PASS_MAX_DAYS: Minimale/maximale Zeitspanne, die zwischen zwei Passwort-änderungen vergehen muss/darf.
- PASS_WARN_AGE: Anzahl von Tagen bevor ein Passwort ausläuft. "0" Warnung wird nur am Tag des Ablaufes angezeigt. Eine negative Zahl bedeutet, dass es keine Warnung gegeben wird.·
- UMASK: Voreinstellung der Zugriffsrechte für neu erstellte Dateien und Verzeichnisse
- GID_MIN, GID_MAX: Minimaler und maximaler Wert für eine Gruppennummer, die das Kommando groupadd automatisch kalkulieren darf.
- UID_MIN, UID_MAX: Minimale bzw. maximale Nummer, die vom Kommando useradd bei der automatischen Vergabe der UID gewählt werden kann. Das Dateiattribut "letzter Zugriff" gibt den Zeitpunkt an, an dem der Vorgang: *Benutzer legt neuen Benutzer an*, gestartet wurde.

(2) Eintrag in Passwortdatei /etc/passwd (Seite 206)

Beschreibung der Quelle:
Die Passwortdatei `/etc/passwd` ist eine Textdatei und wird durch durch Methode 1. 2. und 3. um einen Eintrag ergänzt.

Ergiebigkeit:
Mittel. Aus der Passwortdatei kann man den Loginnamen des neu angelegten Benutzers, Passwort (falls keine `/etc/shadow` existiert) die UserID und GroupID des neu angelegten Benutzers, Info, Namen des Home Directories und die verwendete Shell bekommen.

Der Eintrag für den neuen Benutzer hat folgendes Format:

```
<Username>:<Password>:<UID>:<GID>:<Info>:<Home>:<Shell>
```

Verweis auf andere Vorgänge:

1. *Benutzer ändert Dateiinhalt* (vgl. Seite 88)
2. kein Verweis
3. kein Verweis

(3) Eintrag in Shadowdatei /etc/shadow (Seite 206)

Beschreibung der Quelle:
Die Shadowdatei `/etc/shadow` ist eine Textdatei und wird durch Methode 1. 2. und 3. um einen Eintrag ergänzt.

Ergiebigkeit:
Mittel. Aus der Shadow Datei kann man folgenden Informationen über den neu angelegten Benutzer beziehen:

- Username: wie in `/etc/passwd`
- Passwort: Verschlüsseltes Passwort des neu angelegten Benutzers
- DOC: Day of last change, Tage ab dem 1.1.1970, an dem das Passwort zuletzt geändert wurde
- MinD: Minimale Anzahl Tage, die das Passwort gültig ist
- MaxD: Maximale Anzahl Tage, die das Passwort gültig ist
- Warn: Anzahl der Tage vor Ablauf der Lebensdauer des Passwortes, ab der vor dem Verfall zu warnen ist
- Exp: Expire, wieviele Tage das Passwort trotz Ablauf der MaxD gilt
- Dis: Bis zu diesem Tag (gezählt ab 1.1.1970) ist dieser Account gesperrt
- Res: Reserve, Feld wird derzeit nicht ausgewertet

Der Eintrag für den neuen Benutzer hat folgendes Format:

```
<Username>:<Password>:<DOC>:<MinD>:<MaxD>:<Warn>:<Exp>:<Dis>:<Res>
```

(4) Home- Verzeichnis /$HOME/<newuser>(Seite 235)

Beschreibung der Quelle:
Das Home Verzeichnis /$HOME /<newuser> wird durch Methode 1. 2. und 3. erstellt. Der Pfad des Home Verzeichnisses wird durch den Eintrag für das Home Verzeichnissses in der Passwortdatei `/etc/passwd` festgelegt.

Ergiebigkeit:
Mittel. Aus dem Home Verzeichnis kann man heraus finden, was für Zugriffsrechte gesetzt sind, wer der Eigentümer ist und was für Verzeichnisse und welche Dateien befinden sich in dem Home Verzeichnis, des neu angelegten Benutzers.

Verweis auf andere Vorgänge:

1. - *Benutzer erstellt neues Verzeichnis an* (vgl. Seite 97)
   - *Benutzer ändert Zugriffsrechte des Verzeichnisses* (vgl. Seite 93)
2. kein Verweis
3. kein Verweis

(5) Eintrag in Log- Datei /var/log/messages (Seite 210)

Beschreibung der Quelle:
vgl. Seite 26

Ergiebigkeit:
Gut.

1. kein Eintrag
2. und 3.
   - Time: Wann sich der Benutzer root den neuen Benutzer angelegt hat
   - Host: Rechnername des Hosts
   - pid: Pid vom useradd Programm
   - newuser: Name des neuen Benutzers
   - uid: UID des neuen Benutzers
   - gid: GID des neuen Benutzers
   - home directory: Pfad des Home Directory für den neuen Benutzers
   - shell: Login Shell des neuen Benutzers

   Der Eintrag hat folgendes Format:

   ```
   <time> <host> useradd[<pid>]: newuser: name=<newuser>, uid=<new
   uid>, gid=<gid>, home=<home directory for newuser>, shell=<shell
   from /etc/shells>
   ```

(6) Einträge in History Liste des Benutzers (Seite 221)

Beschreibung der Quelle:
vgl. Seite 26

Ergiebigkeit:
Mittel. Die History Liste enthält folgende Kommandos:

1.
   - vi /etc/passwd
   - vi /etc/shadow oder pwconv
   - mkdir
   - chown <options>
   - passwd
2.
   - useradd <newuser>
   - mkdir <home directory>
   - chown <options>
   - passwd
3. kein Eintrag

6.1.4 Vorgang: Benutzer legt neue Gruppe an

Der Benutzer root kann auf drei verschiedene Wege eine neue Gruppe anlegen.

1. Mit einem Texteditor:
   - Der Benutzer editiert die Systemdateien `/etc/group` fügt eine Zeile die Gruppe betreffend hinzu.
   - Mit dem Kommando `gwconv` wird der entsprechende Eintrag in der Shadowdatei `/etc/gshadow` erstellt.
   - Der Benutzer gibt mit dem Kommando `gpasswd` das Passwort für die Gruppe an. (vgl. Seite 79 Vorgang: *Benutzer ändert Gruppenpasswort*)
2. Mit Hilfe privilegierter Kommandos:
   - `groupadd <options>` und `gpasswd` (vgl. Seite 79 Vorgang: *Benutzer ändert Gruppenpasswort*)
3. Durch Verwendung der graphischen Oberfläche YAST2 → Sicherheit und Benutzer → Gruppen bearbeiten und anlegen

Existenz:
Online und Offline Auswertung:
Methode 1: (1), (2), (3), (5)
Methode 2: (1), (2), (3), (4), (5)
Methode 3: (1), (2), (3), (4)

---

(1) Konfigurationsdatei /etc/login.defs (Seite 205)

Beschreibung der Quelle:
Die Konfigurationsdatei ist eine Textdatei und wird durch Methode 1. 2. und 3. gelesen und ausgewertet.

Ergiebigkeit:
Schlecht. Das Loginverhalten der Shadow Suite wird durch die Konfigurationsdatei /etc/login.defs festgelegt.
Die Konfigurationsdatei enthält folgende Parameter:

- GID_MIN, GID_MAX: Minimaler und maximaler Wert für eine Gruppennummer, die das Kommando groupadd automatisch kalkulieren darf.

Das Dateiattribut "letzter Zugriff" gibt den Zeitpunkt an, an dem der Vorgang: *Benutzer legt neue Gruppe an*, gestartet wurde.

---

(2) Eintrag in Konfigurationsdatei /etc/group (Seite 206)

Beschreibung der Quelle:
Die Konfigurationsdatei ist eine Textdatei und wird durch Methode 1. 2. und 3. um einen Eintrag ergänzt.

Ergiebigkeit:
Mittel. Aus der `/etc/group` kann man den Gruppennamen der neu angelegten Gruppe, Passwort (falls keine `/etc/gshadow` existiert), die GroupID und Liste der Mitglieder erhalten.

Der Eintrag für die neue Gruppe hat folgendes Format:

`<Groupname>:<Password>:<GID>:<List of Members>`

Verweis auf andere Vorgänge:

1. *Benutzer ändert Datei* (vgl. Seite 88)
2. kein Verweis
3. kein Verweis

---

(3) Eintrag in Konfigurationsdatei /etc/gshadow (Seite 206)

Beschreibung der Quelle:
Die Konfigurationsdatei ist eine Textdatei und wird durch Methode 1. 2. und 3. um einen Eintrag ergänzt.

Ergiebigkeit:
Mittel. Aus der `/etc/gshadow` kann man den Gruppennamen der neu angelegten Gruppe, das verschlüsselte Passwort, den Gruppenverwalter und die Liste der Mitglieder erhalten.

Der Eintrag für die neue Gruppe hat folgendes Format:

`<Groupname>:<Password>:<Administrator>:<List of Members>`

Verweis auf andere Vorgänge:

1. *Benutzer ändert Datei* (vgl. Seite 88)
2. kein Verweis
3. kein Verweis

---

(4) Eintrag in Log- Datei /var/log/messages (Seite 210)

Beschreibung der Quelle:
vgl. Seite 26

Ergiebigkeit:
Gut.

1. kein Eintrag
2. und 3.: Der Eintrag enthält folgende Informationen:
   - Time: Wann der Benutzer root die neue Gruppe angelegt hat
   - Host: Rechnername des Hosts
   - pid: Pid vom groupadd Programm
   - name: Name der neuen Gruppe
   - gid: GID der neuen Gruppe

Der Eintrag für die neue Gruppe hat folgendes Format:

```
<time> <host> groupadd[<pid>]: newgroup: name=<newgroup>,
gid=<gid>
```

---

(5) Eintrag in der History Liste des Benutzers (Seite 221)

Beschreibung der Quelle:
vgl. Seite 26

Ergiebigkeit:
Mittel. Die History Liste enthält folgende Kommandos:

1. - `vi /etc/group`
   - `vi /etc/gshadow oder gwconv`
   - `gpasswd`
2. - `groupadd <options> <newgroup>`
   - `chown <options>`
   - `gpasswd`
3. kein Eintrag

6.1.5 Vorgang: Root ändert Benutzereintrag

6.1.5.1 Benutzer root

Der Benutzer root kann auf drei verschiedene Wege einen Benutzereintrag ändern.

1. Mit einem Texteditor: Der Benutzer editiert die Systemdateien `/etc/passwd` und/oder `/etc/shadow` und ändert die betreffenden Felder ab.
2. Mit Hilfe privilegierter Kommandos:
   (a) `chsh -s <shell> <user>`
   (b) `usermod <options> <user>`
   (c) `chfn <options> <user>`
3. Durch Verwendung der graphische Oberfläche YAST2 → Sicherheit und Benutzer → Benutzer bearbeiten und anlegen

Existenz:
Online und Offline Auswertung:
Methode 1: (1), (2), (3), (4)
Methode 2: (1), (2), (3), (4) oder (1), (2), (3), (4), (5)
Methode 3: (1), (2), (3), (5)

(1) Felder in Passwortdatei /etc/passwd (Seite 206)

Beschreibung der Quelle:
Die Passwortdatei ist eine Textdatei und durch Methode 1. 2. und 3. werden einzelne Felder `Username: Password: UID: GID: Info: Home: Shell` geändert.

Ergiebigkeit:
Mittel. Das (die) jeweilige(n) Feld(er), das (die) geändert wurde(n).

Verweis auf andere Vorgänge:

1. *Benutzer ändert Datei* (vgl. Seite 88)
2. kein Verweis
3. kein Verweis

(2) Felder in Shadowdatei /etc/shadow (Seite 206)

Beschreibung der Quelle:
In der Shadowdatei `/etc/shadow` ist eine Textdatei und durch 1. 2. und 3. werden einzelne Felder `Username: Password: DOC: MinD: MaxD: Warn: Exp: Dis: Res` geändert.

Ergiebigkeit:
Mittel. Das (die) jeweilige(n) Feld(er), das (die) geändert wurde(n).

Verweis auf andere Vorgänge:

1. *Benutzer ändert Datei* (vgl. Seite 88)
2. kein Verweis
3. kein Verweis

(3) Konfigurationsdatei /etc/shells (Seite 205)

Beschreibung der Quelle:
Die Konfigurationsdatei ist eine Textdatei und wird von 2. und 3. gelesen und ausgewertet. Dabei wird das Dateiattribut "letzter Zugriff" der Konfigurationsdatei auf die aktuelle Systemzeit und Datum gesetzt.

Ergiebigkeit:
Schlecht. Die Konfigurationsdatei enthält Shells, die ein Benutzer als default Shell verwenden darf. Das Dateiattribut "letzter Zugriff" gibt den Zeitpunkt an, an dem der Vorgang: *Benutzer ändert Benutzereintrag*, gestartet wurde.

(4) Einträge in History Liste des Benutzers (Seite 221)

Beschreibung der Quelle:
vgl. Seite 26

Ergiebigkeit:
Mittel. Die History Liste enthält folgende Kommandos:

1.
   - `vi /etc/passwd`
   - `vi /etc/shadow`
2.
   - `chsh -s <shell>`
   - `usermod <options>`
   - `chfn <options>`
3. kein Eintrag

(5) Eintrag in Log- Datei /var/log/messages (Seite 210)

Beschreibung der Quelle:
vgl. Seite 26

Ergiebigkeit:
Gut.

1. 2.(a), 2.(c), 3. kein Eintrag
2. (b): Der Eintrag enthält folgende Informationen:
   - Time: Wann Benutzer den Benutzereintrag geändert hat
   - Host: Rechnername des Hosts
   - pid: Pid vom usermod Programm
   - User: Name des Benutzers
   - field: Feld in der Passwortdatei bzw. Shadowdatei das gändert wird
   - old value: alter Wert des bestimmten Feldes
   - new value: neuer Wert des geänderten Feldes

   Der Eintrag hat folgendes Format:

   ```
   <time> <host> usermod[<pid>]: change user '<user>' <field> from
   '<old value>' to <new value>
   ```

6.1.5.2 Normaler Benutzer

Der normale Benutzer kann nur seine Shell und den finger Eintrag ändern:

1. Mit Hilfe privilegierter Kommandos:
   - `chsh -s <shell>`
   - `chfn <options>`

Existenz:
Online und Offline Auswertung:
Methode 1: (1), (2), (3)

---

(1) Felder in Passwortdatei /etc/passwd (Seite 206)

Beschreibung der Quelle:
Die Passwortdatei ist eine Textdatei und durch Methode 1. werden einzelne Felder `Info:Shell` geändert.

Ergiebigkeit:
Mittel. Das (die) jeweilige(n) Feld(er), das (die) geändert wurde(n).

---

(2) Konfigurationsdatei /etc/shells (Seite 205)

Beschreibung der Quelle:
Die Konfigurationsdatei ist eine Textdatei und wird durch Methode 1. gelesen und ausgewertet. Dabei wird das Dateiattribut "letzter Zugriff" der Konfigurationsdatei auf die aktuelle Systemzeit und Datum gesetzt.

Ergiebigkeit:
Schlecht. Die Konfigurationsdatei enthält Shells, die ein Benutzer als default Shell verwenden darf. Das Dateiattribut "letzter Zugriff" gibt den Zeitpunkt an, an dem der Vorgang: *Benutzer ändert Benutzereintrag*; gestartet wurde.

---

(3) Einträge in History Liste des Benutzers (Seite 221)

Beschreibung der Quelle:
vgl. Seite 26

Ergiebigkeit:
Gut. Die History Liste enthält folgende Kommandos:

1. `chfn <options> /und/oder chsh -s <shell> und/oder <usermod>`

6.1.6 Vorgang: Benutzer ändert Gruppeneintrag

6.1.6.1 Benutzer root

Der Benutzer kann auf drei verschiedene Wege den Gruppeneintrag ändern:

1. Mit einem Texteditor: Der Benutzer editiert die Systemdateien `/etc/group` und/oder `/etc/gshadow` und ändert die betreffenden Felder ab.
2. Mit Hilfe privilegierter Kommandos:
   - `groupmod <options>` (gid, groupname), `gpasswd <options>` (List of Members, Administrator)
3. Durch Verwendung der graphischen Oberfläche YAST2 → Sicherheit und Benutzer → Gruppen bearbeiten und anlegen

Existenz:
Online und Offline Auswertung:
Methode 1: (1), (2), (3)
Methode 2: (1), (2), (3), (4)
Methode 3: (1), (2), (4)

---

(1) Felder in Konfigurationsdatei /etc/group (Seite 206)

Beschreibung der Quelle:
In der Konfigurationsdatei ist eine Textdatei und durch Methode 1. 2. und 3. werden einzelne Felder `Groupname: Password: Gid: List of Members` geändert.

Ergiebigkeit:
Mittel. Das (die) jeweilige(n) Feld(er), das (die) geändert wurde(n).

Verweis auf andere Vorgänge:

1. *Benutzer ändert Datei* (vgl. Seite 88)
2. kein Verweis
3. kein Verweis

---

(2) Felder in Konfigurationsdatei /etc/gshadow (Seite 206)

Beschreibung der Quelle:
Die Konfigurationsdatei ist eine Textdatei und durch Methode 1. 2. und 3. werden einzelne Felder `Groupname: Password: Administrator: List of Members` geändert.

Ergiebigkeit:
Mittel. Das (die) jeweilige(n) Feld(er), das (die) geändert wurde(n).

Verweis auf andere Vorgänge:

1. *Benutzer ändert Datei* (vgl. Seite 88)
2. kein Verweis

3. kein Verweis

---

(3) Einträge in History Liste des Benutzers (Seite 221)

Beschreibung der Quelle:
vgl. Seite 26

Ergiebigkeit:
Mittel. Die History Liste enthält folgende Kommandos:

1.
   - vi /etc/group
   - vi /etc/gshadow
2. groupmod <options> und/ oder gpasswd <options>
3. kein Eintrag

---

(4) Eintrag in Log- Datei /var/log/messages (Seite 210)

Beschreibung der Quelle:
vgl. Seite 26

Ergiebigkeit:
Gut.

1. kein Eintrag
2. und 3.: Der Eintrag enthält folgende Informationen:
   - Time: Wann der Benutzer den Gruppeneintrag geändert hat
   - Host: Rechnername des Hosts
   - pid: Pid vom groupmod bzw. gpasswd Programm
   - field: Feld in /etc/passwd
   - groupname: Name der Gruppe
   - add/remove: Mitglied hinzufügen oder entfernen
   - user: Name des Gruppenmitgliedes der Administrator der Gruppe wird

   Der Eintrag hat folgendes Format:

   <time> <host> groupmod[<pid>]: change <field> for '<groupname>' to <new value>
   oder
   <time> <host> gpasswd[<pid>]: <add/remove> <user> to group <group name> by root
   oder
   <time> <host> gpasswd[<pid>]: set administrators of <groupname> to <user>

6.1.6.2 Normaler Benutzer

Der normale Benutzer kann den Gruppeneintrag nur ändern falls er Gruppenverwalter der Gruppe ist. Der Gruppenverwalter kann nur den Gruppenmitglieder entfernen oder hinzufügen:

1. Mit Hilfe des privilegierten Kommandos:
   - `gpasswd <options>`

Existenz:
Online und Offline Auswertung:
Methode 1: (1), (2), (3), (4)

---

(1) Felder in Konfigurationsdatei /etc/group (Seite 206)

Beschreibung der Quelle:
In der Konfigurationsdatei ist eine Textdatei und durch Methode 1. wird das Feld List of Members geändert.

Ergiebigkeit:
Mittel. Das jeweilige Feld `List of Members`.

---

(2) Felder in Konfigurationsdatei /etc/gshadow (Seite 206)

Beschreibung der Quelle:
In der Konfigurationsdatei ist eine Textdatei und durch Methode 1. wird das Feld `List of Members` geändert.

Ergiebigkeit:
Mittel. Das jeweilige Feld `List of Members`.

---

(3) Eintrag in History Liste des Benutzers (Seite 221)

Beschreibung der Quelle:
vgl. Seite 26

Ergiebigkeit:
Mittel. Die History Liste enthält folgendes Kommando:

1. `gpasswd <options>`

---

(4) Eintrag in Log- Datei /var/log/messages (Seite 210)

Beschreibung der Quelle:
vgl. Seite 26

Ergiebigkeit:
Gut. Der Eintrag enthält folgende Informationen:

- Time: Wann der Benutzer den Gruppeneintrag geändert hat

- Host: Rechnername des Hosts
- pid: Pid vom gpasswd Programm
- groupname: Name der Gruppe
- add/remove: Mitglied hinzufügen oder entfernen
- administrator: Name des Administrators

Der Eintrag hat folgendes Format:

```
<time> <host> gpasswd[<pid>]: <add/remove> <user> to <groupname>
by <administrator>
```

6.1.7 Vorgang: Benutzer ändert Passwort

6.1.7.1 Benutzer root

Der Benutzer root kann sein Passwort und das Passwort anderer Benutzer folgendermaßen ändern:

1. Mittels dem Kommando `passwd <user>` oder `passwd`
2. Mittels der graphischen Oberfläche YAST2 → Sicherheit und Benutzer → Benutzer bearbeiten und anlegen

Online Auswertung und Offline Auswertung:
Methode 1: (1), (2), (3), (4)
Methode 2: (1), (2), (3), (5)

---

(1) Konfigurationsdatei /etc/login.defs (Seite 205)

Beschreibung der Quelle:
Die Konfigurationsdatei ist eine Textdatei und wird von passwd gelesen und ausgewertet. Dabei wird das Dateiattribut “letzter Zugriff” der Konfigurationsdatei auf die aktuelle Systemzeit und Datum gesetzt.

Ergiebigkeit:
Schlecht. Das Loginverhalten der Shadow Suite wird durch die Konfigurationsdatei `/etc/login.defs` festgelegt. Die Konfigurationsdatei enthält folgende Informationen:·

- OBSCURE_CHECKS_ENAB: “yes” ein neues Passwort wird erst akzeptiert, nachdem es einfachen Test unterzogen wurde (minimale Passwortlänge). Führt Root das Kommando passwd aus, wird die Prüfung ausgesetzt.
- PASS_MIN_LEN, PASS_MAX_LEN: Mindestlänge bzw. maximale Länge eines Passworts.
- PASS_ALWAYS_WARN: Ändert Root ein Passwort, das einer Überprüfung mittels den Mechanismen von OBSCURE_CHECKS_ENAB nicht standhalten würde, wird er gewarnt, falls der Wert des Parameters auf “yes” steht.
- CRACKLIB_DICTPATH: Pfad zu den Cracklib- Wörterbüchern.

- PASS_CHANGE_TRIES: Anzahl Versuche, das Passwort zu ändern, bevor passwd abbricht.

Das Dateiattribut "letzter Zugriff" gibt den Zeitpunkt an, an dem der Vorgang: *Benutzer ändert Passwort*, gestartet wurde.

(2) Passwortdatei /etc/passwd (Seite 205)

Beschreibung der Quelle:
Die Passwortdatei `/etc/passwd` ist eine Textdatei und wird von passwd gelesen und ausgewertet. Dabei wird das Dateiattribut "letzter Zugriff" der Konfigurationsdatei auf die aktuelle Systemzeit und Datum gesetzt.

Ergiebigkeit:
Schlecht. Die Passwortdatei enthält den Loginnamen, Passwort (falls keine `/etc/shadow` existiert) die UserID und GroupID, Info, Namen des Home Directories und die verwendete Shell von root und von dem Benutzer, von dem das Passwort geändert werden soll.
Das Dateiattribut "letzter Zugriff" gibt den Zeitpunkt an, an dem der Vorgang: *Benutzer ändert Passwort*, gestartet wurde.

(3) 2. Feld in Shadowdatei /etc/shadow (Seite 206)

Beschreibung der Quelle:
Die Shadowdatei ist eine Textdatei und durch Methode 1. und 2. wird das 2. Feld des Benutzereintrages geändert.

Ergiebigkeit:
Schlecht. Das 2. Feld des Benutzereintrages enthält das neue Passwort des Benutzers.

(4) Eintrag in History Liste des Benutzers (Seite 221)

Beschreibung der Quelle:
vgl. Seite 26

Ergiebigkeit:
Gut. Die History Liste enthält folgende Kommandos:

1. `passwd oder passwd <user>`
2. kein Eintrag

(5) Eintrag in Log- Datei /var/log/messages (Seite 210)

Beschreibung der Quelle:
vgl. Seite 26

Ergiebigkeit:
Gut.

1. kein Eintrag
2. Der Eintrag enthält folgende Informationen:
   - Time: Wann sich der Benutzer das Passwort geändert hat
   - Host: Rechnername des Hosts
   - pid: Pid vom usermod Programm
   - User: Name des Benutzer von dem das Passwort geändert wurde

   Der Eintrag hat folgendes Format:

   ```
   <time> <host> usermod[<pid>]: change '<user>'password
   ```

6.1.7.2 Normaler Benutzer

Der normale Benutzer kann nur sein eigenes Passwort mit dem Kommando `passwd` ändern.

Existenz:
Online und Offline Auswertung: (1), (2), (3), (4)

---

(1) Konfigurationsdatei /etc/login.defs (Seite 205)

Beschreibung der Quelle:
vgl. Seite 76 Informationsquelle (1)

---

(2) Passwortdatei /etc/passwd (Seite 205)

Beschreibung der Quelle:
Die Passwortdatei `/etc/passwd` ist eine Textdatei und wird von passwd gelesen und ausgewertet. Dabei wird das Dateiattribut "letzter Zugriff" der Konfigurationsdatei auf die aktuelle Systemzeit und Datum gesetzt.

Ergiebigkeit:
Schlecht. Die Passwortdatei enthält den Loginnamen, Passwort (falls keine `/etc/shadow` existiert) die UserID und GroupID, Info, Namen des Home Directories und die verwendete Shell des Benutzers, der sein Passwort ändert.
Das Dateiattribut "letzter Zugriff" gibt den Zeitpunkt an, an dem der Vorgang: *Benutzer ändert Passwort*, gestartet wurde.

---

(3) 2. Feld in Shadowdatei /etc/shadow (Seite 206)

Beschreibung der Quelle:
Die Shadowdatei `/etc/shadow` ist eine Textdatei und durch den Aufruf von `passwd` wird das 2. Feld des Benutzereintrages geändert.

Ergiebigkeit:
Schlecht. Das 2. Feld des Benutzereintrages enthält das neue Passwort des Benutzers.

---

(4) Eintrag in History Liste des Benutzers (Seite 221)

Beschreibung der Quelle:
vgl. Seite 26

Ergiebigkeit:
Mittel. Die History Liste enthält folgendes Kommandos: `passwd`

6.1.8 Vorgang: Benutzer ändert Gruppenpasswort

6.1.8.1 Benutzer root

Der Benutzer kann das Gruppenpasswort von jeder Gruppe folgendermaßen ändern:

1. Mittels dem Kommando `gpasswd <group name>`
2. Mittels der graphischen Oberfläche YAST2 → Sicherheit und Benutzer → Gruppe bearbeiten und anlegen

Existenz:
Online und Offline Auswertung:
Methode 1: (1), (2), (3), (4)
Methode 2: (1), (2)

(1) Konfigurationsdatei /etc/group (Seite 205)

Beschreibung der Quelle:
Die Konfigurationsdatei ist eine Textdatei und wird Methode 1. und 2. gelesen und ausgewertet. Dabei wird das Dateiattribut "letzter Zugriff" der Konfigurationsdatei auf die aktuelle Systemzeit und Datum gesetzt.

Ergiebigkeit:
Schlecht. Die Konfigurationsdatei enthält den Gruppennamen, Passwort (falls keine `/etc/gshadow` existiert), die GroupID und Liste der Mitglieder der Gruppe, deren Passwort geändert wird.
Das Dateiattribut "letzter Zugriff" gibt den Zeitpunkt an, an dem der Vorgang: *Benutzer ändert Gruppenpasswort*, gestartet wurde.

(2) 2. Feld in Konfigurationsdatei /etc/gshadow (Seite 206)

Beschreibung der Quelle:
Die Konfigurationsdatei ist eine Textdatei und durch Methode 1. und 2. wird das 2. Feld des Gruppeneintrages geändert.

Ergiebigkeit:
Schlecht. Das 2. Feld des Gruppeneintrages enthält das neue Passwort der Gruppe.

(3) Eintrag in History Liste des Benutzers (Seite 221)

Beschreibung der Quelle:
vgl. Seite 26

Ergiebigkeit:
Mittel. Die History Liste enthält folgende Kommandos:

1. `gpasswd <group name>`
2. kein Eintrag

(4) Eintrag in Log- Datei /var/log/messages (Seite 210)

Beschreibung der Quelle:
vgl. Seite 26

Ergiebigkeit:
Gut.

1. Der Eintrag enthält folgende Informationen:
   - Time: Wann sich der Benutzer das Gruppenpasswort geändert hat
   - Host: Rechnername des Hosts
   - pid: Pid vom gpasswd Programm
   - User: Name des Benutzer von dem das Passwort geändert wurde
2. kein Eintrag

Der Eintrag hat folgendes Format:

```
<time> <host> gpasswd[<pid>]: change the password for group
<groupname> by root
```

6.1.8.2 Normaler Benutzer

Der normale Benutzer kann das Gruppenpasswort nur ändern, falls er Gruppenverwalter der Gruppe ist, mit dem Kommando `gpasswd <groupname>`.

Existenz:
Online und Offline Auswertung: (1), (2), (3), (4)

(1) Konfigurationsdatei /etc/group (Seite 205)

Beschreibung der Quelle:
Die Konfigurationsdatei ist eine Textdatei und wird durch den Aufruf `gpasswd groupname` gelesen und ausgewertet. Dabei wird das Dateiattribut “letzter Zugriff” der Konfigurationsdatei auf die aktuelle Systemzeit und Datum gesetzt.

Ergiebigkeit:
Schlecht. Die Konfigurationsdatei enthält den Gruppennamen, Passwort (falls keine `/etc/gshadow` existiert), die GroupID und Liste der Mitglieder der Gruppe, deren Passwort geändert wird. Das Dateiattribut "letzter Zugriff" gibt den Zeitpunkt an, an dem der Vorgang: *Benutzer ändert Gruppenpasswort*, gestartet wurde.

---

(2) 2. Feld in Konfigurationsdatei /etc/gshadow (Seite 206)

Beschreibung der Quelle:
Die Konfigurationsdatei ist eine Textdatei und durch den Aufruf `gpasswd groupname` 2. wird das 2. Feld des Gruppeneintrages geändert.

Ergiebigkeit:
Schlecht. Das 2. Feld des Gruppeneintrages enthält das neue Passwort der Gruppe.

---

(3) Eintrag in History Liste des Benutzers (Seite 221)

Beschreibung der Quelle:
vgl. Seite 26

Ergiebigkeit:
Mittel. Die History Liste enthält folgende Kommandos: `gpasswd <group name>`

---

(4) Eintrag in Log- Datei /var/log/messages (Seite 210)

Beschreibung der Quelle:
vgl. Seite 26

Ergiebigkeit:
Gut. Der Eintrag enthält folgende Informationen:

- Time: Wann sich der Benutzer das Gruppenpasswort geändert hat
- Host: Rechnername des Hosts
- pid: Pid vom gpasswd Programm
- group name: Name der Gruppe von der das Passwort geändert wurde
- user: Name des Gruppenverwalters

Der Eintrag hat folgendes Format:

```
<time> <host> gpasswd[<pid>]: change the password for group <group
name> by <user>
```

6.1.9 Vorgang: Root löscht Benutzeraccount

Der Benutzer root kann auf drei verschiedene Wege einen Benutzeraccount löschen.

1. Mit einem Texteditor:
   - Der Benutzer editiert die Systemdateien `/etc/passwd` und `/etc/shadow` und entfernt jeweils eine Zeile den Benutzer betreffend und mit `rm -d /HOME/<user>` das Home- Verzeichnis des Benutzers.
2. Mittels dem Kommando `userdel -r <user>`
3. Mittels der graphischen Oberfläche YAST2 → Sicherheit und Benutzer → Benutzer bearbeiten und anlegen

Nicht- Existenz:
Die Informationsquellen (1), (2), (3), (4) dürfen nicht mehr vorhanden sein.

Existenz:
Online und Offline Auswertung:
Methode 1: (5)
Methode 2: (5), (6)
Methode 3: (5)

---

(1) Eintrag in Passwortdatei /etc/passwd (Seite 206)

Die Zeile des zu löschenden Benutzeraccounts in der Passwortdatei wird entfernt.

(2) Eintrag in Shadowdatei /etc/shadow (Seite 206)

Die Zeile des zu löschenden Benutzeraccounts in der Shadowdatei wird entfernt.

(3) Eintrag in Konfigurationsdatei /etc/group (Seite 206)

Der entsprechende Benutzername wird in der Konfigurationsdatei `/etc/group` entfernt.

(4) Home- Verzeichnis /$HOME/<user>)

Das Home Verzeichnis und die darin liegenden Daten des entsprechenden Benutzers werden entfernt.

---

(5) Eintrag in der Log- Datei /var/log/messages (Seite 210)

Beschreibung der Quelle:
vgl. Seite 26

Ergiebigkeit:
Gut.

1. kein Eintrag
2. und 3.: Der Eintrag enthält folgende Informationen:·
   - Time: Wann sich der Benutzer den Benutzer gelöscht hat
   - Host: Rechnername des Hosts
   - pid: Pid vom userdel Programm

- User: Name des Benutzeraccounts, der gelöscht wird
- group name: Name der Gruppe

Der Eintrag hat folgendes Format:

```
<time> <host> userdel[<pid>]: delete user '<user>'und <time>
<host> userdel[<pid>]: delete '<user>' from group '<group name>'
```

---

(6) Einträge in History Liste des Benutzers (Seite 221)

Beschreibung der Quelle:
vgl. Seite 26

Ergiebigkeit:
Gut. Die History Liste enthält folgende Kommandos:

1. - vi /etc/passwd
   - vi /etc/shadow
   - rm -d /$HOME/<user>
2. userdel -r <user>
3. kein Eintrag

6.1.10 Vorgang: Root löscht Gruppenaccount

Der Benutzer root kann auf drei verschiedene Wege einen Gruppenaccount löschen:

1. Mittels reiner Handarbeit:
   - Der Benutzer editiert die Systemdateien /etc/group und /etc/gshadow und entfernt jeweils eine Zeile die Gruppe betreffend.
2. Mittels dem Kommando groupdel <group name>
3. Mittels der graphischen Oberfläche YAST2 → Sicherheit und Benutzer → Gruppen bearbeiten und anlegen

Nicht- Existenz:
Die Informationsquellen (1), (2) dürfen nicht mehr vorhanden sein.

Existenz:
Online und Offline Auswertung:
Methode 1: (4)
Methode 2: (3), (4)
Methode 3: (3)

---

(1) Eintrag in Konfigurationsdatei /etc/group (Seite 206)

Die Zeile des zu löschenden Gruppenaccounts in /etc/group wird entfernt.

(2) Eintrag in Konfigurationsdatei /etc/gshadow (Seite 206)

Die Zeile des zu löschenden Gruppenaccounts in /etc/gshadow wird entfernt.

---

(3) Eintrag in Log- Datei /var/log/messages (Seite 210)

Beschreibung der Quelle:
vgl. Seite 26

Ergiebigkeit:
Gut.

1. kein Eintrag
2. und 3.: Der Eintrag Der Eintrag enthält folgende Informationen:
   - Time: Wann der Benutzer die Gruppe gelöscht hat
   - Host: Rechnername des Hosts
   - pid: Pid vom groupdel Programm
   - group name: Name der Gruppe

   Der Eintrag hat folgendes Format:

   ```
   <time> <host> groupdel[<pid>]: remove group '<group name>'
   ```

---

(4) Einträge in History Liste des Benutzers (Seite 221)

Beschreibung der Quelle:
vgl. Seite 26

Ergiebigkeit:
Gut. Die History Liste enthält folgende Kommandos:

1. - vi /etc/group
   - vi /etc/gshadow
   - rm -d /$HOME/<user>
2. userdel -r <user>
3. kein Eintrag

6.1.11 Vorgang: Benutzer fügt Gruppenmitglied hinzu

Vgl. Seite 73 Vorgang: Benutzer ändert Gruppeneintrag

6.1.12 Vorgang: Benutzer entfernt Gruppenmitglied(er)

Vgl. Seite 73 Vorgang: *Benutzer ändert Gruppeneintrag*

6.1.13 Vorgang: Benutzer nimmt einen Gruppenwechsel vor

(1) vgl. Seite 84 Vorgang: *Benutzer entfernt Gruppenmitglied*

(2) vgl. Seite 84 Vorgang: *Benutzer fügt Gruppenmitglied hinzu*

6.1.14 Vorgang: Benutzer ändert Gruppenverwalter

Vgl. Seite 32 Vorgang: *Benutzer ändert Gruppeneintrag*

6.2 Dateiverwaltung

Linux bietet mehrere Dateisysteme an. Die wichtigsten und am häufigsten verwendeten Dateisysteme sind ext2, ext3 und ReiserFS. Das Virtual File System (VFS) ist die übergeordnete Schnittstelle im Systemkern zwischen den einzelnen Dateisystemen und dem Rest des Systemkerns. Die Unterstützung der Dateisysteme ist Aufgabe des Kernels, d.h. jedes Dateisystem, auf das das System zu zugreifen soll, muss in den Kernel einkompiliert oder als Modul realisiert sein. In der Datei `/proc/filesystems` kann entnommen werden, welche Dateisysteme momentan unterstützt werden. Liegt der Treiber für ein Dateisystem als Modul vor, so erscheint es erst in `/proc/filesystems`, wenn sein Modul geladen wurde. Im Verzeichnis `/lib/modules/<kernel_version>/kernel/fs` liegen die Dateisystemtreiber, die durch Module realisiert sind. In allen drei Linux Dateisystemen werden die Verwaltungsinformationen von den eigentlichen Daten getrennt gespeichert. Verwaltungsdaten (Attribute) einer Datei werden in einem Inode gespeichert. Nur der Name der Datei selbst wird nicht im Inode gespeichert, sondern steht

selbst in einem Verzeichnis und verweist auf den zugehörigen Inode. Ein Verzeichnis ist wiederum auch nur eine Datei und wird durch einen Inode repräsentiert. Die Daten der Verzeichnisdatei sind die enthaltenen Verzeichniseinträge.

6.2.1 Vorgang: Benutzer öffnet Datei zum Lesen

Der Benutzer öffnet die Datei entweder:

1. Mit einem Texteditor z.B. `vi`
2. Mit einem Kommando, wie `cat` um den Inhalt der Datei zu lesen. Der Inhalt der Datei wird mit `cat` auf dem Standard Output ausgegeben.

Existenz:
Online und Offline Auswertung: (1), (2), (3), (4)

---

(1) Dateiattribut "letzter Zugriff" (Seite 230)

Beschreibung der Quelle:
Das Dateiattribut "letzter Zugriff" der Datei auf die aktuelle Datum und Zeit des Systems gesetzt.

Ergiebigkeit:
Mittel. Wann auf die Datei das letzte Mal lesend zugriffen wurde.

---

(2) Dateiattribut "letzter Zugriff" des P.D. (Seite 230)

Beschreibung der Quelle:
Wenn der Benutzer den Vorgang auf die Datei ausführt, wird das Dateiattribut "letzter Zugriff" des Parent Directory auf das aktuelle Datum und die aktuelle Zeit des Systems gesetzt.

Ergiebigkeit:
Mittel. Wann der Vorgang auf die Datei stattgefunden hat.

---

(3) Datei <filename>(Seite 233)

Beschreibung der Quelle:
Diese Datei wird durch den Vorgang geöffnet.

Ergiebigkeit:
Gut. Die Datei enthält folgende gesetzen Dateiattribute:

- File: Name der geöffneten Datei
- Device: Gerätename
- Inode: Nummer des Inodes der geöffneten Datei
- Access: lesend auf reguläre Datei (d)
- Links: 1 Hardlink
- Uid: User ID des Benutzers
- Gid: Group ID des Benutzers, Benutzer ist Mitglied dieser Gruppe
- Device Type: Typ des Gerätes
- Size: Gesamtgröße in Bytes
- IO Block: Blockgröße vom Dateisystem IO
- Access: aktuelles Datum und Systemzeit → atime
- Modify: Datum des letzten schreibenden Zugriffs → mtime
- Change: Datum der letzten Statusänderung → ctime

(4) Einträge in History Liste des Benutzers (Seite 221)

Beschreibung der Quelle:
vgl. Seite 26

Ergiebigkeit:
Mittel. Die History Liste enthält folgende Kommandos:

1. `vi <filename>`
2. `cat <filename>`

6.2.2 Vorgang: Benutzer erstellt Datei

Der Benutzer kann eine Datei folgendermaßen erstellen:

1. Mittels dem Kommando `touch <options> <filename>`
2. Mittels der graphischen Oberfläche Konqueror → Neu → Datei

Existenz:
Online und Offline Auswertung:
Methode 1.: (1), (2), (3), (4)
Methode 2.: (1), (2), (3)

---

(1) Datei <filename>(Seite 233)

Beschreibung der Quelle:
Wenn eine neue Datei angelegt wird, wird für diese ein neuer angelegt und die im Inode gespeicherten Attribute werden gesetzt.

Ergiebigkeit:
Gut. Die Datei enthält folgende gesetzen Dateiattribute:

- File: Name der neu erstellten Datei
- Device: Gerätename
- Inode: Nummer des neuen Inodes
- Access: reguläre Datei mit den Zugriffsrechten, die sich in der `umask` befinden
- Links: Anzahl der Dateinamen (Hardlinks)
- Uid: User ID des Benutzers
- Gid: Group ID des Benutzers
- Device Type: Gerätetyp
- Size: Gesamtgröße in Bytes (0)
- IO Block: Blockgröße vom Dateisystem IO
- Access: aktuelle Datum und aktuelle Zeit des Systems
- Modify: aktuelle Datum und aktuelle Zeit des Systems
- Change: aktuelle Datum und aktuelle Zeit des Systems

---

(2) Dateiattribut "letzter schreibender Zugriff" des P.D. (Seite 230)

Beschreibung der Quelle:
Wenn der Benutzer den Vorgang auf die Datei ausführt, wird das Dateiattribut "letzter schreibender Zugriff" des Parent Directory auf das aktuelle Datum und die aktuelle Zeit des Systems gesetzt.

Ergiebigkeit:
Wann das letzte Mal ein lesender oder schreibender Zugriff auf die Datei stattgefunden hat.

---

(3) Dateiattribut "letzte Statusänderung" des P.D. (Seite 230)

Beschreibung der Quelle:
Wenn der Benutzer den Vorgang auf die Datei ausführt, wird das Dateiattribut "letzte Statusänderung" des Parent Directory auf das aktuelle Datum und die aktuelle Zeit des Systems gesetzt.

Ergiebigkeit:
Wann der Vorgang auf die Datei das letzte Mal eine Statusänderung ausgelöst hat.

(4) Eintrag in History Liste des Benutzers (Seite 221)

Beschreibung der Quelle:
vgl. Seite 26

Ergiebigkeit:
Mittel. Die History Liste enthält folgende Kommandos:

1. `touch <options> filename`
2. kein Eintrag

6.2.3 Vorgang: Benutzer ändert Dateiinhalt

Der Benutzer kann den Dateiinhalt mittels eines Texteditor ändern.

1. Nicht graphischer Editor: Texteditoren `vi` und
2. Graphischer Texteditor: `Kwrite`

Kwrite und vi zeigen einige Unterschiede im Setzen der MAC Zeiten.

Existenz:
Online und Offline Auswertung:
Methode 1: (1), (2), (3), (4), (5)
Methode 2: (1), (2), (3) und (4)

(1) Dateiattribut "letzter schreibender Zugriff"(Seite 230)

Beschreibung der Quelle:
Das Dateiattribut "letzter schreibender Zugriff" der Datei wird auf das aktuelle Datum und die aktuelle Zeit des Systems gesetzt.

1. Unter Verwendung des Editors `vi` wird das Dateiattribut "letzter schreibender Zugriff" auf die aktuelle Zeit gesetzt, wann die Datei abgespeichert wurde.
2. Unter Verwendung des Editors `kwrite` wird das Dateiattribut "letzter schreibender Zugriff" auf die aktuelle Zeit gesetzt, wann die Datei gespeichert wurde.

Ergiebigkeit:
Mittel. Vi und kwrite: Wann auf die Datei das letzte Mal zugegriffen wurde bzw. der geänderte Dateiinhalt gespeichert wurde.

---

(2) Dateiattribut "letzter Zugriff" (Seite 230)

Beschreibung der Quelle:
Das Dateiattribut "letzter Zugriff" der Datei wird auf das aktuelle Datum und die aktuelle Zeit des Systems gesetzt.

1. Unter Verwendung des Editors `vi` wird das Dateiattribut "letzter Zugriff" auf die aktuelle Zeit gesetzt, wann die Datei gespeichert wurde.
2. Unter Verwendung des Editors `kwrite` wird das Dateiattribut "letzter Zugriff" auf die aktuelle Zeit gesetzt, wann die Datei mit dem Editor geöffnet wurde.

Ergiebigkeit:
Mittel. Wann auf die Datei das letzte Mal zugegriffen wurde:

1. Mit `vi`: wann die Datei das letzte Mal abgespeichert wurde.
2. Mit `kwrite`: wann die Datei das letzte Mal geöffnet wurde.

---

(3) Dateiattribut "letzte Statusänderung" (Seite 230)

Beschreibung der Quelle:
Das Dateiattribut "letzte Statusänderung" wird auf das aktuelle Datum und die aktuelle Zeit des Systems gesetzt z.B. beim Wechsel der Zugriffsrechte, des Eigentümers, der Gruppe usw.

1. Unter Verwendung des Editors `vi` wird das Dateiattribut "letzte Statusänderung" auf die aktuelle Zeit gesetzt, wann die Datei gespeichert wurde.
2. Unter Verwendung des Editors `kwrite` wird das Dateiattribut "letzte Statusänderung" auf die aktuelle Zeit gesetzt, wann die Datei mit dem Editor gespeichert wurde.

Ergiebigkeit:
Mittel. Vi und kwrite: Wann auf die Datei das letzte Mal eine Statusänderung ausgeführt wurde bzw. der geänderte Dateiinhalt gespeichert wurde.

---

(4) Datei <filename>(Seite 233)

Beschreibung der Quelle:
Durch Hinzufügen von Zeichen in die Datei, nimmt die Größe der Datei zu, im anderen Falle ab. Die beiden Dateiattribute werden bei dem Vorgang neu gesetzt. Des weiteren ändert sich die Inodenummer.

Ergiebigkeit:
Mittel. Dateiattribute im Inode:

- File: Name der geänderten Datei
- Device: Gerätename
- Inode: Nummer des Inodes
- Access: reguläre Datei
- Links: Anzahl der Dateinamen (Hardlinks)
- Uid: User ID des Benutzers
- Gid: Group ID des Benutzers
- Device Type: Gerätetyp
- Size: Gesamtgröße in Bytes wird größer, wenn Zeichen hinzugefügt werden und kleiner falls Zeichen entfernt werden
- IO Block: Blockgröße vom Dateisystem IO (4096)
- Access: siehe Informationsquelle (2)
- Modify: siehe Informationsquelle (1)
- Change: siehe Informationsquelle (3)

---

(5) Eintrag in History Liste des Benutzers (Seite 221)

Beschreibung der Quelle:
vgl. Seite 26

Ergiebigkeit:
Mittel. Die History Liste enthält folgende Kommandos:

1. `vi <filename>`
2. kein Eintrag

6.2.4 Vorgang: Benutzer kopiert Datei

Der Benutzer kopiert eine Datei:

1. Mittels dem `cp` Kommando: `cp <filename> <source directory>`
2. Mittels Graphischer Oberfläche Konqueror → Kopieren

Existenz:
Online und Offline Auswertung:
Methode 1.: (1), (2), (3), (4), (5), (6)
Methode 2.: (1), (2), (3), (4), (5)

(1) Dateiattribut “letzter Zugriff” (Seite 230)

Beschreibung der Quelle:
Dieses Dateiattribut wird sowohl bei der Originaldatei und der Kopie neu gesetzt.

1. Bei der Originaldatei und der Kopie wird das Dateiattribut “letzter Zugriff” auf die aktuelle Systemzeit und Datum gesetzt.

2. Bei der Originaldatei wird das Dateiattribut auf “letzter Zugriff” auf die aktuelle Systemzeit und Datum gesetzt. Bei der Kopie wird das Dateiattribut “letzter Zugriff” auf die Zeit des letzten Zugriffes auf die Originaldatei gesetzt, d.h. bevor der Kopiervorgang stattgefunden hat.

Ergiebigkeit:
Mittel.

1. Wann die Originaldatei kopiert wurde.

2. Wann die Originaldatei kopiert wurde und Zeitpunkt des letzten Zugriffes vor dem Kopiervorgang

(2) Dateiattribut “letzter schreibender Zugriff” (Seite 230)

Beschreibung der Quelle:
Dieses Dateiattribut wird nur bei der Kopie gesetzt.

1. Bei der Originaldatei wird das Dateiattribut auf “letzter schreibender Zugriff” nicht neu gesetzt. Bei der Kopie wird das Dateiattribut “letzter schreibender Zugriff” auf die aktuelle Systemzeit und Datum gesetzt.

2. Bei der Originaldatei und der Kopie wird das Dateiattribut auf “letzter schreibender Zugriff” nicht neu gesetzt.

Ergiebigkeit:
Mittel.

1. Wann die Kopie entstanden ist.

2. Wann die Originaldatei das letzte Mal modifiziert wurde.

(3) Dateiattribut “letzte Statusänderung” (Seite 230)

Beschreibung der Quelle:

1. und 2.: Bei der Originaldatei wird das Dateiattribut auf “letzte Statusänderung” nicht neu gesetzt. Bei der Kopie wird das Dateiattribut “letzte Statusänderung” auf die aktuelle Systemzeit und Datum gesetzt.

Ergiebigkeit:
Mittel. 1. und 2. Wann die Kopie entstanden ist.

(4) Quelldatei <filename>(Seite 233)

Beschreibung der Quelle:
vgl. Seite 27

Ergiebigkeit:
Gut. Die Datei enthält folgende gesetzte Dateiattribute:

- File: Name der kopierten Datei
- Device: Gerätename
- Inode: Nummer des Inodes der kopierten Datei
- Access: Zugriffsmodus und Dateiart
- Links: 1 Hardlink
- Uid: User ID des Benutzers
- Gid: Group ID des Benutzers
- Device Type: Typ des Gerätes
- Size: Gesamtgröße in Bytes
- IO Block: Blockgröße vom Dateisystem IO
- Access: siehe Informationsquelle (1)
- Modify: siehe Informationsquelle (2)
- Change: siehe Informationsquelle (3)

(5) Zieldatei (Seite 233)

Beschreibung der Quelle:
Die Zieldatei wird beim Kopiervorgang neu erstellt. Es handelt sich hierbei um eine 1:1 Kopie.

Ergiebigkeit:
Gut. Die Datei enthält folgende Informationen:

- File: Name der Kopie
- Device: Gerätename
- Inode: Nummer des Inodes der Kopie
- Access: Zugriffsmodus und Dateiart, wie bei der Quelldatei
- Links: 1 Hardlink
- Uid: User ID des Eigentümers
- Gid: Group ID des Eigentümers
- Device Type: Typ des Gerätes, wie bei Quelldatei

- Size: Gesamtgröße in Bytes, wie bei Quelldatei
- IO Block: Blockgröße vom Dateisystem IO, wie bei Quelldatei
- Access: aktuelles Datum und Systemzeit
- Modify: aktuelles Datum und Systemzeit
- Change: aktuelles Datum und Systemzeit

---

(6) Eintrag in History Liste des Benutzers (Seite 221)

Beschreibung der Quelle:
vgl. Seite 26

Ergiebigkeit:
Mittel. Die History Liste enthält folgende Kommandos:

1. `cp <filename> <source directory>`
2. kein Eintrag

6.2.5 Vorgang: Benutzer ändert Zugriffsrechte der Datei

Der Benutzer ändert Zugriffsrechte der Datei:

1. Mittels dem Kommando: `chmod, chown, chgrp` Mit `chmod` werden die Lese-, Schreibrechte und die Rechte zur Ausführung der Datei erteilt. Mit `chown` kann der Eigentümer geändert werden, wobei nur der Eigentümer der Datei oder root dieses Kommando auf die Datei anwenden können. Das letztere Kommando ändert die Gruppe der Datei.
2. Mittels graphischer Oberfläche Konqueror → Eigenschaften

Existenz:
Online und Offline Auswertung:
Methode 1.: (1), (2)
Methode 2.: (1), (2), (3)

---

(1) Dateiattribut "letzte Statusänderung" (Seite 230)

Beschreibung der Quelle:
Das Dateiattribut "letzte Statusänderung" wird auf das aktuelle Datum und die aktuelle Zeit des Systems gesetzt.

Ergiebigkeit:
Mittel. Wann auf die Datei das letzte Mal eine Statusänderung ausgeführt wurde.

---

(2) Datei <filename>(Seite 233)

Beschreibung der Quelle:
Die Zugriffsrechte auf diese Datei werden durch Methode 1. und 2. geändert.

Ergiebigkeit:
Gut. Die Datei enthält folgende gesetzte Dateiattribute

- File: Name der Datei, bei der die Zugriffsrechte geändert wurden
- Device: Gerätename
- Inode: Nummer des Inodes der geänderten Datei
- Access: neuen Zugriffsrechte und Dateiart
- Links: 1 Hardlink
- Uid: User ID des Benutzers
- Gid: Group ID des Benutzers
- Device Type: Typ des Gerätes
- Size: Gesamtgröße in Bytes
- IO Block: Blockgröße vom Dateisystem IO (4096)
- Access: Datum des letzten Zugriffs (auch lesender) → atime
- Modify: Datum des letzten schreibenden Zugriffs → mtime
- Change: siehe Informationsquelle (1)

(3) Eintrag in History Liste des Benutzers (Seite 221)

Beschreibung der Quelle:
vgl. Seite 26

Ergiebigkeit:
Gut. Die History Liste enthält folgende Kommandos:

1. `chmod oder chown oder chgrp`
2. kein Eintrag

6.2.6 Vorgang: Benutzer löscht Datei

Der Benutzer kann die Datei folgendermaßen löschen:

1. Mittels dem Kommando `rm <filename>`
2. Mittels graphischer Oberfläche Konqueror:
   (a) In den Mülleimer verschieben und dann den Mülleimer leeren
   (b) Datei direkt löschen

Nicht- Existenz:
Die Informationsquellen (1), (2), (3), (4) dürfen nicht mehr vorhanden sein.

Existenz:
Online und Offline Auswertung
Methode 1.: (1), (2), (3), (4), (5), (6)
Methode 2.: (1), (2), (3), (4), (5) oder (2), (3), (4), (5), (6) oder (2), (3), (4), (5)

---

(1) Papierkorb /$HOME/user/Desktop/Trash (Seite 235)

Beschreibung der Quelle:
Der Papierkorb ist bei Suse Linux das Verzeichnis /home/user/Desktop/Trash. In den Papierkorb wird durch Methode 2. verschoben.

Ergiebigkeit:
Gut. Die zu löschende Datei selbst und ihre Dateiattribute bei 2 (b).

---

(2) Dateiattribut "letzter Zugriff" des P.D. (Seite 230)

---

vgl. Seite 85 Informationsquelle (2)

---

(3) Dateiattribut "letzter schreibender Zugriff" des P.D. (Seite 230)

---

vgl. Seite 80 Informationsquelle (2)

---

(4) Dateiattribut "letzte Statusänderung" des P.D. (Seite 230)

---

vgl. Seite 88 Informationsquelle (3)

---

(5) Datei <filename>(Seite 233)

Die Datei ist dann nicht mehr vorhanden.

---

(6) Eintrag in History Liste des Benutzers (Seite 221)

Beschreibung der Quelle:
vgl. Seite 26

Ergiebigkeit:
Mittel. Die History Liste enthält folgende Kommandos:

1. `rm <filename>`
2. kein Eintrag

6.2.7 Vorgang: Benutzer benennt Datei um oder verschiebt Datei

Der Benutzer benennt Datei um oder verschiebt Datei:

1. Mittels dem Kommando `mv <options>`
2. Mittels graphischer Oberfläche Konqueror → Move to oder Umbenennen

Existenz:
Online und Offline Auswertung:
Methode 1.: (1), (2), (3), (4), (5)
Methode 2.: (1), (2), (3), (5)

(1) Dateiattribut "letzter Zugriff" des D.D. und S.D. (Seite 230)

Beschreibung der Quelle:
Wenn der Vorgang ausgeführt wird, wird das Dateiattribut "letzter Zugriff" des Destination und Source Directory auf das aktuelle Datum und die aktuelle Zeit des Systems gesetzt.

Ergiebigkeit:
Mittel. Wann der Vorgang stattgefunden hat und wann auf das Destination Directory das letzte Mal zugegriffen wurde.

(2) Dateiattribut "letzter schreibender Zugriff" des D.D. und S.D. (Seite 230)

Beschreibung der Quelle:
Wenn der Vorgang ausgeführt wird, wird das Dateiattribut "letzter schreibender Zugriff" des Destination Directory auf das aktuelle Datum und die aktuelle Zeit des Systems gesetzt.

Ergiebigkeit:
Mittel. Wann der Vorgang stattgefunden hat und wann auf das Destination Directory das letzte Mal ein schreibender Zugriff stattgefunden hat.

(3) Dateiattribut “letzte Statusänderung” des D.D. und S.D. (Seite 230)

Beschreibung der Quelle:
Wenn der Vorgang ausgeführt wird, wird das Dateiattribut “letzte Statusänderung” des Destination Directory und Source Directory auf das aktuelle Datum und die aktuelle Zeit des Systems gesetzt.

Ergiebigkeit:
Mittel. Wann der Vorgang stattgefunden hat und wann auf das Destination Directory das letzte Mal eine Statusänderung stattgefunden hat.

(4) Eintrag in der History Liste des Benutzers (Seite 221)

Beschreibung der Quelle:
vgl. Seite 26

Ergiebigkeit:
Mittel. Die History Liste enthält folgende Kommandos:

1. `mv <options>`
2. kein Eintrag

(5) Datei <filename>(Seite 233)

Beschreibung der Quelle:
Diese Datei wird umbenannt bzw. verschoben.

Ergiebigkeit:
Gut. Die Datei enthält folgende Informationen:

- File: neuer Name für die umbenannte bzw. verschobene Datei
- Device: Gerätename
- Inode: Nummer des Inodes unverändert
- Access: Zugriffsmodus und Dateiart unverändert
- Links: 1 Hardlink
- Uid: User ID des Benutzers
- Gid: Group ID des Benutzers
- Device Type: Typ des Gerätes
- Size: Gesamtgröße in Bytes
- IO Block: Blockgröße vom Dateisystem IO
- Access: Datum des letzten Zugriffs (auch lesender) unverändert
- Modify: Datum des letzten schreibenden Zugriffs unverändert
- Change: Datum der letzten Statusänderung unverändert

6.2.8 Vorgang: Benutzer erstellt neues Verzeichnis

Der Benutzer erstellt ein neues Verzeichnis:

1. Mittels dem Kommando `mkdir <options> <directory>`
2. Mittels graphischer Oberfläche Konqueror → Neu erstellen → Verzeichnis

Existenz:
Online und Offline Auswertung:
Methode 1.: (1), (2), (3), (4), (5)
Methode 2.: (1), (2), (3), (4)

---

(1) Datei <filename>(Seite 233)

---

vgl. Seite 87 Informationsquelle (1)

---

(2) Dateiattribut "letzter Zugriff" des P.D. (Seite 230)

---

vgl. Seite 85 Informationsquelle (2)

---

(3) Dateiattribut "letzter schreibender Zugriff" des P.D. (Seite 230)

---

vgl. Seite 80 Informationsquelle (2)

---

(4) Dateiattribut "letzte Statusänderung" des P.D. (Seite 230)

---

vgl. Seite 87 Informationsquelle (3)

---

(5) Eintrag in History Liste des Benutzers (Seite 221)

Beschreibung der Quelle:
vgl. Seite 26

Ergiebigkeit:
Mittel. Die History Liste enthält folgende Kommandos:

1. `mkdir <options> directory`
2. kein Eintrag

6.2.9 Vorgang: Benutzer kopiert Verzeichnis

Der Benutzer kopiert das Verzeichnis:

1. Mittels dem Kommando `cp <options> <source> <directory>`
2. Mittels graphischer Oberfläche Konqueror → Copy to

vgl. Seite 90 Vorgang: Benutzer kopiert Datei

6.2.10 Vorgang: Benutzer benennt Verzeichnis um oder verschiebt Verzeichnis

Der Benutzer benennt das Verzeichnis um oder verschiebt das Verzeichnis:

1. Mittels dem Kommando `mv <source> <destination>`
2. Mittels graphischer Oberfläche Konqueror → Move to oder Umbenennen

vgl. Seite 96 Vorgang: Benutzer benennt Datei um oder verschiebt Datei

---

6.2.11 Vorgang: Benutzer ändert Zugriffrechte des Verzeichnisses

Der Benutzer ändert Zugriffsrechte des Verzeichnisses:

1. Mittels dem Kommando `chmod, chown, chgrp`
2. Mittels graphischer Oberfläche Konqueror → Eigenschaften

vgl. Seite 93 Vorgang: Benutzer ändert Zugriffrechte des Verzeichnisses

---

6.2.12 Vorgang: Benutzer löscht Verzeichnis

Der Benutzer kann das Verzeichnis folgendermaßen löschen:

1. Mittels dem Kommando `rm -d <directory>`
2. Mittels graphischer Oberfläche Konqueror:
   (a) In den Mülleimer verschieben und dann den Mülleimer leeren
   (b) Verzeichnis direkt löschen

vgl. Seite 94 Vorgang: Benutzer löscht Datei

6.3 Partitionsverwaltung

Festplatten können in eine oder mehrere logische Partitionen unterteilt werden. Die Unterteilung befindet sich in der Partitionstabelle, die sich im Sektor 0 der Festplatte befindet. Die Partionstabelle kann maximal vier primäre Partitionen enthalten. Eine primäre Partition kann eine erweiterte Partition sein, die logische Partitionen enthält.

6.3.1 Vorgang: Root legt neue Partition an

Der Benutzer root kann eine Partition folgendermaßen anlegen:

1. Mittels dem Kommando `fdisk <options> [Platte]`
2. Mittels graphischer Oberfläche Konqueror → System → Partitionieren

Existenz:
Online und Offline Auswertung:
Methode 1.: (1), (2), (3)
Methode 2.: (1), (2)

---

(1) Virtuelle Datei /proc/filesystems (Seite 224)

Beschreibung der Quelle:
Die virtuelle Datei `/proc/filesystems` ist eine Textdatei und durch Methode 1. und 2. gelesen und ausgwertet. Dabei wird das Dateiattribut “letzter Zugriff“ der virtuellen Datei auf die aktuelle Systemzeit und Datum gesetzt.

Ergiebigkeit:
Schlecht. Die Datei enthält alle Dateisysteme, die der Kernel unterstützt. Das Dateiattribut “letzter Zugriff” gibt den Zeitpunkt an, an dem der Vorgang: *Root legt neue Partition an*, gestartet wurde.

---

(2) Eintrag in virtueller Datei /proc/partitions (Seite 225)

Beschreibung der Quelle:
Die virtuelle Datei `/proc/partitions` ist eine Textdatei und wird durch Methode 1. und 2. um einen Eintrag erweitert.

Ergiebigkeit:
Gut. Der Eintrag enthält folgenden Informationen über die neu angelegt Partion:

- Major: Major- Nummer des Gerätes, auf dem diese Partition liegt.
- Minor: Minor- Nummer des Geräts, auf dem diese Partition liegt. Diese dient dazu, die Partitionen auf verschiedene physische Geräte aufzuteilen und hängt mit der Zahl am Ende des Partitionsnamens zusammen.
- #blocks: Anzahl von Plattenblöcken auf, die in einer bestimmten Partition enthalten sind.
- Name: Name der Partition.

Die Datei hat folgendes Format:

```
Major minor #blocks name
```

---

(3) Eintrag in History Liste des Benutzers (Seite 221)

Beschreibung der Quelle:
vgl. Seite 26

Ergiebigkeit:
Mittel. Die History Liste enthält folgende Kommandos:

1. `fdisk <options> [Platte]`
2. kein Eintrag

6.3.2 Vorgang: Benutzer erstellt Dateisystem

Der Benutzer root kann ein Dateisystem auf einer bestimmten Partition erstellen.

1. Mittels Kommando:
   - Der Benutzer legt eine neue Partition an (vgl. Seite 100 Vorgang: Benutzer legt neue Partition an).
   - Mit den nachfolgenden Kommandos für die verschiedenen unterstützten Dateisysteme kann der Benutzer die Partition formatieren.
     (a) Ext2: `mke2fs <options> device <blocks>`
     (b) Ext3: `mke2fs -j device <blocks>`
     (c) Minix: `mkfs.minix <options> device`
     (d) FAT: `mkdos <options> device <blocks>`
     (e) ReiserFS: `mkreiserfs <options> device <blocks>`
     (f) Swap: `mkswap device`
2. Mittels graphischer Oberfläche Konqueror → System → Partitionieren

Existenz:
Online Auswertung:
Methode 1.: (1), (2), (3), (4), (5)
Methode 2:: (1), (2), (3), (4)
Offline Auswertung:
Methode 1.: (2), (5)
Methode 2.: (2)

(1) Virtuelle Datei /proc/filesystems (Seite 224)

Beschreibung der Quelle:
Die virtuelle Datei `/proc/filesystems` ist eine Textdatei und durch Methode 1. und 2. gelesen und ausgwertet. Dabei wird das Dateiattribut "letzter Zugriff" der virtuellen Datei auf die aktuelle Systemzeit und Datum gesetzt.

Ergiebigkeit:
Schlecht. Die Datei enthält alle Dateisysteme, die der Kernel unterstützt. Das Dateiattribut "letzter Zugriff" gibt den Zeitpunkt an, an dem der Vorgang: *Root legt neue Partition an*, gestartet wurde.

(2) Eintrag in Konfigurationsdatei /etc/fstab (Seite 206)

Beschreibung der Quelle:
Die Konfigurationsdatei ist eine Textdatei und durch Methode 1. und 2. wird ein Eintrag hinzugefügt.

Ergiebigkeit:
Mittel. Die Konfigurationsdatei enthält folgende Informationen über das neu erstellte Dateisystem:

- Device: Blockdevice
- Mountpoint: Verzeichniseintrag, in dem das neu erstellte Dateisystem erscheinen soll
- Type: Typ des Dateisystems
- Options: Optionen
- Dump: Ob das neu erstellte Dateisystem vom Kommando dump zu sichern ist, nur beim Dateisystemtyp ext2 bewertet
- Check: Ob das Dateisystem vor dem Mounten zu überprüfen ist

Der Eintrag hat folgendes Format:

```
<Device> <Mountpunkt> <Type> <Options> <Dump> <Check>
```

(3) Eintrag in Konfigurationsdatei /etc/mtab (Seite 206)

Beschreibung der Quelle:
Die Konfigurationsdatei ist eine Textdatei und wird durch Methode 2. um einen Eintrag erweitert.

Ergiebigkeit:
Mittel. Die Konfigurationsdatei enthält für das neu erstellte Dateisystem folgende Informationen:

- Device: Blockdevice
- Mountpoint: Verzeichniseintrag, in dem das Dateisystem erscheinen soll
- Type: Typ des Dateisystems
- Options: Optionen für den Mountvorgang
- Dump: Gibt an, ob das Dateisystem vom Kommando dump zu sichern ist. Dieser Eintrag wird derzeit nur beim Dateisystemtyp ext2 bewertet.
- Check: Gibt an, ob das Dateisystem vor dem Mounten zu überprüfen ist. Beim Root-Dateisystem sollte hier eine "1" stehen und bei allen anderen entweder eine "0" (keine Prüfung) oder eine "2". Dateisysteme mit gleicher Nummer werden parallel überprüft, das Root- Dateisystem sollte immer allein und als erstes getestet werden.

Der Eintrag hat folgendes Format:

`<Device> <Mountpunkt> <Type> <Options> <Dump> <Check>`

---

(4) Eintrag in virtueller Datei /proc/mounts (Seite 225)

Beschreibung der Quelle:
Die virtuelle Datei ist eine Textdatei und durch Methode 2. um einen Eintrag erweitert.

Ergiebigkeit:
Mittel. Die virtuelle Datei `/proc/mounts` ist sehr ähnlich zum Inhalt von der Konfigurationsdatei `/etc/mtab` (vgl. Seite 101 Informationsquelle Nr. (3)), nur dass sie aktueller sein kann.

---

(5) Eintrag in History Liste des Benutzers (Seite 221)

Beschreibung der Quelle:
vgl. Seite 26

Ergiebigkeit:
Mittel. Die History Liste enthält folgende Kommandos:

1. `Kommando aus 1. (a) - (f)`
2. kein Eintrag

6.3.3 Vorgang: Benutzer bindet Dateisystem ein

Der Benutzer, normalerweise nur der Superuser kann Dateisysteme einhängen. Falls in der `/etc/fstab` die Option `user` in einer Zeile aufgeführt ist, kann auch der normale Benutzer das korespondierende Dateisystem einhängen. Das Dateisystem kann folgendermaßen eingehängt werden:

1. Mittels dem Kommando `mount [options] <mountpoint> or <device>`
2. Icon Menü → Laufwerk einbinden

Existenz:
Online Auswertung:
Methode 1.: (1), (2), (3), (4), (5), (6), (7)
Methode 2.: (1), (2), (3), (4), (5), (6)
Offline Auswertung:
Methode 1.: (1), (4), (7)
Methode 2.: (1), (4)

---

(1) Konfigurationsdatei /etc/fstab (Seite 205)

Beschreibung der Quelle:
Die Konfigurationsdatei ist eine Textdatei und wird durch Methode 1. und 2. gelesen und ausgewertet. Dabei wird das Dateiattribut “letzter Zugriff“ der

Konfigurationsdatei auf die aktuelle Systemzeit und Datum gesetzt.

Ergiebigkeit:
Schlecht. Die Konfigurationsdatei enthält folgende Informationen über die Dateisysteme, die vom Benutzer eingebunden werden dürfen:

- Device: Blockdevice
- Mountpoint: Verzeichniseintrag, in das Dateisystem eingebunden werden soll
- Type: Typ des Dateisystems
- Options: Optionen
- Dump: Ob das Dateisystem vom Kommando dump zu sichern ist, nur beim Dateisystemtyp ext2 bewertet
- Check: Ob das Dateisystem vor dem Mounten zu überprüfen ist

Die Datei hat hat folgendes Format:

```
Device Mountpunkt Type Options, user Dump Check
```

Das Dateiattribut "letzter Zugriff" gibt den Zeitpunkt an, an dem der Vorgang: *Benutzer bindet Dateisystem ein*, gestartet wurde.

---

(2) Virtuelle Datei /proc/partitions (Seite 224)

Beschreibung der Quelle:
Die virtuelle Datei ist eine Textdatei und wird durch den Aufruf von `mount` mit der Option `-U uuid` gelesen und ausgewertet. Dabei wird das Dateiattribut "letzter Zugriff" der virtuellen Datei auf die aktuelle Systemzeit und Datum gesetzt.

Ergiebigkeit:
Schlecht. Die virtuelle Datei enthält folgenden Informationen über die Partition auf der das einzubindende Dateisystem liegt:

- Major: Major- Nummer des Gerätes, auf dem diese Partition liegt.
- Minor: Minor- Nummer des Geräts, auf dem diese Partition liegt. Diese dient dazu, die Partitionen auf verschiedene physische Geräte aufzuteilen und hängt mit der Zahl am Ende des Partitionsnamens zusammen.
- #blocks: Anzahl von Plattenblöcken auf, die in einer bestimmten Partition enthalten sind.
- Name: Name der Partition.

Die Datei hat folgendes Format:

```
Major minor #blocks
```

Das Dateiattribut "letzter Zugriff" gibt den Zeitpunkt an, an dem der Vorgang: *Benutzer bindet Dateisystem ein*, gestartet wurde.

---

(3) Virtuelle Datei /proc/filesystems (Seite 224)

Beschreibung der Quelle:
Die virtuelle Datei `/proc/filesystems` ist eine Textdatei und wird durch den Aufruf von `mount -t auto <device> or <mountpoint>` gelesen und ausgewertet. Die Datei wird nur gelesen, wenn das Dateisystem nicht in der Konfigurationsdatei `/etc/filesystems` vorkommt oder `/etc/filesystems` enthält in der letzten Zeile ein "*". Dabei wird das Dateiattribut "letzter Zugriff" der virtuellen Datei auf die aktuelle Systemzeit und Datum gesetzt.

Ergiebigkeit:
Schlecht. Die virtuelle Datei enthält eine Liste von Dateisystemen, die der Kernel unterstützt.
Das Dateiattribut "letzter Zugriff" gibt den Zeitpunkt an, an dem der Vorgang: *Benutzer bindet Dateisystem ein*, gestartet wurde.

(4) Konfigurationsdatei /etc/filesystems (Seite 205)

Beschreibung der Quelle:
Die Konfigurationsdatei ist eine Textdatei und wird durch den Aufruf von `mount -t auto <device> or <mountpoint>` gelesen und ausgewertet. Dabei wird das Dateiattribut "letzter Zugriff" der Konfigurationsdatei auf die aktuelle Systemzeit und Datum gesetzt.

Ergiebigkeit:
Schlecht. Die Konfigurationsdatei enthält eine Liste von Dateisystemen, die der Reihe nach gelesen werden, falls `mount` mit der `auto` Option aufgerufen wird. Das Dateiattribut "letzter Zugriff" gibt den Zeitpunkt an, an dem der Vorgang: *Benutzer bindet Dateisystem ein*, gestartet wurde.

(5) Eintrag in Konfigurationsdatei /etc/mtab (Seite 206)

Beschreibung der Quelle:
Die Konfigurationsdatei ist eine Textdatei und durch den Aufruf des Kommandos `mount` wird ein Eintrag in die Datei hinzugefügt. Der Eintrag wird nur hinzugefügt, falls `mount` nicht mit der Option `-n` aufgerufen wurde.

Ergiebigkeit:
Mittel. Die Konfigurationsdatei enthält für das eingehängte Dateisystem folgende Informationen:

- Device: Blockdevice
- Mountpoint: Verzeichniseintrag, in dem das Dateisystem erscheinen soll
- Type: Typ des Dateisystems
- Options: Optionen für den Mountvorgang
- Dump: Gibt an, ob das Dateisystem vom Kommando `dump` zu sichern ist. Dieser Eintrag wird derzeit nur beim Dateisystemtyp ext2 bewertet.

- Check: Gibt an, ob das Dateisystem vor dem Mounten zu überprüfen ist. Beim Root-Dateisystem sollte hier eine "1" stehen und bei allen anderen entweder eine "0" (keine Prüfung) oder eine "2". Dateisysteme mit gleicher Nummer werden parallel überprüft, das Root- Dateisystem sollte immer allein und als erstes getestet werden.

Der Eintrag hat folgendes Format:

`<Device> <Mountpunkt> <Type> <Options> <Dump> <Check>`

---

(6) Eintrag in virtueller Datei /proc/mounts (Seite 225)

Beschreibung der Quelle:
Die virtuelle Datei ist eine Textdatei und durch Methode 1. und 2. wird ein Eintrag hinzugefügt.

Ergiebigkeit:
Mittel. Die virtuelle Datei `/proc/mounts` ist sehr ähnlich zum Inhalt von der Konfigurationsdatei `/etc/mtab` (vgl. Seite 103 Informationsquelle (5)), nur dass sie aktueller sein kann.

---

(7) Eintrag in History Liste des Benutzers (Seite 221)

Beschreibung der Quelle:
vgl. Seite 26

Ergiebigkeit:
Mittel. Die History Liste enthält folgende Kommandos:

1. `mount [options] <device> or <mountpoint>`
2. kein Eintrag

6.3.4 Vorgang: Benutzer löst Dateisystem

Der Benutzer, normalerweise nur der Superuser kann Dateisysteme lösen. Falls in der `/etc/fstab` die Option user in einer Zeile aufgeführt ist, kann der normale Benutzer das korespondierende Dateisystem aushängen. Das Dateisystem kann folgendermaßen ausgehängt werden:

1. Mittels dem Kommando `umount <mountpoint> or <device>`
2. Icon Menü → Laufwerk- Einbindung lösen

Nicht- Existenz:
Die Informationsquellen (1), (2) dürfen nicht mehr vorhanden sein.

Existenz:
Online und Offline Auswertung:
Methode 1. und 2.: (3)

(1) Eintrag in Konfigurationsdatei /etc/mtab (Seite 206)

Die Zeile des zu lösenden Dateisystems wird aus der Konfigurationsdatei `/etc/mtab` nur entfernt, wenn `mount` nicht mit der Option `-n` aufgerufen wird.

(2) Eintrag in virtueller Datei /proc/mounts (Seite 225)

Die Zeile des zu lösenden Dateisystems wird aus der virtuellen Datei `/proc/mounts` entfernt.

(3) Eintrag in History Liste des Benutzers (Seite 221)

Beschreibung der Quelle:
vgl. Seite 27

Ergiebigkeit:
Gut. Die History Liste enthält folgende Kommandos:

1. `mount <device> <mountpoint>`
2. kein Eintrag

6.4 Prozessverwaltung

Das Prozessdateisystem stellt zur Laufzeit des Betriebssystems Daten aus dem Kernel in der Form eines normalen Dateisystems dar. Es belegt dabei keinen Platz auf der Festplatte, die Verzeichnisse und Dateien existieren allein im Arbeitsspeicher. Der Kernel fängt Zugriffe auf das `/proc` Dateisystem ab und erzeugt die Verzeichnis- und Dateiinhalte bei Bedarf.

6.4.1 Benutzer startet Prozess

Es können vom Benutzer zwei verschiedene Arten von Prozessen gestartet werden: Daemon- Prozesse und Benutzerprozesse. Daemonen sind Prozesse, die selbst kein Terminal besitzen und benutzen. Benutzerprozesse sind einem bestimmten Terminal zugeordnet.

1. Benutzer startet Daemon-Prozess mit `rc<daemon name> start`

2. Benutzer startet Benutzerprozess `<command>`

Existenz:
Online Auswertung: (1), (2), (3), (4), (5)
Offline Auswertung: (5)

---

(1) /proc/<pid>Verzeichnis (Seite 226)

Beschreibung der Quelle:
Wenn ein neuer Prozess gestartet wird, wird ein neuer Ordner im Verzeichnis `/proc` erstellt. Der Ordner wird mit der Prozess ID bezeichnet.

Ergiebigkeit:
Gut. Der Ordner enthält folgende Dateien und Ordner des gestarteten Prozess:

- cmdline: Kommandozeile für den Prozess, wenn der Prozess nicht vollständig in den Swapbereich ausgelagert ist
- cwd: Link auf das aktuelle Arbeitsverzeichnis des Prozesses
- environ: Daten aus der Prozessumgebung (Environment Variablen)
- exe: Link auf die ausführbare Programmdatei des Prozesses
- Fd: Links auf alle offene Dateien des Prozesses
- maps: Adreßbereich, die Attribute (rwxp) und den Offset, die Gerätenummer und I- Node der mit mmap eingeblendeten virtuellen Speicherbereich des Prozesses
- mem: Zugang zum Speicherbereich des Prozesses
- mounts: Aktuell gemounteter Dateisystem (wie `/etc/mtab`)
- Root: Link auf das root- Verzeichnis des Prozesses
- Stat und statm: Statusinformationen zum Prozess aus der Prozesstabelle

---

(2) Environment Liste (Seite 226)

Beschreibung der Quelle:
Jeder Unix Prozess besitzt seine eigene Umgebung (environment). Diese Environment liegt in Form einer Liste vor, die ihm vor der Startup-Routine übergeben wird. Die Environment ist die Datei `/proc/<pid>/environment`.

Ergiebigkeit:
Gut. Environment Parametern mit denen der Prozess arbeitet.

(3) Eintrag in Prozesstabelle (Seite 226)

Beschreibung der Quelle:
Wenn der Prozess gestartet wird, wird ein Eintrag in die Prozesstabelle, die in `<linux/sched.h>` deklariert ist. Die Informationen zu einem Prozess werden in der Struktur `task_struct` gehalten.

Ergiebigkeit:
Gut. In die Prozesstabelle wird ein Eintrag mit folgenden Informationen zum gestarteten Prozess hinzugefügt.

- COMMAND: Name des Kommandos; Prozesse, die komplett in den Swapbereich ausgelagert sind, werden in Klammern angezeigt
- TIME: verbrauchte Rechenzeit des Prozesses (Summe User- und Kernelmodus) im Format MM:SS
- TT: die Nummer des kontrollierenden Terminals
- UID: die Benutzer-ID des Eigentümers dieses Prozesses
- PID: die Prozeßnummer dieses Prozesses
- PPID: die Prozeßnummer des Elternprozesses
- PGID: die Prozeßgruppe dieses Prozesses
- TPGID: die Prozeßgruppe, der z. Z. das kontrollierende Terminal zu diesem Prozeß gehört
- SID: die Session-ID des Prozesses (ID der Loginshell)
- STAT: Status des Prozesses; folgende Symbole sind möglich: R lauffähig S schlafend D nicht störbarer Schlaf T angehalten oder verfolgt Z Zombie: beendeter Prozess, der noch nicht aus der Prozesstabelle entfernt wurde. W der Prozeß belegt keine Seiten im Arbeitsspeicher I (idle) der Prozeß läuft leer P der Prozeß wird gerade in den Swapbereich ausgelagert x der Prozeß wird mit dem Debugger verfolgt X die System-Calls werden verfolgt s der Prozeß führt eine neue Session an + der Prozeß ist in einer Prozeßgruppe im Vordergrund <kennzeichnet die Prozesse mit höherer Priorität N kennzeichnet die Prozesse mit verminderter Priorität

- F: Flags des Prozesses 10 der Prozeß wird verfolgt (traced); das ist z. B. im Debugger der Fall 20 die System-Calls des Prozesses werden verfolgt 40 der Prozeß hat sich selbst von seinem Elternprozeß gelöst (häufig bei Dämonen) 100 der Prozeß läuft mit Rootrechten 200 der Prozeß hat einen Coredump ausgelöst 400 der Prozeß wird durch ein Signal beendet 100000 Prozeß hat die FPU (den mathematischen Coprozessor) benutzt
- PRI: aktuelle Maximum an Rechenzeit (in Millisekunden), das dem Prozeß zugeteilt wird; wenn der Prozeß gerade läuft, ist das der Rest von der Zeitscheibe
- NI: Nicewert des Prozesses; dieser Wert kann die Geschwindigkeit erhöhen oder verringern, in der die Zeitscheibe eines Prozesses verbraucht wird
- MAJFLT: (auch PAGEIN) Anzahl der "major page faults" (das sind die Versuche, auf eine ausgelagerte Seite zuzugreifen)
- MINFLT: Anzahl der "minor page faults"; diese Zugriffe hatten keine Hardwareaktion zur Folge
- TSIZ: (Textsize) die Größe des Textsegmentes der ausführbaren Datei
- DSIZ: (Datasize) die Differenz aus vsize (virtuelle Größe des Prozesses) und TSIZ
- RSS: (Resident Set Size) ist die Größe des Programms im Arbeitsspeicher; dieser Wert wird aus dem `task_struct` errechnet und ist nicht mit dem RSS Feld in statm identisch
- SIZE: der "virtuelle" Speicherbereich des Prozesses
- STACK: zeigt auf die Basis des nach unten wachsenden Stack
- LIM: zeigt den mit `ulimit -m` eingestellten Grenzwert für den Resident Stack Size %MEM
- ESP: (Extended Stack Pointer) zeigt auf die Spitze des nach unten wachsenden Stack
- EIP: (Extended Instruction Pointer) zeigt auf den aktuellen Maschinenbefehl im Programmtext (<0x60000000) oder in den Shared Libraries (>0x60000000)
- TMOUT: zeigt den Wert eines eventuell gesetzten Timeouts
- ALARM: zeigt den Wert eines eventuell gesetzten Alarmtimers (z. B. von sleep)
- SIGNAL: zeigt ein eventuell gerade anliegendes Signal
- BLOCKED: Bitmaske der blockierten Signale
- IGNORED: Bitmaske der Signale, die ignoriert werden
- CATCHED: Bitmaske für Signale, die abgefangen werden
- WCHAN: Name der Kernelfunktion, in der der Prozeß schläft
- TRS: (Text Resident Size) Größe des Textsegments (enthält keine shared Libraries)
- DRS: (Data Resident Size, auch DSIZ) Größe des Datensegments (enthält benutzte Libraryseiten)

- SIZE: die Speicherbelegung des Prozesses; Text, Daten und Stack unabhängig davon, ob sie sich im physikalischen Speicher befinden
- SWAP: ausgelagerte Speicherseiten in Kilobyte (oder Seiten mit -p); ist einfach die Differenz aus SIZE und RSS
- RSS: (Resident Set Size)
- SHRD: Größe des mit mindestens einem anderen Prozeß gemeinsam benutzten Speicherbereiches
- LIB: (Library Resdident Size) die gesamte Größe der vom Prozeß benutzten Libraryseiten im Arbeitsspeicher
- DT: (Dirty) benutzte Libraryseiten in Kilobyte (oder Seiten mit -p)

---

(4) Datei- Descriptoren in /proc/<pid>/fd (Seite 226)

Beschreibung der Quelle:
Im Verzeichnis `/proc/<pid>/fd` befinden sich die Dateidescriptoren der Dateien, die der Prozess öffnet, nachdem er gestartet wurde.

Ergiebigkeit:
Mittel. Der Eintrag per geöffneter Datei enthält folgendende Informationen:

- Zugriffsart:
- Positionszeiger der Datei:
- Verweis auf I- Node Tabelle:

---

(5) Eintrag in History Liste des Benutzers (Seite 221)

Beschreibung der Quelle:
vgl. Seite 26

Ergiebigkeit:
Gut. Die History Liste enthält folgende Kommandos:

1. `rc<daemon> start`
2. `<command>`

6.4.2 Vorgang: Benutzer beendet Prozess

1. Der Benutzer kann einen Prozess mit dem Kommando kill [pid] wenn sich der Prozess nicht beenden lässt, beenden
2. bei Dämonen mit /etc/init.d/<daemon> stop oder rc<daemon> stop.

Nicht- Existenz:
Die Informationsquellen (1), (2), (3) dürfen nicht mehr vorhanden sein.

Existenz:
Online und Offline Auswertung: (4)

---

(1) Verzeichnis /proc/<pid>(Seite 226)

Der Ordner mit dem Namen der Prozess-ID (pid) des Prozesses, der beendet wurde, wird aus dem /proc Verzeichnis entfernt.

(2) Environment Liste (Seite 226)

Dem Prozess wird die Environment Liste entzogen.

(3) Eintrag in Prozesstabelle (Seite 226)

Der Eintrag für den Prozess wird aus der Prozesstabelle gelöscht, falls der Prozess normal beendet wird.

---

(4) Eintrag in der History Liste des Benutzers (Seite 221)

Beschreibung der Quelle:
vgl. Seite 26

Ergiebigkeit:
Mittel. Die History Liste enthält folgende Kommandos:

1. kill [pid]
2. /etc/init.d/<daemon> stop oder rc <daemon> stop

6.5 Automatisierung von Vorgängen

Der Benutzer kann Jobs zu einem späteren Zeitpunkt oder zyklisch ausführen lassen, um nicht die ganze Zeit am System sein zu müssen.

6.5.1 Vorgang: Benutzer startet Prozess zu bestimmter Zeit

Mit dem Kommando `at` kann der Benutzer Jobs zu einem späteren Zeitpunkt starten lassen. Die Jobs werden nur einmal ausgeführt. Der Benutzer ruft das Kommando `at <options> <ZEIT>` auf. At liest liest die zu startenden Kommandos entweder:

1. Von der Standardeingabe: `at <ZEIT>`
2. Aus einer Datei: `at -f <filename> ZEIT`

Existenz:
Online und Offline Auswertung:
Methode 1.: (1), (3), (4), (5) oder (1), (3), (5)
Methode 2.: (1), (2), (3), (4), (5) oder (1), (2), (3), (5)

---

(1) Zugriffskontrolldateien /etc/at.{allow, deny} (Seite 205)

Beschreibung der Quelle:
Die beiden Dateien `/etc/at.allow` und `/etc/at.deny` sind Textdateien, die beim Aufruf von at gelesen werden und so die Berechtigung zur Benutzung von at regeln. Dabei wird das Dateiattribut "letzter Zugriff" der Konfigurationsdatei auf die aktuelle Systemzeit und Datum gesetzt.

Ergiebigkeit:
Mittel. Die Datei `/etc/at.allow` enthält Benutzer, die das Kommando verwenden dürfen. Die Datei `/etc/at.deny` enthält Benutzer, die das Kommando nicht verwenden dürfen. Das Dateiattribut "letzter Zugriff" gibt den Zeitpunkt an, an dem der Vorgang: *Benutzer startet Prozess zur bestimmter Zeit*, gestartet wurde.

---

(2) At- Datei <filename>(Seite 233)

Beschreibung der Quelle:
Die `at- Datei` ist eine Textdatei und wird durch Methode 1. gelesen und ausgewertet. Dabei wird das Dateiattribut "letzter Zugriff" der At-Datei auf die aktuelle Systemzeit und Datum gesetzt.

Ergiebigkeit:
Mittel. Die `at- Datei` enthält die Kommandos des Benutzers, die zu einem späteren Zeitpunkt gestartet werden sollen. Das Dateiattribut "letzter Zugriff" gibt den Zeitpunkt an, an dem der Vorgang: *Benutzer startet Prozess zur bestimmter Zeit*, gestartet wurde.

---

(3) Warteschlange /var/spool/atjobs (Seite 213)

Beschreibung der Quelle:
Die Jobs werden in eine Warteschlange (Job Queue) eingereiht. Dabei wird eine Textdatei des Jobs angelegt. Die Datei bekommt dann einen bestimmten

Jobnamen. Die Warteschlange für die Jobs ist schon existent. Die Jobdatei ist noch nicht existent. Sie entsteht erst wenn at aufgerufen wurde.

Ergiebigkeit:
Mittel. Die Jobdatei enthält folgende Informationen:
Verwendete Shell, Environment Parametern die gesetzt werden, Benutzernamen, uid, gid, Liste von Kommandos, die zu einem bestimmten Zeitpunkt ausgeführt werden sollen.In der Warteschlange sind die Jobs nach Nummer, Datum, Stunde und Jobklasse eingereiht.

---

(4) Mail in Datei /var/spool/mail/<user>(Seite 213)

Beschreibung der Quelle:
Die Datei /var/spool/mail/user ist eine Textdatei, in die Mails von root, at Jobs des Benutzers betreffend hinzugefügt werden. Dem User werden alle Ausgaben (STDOUT und STDERR) per Mail zugeschickt.

Ergiebigkeit:
Gut. Welche(s) Kommando(s) der Benutzer ausgeführt hat, die STDOUT und STDERR als Ausgabe hatten.

---

(5) Eintrag in History Liste des Benutzers (Seite 221)

Beschreibung der Quelle:
vgl. Seite 26

Ergiebigkeit:
Mittel. Die History Liste enthält folgende Kommandos: `at -f <filename> ZEIT oder at ZEIT`.

6.5.2 Vorgang: Benutzer lässt Jobs zyklisch ausführen

Cron ist ein Dämon der verwendet wird, um die Ausführung wiederkehrender Tasks (Jobs) zu bestimmten Zeiten zu regeln. *Cron* führt im Gegensatz zu at die Jobs wiederholt aus. Cron sucht zuerst im Verzeichnis `/var/spool/tabs` nach Crontab Dateien, die nach den Accounts in der Passwortdatei /etc/passwd benannt sind. Des weiteren sucht er nach seiner Hauptkonfigurationsdatei /etc/crontab und in den Verzeichnisse `/etc/cron.hourly, daily, weekly, monthly` nach Scripten, die auszuführen sind. Der Cron Dämon erwacht jede Minute, um alle gespeicherten Crontabs zu untersuchen, ob sie in der aktuellen Minute ausgeführt werden müssen. Zusätzlich überprüft er jede Minute, ob sich die Modifikationszeit des spool- Verzeichnisses oder die Modifikationszeit der Hauptkonfigurationszeit sich geändert hat. Falls sie sich geändert hat überprüft cron die Modifikationszeit aller Crontabs und lädt diese neu, die sich geändert haben. Deshalb muss Cron nie neu gestartet werden, wenn eine Crontab verändert wird.
Bei cron werden benutzerdefinierte und systemdefinierte Crontab unterschieden.

6.5.2.1 Systemdefinierte Crontab

Die systemdefinierte Crontab kann nur durch root durch editieren der Hauptkonfigurationsdatei `/etc/crontab` um auszuführende Tasks erweitert werden.

Existenz:
Online und Offline Auswertung: (1), (2), (3)

---

(1) Zugriffskontrolldateien /etc/cron.{allow, deny} (Seite 205)

Beschreibung der Quelle:
Die beiden Dateien `/etc/cron.allow` und `/etc/cron.deny` sind Textdateien, die beim Aufruf von at gelesen werden und so die Berechtigung zur Benutzung von at regeln. Dabei wird das Dateiattribut "letzter Zugriff" der Konfigurationsdatei auf die aktuelle Systemzeit und Datum gesetzt.

Ergiebigkeit:
Mittel. Die Datei `/etc/cron.allow` enthält Benutzer, die das Kommando verwenden dürfen. Die Datei `/etc/cron.deny` enthält Benutzer, die das Kommando nicht verwenden dürfen. Das Dateiattribut "letzter Zugriff" gibt den Zeitpunkt an, an dem der Vorgang: *Benutzer lässt Jobs zyklisch ausführen*, gestartet wurde.

---

(2) Eintrag in der Konfigurationsdatei /etc/crontab (Seite 206)

Beschreibung der Quelle:
Die Konfigurationsdatei ist eine Textdatei und wird um Einträge ergänzt.

Ergiebigkeit:
Mittel.

- Environment Parametern:
  - SHELL: welche Shell
  - PATH: Pfad, der zur Ausführung der Kommandos verwendet wird
  - MAILTO: Ausgaben von cron werden an den Benutzer, der bei MAILTO angegeben wird geschickt.
  - HOME: Home Verzeichnis, das zum Ausführen der Kommandos und Scripte verwendet werden soll.
    Jede Zeile: Minute, Stunde, Tag, Monat, Wochentag, User, Kommando
- Optional:
  - `/etc/cron.daily`: Skripte in diesem Verzeichnis werden täglich ausgeführt
  - `/etc/cron.hourly`: Skripte in diesem Verzeichnis werden stündlich ausgeführt
  - `/etc/cron.weekly`: Skripte in diesem Verzeichnis werden wöchentlich ausgeführt
  - `/etc/cron.monthly`: Skripte in diesem Verzeichnis werden monatlich ausgeführt
  - `/etc/cron.d`: Enthält cronjobs, die nicht täglich, stündlich, wöchentlich oder monatlich ausgeführt werden sollen.

---

(3) Eintrag in History Liste des Benutzers (Seite 221)

Beschreibung der Quelle:
vgl. Seite 26

Ergiebigkeit:
Mittel. Die History Liste enthält folgende Kommandos: `cp <script> /etc/cron.`{`daily, hourly, weekly, monthly, d`}.

6.5.3 Benutzerdefinierte Crontabs

Der normale Benutzer kann auch Tasks zyklisch ausführen lassen. Dabei ruft er das Kommando `crontab -e` auf und kann so seine eigene Crontab editieren.

Existenz:
Online und Offline Auswertung: (1), (2), (3)

(1) Zugriffskontrolldateien /etc/cron.{allow, deny}

vgl. Seite 115 Informationsquelle (1)

(2) Eintrag in der Crontab /var/spool/tabs/<user>(Seite 213)

Beschreibung der Quelle:
Die benutzerdefinierte Crontab ist eine Textdatei und wird vom Benutzer um Einträge ergänzt. Mit der Ausführung von `crontab -e` vom Benutzer wird ein Texteditor geöffnet um die Crontab zu editieren. Die benutzerdefinierte Crontab hat das gleiche Format, wie die Konfigurationsdatei `/etc/crontab`. Das Crontab `/var/spool/cron/tabs/<user>` wird dann angelegt, aber nur wenn die benutzerdefinierte crontab das richige Format hat.

Ergiebigkeit:
Mittel. Wann der Benutzer, welches Kommando bzw. Skripte automatisch ausführen lassen will.

(3) Eintrag in der History Liste des Benutzers (Seite 221)

Beschreibung der Quelle:
vgl. Seite 26

Ergiebigkeit:
Mittel. Die History Liste enthält folgendes Kommando: `crontab -e`

6.6 Netzwerk

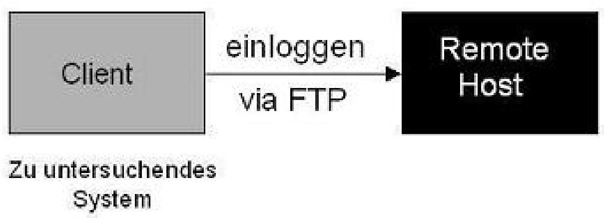
6.6.1 File Transfer Protocol

Der klassische Internet- Dienst FTP dient dazu, Dateien zwischen zwei Rechnern auszutauschen. Der FTP- Server dient zum Bereitstellen von Dateien zum Fernladen (download) durch Benutzer und Aufnehmen von Dateien zum Fernspeichern (upload) durch Benutzer. Der FTP- Server wird durch die globale Konfigurationsdatei `/etc/vsftpd.conf` gesteuert. FTP- Clients nutzen diese Dienste anonym oder mit dem Benutzernamen und Passwort. Bei einem anonymen Zugang gibt der Benutzer den Benutzernamen "anonymous" und als Passwort seine Benutzerkennung ein. Bei einem benutzerauthentifizierten Zugang gibt der Benutzer seinen Username und Passwort vom Server ein, wie beim normalen Einloggen. FTP benötigt zwei separate TCP- Verbindungen: eine um Kommandos und Antworten zwischen Client und Server zu übertragen (*command channel*) und eine weitere um die Daten auszutauschen (*data channel*). Auf der Serverseite wird für den command channel standardmäßig Port 21 verwendet und für den data channel Port 20. Um eine FTP-Verbindung aufzubauen wird entweder der *active mode* oder *passive mode* verwendet (siehe Seite 239).

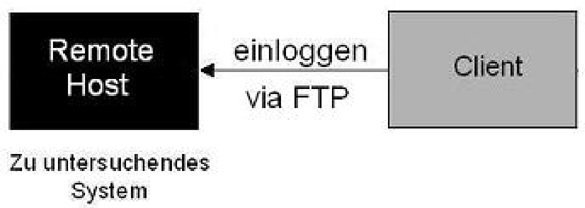
6.6.1.1 Vorgang: Benutzer loggt sich via FTP ein

Der Benutzer kann sich folgendermaßen via FTP einloggen:

1. Der Benutzer ist/ war auf dem zu untersuchenden System eingeloggt und will sich von diesem Rechner aus via FTP an einem anderen Rechner einloggen.



2. Der Benutzer ist/ war nicht auf dem untersuchenden System eingeloggt und will sich via FTP auf dem zu untersuchenden System einloggen.



6.6.1.1.1 FTP- Client auf zu untersuchenden System

Der Benutzer loggt sich vom untersuchenden System aus auf einen anderen Rechner ein. Der Benutzer kann sich auf dem Server folgendermaßen einloggen:

1. Anonyme, das Verzeichnis `/srv/ftp` ist als Root- Verzeichnis eingestellt, falls in der Konfigurationsdatei `/etc/vsftpd.conf` der Parameter `anonymous_enable` auf "yes" gesetzt ist, oder
2. Als Systembenutzer (lokaler Benutzer), der nicht in der Datei `vsftpd.chroot_list` aufgeführt ist, bei ihm ist das jeweilige Home- Verzeichnis als Root- Verzeichnis eingestellt, falls in der Konfigurationsdatei `/etc/vsftpd.conf` die beiden Parameter `chroot_list_enable` auf "yes" und `chroot_list_file` auf die Chroot Liste `/etc/vsftpd.chroot_list` gesetzt sind, oder
3. Als Systembenutzer (lokaler Benutzer), der in der Datei `vsftpd.chroot_list` aufgeführt ist, behält vollen Zugriff auf das Dateisystem, falls in der Konfigurationsdatei `/etc/vsftpd.conf` die beiden Parameter `chroot_list_enable` auf "yes" und `chroot_list_file` auf die Chroot Liste `/etc/vsftpd.chroot_list` gesetzt sind.

Existenz:
Online Auswertung: (1), (2)
Offline Auswertung: (2)

---

(1) Ausgehende TCP- Verbindung (Seite 231)

Beschreibung der Quelle:
Eine ausgehende TCP- Verbindung wird beim Einloggen via FTP erzeugt.

Ergiebigkeit:
Gut. Die ausgehende TCP- Verbindung birgt folgendene Informationen:

- Local Adress: IP- Adresse des Clients
- Local Port: >1024
- Foreign Adress: IP- Adresse des Remote Hosts (Server)
- Foreign Port: 21
- State: Verbindung hergestellt

---

(2) Eintrag in History Liste des Benutzers (Seite 221)

Beschreibung der Quelle:
vgl. Seite 26

Ergiebigkeit:
Gut. Die History Liste enthält folgendes Kommando: `ftp <Rechnername>`

6.6.1.1.2 FTP- Server auf zu untersuchenden System

Das Einloggen erfolgt hier genauso, wie in Abschnitt 6.6.1.1.

Existenz:
Online Auswertung:
Methode 1.: (1), (6), (7), (8), (9)
Methode 2.: (1), (2), (3), (4), (5), (7), (8), (9)
Methode 3.: (1), (2), (3), (4), (7), (8), (9)
Offline Auswertung: Methode 1.: (1), (6), (8), (9)
Methode 2.: (1), (2), (3), (4), (5), (8), (9)
Methode 3.: (1), (2), (3), (4), (8), (9)

---

(1) Konfigurationsdatei /etc/vsftpd.conf (Seite 205)

Beschreibung der Quelle:
Die Konfigurationsdatei `/etc/vsftpd.conf` ist eine Textdatei und wird von *vsftpd* gelesen und ausgewertet. Dabei wird das Dateiattribut "letzter Zugriff" der Konfigurationsdatei auf die aktuelle Systemzeit und Datum gesetzt.

Ergiebigkeit:
Schlecht. Die Konfigurationsdatei enthält folgenden Parameter:

- `anonymous_enable (NO)`: Ist diese Option gesetzt, ist der anonyme FTP-Zugang zugelassen.
- `chroot_list_enable (NO)`: Bei Aktivierung werden lokale Benutzer, die in einer Datei `/etc/vsftp.chroot_list` aufgeführt sind, vom chroot- Wechsel in ihr Heimatverzeichnis ausgenommen (in Verbindung mit der Option `chroot_local_usr` sinnvoll). Die zu verwendende Konfigurationsdatei kann mittels der Option `chroot_list_file` geändert werden.
- `log_ftp_protocol (NO)`: Bei aktiver Option werden alle FTP-Anforderungen und -Antworten protokolliert.
- `text_userdb_names (NO)`: Eigentümer/Gruppen werden beim Dateilisting als Namen anstatt als numerische ID's dargestellt.
- `xferlog_enable (NO)`: Ermöglicht die detaillierte Protokollierung von Downloads und Uploads. Die Daten landen in der in `xferlog_file` benannten Datei (Voreinstellung `/var/log/vsftpd.log`).
- `anon_root (none)`: Bei anonymen Zugang erfolgt eine Wechsel in das angegebene Verzeichnis (via chroot).
- `chroot_list_file (/etc/vsftpd.chroot_list)`: Existiert die angegebene Datei und sind die Optionen `chroot_list_enable` aktiv bzw. `chroot_local_user` nicht aktiv, so werden die in der Datei benannten lokalen Benutzer via chroot bei Anmeldung in ihr Heimatverzeichnis verbannt.
- `guest_username (ftp)`: Der Benutzername für den anonymen Zugang, falls dieser auf ein spezielles "Gastlogin" gemappt ist. Die Option wird nur betrachtet, wenn `guest_enable` gesetzt ist.

- `ftp_username (ftp)`: Der Benutzername für den anonymen Zugang. Das Heimatverzeichnis ist i.d.R. ein spezielles FTP-Verzeichnis, das mittels eines chroot- Umgebung betreten wird. Der Unterschied zum Gast-Zugang (vergleiche `guest_username`) ist im Wesentlichen, das letzterer nicht zwingend in einer chroot-Umgebung gefangen ist.
- `xferlog_file (/var/log/vsftpd.log)`: Protokolldatei für die Transferstatistik

Das Dateiattribut "letzter Zugriff" gibt den Zeitpunkt an, an dem der Vorgang: *Benutzer loggt sich via FTP ein*, gestartet wurde.

(2) Chroot Liste /etc/vsftpd.chroot_list (Seite 205)

Beschreibung der Quelle:
Die Konfigurationsdatei eine Textdatei und wird nur durch Methode 2. und 3. gelesen. Die Konfigurationsdatei `/etc/vsftpd.chroot_`list ist nur vorhanden, wenn der Parameter `chroot_list_file` in der Konfigurationsdatei `/etc/vsftpd.conf` auf diese Konfigurationsdatei gesetzt ist. Dabei wird das Dateiattribut "letzter Zugriff" der Konfigurationsdatei auf die aktuelle Systemzeit und Datum gesetzt.

Ergiebigkeit:
Gut. Die Chroot- Liste enthält Benutzer, die vollen Zugriff auf das Dateisystem haben. Das Dateiattribut "letzter Zugriff" gibt den Zeitpunkt an, an dem der Vorgang: *Benutzer loggt sich via FTP ein*, gestartet wurde.

(3) Passwortdatei /etc/passwd (Seite 205)

Beschreibung der Quelle:
Die Passwortdatei ist eine Textdatei und wird von 2. und 3. gelesen und ausgewertet. Dabei wird das Dateiattribut "letzter Zugriff" der Konfigurationsdatei auf die aktuelle Systemzeit und Datum gesetzt.

Ergiebigkeit:
Gut. Die Passwortdatei enthält unteranderem den Loginnamen, Passwort (falls keine `/etc/shadow` existiert) die UserID und GroupID aller Benutzer, Info, Namen des Home Directories und die verwendete Shell des Benutzers, der sich via FTP einloggt.
Das Dateiattribut "letzter Zugriff" gibt den Zeitpunkt an, an dem der Vorgang: *Benutzer loggt sich via FTP ein*, gestartet wurde.

(4) Konfigurationsdatei /etc/shadow (Seite 205)

Beschreibung der Quelle:
Die Shadowdatei ist eine Textdatei und wird vom login Programm gelesen und ausgewertet. Die Datei ist nur vorhanden, falls in der Passwortdatei `/etc/passwd` an 2. Stelle ein "x" steht. Dabei wird das Dateiattribut "letzter Zugriff" der Konfigurationsdatei auf die aktuelle Systemzeit und Datum gesetzt.

Ergiebigkeit:
Mittel. Aus der Shadow Datei enthält unteranderem Informationen über den Benutzer, der sich via FTP einloggt.

- Username: wie in /etc/passwd
- Passwort: Verschlüsseltes Passwort
- DOC: Day of last change, Tag ab dem 1.1.1970, an dem das Passwort zuletzt geändert wurde
- MinD: Minimale Anzahl Tage, die das Passwort gültig ist
- MaxD: Maximale Anzahl Tage, die das Passwort gültig ist
- Warn :Anzahl der Tage vor Ablauf der Lebensdauer des Passwortes, ab der vor dem Verfall zu warnen ist
- Exp: Expire, wieviele Sekunden das Passwort trotz Ablauf der MaxD gilt
- Dis: Bis zu diesem Tag (gezählt ab 1.1.1970) ist dieser Account gesperrt
- Res: Reserve, Feld wird derzeit nicht ausgewertet

Das Dateiattribut "letzter Zugriff" gibt den Zeitpunkt an, an dem der Vorgang: *Benutzer loggt sich via FTP ein*, gestartet wurde.

---

(5) Home- Verzeichnis /$HOME/<user>(Seite 235)

Beschreibung der Quelle:
Wenn der Benutzer ein lokaler Benutzer ist, wird nach beim Einloggen die Change-Root Umgebung auf das Home- Verzeichnis des Benutzers gesetzt. Die Change-Root Umgebung ist das Home- Verzeichnis, falls in der Konfigurationsdatei `/etc/vsftpd.conf` der Parameter `chroot_local_user` auf "yes" gesetzt ist. Dabei wird das Dateiattribut "letzter Zugriff" des Home- Verzeichnisses auf die aktuelle Systemzeit und Datum gesetzt.

Ergiebigkeit:
Gut. Welche Dateien sich in dem Home Verzeichnis des Benutzers befinden. Das Verzeichnis selbst hat folgende gesetzte Dateiattribute:

- File: /home/<user>
- Device: Gerät
- Inode: Nummer des Inodes des Home Verzeichnisses
- Access: Zugriffsmodus und Verzeichnis (d)
- Links: 1 Hardlink
- Uid: User ID des Benutzers
- Gid: Group ID des Benutzers
- Device Type: Gerätetyp
- Size: Gesamtgröße in Bytes
- IO Block: Blockgröße vom Dateisystem IO
- Access: aktuelles Datum und Systemzeit
- Modify: Datum des letzten schreibenden Zugriffs

- Change: Datum der letzten Statusänderung

Das Dateiattribut "letzter Zugriff" gibt den Zeitpunkt an, an dem der Vorgang: *Benutzer loggt sich via FTP ein*, gestartet wurde.

(6) Chroot Directory /srv/ftp (Seite 235)

Beschreibung der Quelle:
Wenn der Benutzer sich anonym einloggt, wird die Change- Root Umgebung auf das Verzeichnis `/srv/ftp` gesetzt. Dabei wird das Dateiattribut "letzter Zugriff" das Verzeichnis `/srv/ftp` auf die aktuelle Systemzeit und Datum gesetzt.

Ergiebigkeit:
Gut. Das Chroot Verzeichnis enthält die gleichen Informationen wie das Home-Verzeichnis des Benutzers. Das Dateiattribut "letzter Zugriff" gibt den Zeitpunkt an, an dem der Vorgang: *Benutzer loggt sich via FTP ein*, gestartet wurde.

(7) Eingehende TCP- Verbindung (Seite 231)

Beschreibung der Quelle:
Eine eingehende TCP- Verbindung wird beim Einloggen des Benutzers via FTP erzeugt.

Ergiebigkeit:
Gut. Die eingehende TCP- Verbindung birgt folgendene Informationen:

- Local Adress: IP- Adresse des FTP- Servers
- Local Port: 21
- Foreign Adress: IP- Adresse des Clients
- Foreign Port: >1023
- State: Verbindung hergestellt

(8) Eintrag in Log- Datei /var/log/vsftpd.log (Seite 210)

Beschreibung der Quelle:
vgl. Seite 27

Ergiebigkeit:
Gut.

1. Der Eintrag enthält folgende Informationen:
   - Time: Wann sich der Benutzer am Host eingeloggt hat
   - IP- address: IP- Adresse des Client Rechner
   - pid: Pid vom vsftpd Dämon
   - e- mail: e- mail Adresse des Benutzers

Der Eintrag hat folgendes Format:

```
<time> ftp[<pid>]: OK LOGIN: Client ‘‘<IP- address>’’,
anonpassword <e-mail address>
```

2. Der Eintrag enthält folgende Informationen:
   - Time: Wann sich der Benutzer am Host eingeloggt hat
   - IP- address: IP- Adresse des Client Rechner
   - pid: Pid vom vsftpd Dämon
   - user: Name des Benutzers

   Der Eintrag hat folgendes Format:

```
<time> ftp[<pid>]:[<user>]OK LOGIN: Client ‘‘<IP- address>’’
```

---

(9) Eintrag in Log- Datei /var/log/messages (Seite 210)

Beschreibung der Quelle:
vgl. Seite 26

Ergiebigkeit:
Gut. Der Eintrag enthält folgende Informationen:

- Time: Wann sich der Benutzer am Host eingeloggt hat
- IP- address: IP- Adresse des Client Rechner
- pid: Pid vom vsftpd Dämon
- Host IP- address: IP- Adresse des Host Rechners

Der Eintrag hat folgendes Format:

```
<time> vsftpd[<pid>]: connect from ‘‘<IP- address>’’ (<Host IP-
address>)
```

6.6.1.2 Vorgang: Benutzer macht Download

Der Benutzer ist entweder auf dem zu untersuchenden System eingeloggt und macht von einem anderen Rechner einen Download oder ist auf einem anderen Rechner eingeloggt und macht vom untersuchenden System, das der Server ist, einen Download.

6.6.1.2.1 FTP- Client auf zu untersuchenden System

Der Benutzer ist auf dem zu untersuchenden System eingeloggt und hat eine Verbindung zu dem FTP- Server aufgebaut und macht einen Download:

1. Als anonymer Benutzer
2. Als lokaler Benutzer

Existenz:
Online Auswertung:
Methode 1. und 2.: (1), (2), (3)
Offline Auswertung:
Methode 1. und 2.: (1)

---

(1) Datei <filename>(Seite 233)

Beschreibung der Quelle:
Die Datei wird vom Benutzer vom FTP- Server heruntergeladen.

Ergiebigkeit:
Mittel. Die Datei enthält folgende gesetzte Dateiattribute:

- File: Name der heruntergeladenen Datei
- Device: Gerät des Client Rechners
- Inode: Nummer des Inodes der heruntergeladenen Datei
- Access: Zugriffsmodus und Dateiart, wie Datei auf dem Server
- Links: 1 Hardlink
- Uid: User ID des Benutzers
- Gid: Group ID des Benutzers
- Device Type: Gerätetyp
- Size: Gesamtgröße in Bytes
- IO Block: Blockgröße vom Dateisystem IO
- Access: aktuelles Datum und Systemzeit des gestarteten Downloads
- Modify: aktuelles Datum und Systemzeit des gestarteten Downloads
- Change: aktuelles Datum und Systemzeit des gestarteten Downloads

(2) Eingehende TCP- Verbindung (Seite 231)

Beschreibung der Quelle:
Eine eingehende TCP- Verbindung wird beim Download einer Datei via aktiven FTP erzeugt.

Ergiebigkeit:
Gut. Die eingehende TCP- Verbindung birgt folgendene Informationen:

- Local Adress: IP- Adresse des Clients
- Local Port: >1023
- Foreign Adress: IP- Adresse des FTP- Servers
- Foreign Port: 20
- State: Verbindung hergestellt

(3) Ausgehende TCP- Verbindung (Seite 231)

Beschreibung der Quelle:
Eine ausgehende TCP- Verbindung wird beim beim Download einer Datei via passiven FTP erzeugt.

Ergiebigkeit:
Gut. Die ausgehende TCP- Verbindung birgt folgendene Informationen:

- Local Adress: IP- Adresse des Clients
- Local Port: >1023
- Foreign Adress: IP- Adresse des FTP- Servers
- Foreign Port: >1023
- State: Verbindung hergestellt

6.6.1.2.2 FTP- Server auf dem zu untersuchenden System

Der Benutzer loggt sich via FTP auf den zu untersuchenden System ein (vgl. Vorgang: Benutzer loggt sich via FTP ein Seite) und macht einen Download:

1. Als anonymer Benutzer
2. Als lokaler Benutzer

Existenz:
Online Auswertung:
Methode 1. und 2.: (1), (2), (3), (4), (5)
Offline Auswertung:
Methode 1. und 2.: (1), (2), (5)

(1) Konfigurationsdatei /etc/vsftpd.conf (Seite 205)

Beschreibung der Quelle:
Die Konfigurationsdatei ist eine Textdatei und wird von *vsftpd* gelesen und ausgewertet. Dabei wird das Dateiattribut "letzter Zugriff" der Konfigurationsdatei auf die aktuelle Systemzeit und Datum gesetzt.

Ergiebigkeit:
vgl. Seite 119 Informationsquelle (1)

(2) Eintrag in Log- Datei /var/log/vsftpd.log (Seite 210)

Beschreibung der Quelle:
vgl. Seite 27

Ergiebigkeit:
Gut. Der Eintrag enthält folgende Informationen:

- Time: Wann der Benutzer vom Host einen Download vorgenommen hat
- IP- address: IP- Adresse des Client Rechner
- pid: Pid vom vsftpd Dämon
- Path: Pfad der Datei
- Size: Größe der heruntergeladenen Datei
- Speed: Übertragungsrate in Kbyte/sec

Der Eintrag hat folgendes Format:

```
<time> ftp[<pid>]: OK DOWNLOAD: Client ‘‘<IP- address>’’,
‘‘/<path>/<filename>’’, <size>, <speed> Kbyte/sec
```

(3) Ausgehende TCP- Verbindung (Seite 231)

Beschreibung der Quelle:
Eine augsehende TCP- Verbindung wird beim beim Download einer Datei via aktiven FTP erzeugt.

Ergiebigkeit:
Gut. Die ausgehende TCP- Verbindung birgt folgendene Informationen:

- Local Adress: IP- Adresse des FTP- Servers
- Local Port: >1023
- Foreign Adress: IP- Adresse des Clients
- Foreign Port: 20
- State: Verbindung hergestellt

(4) Eingehende TCP- Verbindung (Seite 231)

Beschreibung der Quelle:
Eine eingehende TCP- Verbindung wird beim Download einer Datei via passiven FTP erzeugt.

Ergiebigkeit:
Gut. Die ausgehende TCP- Verbindung birgt folgendene Informationen:

- Local Adress: IP- Adresse des FTP- Servers
- Local Port: >1023
- Foreign Adress: IP- Adresse des Clients
- Foreign Port: >1023
- State: Verbindung hergestellt

(5) Datei <filename>(Seite 233)

Beschreibung der Quelle:
Diese Datei wird vom Benutzer vom FTP Server heruntergeladen. Den Dateinamen bekommt man aus der Log- Datei `/var/log/vsftpd.log` des Servers.

Ergiebigkeit:
Mittel. Die Datei enthält folgende gesetzte Dateiattribute:

- File: Name der Datei, die zum Download steht
- Device: Gerät des Server Rechners
- Inode: Nummer des Inodes der Datei, die zum Download steht
- Access: Zugriffsmodus und Dateiart der
- Links: 1 Hardlink
- Uid: User ID des FTP Servers (ftp)
- Gid: Group ID des FTP Servers (ftp)
- Device Type: Gerätetyp
- Size: Gesamtgröße in Bytes
- IO Block: Blockgröße vom Dateisystem IO
- Access: aktuelles Datum und Systemzeit des gestarteten Downloads
- Modify: Datum des letzten schreibenden Zugriffs, unverändert
- Change: Datum der letzten Statusänderung, unverändert

6.6.1.3 Vorgang: Benutzer macht Upload

Der Benutzer ist entweder auf dem zu untersuchenden System eingeloggt und macht von einem anderen Rechner einen Upload oder ist auf einem anderen Rechner eingeloggt und macht vom untersuchenden System, das der Server ist, einen Upload.

6.6.1.3.1 FTP- Client auf zu untersuchenden System

Der Benutzer ist auf dem zu untersuchenden System eingeloggt (vgl. Seite 117 Vorgang: *Benutzer loggt sich via FTP*) und hat eine Verbindung zu dem FTP-Server aufgebaut und macht einen Upload:

1. Als anonymer Benutzer
2. Als lokaler Benutzer

Existenz:
Online Auswertung:
Methode 1. und 2.: (1), (2), (3), (4)
Offline Auswertung:
Methode 1. und 2.: (1), (4)

---

(1) Datei <filename>(Seite 233)

Beschreibung der Quelle:
Die Datei wird vom Benutzer auf den FTP- Server übertragen.

Ergiebigkeit:
Mittel. Die Datei enthält folgende gesetzte Dateiattribute:

- File: Name der Datei, die der Benutzer auf den Server laden möchte
- Device: Gerät des Client Rechners
- Inode: Nummer des Inodes der Datei, die auf den Server geladen wird
- Access: Zugriffsmodus und Dateiart
- Links: 1 Hardlink
- Uid: User ID des Benutzers
- Gid: Group ID des Benutzers
- Device Type: Gerätetyp
- Size: Gesamtgröße in Bytes
- IO Block: Blockgröße vom Dateisystem IO
- Access: aktuelles Datum und Systemzeit des gestarteten Uploads
- Modify: Datum des letzten schreibenden Zugriffs
- Change: Datum der letzten Statusänderung

(2) Eingehende TCP- Verbindung (Seite 231)

Beschreibung der Quelle:
Eine eingehende TCP- Verbindung wird beim Upload einer Datei via aktiven FTP erzeugt.

Ergiebigkeit:
Gut. Die eingehende TCP- Verbindung birgt folgendene Informationen:

- Local Adress: IP- Adresse des Clients
- Local Port: >1023
- Foreign Adress: IP- Adresse des FTP- Servers
- Foreign Port: 20
- State: Verbindung hergestellt

(3) Ausgehende TCP- Verbindung (Seite 231)

Beschreibung der Quelle:
Eine ausgehende TCP- Verbindung wird beim beim Upload einer Datei via passiven FTP erzeugt.

Ergiebigkeit:

Gut. Die ausgehende TCP- Verbindung birgt folgendene Informationen:

- Local Adress: IP- Adresse des Clients
- Local Port: >1023
- Foreign Adress: IP- Adresse des FTP- Servers
- Foreign Port: >1023
- State: Verbindung hergestellt

(4) Eintrag in History Liste des Benutzers (Seite 221)

Beschreibung der Quelle:
vgl. Seite 27

Ergiebigkeit:
Mittel. Die History Liste enthält folgendes Kommando: `ftp <servername>`

6.6.1.3.2 FTP- Server auf zu untersuchenden System

Der Benutzer loggt sich via FTP auf den zu untersuchenden System ein (vgl. Seite 117 Vorgang: *Benutzer loggt sich via FTP ein*) und macht einen Upload:

1. Als anonymer Benutzer
2. Als lokaler Benutzer

Existenz:
Online Auswertung:
Methode 1. und 2.: (1), (2), (3), (4), (5)
Offline Auswertung:
Methode 1. und 2.: (1), (3), (5)

(1) Konfigurationsdatei /etc/vsftpd.conf (Seite 205)

vgl. Seite 119 Informationsquelle (1)

(2) Eingehende TCP- Verbindung (Seite 231)

Beschreibung der Quelle:
Eine eingehende TCP- Verbindung wird beim Upload einer Datei via passiven FTP erzeugt.

Ergiebigkeit:
Gut. Die eingehende TCP- Verbindung birgt folgendene Informationen:

- Local Adress: IP- Adresse des FTP- Servers
- Local Port: >1023
- Foreign Adress: IP- Adresse des Clients
- Foreign Port: >1023
- State: Verbindung hergestellt

(3) Datei <filename>(Seite 233)

Beschreibung der Quelle:
Die Datei bekommt der FTP Server neu hinzu. Den Dateinamen bekommt man aus der Log- Datei `/var/log/vsftpd.log`

Ergiebigkeit:
Mittel. Die Datei enthält folgende gesetzte Dateiattribute:

- File: Name der Datei, die auf den Server geladen wurde
- Device: Gerät des Server Rechners
- Inode: Nummer des Inodes der Datei, die auf den Server geladen wurde

- Access: Zugriffsmodus und Dateiart
- Links: 1 Hardlink
- Uid: User ID des FTP Servers (ftp)
- Gid: Group ID des FTP Servers (ftp)
- Device Type: Gerätetyp
- Size: Gesamtgröße in Bytes
- IO Block: Blockgröße vom Dateisystem IO
- Access: aktuelles Datum und Systemzeit des gestarteten Downloads
- Modify: aktuelles Datum und Systemzeit des gestarteten Downloads
- Change: aktuelles Datum und Systemzeit des gestarteten Downloads

(4) Ausgehende TCP- Verbindung (Seite 231)

Beschreibung der Quelle:
Eine ausgehende TCP- Verbindung wird beim beim Upload einer Datei via aktiven FTP erzeugt.

Ergiebigkeit:
Gut. Die ausgehende TCP- Verbindung birgt folgendene Informationen:

- Local Adress: IP- Adresse des FTP- Servers
- Local Port: >1023
- Foreign Adress: IP- Adresse des Clients
- Foreign Port: 20
- State: Verbindung hergestellt

(5) Eintrag in Log- Datei /var/log/vsftpd.log (Seite 210)

Beschreibung der Quelle:
vgl. Seite 27

Ergiebigkeit:
Gut. Der Eintrag enthält folgende Informationen:

- Time: Wann der Benutzer vom Host einen Upload vorgenommen hat
- IP- address: IP- Adresse des Client Rechner
- pid: Pid vom vsftpd Dämon
- Path: Pfad der Datei
- Size: Größe der heruntergeladenen Datei

- Speed: Übertragungsrate in Kbyte/sec

Der Eintrag hat folgendes Format:

```
<time> ftp[<pid>]: OK UPLOAD: Client ‘‘<IP- address>’’,
‘‘/<path>/<filename>’’, <size>, <speed> Kbyte/sec
```

6.6.2 Telnet (Terminal Emulation over Network

Der Telnet Dienst ermöglicht das Eröffnen einer Sitzung auf einem entfernten Rechner. Das Programm (und das gleichnahmige Protokoll) definieren ein “Network Virtual Terminal”, über das Programme bedient oder andere Dienste angesprochen werden können. Auf dem entfernten Rechner (Server) muss der telnetd- Dämon gestartet sein und in der Konfigurationsdatei `/etc/inetd.conf` des Internet Service Dämons inetd Einträge über Telnet und die Remote Shell vorhanden sein.

6.6.2.1 Vorgang: Benutzer loggt sich beim Remote Host ein

Der Benutzer ist/ war entweder auf dem zu untersuchenden System eingeloggt und will sich von diesem Rechner aus auf einem Remote Host via telnet einloggen oder ist/ war nicht auf dem zu untersuchenden System eingeloggt und will sich via telnet auf dem zu untersuchenden System via telnet einloggen.

6.6.2.1.1 Client auf zu untersuchenden System

Der Benutzer ist auf dem zu untersuchenden System eingeloggt und ruft das Kommando `telnet <server>` auf.

Existenz:
Online Auswertung: (2), (3), (4)
Offline Auswertung: (2), (4)

---

(1) Konfigurationsdatei /etc/iptos (Seite 205)

Beschreibung der Quelle:
Die Konfigurationsdatei ist eine Textdatei und wird durch den Aufruf von `telnet` mit der Option `-a tos` gelesen und ausgewertet. Dabei wird das Dateiattribut “letzter Zugriff“ der Konfigurationsdatei auf die aktuelle Systemzeit und Datum gesetzt.

Ergiebigkeit:
Schlecht. Die Konfigurationsdatei enthält Type Of Service (TOS) Namen, die das IP- Protokoll braucht:

- Symbolische Namen für TOS Optionen Bits:
  - none 0x00
  - delay 0x10
  - throughput 0x08
  - reliability 0x04
  - reserved1 0x02
  - reserved2 0x01
- Symbolische Namen für TOS Precedence Werte:
  - netcontrol 0xe0
  - internetcontrol 0xc0
  - critic/ecp 0xa0
  - flashoverride 0x80

- flash 0x60
- immediate 0x40
- priority 0x20
- routine 0x00

Das Dateiattribut “letzter Zugriff” gibt den Zeitpunkt an, an dem der Vorgang: *Benutzer loggt sich beim Remote Host ein*, gestartet wurde.

---

(2) Konfigurationsdatei /etc/securetty (Seite 205)

Beschreibung der Quelle:
Die Konfigurationsdatei ist eine Textdatei und wird vom *telnetd* gelesen und ausgewertet. Dabei wird das Dateiattribut “letzter Zugriff“ der Konfigurationsdatei auf die aktuelle Systemzeit und Datum gesetzt.

Ergiebigkeit:
Schlecht. Die Konfigurationsdatei enthält vertrauenswürdige Terminals über die sich root einloggen kann. Das Dateiattribut “letzter Zugriff” gibt den Zeitpunkt an, an dem der Vorgang: *Benutzer loggt sich beim Remote Host ein*, gestartet wurde.

---

(3) TCP- Verbindung (Seite 231)

Beschreibung der Quelle:
Eine TCP- Verbindung wird vom *telnet Client* erzeugt

Ergiebigkeit:
Gut. Die TCP- Verbindung birgt folgendene Informationen:

- Local Adress: IP- Adresse des Clients
- Local Port: >1024
- Foreign Adress: IP- Adresse des Remote Hosts (Server)
- Foreign Port: 23
- State: aktive Verbindung oder Listen Zustand

---

(4) Eintrag in History Liste des Benutzers (Seite 221)

Beschreibung der Quelle:
vgl. Seite 26

Ergiebigkeit:
Mittel. Die History Liste enthält folgendes Kommando: `telnet <server>`

6.6.2.1.2 Telnetd auf zu untersuchenden System

Der Benutzer ist auf einem anderen Rechner eingeloggt und will sich auf untersuchenden System einloggen, in dem er das Kommando `telnet <server>` aufruft.

Existenz:
Online Auswertung:
Anmeldung mit /etc/passwd: (1), (2), (7), (8), (10), (11), (12), (13), (14), (15)
Anmeldung mit Passwd Map: (3), (4), (7), (8), (10), (11), (12), (13), (14), (15)
Offline Auswertung:
Anmeldung mit /etc/passwd: (1), (2), (7), (8), (10), (12)
Anmeldung mit Passwd Map: (3), (4), (7), (8), (10), (12)

---

(1) Passwort /etc/passwd (Seite 205)

Beschreibung der Quelle:
Die Passwortdatei ist eine Textdatei und wird vom `telnetd` gelesen und ausgewertet, falls in der Konfigurationsdatei `/etc/nsswitch.conf` für die Servicedatei passwd der Parameter `files` eingetragen ist, d.h. die lokale Passwortdatei wird beim Einloggen via Telnet verwendet. Dabei wird das Dateiattribut "letzter Zugriff" der Konfigurationsdatei auf die aktuelle Systemzeit und Datum gesetzt.

Ergiebigkeit:
Gut. Die Passwortdatei enthält unteranderem den Loginnamen, Passwort (falls keine `/etc/shadow` existiert) die UserID und GroupID, Info, Namen des Home Directories und die verwendete Shell des Benutzers, der sich mit telnet auf dem Rechner einloggt.
Das Dateiattribut "letzter Zugriff" gibt den Zeitpunkt an, an dem der Vorgang: *Benutzer loggt sich beim Remote Host ein*, gestartet wurde.

---

(2) Konfigurationsdatei /etc/shadow (Seite 205)

Beschreibung der Quelle:
Die Shadowdatei ist eine Textdatei und wird vom telnet Programm gelesen und ausgewertet. Die Datei ist nur vorhanden, falls in der Passwortdatei `/etc/passwd` an 2. Stelle ein "x" steht. Dabei wird das Dateiattribut "letzter Zugriff" der Konfigurationsdatei auf die aktuelle Systemzeit und Datum gesetzt.

Ergiebigkeit:
Mittel. Aus der Shadow Datei kann man folgenden Informationen über den Benutzer beziehen, der sich mit telnet auf dem Rechner einloggt:

- Username: wie in /etc/passwd
- Passwort: Verschlüsseltes Passwort
- DOC: Day of last change, Tag ab dem 1.1.1970, an dem das Passwort zuletzt geändert wurde
- MinD: Minimale Anzahl Tage, die das Passwort gültig ist
- MaxD: Maximale Anzahl Tage, die das Passwort gültig ist

- Warn: Anzahl der Tage vor Ablauf der Lebensdauer des Passwortes, ab der vor dem Verfall zu warnen ist
- Exp: Expire, wieviele Sekunden das Passwort trotz Ablauf der MaxD gilt
- Dis: Bis zu diesem Tag (gezählt ab 1.1.1970) ist dieser Account gesperrt
- Res: Reserve, Feld wird derzeit nicht ausgewertet

Das Dateiattribut "letzter Zugriff" gibt den Zeitpunkt an, an dem der Vorgang: *Benutzer loggt sich beim Remote Hos ein*, gestartet wurde.

(3) /var/yp/<nisdomain>/passwd.{byname, byuid} (Seite 217)

Beschreibung der Quelle:
Die passwd- Map ist eine Datei, in der die Daten im DBM-Format (Database Management) organisiert sind. Die passwd Map wird von einem NIS- Server auf das System übertragen falls in der Konfigurationsdatei `/etc/nsswitch.conf` für die Servicedatei passwd der Parameter `nis` eingetragen ist. Es wird entweder die Map `passwd.byname` oder `passwd.uid` angefordert.

Ergiebigkeit:
Mittel. Aus der `Passwd Map` kann man Informationen über die Benutzer bekommen, die sich via telnet auf dem System einloggen dürfen: Loginnamen des Benutzers, Passwort (falls keine `Shadow Map` existiert) die UserID und GroupID des Benutzers, Info, Namen des Home Directories und die verwendete shell.

(4) /var/yp/<nisdomain>/shadow.byname (Seite 217)

Beschreibung der Quelle:
Die shadow.byname- Map ist eine Datei, in der die Daten im DBM-Format (Database Management) organisiert sind. Die Shadow Map wird vom NIS- Server auf das System übertragen, falls in der Konfigurationsdatei `/etc/nsswitch.conf` für die Servicedatei passwd der Parameter `nis` eingetragen ist.

Ergiebigkeit:
Mittel. Die Shadow Map enthält folgende Informationen:

- Username: wie in `/etc/passwd`
- Passwort: Verschlüsseltes Passwort
- DOC: Day of last change, Tag ab dem 1.1.1970, an dem das Passwort zuletzt geändert wurde
- MinD: Minimale Anzahl Tage, die das Passwort gültig ist
- MaxD: Maximale Anzahl Tage, die das Passwort gültig ist
- Warn: Anzahl der Tage vor Ablauf der Lebensdauer des Passwortes, ab der vor dem Verfall zu warnen ist
- Exp: Expire, wieviele Sekunden das Passwort trotz Ablauf der MaxD gilt
- Dis: Bis zu diesem Tag (gezählt ab 1.1.1970) ist dieser Account gesperrt

- Res: Reserve, Feld wird derzeit nicht ausgewertet

---

(5) Konfigurationsdatei /etc/services (Seite 205)

Beschreibung der Quelle:
Die Konfigurationsdatei ist eine Textdatei und wird vom *telnetd* gelesen und ausgewertet. Dabei wird das Dateiattribut “letzter Zugriff“ der Konfigurationsdatei auf die aktuelle Systemzeit und Datum gesetzt.

Ergiebigkeit:
Schlecht. Die Konfigurationsdatei enthält Informationen bzgl. der bekannten Dienste, die vom DARPA Internet bereitgestellt werden.

- Official service name: telnet
- Port number: Port Nummer des Dienstes, an dem er auf Anfragen wartet, bei Telnet standardmäßig Port 23
- Protocol name: TCP
- Aliases: anderer Name für den Dienst

Die Konfigurationsdatei hat folgendes Format:

```
official_service_name port_number/protocol_name aliases
```

Das Dateiattribut “letzter Zugriff” gibt den Zeitpunkt an, an dem der Vorgang: *Benutzer loggt sich beim Remote Host ein*, gestartet wurde.

---

(6) Konfigurationsdatei /etc/protocols (Seite 205)

Beschreibung der Quelle:
Die Konfigurationsdatei ist eine Textdatei und wird von *telnetd* gelesen und ausgewertet. Dabei wird das Dateiattribut “letzter Zugriff“ der Konfigurationsdatei auf die aktuelle Systemzeit und Datum gesetzt.

Ergiebigkeit:
Schlecht. Die Konfigurationsdatei enthält die Protokollnummern der Transportprotokolle, die vom TCP/ IP Subsystem bereitgestellt werden. Die Konfigurationsdatei hat folgendes Format:

```
Protocol name Number of Protocol
```

Das Dateiattribut “letzter Zugriff” gibt den Zeitpunkt an, an dem der Vorgang: *Benutzer loggt sich beim Remote Host ein*, gestartet wurde.

---

(7) Konfigurationsdatei /etc/hosts (Seite 205)

Beschreibung der Quelle:
Die Konfigurationsdatei ist eine Textdatei und wird von *telnetd* gelesen und ausgewertet. Dabei wird das Dateiattribut "letzter Zugriff" der Konfigurationsdatei auf die aktuelle Systemzeit und Datum gesetzt.

Ergiebigkeit:
Mittel. Die Konfigurationsdatei enthält eine Liste von Hostname zu Adresse Abbildungen für das TCP/IP Subsystem. Die Konfigurationsdatei hat folgendes Format:

```
IP- Address Full-Qualified-Hostname Short-Hostname
```

Das Dateiattribut "letzter Zugriff" gibt den Zeitpunkt an, an dem der Vorgang: *Benutzer loggt sich beim Remote Host ein*, gestartet wurde.

(8) Konfigurationsdatei /etc/host.conf (Seite 205)

Beschreibung der Quelle:
Die Konfigurationsdatei ist eine Textdatei und wird von `telnetd` gelesen und ausgewertet. Dabei wird das Dateiattribut "letzter Zugriff" der Konfigurationsdatei auf die aktuelle Systemzeit und Datum gesetzt.

Ergiebigkeit:
Mittel. Die Konfigurationsdatei enthält Informationen, darüber wie Hostnames aufgelöst werden.

```
order hosts,bindmulti on
```

Zuerst wird die `/etc/hosts` für eventuelle Namens Abfragen überprüft, und dann wird ein Nameserver gefragt, falls einer vorhanden ist. Der Eintrag multi erlaubt mehrere IP- Adresse für eine bestimmte Maschine.
Das Dateiattribut "letzter Zugriff" gibt den Zeitpunkt an, an dem der Vorgang: *Benutzer loggt sich beim Remote Host ein*, gestartet wurde.

(9) Konfigurationsdatei /etc/nsswitch.conf(Seite 205)

Beschreibung der Quelle:
Die Konfigurationsdatei ist eine Textdatei und wird von `telnetd` gelesen und ausgewertet. Dabei wird das Dateiattribut "letzter Zugriff" der Konfigurationsdatei auf die aktuelle Systemzeit und Datum gesetzt.

Ergiebigkeit:
Schlecht. In der Konfigurationsdatei werden die Mechanismen für die meisten auch über Netzwerk verteilbare Servicedateien (z.B. passwd, group) festgelegt.

- Name der Datenbank: Namen der lokalen Konfigurationsdatei
- Mechanismen: Mechanismen, die in gegebener Reihenfolge, angewandt werden, um eine Information aus der betreffenden Datenbank zu gewinnen. Liefert ein Versuch kein Ergebnis, wird in der Voreinstellung das nächste Verfahren versucht.

Die folgenden Mechanismen sind möglich:

- Files: mittels lokaler Dateien (z.B. `/etc/passwd`)
- Dns: mittels DNS
- Nis oder yp: mittels NIS [NOTFOUND=return]: bewirkt, daß die Suche gestoppt wird, wenn der vorhergehenden Mechanismus zu keinem Ergebnis kommt.
- Compat: aktiviert die Auswertung einer erweiterten Syntax in den Datenbanken passwd, shadow und group

Die Konfigurationsdatei hat folgendes Format:

```
Name der Datenbank: Mechanismen
```

Das Dateiattribut "letzter Zugriff" gibt den Zeitpunkt an, an dem der Vorgang: *Benutzer loggt sich beim Remote Host ein*, gestartet wurde.

---

(10) Konfigurationsdatei /etc/resolv.conf (Seite 205)

Beschreibung der Quelle:
Die Konfigurationsdatei ist eine Textdatei und wird von *telnetd* gelesen und ausgewertet. Dabei wird das Dateiattribut "letzter Zugriff" der Konfigurationsdatei auf die aktuelle Systemzeit und Datum gesetzt.

Ergiebigkeit:
Schlecht. Die Konfigurationsdatei enthält IP- Adressen und die Domain Suchliste. Die Konfigurationsdatei hat folgendes Format:

```
search <domain 1> ... <domain n> nameserver <IP- Adresse>
```

Das Dateiattribut "letzter Zugriff" gibt den Zeitpunkt an, an dem der Vorgang: *Benutzer loggt sich beim Remote Host ein*, gestartet wurde.

---

(11) TCP- Verbindung (Seite 231)

Beschreibung der Quelle:
Eine TCP- Verbindung wird vom *telnetd* erzeugt.

Ergiebigkeit:
Gut. Die TCP- Verbindung birgt folgende Informationen:

- Local Adress: IP- Adresse des Servers
- Local Port: 23
- Foreign Adress: IP- Adresse des Clients
- Foreign Port: >1024
- State: aktive Verbindung

(12) Eintrag in Log- Datei /var/log/messages (Seite 210)

Beschreibung der Quelle:
vgl. Seite 26

Ergiebigkeit:
Gut. Der Eintrag enthält folgende Informationen:

- Time: Wann sich der Benutzer am Host eingeloggt hat
- Local Hostname: Rechnername
- IP- address: IP- Adresse des Client Rechner
- pid: Pid vom in.telnetd Dämon
- Host IP- address: IP- Adresse des Host Rechners

Der Eintrag hat folgendes Format:

```
<time> <local hostname> in.telnetd[<pid>]: connect from ‘‘<IP-
address>’’ (<Host IP- address>)
```

(13) Eintrag in Log- Datei /var/log/lastlog (Seite 210)

Beschreibung der Quelle:
vgl. Seite 27

Ergiebigkeit:
Gut. Der Eintrag in der `/var/log/lastlog` enthält den Benutzernamen, Portnummer, und die Zeit des letzten erfolgreichen Einloggens.

(14) Eintrag in Log- Datei /var/log/wtmp (Seite 210)

Beschreibung der Quelle:
vgl. Seite 27

Ergiebigkeit:
Gut. Der Eintrag in der Log- Datei enthält den Benutzernamen, den Terminal, die Loginzeit und von wo aus sich der Benutzer eingeloggt hat.

(15) Eintrag in Log- Datei /var/log/utmp (Seite 210)

Beschreibung der Quelle:
vgl. Seite 27

Ergiebigkeit:
Gut. Der Eintrag enthält folgende Informationen den Benutzer folgenden Informationen:

- Ut_type: Typ des Logins (Login_Prozess)

- Ut_pid: Prozess- ID des Login Prozesses
- Ut_line: Gerätename des Terminals (pts)
- Ut_id: ID des Init Prozesses oder Name des Terminals
- Ut_user: Name des Benutzers
- Ut_host: Rechnername
- Ut_session: Session ID
- Ut_addr_v6: IP Adresse des remote Host

6.6.3 Network File System (NFS)

Mit dem Network File System können verschiedene Rechner Dateien gemeinsam nutzen. Auf dem Serversystem wird ein NFS- Server installiert, der die Verzeichnisse (Network Volumes) lokal zur Verfügung stellt (exportiert). Der Benutzer kann mit Hilfe des NFS- Clients, der auf seinem System läuft, Daten des entfernten Rechners mitbenutzen (importieren). Das Network File System ist auf der Basis von Remote Procedure Calls (RPCs) implementiert.

6.6.3.1 Vorgang: Root exportiert Dateisystem

Root kann ein Dateisystem folgendermaßen exportieren.

1. Manuelles Exportieren:
   - Der Benutzer editiert die Konfigurationsdatei `/etc/exports`.
   - Der Benutzer ruft das Kommando `exportfs <options>` auf.
2. Exportieren von Dateisystemen mit der graphischen Oberfläche YAST2 → Netzwerk/ Erweitert → NFS- Server → Verzeichnisse

Existenz:
Online Auswertung und Offline Auswertung:
Methode 1: (1), (2), (3), (4), (5)
Methode 2: (1), (2), (3), (4)

(1) Eintrag in Konfigurationsdatei /etc/exports (Seite 206)

Beschreibung der Quelle:
Die Konfigurationsdatei ist eine Textdatei und wird durch Methode 1. und 2. um Einträge ergänzt.

Ergiebigkeit:
Mittel. Der Eintrag in der Konfigurationsdatei enthält folgende Informationen:

- Verzeichnis: zu exportierendes Verzeichnis
- Client: Liste von Clients, die zum Importieren dieses Verzeichnisses befugt sind.

Optionen für die Clients:

- Secure, insecure: Client-Anfragen werden nur von vertrauenswürdigen Ports akzeptiert, mit "insecure" werden auf Anfragen an höhere Ports akzeptiert.
- Ro, Rw: Das Verzeichnis wird schreibgeschützt ("read only") bzw. mit vollen Lese- und Schreibrechten für den Client ("read/write") exportiert.
- Sync, async: Der Server darf den Vollzug eines Schreibvorgang dem Client erst melden, wenn die Daten tatsächlich auf die Platte geschrieben wurden (Ausschalten des Plattencaches).
- wdelay, no_wdelay: Die Option wird nur in Zusammenhang mit "sync" beachtet und erlaubt dem Server die Bestätigung eines Schreibvorgangs zu verzögern, falls mehrere Schreibvorgänge von einem Client zur gleichen Zeit im Gange sind. Anstatt jeden zu bestätigen, sendet der Server nur eine einzige Antwort nach Vollzug aller Schreiboperationen (betrifft "wdelay").

Der Eintrag hat folgendes Format:

`Verzeichnis Client Optionen`

Verweis auf andere Vorgänge:

1. *Benutzer ändert Datei* (vgl. Seite 88)
2. kein Verweis
3. kein Verweis

---

(2) Eintrag in Datei /var/lib/nfs/xtab (Seite 215)

Beschreibung der Quelle:
Die Datei ist eine Textdatei und wird durch das Kommando `exportfs -a` (alle Volumes aus der Datei `/etc/exports`) oder `exports -r` (reexport) durch Änderung der `/etc/exports` um einen Eintrag ergänzt.

Ergiebigkeit:
Mittel. Die Datei enthält eine Liste aller Dateisysteme, die in der Exportdatei `/etc/exports` angegeben sind, d.h. aktuell exportiert werden.

---

(3) Eintrag in virtueller Datei /proc/fs/nfs/exports (Seite 225)

Beschreibung der Quelle:
Die virtuelle Datei ist eine Textdatei und wird durch Methode 1. und 2. um einen Eintrag ergänzt

Ergiebigkeit:
Mittel. Die Datei enthält folgende Informationen über das zu exportierende Dateisystem:

- Verzeichnis: zu exportierende Verzeichnis
- Client: Liste von Clients, die zum Importieren des Verzeichnisses befugt sind.
- IP- Adresse: Adresse des Clients

---

(4) Eintrag in Datei /var/lib/nfs/etab (Seite 215)

Beschreibung der Quelle:
Die Datei ist eine Textdatei und wird durch Methode 1. und 2. um einen Eintrag ergänzt.

Ergiebigkeit:
Mittel. Die Datei enthält Informationen über das zu exportierende Dateisystem, Dateisystemast und detaillierte Informationen über die Optionen, die benutzt werden, wenn ein Dateisystem zu einem einzelnen Client exportiert wird.

---

(5) Eintrag in History Liste des Benutzers (Seite 221)

Beschreibung der Quelle:
vgl. Seite 26

Ergiebigkeit:
Mittel. Die History Liste enthält folgende Kommandos:

1. - `vi /etc/exports`
   - `exportfs <options>`
2. kein Verweis

6.6.3.2 Vorgang: Benutzer importiert Dateisystem

Der Benutzer ist/ war entweder am zu untersuchenden System eingeloggt und importiert von diesem Rechner aus ein Dateisystem oder ist/ war nicht im zu untersuchenden System eingeloggt und will vom untersuchenden System Dateien importieren. Das Importieren des Dateisystem kann in beiden Fällen folgendermaßen geschehen:

1. Durch Modifizieren der Konfigurationsdatei /etc/fstab
   - Manuelles Importieren von Dateisystemen durch Editieren der `/etc/fstab` und dann `mount -a -t nfs` aufrufen oder `mount -t nfs <nfs-server>:NFS-Volume /mnt- point`
   - Mittels der graphischen Oberfläche YAST2 → NFS- Client → Hinzufügen
2. Mittels dem autofs Dienst

6.6.3.2.1 NFS- Client auf zu untersuchenden System

Der Benutzer ist/ war am zu untersuchenden System eingeloggt und importiert von diesem Rechner aus ein Dateisystem. **Existenz:**
Online Auswertung:
Methode 1: (1), (2), (3), (6), (7), (8)
Methode 2: (1), (2), (3), (4), (5), (6), (7)
Offline Auswertung:
Methode 1: (2), (3), (7), (8)
Methode 2: (2), (3), (4), (6), (7)

---

(1) Virtuelle Datei /proc/filesystems (Seite 224)

Beschreibung der Quelle:
Die Datei ist eine Textdatei und wird beim Importieren des Dateisystems gelesen, ob `nfs` unterstützt wird. Dabei wird das Dateiattribut "letzter Zugriff" der virtuellen auf die aktuelle Systemzeit und Datum gesetzt.

Ergiebigkeit:

Mittel. Die Datei enthält alle Dateisysteme, die der Kernel unterstützt. Das Dateiattribut “letzter Zugriff” gibt den Zeitpunkt an, an dem der Vorgang: *Benutzer importiert Dateisystem*, gestartet wurde.

---

(2) Eintrag in Konfigurationsdatei /etc/fstab (Seite 206)

Beschreibung der Quelle:
Die Konfigurationsdatei ist eine Textdatei und wird durch Methode 1. um einen Eintrag bzgl. nfs ergänzt.

Ergiebigkeit:
Mittel. Die Konfigurationsdatei enthält folgende Informationen über das importierte Dateisystem:

- Server: Hostnamen, die IP-Adresse oder den kompletten Domain-Name des Servers, der das Dateisystem exportiert.
- /Pfad/zum/gemeinsam genutzten/Verzeichnis: was gemountet werden soll
- /lokaler/Mount-Punkt: legt fest, wo das exportierte Verzeichnis im lokalen Dateisystem gemountet werden soll.
- nfs: Typ des gemounteten Dateisystems
- Optionen:
  - Mount-Optionen für das Dateisystem)

Der Eintrag hat folgendes Format:

```
<server>:</path/of/dir> </local/mnt/point> nfs <options> 0 0
```

---

(3) Eintrag in Konfigurationsdatei /etc/mtab (Seite 206)

Beschreibung der Quelle:
Die Konfigurationsdatei ist eine Textdatei und wird durch Methode 1. und 2. um einen Eintrag ergänzt.

Ergiebigkeit:
Mittel. Die Konfigurationsdatei enthält für das importierte Dateisystem folgenden Informationen:

- Device: entferntes Dateisystem
- Mountpoint: Verzeichniseintrag, in dem das Dateisystem erscheinen soll
- Type: Typ des Dateisystems → nfs
- Options: Optionen für den Mountvorgang
- Dump: Gibt an, ob das Dateisystem vom Kommando dump zu sichern ist. Dieser Eintrag wird derzeit nur beim Dateisystemtyp ext2 bewertet.
- Check: Gibt an, ob das Dateisystem vor dem Mounten zu überprüfen ist. Beim Root-Dateisystem sollte hier eine “1” stehen und bei allen anderen entweder eine “0” (keine Prüfung) oder eine “2”. Dateisysteme mit gleicher Nummer werden parallel überprüft, das Root- Dateisystem sollte immer allein und als erstes getestet werden.

Der Eintrag hat folgendes Format:

```
<Device> <Mountpunkt> nfs <Options> 0 0
```

(4) Eintrag in Konfigurationsdatei /etc/auto.master (Seite 206)

Beschreibung der Quelle:
Die Konfigurationsdatei ist eine Textdatei und durch Methode 2. wird ein Eintrag hinzugefügt.

Ergiebigkeit:
Mittel. Die Konfigurationsdatei enthält für das zu importierende Dateisystem folgende Informationen:

- Mount point: wo das Gerät oder das exportierte Dateisystem in Ihrem lokalen Dateisystem gemountet werden soll
- Zuordnungstyp (Map- Type): eine Datei, die als Zuordnungstyp für einen bestimmten Mount-Punkt verwendet wird. Die Zuordnungsdatei hat die Bezeichnung `auto.<mount point>`.

Der Eintrag hat folgendes Format:

```
<mount-point> <map-type>
```

(5) Zuordnungsdatei /etc/auto.<mountpoint>(Seite 206)

Beschreibung der Quelle:
Die Zuordnungsdatei ist eine Textdatei und wird durch Methode 2. neu erstellt. Der Name der Datei wird durch den `<mount point>`, der in der Datei `/etc/auto.master` angegeben ist spezifiziert.

Ergiebigkeit:
Mittel. Die Zuordnungsdatei enthält folgende Informationen über das zu importierende Dateisystem:

- Verzeichnis: Verzeichnis im Mount-Punkt, wo das exportierte Dateisystem gemountet werden soll.
- Host: <exportiertes-Dateisystem >: Standardbefehl mount, der Host, der das Dateisystem exportiert das exportierte Dateisystem eingegeben.
- Mount- Optionen: bestimmte Optionen, die beim Mounten des exportierten Dateisystems verwendet werden.

Die Zuordnungsdatei hat folgendes Format:

```
<directory> <mount-options> <host:exported-file-system>
```

(6) TCP- Verbindung (Seite 231)

Beschreibung der Quelle:
Eine TCP- Verbindung wird vom *nfs Client* erzeugt.

Ergiebigkeit:
Gut. Die TCP- Verbindung birgt folgende Informationen:

- Local Adress: IP- Adresse des Client
- Local Port: >1024
- Foreign Adress: IP- Adresse des Servers
- Foreign Port: <1024
- State: aktive Verbindung

(7) Eintrag in Log- Datei /var/log/messages (Seite 210)

Beschreibung der Quelle:
vgl. Seite 26

Ergiebigkeit:
Gut. 1. und 2. Der Eintrag enthält folgende Informationen:

- Time: Wann sich der Benutzer die mount Anfrage gestellt hat
- Local Hostname: Rechnername des Clients
- remote host: Name des NFS Servers
- rights: Zugriffsrechte
- requested directory: angefragtes Verzeichnis

Der Eintrag hat folgendes Format:

```
<time> <local hostname> rpc.mountd: authenticated mount request
from <remote host>:<rights> <requested directory> (<<requested
directory>)
```

(8) Eintrag in History Liste des Benutzers (Seite 221)

Beschreibung der Quelle:
vgl. Seite 26

Ergiebigkeit:
Mittel. Die History Liste enthält folgende Kommandos:

1. `mount -a -t nfs` oder `mount -t nfs nfs-server:NFS-Volume /mnt-point`
2. kein Eintrag

6.6.3.2.2 NFS- Server auf zu untersuchenden System

Der Benutzer ist/war auf einem anderen Rechner eingeloggt und importiert ein Dateisystem vom NFS- Server.

Existenz:
Online Auswertung: (1), (2), (3), (4), (5), (6), (7)
Offline Auswertung: (1), (2), (7)

---

(1) Zugriffskontrolldateien /etc/hosts.{allow, deny} (Seite 205)

Beschreibung der Quelle:
Die beiden Dateien `/etc/hosts.allow` und `/etc/hosts.deny` sind Textdateien, die beim Aufruf von at gelesen werden und so die Berechtigung zur Benutzung von *nfs* regeln. Dabei wird das Dateiattribut "letzter Zugriff" der Datei auf die aktuelle Systemzeit und Datum gesetzt.

Ergiebigkeit:
Mittel. Die Datei `/etc/hosts.allow` enthält Benutzer, die das Kommando verwenden dürfen. Die Datei `/etc/hosts.deny` enthält Benutzer, die das Kommando nicht verwenden dürfen.

---

(2) Eintrag in Datei /var/lib/nfs/rmtab (Seite 215)

Beschreibung der Quelle:
Die Datei ist eine Textdatei und wird durch eine Mountanfrage, die von einem NFS-Client erhalten wurde, um einen Eintrag erweitert.

Ergiebigkeit:
Mittel. Der Eintrag enthält den Namen des Clienten und das Dateisystem, das von diesem Server importiert wird. Das Dateiattribut "letzter Zugriff" gibt den Zeitpunkt an, an dem der Vorgang: *Benutzer importiert Dateisystem*, gestartet wurde.

---

(3) Datei /var/lib/nfs/etab (Seite 215)

Beschreibung der Quelle:
Die Datei ist eine Textdatei und wird durch eine Mountanfrage gelesen, die von einem NFS- Client erhalten wurde. Dabei wird das Dateiattribut "letzter Zugriff" der Datei auf die aktuelle Systemzeit und Datum gesetzt.

Ergiebigkeit:
Mittel. Die Datei enthält den Pfad des momentan exportierten Dateisystems, den Client und wie das Dateisystem beim Client eingehängt werden soll. Die Datei hat folgendes Format:

```
exported-file-system client mount-options
```

(4) Eintrag in Datei /var/lib/nfs/xtab (Seite 215)

Beschreibung der Quelle:
Die Datei ist eine Textdatei und durch eine Mountanfrage wird ein Eintrag hinzugefügt.

Ergiebigkeit:
Gut. Die Datei enthält den Pfad des exportierten Dateisystems, den Client, wie das Dateisystem beim Client eingehängt werden soll und die IP- Adresse des Clients. Die Datei hat folgendes Format:

```
exported-file-system client mount-options client-ip
```

Das Dateiattribut "letzter Zugriff" gibt den Zeitpunkt an, an dem der Vorgang: *Benutzer importiert Dateisystem*, gestartet wurde.

(5) Eintrag in virtueller Datei /proc/fs/nfs/exports (Seite 225)

Beschreibung der Quelle:
Die virtuelle Datei ist eine Textdatei und durch eine Mountanfrage wird ein Eintrag hinzugefügt.

Ergiebigkeit:
Mittel. Die Datei enthält den Pfad des exportierten Dateisystems, den Client, wie das Dateisystem beim Client eingehängt werden soll und die IP- Adresse des Clients. Die Datei hat folgendes Format:

```
<exported-file-system> <client> <mount-options> <client-ip>
```

(6) TCP- Verbindung (Seite 231)

Beschreibung der Quelle:
Eine TCP- Verbindung wird vom *nfs Client* erzeugt.

Ergiebigkeit:
Gut. Die TCP- Verbindung birgt folgende Informationen:

- Local Adress: IP- Adresse des NFS Servers
- Local Port: <1024
- Foreign Adress: IP- Adresse des Clients
- Foreign Port: >1024
- State: aktive Verbindung

(7) Eintrag in Log- Datei /var/log/messages (Seite 210)

Beschreibung der Quelle:
vgl. Seite 26

Ergiebigkeit:
Gut. Der Eintrag enthält folgende Informationen:

- Time: Wann sich der Benutzer die mount Anfrage gestellt hat
- Local Hostname: Rechnername des Servers
- pid: Prozess ID von mountd
- ip- address: IP Adresse des Clients, der das Verzeichnis anfordert

Der Eintrag hat folgendes Format:

```
<time> <local hostname> mountd[pid]: export request from <ip
address>
```

6.6.4 Domain Name Service (DNS)

Der Domain Name Service löst Domain- und Rechnernamen in IP- Adressen auf. Auf dem Serversystem läuft das Programm BIND9.

6.6.4.1 Vorgang: Benutzer macht DNS- Anfrage

Der Benutzer ist/ war entweder am zu untersuchenden System eingeloggt und stellt von diesem Rechner aus eine DNS- Anfrage oder ist/ war nicht am zu untersuchenden System eingeloggt und stellt eine DNS- Anfrage an das zu untersuchende System.

6.6.4.1.1 DNS- Client auf zu untersuchenden System

Der Benutzer ist auf dem zu untersuchenden System eingeloggt und kann eine DNS- Anfrage folgendermaßen stellen.

1. Direkt mit dem Kommando `dig <servername> <name> <type>` um einen DNS- Look up zu machen. Servername ist der Nameserver, falls

   es ein Hostname ist, wird dieser Name aufgelöst und Type ist der Typ von Resource Record.

   (a) Zuordnung: Rechnername → IP- Adresse
   (b) Zuordnung: IP- Adresse → Rechnername

2. Indirekt durch starten eines Webdienstes

   (a) Zuordnung: Rechnername → IP- Adresse
   (b) Zuordnung: IP- Adresse → Rechnername

Existenz:
Online Auswertung:
Methode 1.:(1), (2), (3), (4)
Methode 2.: (1), (2), (3)
Offline Auswertung:
Methode 1.: (1), (2), (4)
Methode 2.: (1), (2)

(1) Konfigurationsdatei /etc/resolv.conf (Seite 205)

Beschreibung der Quelle:
Die Konfigurationsdatei ist eine Textdatei und wird durch Methode 1. und 2. gelesen und ausgewertet. Dabei wird das Dateiattribut "letzter Zugriff" der Konfigurationsdatei auf die aktuelle Systemzeit und Datum gesetzt.

Ergiebigkeit:
vgl. Seite 139 Informationsquelle (10)

(2) Konfigurationsdatei /etc/nsswitch.conf (Seite 205)

Beschreibung der Quelle:
Die Konfigurationsdatei ist eine Textdatei und wird durch Methode 1. und 2. gelesen und ausgewertet. Dabei wird das Dateiattribut "letzter Zugriff" der Konfigurationsdatei auf die aktuelle Systemzeit und Datum gesetzt.

Ergiebigkeit:
vgl. Seite 138 Informationsquelle (9)

(3) UDP- Verbindung (Seite 231)

Beschreibung der Quelle:
Eine UDP- Verbindung wird von *dig* erzeugt.

Ergiebigkeit:
Gut. Die UDP - Verbindung birgt folgende Informationen:

- Local Adress: IP- Adresse des Client
- Local Port: >1024
- Foreign Adress: IP- Adresse des Servers
- Foreign Port: 53
- State: aktive Verbindung

(4) Eintrag in der History Liste (Seite 221)

Beschreibung der Quelle:
vgl. Seite 26

Ergiebigkeit:
Mittel. Die History Liste enthält folgende Kommandos:

1. `dig <servername> <name> <type>`
2. kein Eintrag

6.6.4.1.2 DNS- Server auf zu untersuchenden System

Der Benutzer ist auf einem anderen Rechner eingeloggt und will eine DNS- Anfrage an das zu untersuchenden System stellen. Auf dem zu untersuchenden System läuft das Programm BIND9 (named).

Existenz:
Online Auswertung:
Methode 1.: (1), (2), (4)
Methode 2.: (1), (3), (4)
Offline Auswertung: Methode 1.: (1), (3)
Methode 2: (1), (2)

(1) Konfigurationsdatei /etc/named.conf (Seite 205)

Beschreibung der Quelle:
Die Konfigurationsdatei wird von *BIND9* gelesen und ausgewertet. Dabei wird das Dateiattribut "letzter Zugriff" der Konfigurationsdatei auf die aktuelle Systemzeit und Datum gesetzt.

Ergiebigkeit:
Mittel. Die Konfigurationsdatei lässt sich grob in zwei Bereiche unterteilen:
Options:

- allow-query: Welche Hosts diesen Nameserver abfragen dürfen. Standardmäßig sind alle Hosts dazu berechtigt. Mit Hilfe einer Access- Controll-List, einer Sammlung von IP-Adressen oder Netzwerken kann festgelegt werden, dass nur bestimmte Hosts den Nameserver abfragen dürfen.
- allow-recursion: rekursive Abfragen der Hosts
- blackhole: Welchen Hosts es nicht erlaubt ist Anfragen an den Server zu stellen.
- directory: named- Arbeitsverzeichnis, so dass es sich von dem Default, `/var/named`, unterscheidet.
- forward: Kontrolliert das Verhalten beim Weiterleiten einer forwarders Direktive. Nimmt die folgenden Optionen an:
  - first: Nameserver, die in der forwarders- Option festgelegt sind, zuerst nach Informationen abgefragt werden, sollten anschließend keine Informationen vorhanden sein, versucht *named* die Auflösung selbst durchzuführen.
  - Only: named versucht nichtdie Auflösung selbst durchzuführen, wenn die forwarders Direktive nicht erfolgreich war.
  - forwarders: Liste von Nameservern, bei denen Abfragen für Auflösungen weitergeleitet werden.
  - listen-on: Netzwerk-Schnittstelle, die named verwendet, um Anfragen zu prüfen.
- Notify: Kontrolliert, ob named die Slave-Server informiert, wenn eine Zone aktualisiert wird. Nimmt die folgenden Optionen an:
  - yes: Informiert Slave-Server.
  - no: Informiert Slave-Server nicht.
- Explicit: Informiert Slave-Server nur dann, wenn diese in einer also-notify List innerhalb des Zonen Statement angegeben sind.
- pid-file: Ort für die Prozess- ID- Datei, die named erstellt (`/var/named/pid`).
- statistics-file: Ort in welcher die Statistik- Dateien abgelegt werden (`/var/named`)

Das Option Statement hat folgendes Format:

```
options  <option>;
[<option>; ...];
```

Zonen- Einträge:

- allow-query: Welche Clients Informationen über diese Zone anfordern dürfen.
- allow-transfer: Slave-Server, die den Transfer der Informationen über die Zonen anfordern dürfen. Standardmäßig sind alle Transfer-Anfragen zulässig.
- allow-update: Hosts, die Informationen in ihrer Zone dynamisch aktualisieren dürfen.
- File: Namen der Datei im named-Arbeitsverzeichnis, die die Zone-Konfigurationsdateien enthält.
- masters: IP Adressen von denen authoritative zone informationen abgefragt werden können.
- Notify: Wird verwendet, wenn die Zone als Slave type festgelegt ist. Die masters- Option teilt dem named eines Slaves die IP-Adressen mit, von denen maßgebliche Informationen über die Zone angefragt werden.
  - yes: Informiert Slave Server.
  - no: Informiert Slave Server nicht.
- Explicit: Informiert Slave-Server nur dann, wenn diese in einer also-notify List innerhalb des Zonen Statement angegeben sind.
- type: Typ der ZoneNimmt folgende Optionen an:
  - forward: Weist den Nameserver an, alle Anfragen zu Informationen über die Zone an andere Nameserver weiterzuleiten.
  - hint: Root- Nameserver, die verwendet werden, um Abfragen zu lösen, wenn eine Zone ansonsten unbekannt ist.
  - master: Nameserver, der für diese Zone maßgeblich ist. Wenn die Konfigurationsdateien für diese Zone auf Ihrem System sind, sollte der master-Typ eingestellt werden.
  - slave: Nameserver, der für diese Zone der Slave-Server ist und der named mitteilt, die Zonen-Konfigurationsdateien für diese Zone von der IP-Adresse des Master-Nameservers abzufragen.
  - zone-statistics: Ort an der die Statistiken über diese Zone aufbewahrt werden.

Die Zonen- Einträge haben folgendes Format:

```
zone <zone-name> <zone-class>
<zone-options>;
[<zone-options>; ...];
```

Das Dateiattribut “letzter Zugriff” gibt den Zeitpunkt an, an dem der Vorgang: *Benutzer macht DNS- Anfrage*, gestartet wurde.

(2) Forward Zonendatei /var/named/<domain>.zone (Seite 212)

Beschreibung der Quelle:
Die Forward Zonendatei wird bei einer Anfrage des Clients durch Methode 1 (a) und 2 (a) von BIND9 gelesen und ausgewertet. Dabei wird das Dateiattribut “letzter Zugriff“ der Zonendatei auf die aktuelle Systemzeit und Datum gesetzt.

Ergiebigkeit:
Mittel. Die Zonendatei enthält folgende Informationen:

Zone-Dateien-Direktive:

- $INCLUDE: Weist named an, in diese Zone-Datei an Stelle der Anweisung eine andere Zone-Datei einzufügen. Dadurch können zusätzliche Einstellungen der Zone getrennt von der Haupt-Zone-Datei gespeichert werden.
- $ORIGIN: Stellt den Domain-Name so ein, dass er an alle ungeeigneten Records angefügt wird.
- $TTL: Legt den Standard-Time to Live (TTL)-Wert für die Zone fest. Dieser Wert legt für die Name-Server in Sekunden fest, wie lange das Resource-Record für die Zone gültig ist.

Resource Records:

- A: Adressen-Record, das einem Namen eine IP-Adresse zuweist. Format des Adressen Record:

  ```
  <host> IN A <IP-address>
  ```

- CNAME: Name-Record, welcher Namen untereinander zuordnet. Format des Name- Records:

  ```
  <alias-name> IN CNAME <real-name>
  ```

- MX: Mail eXchange- Record, welchen Weg eine Mail nimmt, die an ein bestimmtes Namespace gesendet und von dieser Zone kontrolliert wurde. Format des Mail Exchange Records:

  ```
  IN MX <preference-value> <email-server-name>
  ```

- NS: Name-Server-Record, der die maßgeblichen Name-Server für eine bestimmte Zone anzeigt. Format des Name-Server Record:

  ```
  IN NS <nameserver-name>
  ```

- PTR: PoinTeR-Record verweist auf einen anderen Teil des Namespace.
- SOA: Start Of Authority-Record, gibt wichtige maßgebliche Informationen über den Namespace an den Name-Server. Format des Start of Authority-Record:

  ```
  @ IN SOA <primary-name-server> <hostmaster-email>
  (<serial-number> <time-to-refresh> <time-to-retry>
  <time-to-expire> <minimum-TTL> )
  ```

- primary-Nameserver: erster für diese Domain maßgebliche verwendete Name- Server
- hostmaster-email: E-Mail der über diesen Namespace zu kontaktierenden Person
- serial-number wird bei jeder Änderung der Zone-Datei erhöht, so dass named erkennt, dass diese Zone neu geladen werden kann.
- time-to-refresh: teilt den Slave-Servern mit, wie lange sie warten müssen, bevor sie beim Master-Nameserver anfragen, ob alle Änderungen für die Zone durchgeführt wurden.
- time-to-retry: Zeitraum, nach dem eine neue Anfrage bezüglich der Aktualisierung durchgeführt werden soll, wenn der Master-Nameserver auf die letzte Anfrage nicht reagiert hat.
- time-to-expire: Zeitraum, nachdem ein sekundärer Nameserver die gecachten Daten verwirft, wenn er keinen Kontakt zum primärer Server mehr bekommen hat.
- minimum-TTL: ist die Zeit, die anderen Nameservern zum Verarbeiten der Zonen-Informationen mindestens zur Verfügung steht (in Sekunden).

Das Dateiattribut "letzter Zugriff" gibt den Zeitpunkt an, an dem der Vorgang: *Benutzer macht DNS- Anfrage*, gestartet wurde.

---

(3) Reverse Zonendatei /var/named/<reverse domain>.zone(Seite 212)

Beschreibung der Quelle:
Die Reverse Zonendatei wird bei einer Anfrage des Clients durch Methode 1 (b) und 2 (b) von BIND9 gelesen und ausgewertet. Dabei wird das Dateiattribut "letzter Zugriff" der Zonendatei auf die aktuelle Systemzeit und Datum gesetzt. Der Name von der Reverse Zonendatei hängt von dem Nameder er Forward Zonendatei ab, wobei die IP- Adresskomponenten in umgekehrter Reihenfolge vorkommen.

Ergiebigkeit:
Gut. Die Zonendatei enthält folgende Informationen:

Zone- Dateien- Direktive:

- $INCLUDE: Weist named an, in diese Zone-Datei an Stelle der Anweisung eine andere Zone-Datei einzufügen. Dadurch können zusätzliche Einstellungen der Zone getrennt von der Haupt-Zone-Datei gespeichert werden.
- $ORIGIN: Stellt den Domain-Name so ein, dass er an alle ungeeigneten Records angefügt wird.
- $TTL: Legt den Standard-Time to Live (TTL)-Wert für die Zone fest. Dieser Wert legt für die Name-Server in Sekunden fest, wie lange das Resource-Record für die Zone gültig ist.

Resource- Record:

- SOA: Start Of Authority-Record, gibt wichtige maßgebliche Informationen über den Namespace an den Name-Server. Format des Start of Authority-Record:

```
@ IN SOA <primary-name-server> <hostmaster-email> (
<serial-number> <time-to-refresh> <time-to-retry>
<time-to-expire> <minimum-TTL> )
```

  - primary-Nameserver: erster für diese Domain maßgebliche verwendete Name- Server
  - hostmaster-email: E-Mail der über diesen Namespace zu kontaktierenden Person
  - serial-number wird bei jeder Änderung der Zone-Datei erhöht, so dass named erkennt, dass diese Zone neu geladen werden kann.
  - time-to-refresh: teilt den Slave-Servern mit, wie lange sie warten müssen, bevor sie beim Master-Nameserver anfragen, ob alle Änderungen für die Zone durchgeführt wurden.
  - time-to-retry: Zeitraum, nach dem eine neue Anfrage bezüglich der Aktualisierung durchgeführt werden soll, wenn der Master-Nameserver auf die letzte Anfrage nicht reagiert hat.
  - time-to-expire: Zeitraum, nachdem ein sekundärer Nameserver die gecachten Daten verwirft, wenn er keinen Kontakt zum primärer Server mehr
    bekommen hat.
  - minimum-TTL: ist die Zeit, die anderen Nameservern zum Verarbeiten der Zonen-Informationen mindestens zur Verfügung steht (in Sekunden)

- PTR:
  - last-IP-digit: letzte Ziffer in einer IP-Adresse, mit der auf einen bestimmten FQDN im System hingewiesen wird. Format des PTR Resource Record:

    ```
    <last-IP-digit> IN PTR <FQDN-of-system>
    ```

Das Dateiattribut “letzter Zugriff” gibt den Zeitpunkt an, an dem der Vorgang: *Benutzer macht DNS- Anfrage*, gestartet wurde.

---

(4) UDP- Verbindung (Seite 231)

Beschreibung der Quelle:
Eine TCP- Verbindung wird bei einer Anfrage eines Clients erzeugt.

Ergiebigkeit:
Gut. Die UDP- Verbindung birgt folgende Informationen:

- Local Adress: IP- Adresse des DNS Servers
- Local Port: 53
- Foreign Adress: IP- Adresse des Clients, der die Anfrage stellt
- Foreign Port: >1024
- State: aktive Verbindung

6.6.5 Network Information Service (NIS)

Das Network Information System ist ein Verzeichnisdienst, der die Benutzung bestimmter Konfigurationsdateien von verschiedenen Rechnern aus ermöglicht. Insbesondere in Verbindung mit dem Network File System wird NIS verwendet, um Benutzern netzwerkweit dieselbe Umgebung zur Verfügung zu stellen. Auf dem Client Rechner läuft ein NIS- Client Programm *ypbind*.

6.6.5.1 Vorgang: Root erzeugt Map

Root kann Maps folgendermaßen erzeugen.

1. Manuelles Erzeugen der Maps:
   - Das Kommando zum Erzeugen einer Map aus einer Master- Dtei ist makedbm. (`/usr/lib/yp/makedbm [options] [Master file mapname]`).
2. Automatisches Erzeugen der Maps:
   - `vi /var/yp/Makefile` und nach dem Eintrag `all:` suchen. Die NIS-Domäne wird noch gesetzt und `make -C /var/yp` wird aufgerufen, um die Maps zu erzeugen.
   - Mit `ypinit -m` werden die Maps zusätzlich für weitere NIS- Server der gleichen Domain NIS- Slave- Server erzeugt.

Existenz:
Online und Offline Auswertung:
Methode 1. und 2.: (1), (2), (3), (4), (5), (6), (7)

(1) Eintrag in Datei /var/yp/nicknames (Seite 218)

Beschreibung der Quelle:
Die Datei ist eine Textdatei und wird vom Benutzer um einen Mapeintrag ergänzt.

Ergiebigkeit:
Mittel. Die Datei enthält den Alias Namen der neuen Map und zugehörige Map. Die Datei hat folgendendes Format:

```
Alias Map Name.[option]
```

Als Option wird angeben wie die anderen Benutzer an die Mapinformation kommt: per Nummer (`bynumber`), per Name (`byname`), per Alias (`aliases`), per Adresse (`byaddr`).

(2) Map /var/yp/<nisdomainname>/<mapname>(Seite 218)

Beschreibung der Quelle:
Die Daten der Maps sind im DBM-Format (Database Management) organisiert und in dem Verzeichnis `/var/yp/<nisdomainname>` gespeichert, so dass der Zugriff mittels Hashing- Funktionen enorm beschleunigt erfolgt. Die Map wird durch den Benutzer neu erstellt.

Ergiebigkeit:
Mittel. Die Standardmaps sind folgende:

- */etc/ethers: ethers.byname und ethers.byaddr* Die Map enthält für jeden Eintrag die 48 Bit Ethernet Adresse (x:x:x:x:x:x) und ihre zugehörige IP- Adresse (Hostname).
- */etc/hosts: hosts.byname und hosts.byaddr* Die Map enthält für jeden Eintrag die IP- Adresse, den kanonischen Hostname und den Short Hostname (Aliase).
- */etc/networks: networks.byaddr und networks.byname* Die Map enthält für jeden Eintrag den Netznamen und die dazugehörige IP- Adresse.
- */etc/protocols: protocols.bynumber und protocols.byname* Die Map enthält für jeden Eintrag den Protokollnamen und die Protokollnummer.
- *etc/rpc: rpc.byname und rpc.bynumber* Die Map enthält für jeden Eintrag den Namen des RPC Programms und die Programmnummer.
- */etc/services: services.byname* Die Map enthält für jeden Eintrag den Namen des Dienstes (Client Programm), Portnummer, der für den Dienst verwendet wird, das verwendete Protokoll (tcp, udp) und andere Namen für den Dienst (Aliases).
- */etc/passwd: passwd.byname und passwd.byuid* Die Map enthält für jeden Eintrag den Benutzernamen, Passwort (falls keine `shadow.byname` existiert) die UserID und GroupID, Info, Name des Home Directories und die verwendete Shell.
- */etc/group: group.byname und group.bygid* Die Map enthält für jeden Eintrag den Gruppennamen, Passwort (falls keine `gshadow.byname` existiert), die GroupID und Liste der Mitglieder.
- */etc/netid: netid.byname* Die Map enthält für jeden Eintrag den Benutzernamen, den Gruppennamen und den Hostname.
- */etc/netgroup: netgroup.byhost und netgroup.byuser* Die Map enthält für jeden Eintrag den Netgroup Namen, Liste von Mitglieder, die entweder eine weitere Netgroup ist oder aus dem Tupel (Hostname, Benutzername, Domain) besteht.
- */etc/shadow: shadow.byname* Die Map enthält für jeden Eintrag den Benutzernamen, das verschlüsselte Passwort, Day of last change, Tag ab dem 1.1.1970, an dem das Passwort zuletzt geändert wurde (DOC), minimale Anzahl Tage (MIND) und maximale Anzahl Tage, die das Passwort gültig ist, Anzahl der Tage vor Ablauf der Lebensdauer des Passwortes, ab der vor dem Verfall zu warnen ist (WARN), wieviele Sekunden das Passwort trotz Ablauf der MaxD gilt (EXP), und den Tagn bis zu diesem (gezählt ab 1.1.1970) der Account gesperrt ist.
- */etc/gshadow: gshadow.byname* Die Map enthält für jeden Eintrag den Gruppennamen, das verschlüsselte Passwort, den Gruppenverwalter und die Mitglieder.

(3) Datei /var/yp/ypservers (Seite 217)

Beschreibung der Quelle:
Die Datei ist eine Textdatei und wird beim Aufruf von `make -C /var/yp` gelesen. Dabei wird das Dateiattribut "letzter Zugriff" der Datei auf die aktuelle Systemzeit und Datum gesetzt.

Ergiebigkeit:
Schlecht. Die Datei enthält den Namen aller NIS- Server zu einer Domain (auch NIS- Slave- Server genannt). Das Dateiattribut "letzter Zugriff" gibt den Zeitpunkt an, an dem der Vorgang: *Root erzeugt Map*, gestartet wurde.

(4) Datei /var/yp/Makefile (Seite 217)

Beschreibung der Quelle:
Die Datei ist eine Textdatei und wird beim Erstellen der Map vom Benutzer aufgerufen. Dabei wird das Dateiattribut "letzter Zugriff" der Makefile auf die aktuelle Systemzeit und Datum gesetzt.

Ergiebigkeit:
Mittel. Die Makefile enthält folgende Informationen:

- NOPUSH: Die Maps werden nicht zu einem Slave Server übertragen.
- MINUID: Minimum UID, die in der Passwd Map eingeschlossen werden soll
- MINGID: Minimum GID, die in der Group Map eingeschlossen werden soll
- MERGE_PASSWD: Soll die Passwortdatei mit der Shadow Datei zusammen geführt werden
- MERGE_GROUP: Soll die Gruppendatei mit der gshadow Datei zusammen geführt werden

Source Directory für die NIS- Dateien:

- YPSRCDIR: Ort an dem sich die lokalen Dateien befinden
- YPPWDDIR: Verzeichnis für passwd, group und shadow
- YPBINDIR: standardmäßig `/usr/lib/yp`
- YPSBINDIR: standardmäßig `/usr/sbin`
- YPDIR: standardmäßig `/var/yp`
- YPMAPDIR: $(YPDIR)/$(DOMAIN)Lokale Dateien, von denen die NIS- Maps erstellt werden: Z.B.:
  - GROUP: $(YPPWDDIR)/group
  - PASSWD: $(YPPWDDIR)/passwd
  - ETHERS: $(YPSRCDIR)/ethers
  - NETGROUP: $(YPSRCDIR)/netgroup
  - HOSTS: $(YPSRCDIR)/hosts

  – NETWORKS: $(YPSRCDIR)/networks
  – PROTOCOLS: $(YPSRCDIR)/protocols
  – PUBLICKEYS: $(YPSRCDIR)/publickey
  – RPC: $(YPSRCDIR)/rpc
  – SERVICES: $(YPSRCDIR)/services
  – NETGROUP: $(YPSRCDIR)/netgroup
  – NETID: $(YPSRCDIR)/netid

Maps, die erstellt werden sollen:
`all: passwd ...`

Skripten, die für jede lokale Datei die NIS- Map erstellt.

Das Dateiattribut "letzter Zugriff" gibt den Zeitpunkt an, an dem der Vorgang: *Root erzeugt Map*, gestartet wurde.

---

(5) Eintrag in /var/yp/Makefile (Seite 218)

Beschreibung der Quelle:
Die Datei ist eine Textdatei und wird durch Methode 2. um einen Eintrag erweitert.

Ergiebigkeit:
Mittel. Der Eintrag all: in der Makefile enthält die Maps die erstellt werden sollen und zusätzlich die neue Map. Der Eintrag hat folgendes Format:

```
all: <mapname 1>...<mapname n>
```

---

(6) Lokale Datei (Seite 233)

Beschreibung der Quelle:
Die lokale Datei ist eine Textdatei und wird beim Erstellen der Map gelesen. Aus der lokalen Datei wird die Map erzeugt. Dabei wird das Dateiattribut "letzter Zugriff" der Konfigurationsdatei auf die aktuelle Systemzeit und Datum gesetzt.

Ergiebigkeit:
Mittel. Die lokalen Dateien, die standardmäßig verwendet werden:

- `/etc/passwd`: Die lokale Datei enthält in jeder Zeile den Benutzernamen, Passwort (falls keine shadow.byname existiert) die UserID und GroupID, Info, Name des Home Directories und die verwendete Shell.

- `/etc/shadow`: Die lokale Datei enthält für jeden Eintrag den Benutzernamen, das verschlüsselte Passwort, Day of last change, Tag ab dem 1.1.1970, an dem das Passwort zuletzt geändert wurde (DOC), minimale Anzahl Tage (MIND) und maximale Anzahl Tage, die das Passwort gültig ist, Anzahl der Tage vor Ablauf der Lebensdauer des Passwortes, ab der vor dem Verfall zu warnen ist (WARN), wieviele Sekunden das Passwort trotz Ablauf der MaxD gilt (EXP), und den Tagn bis zu diesem (gezählt ab 1.1.1970) der Account gesperrt ist.

- `/etc/group`: Die lokale Datei enthält in jeder Zeile den Gruppennamen, Passwort (falls keine `/etc/gshadow` existiert), die GroupID und Liste der Mitglieder.
- `/etc/gshadow`: Die lokale Datei enthält in jeder Zeile den Gruppennamen, das verschlüsselte Passwort, den Gruppenverwalter und die Mitglieder.
- `/etc/netgroup`: Die lokale Datei enthält in jeder Zeile den Netgroup Namen, Liste von Mitglieder, die entweder eine weitere Netgroup ist oder aus dem Tupel (Hostname, Benutzername, Domain) besteht.· /etc/networks:Die lokale Datei enthält in jeder Zeile den Netznamen und die dazugehörige IP- Adresse.
- `/etc/hosts`: Die lokale Datei enthält in jeder Zeile die IP- Adresse, den kanonischen Hostname und den Short Hostname (Aliase).
- `/etc/ethers`: Die lokale Datei enthält in jeder Zeile eine 48 Bit Ethernet Adresse (x:x:x:x:x:x) und ihre zugehörige IP- Adresse (Hostname).
- `/etc/netid`: Die lokale Datei enthält in jeder Zeile den Benutzernamen, den Gruppennamen und den Hostname.
- `/etc/services`: Die lokale Datei enthält in jeder Zeile den Namen des Dienstes (Client Programm), Portnummer, der für den Dienst verwendet wird, das verwendete Protokoll (tcp, udp) und andere Namen für den Dienst (Aliases).
- `/etc/protocols`: Die lokale Datei enthält in jeder Zeile den Protokollnamen und die Protokollnummer.
- `/etc/rpc`: Die lokale Datei enthält in jeder Zeile den Namen des RPC Programms und die Programmnummer.

Das Dateiattribut “letzter Zugriff” gibt den Zeitpunkt an, an dem der Vorgang: *Root erzeugt Map*, gestartet wurde.

---

(7) Eintrag in History Liste des Benutzers (Seite 221)

Beschreibung der Quelle:
vgl. Seite 26

Ergiebigkeit:
Gut. Die History Liste enthält folgende Kommandos:

1. `/usr/lib/yp/makedbm [options] [Master file map-name]`
2. - `vi /var/yp/Makefile`
   - `make -C /var/yp`
   - `ypinit -m`

6.6.5.2 Vorgang: Benutzer frägt Informationen über NIS ab

Der Benutzer ist/ war entweder am zu untersuchenden System eingeloggt und will sich von diesem Rechner aus eine Map Anfrage an einen NIS Server stellen oderist/ war nicht am zu untersuchenden System eingeloggt und will dem zu untersuchenden System eine Map Anfrage stellen.

6.6.5.2.1 NIS- Client auf zu untersuchenden System

Der Benutzer ist auf dem zu untersuchenden System eingeloggt und sucht einen für die NIS- Domain zuständigen Server via broadcast mit dem Kommando ypbind -broadcast oder mittels der Konfigurationsdatei /etc/yp.conf oder konfiguriert den NIS- Client mit dem YAST2 Kontrollzentrum → Netzwerk → NIS- Client. Der Benutzer kann mit dem Kommando ypset -d <domain> -h <hostname server> sich an einen bestimmten NIS- Server binden. Mit dem Kommando ypcat -d <domain> -h <hostname> <mapname> kann der Benutzer die Werte aller Schlüssel der NIS- Datenbank nach Mapnamen ausgeben. Um die Werte eines bestimmten Schlüssels der NIS- Datenbank auszugeben verwendet der Benutzer das Kommando ypmatch -d <domain> <key> ... <mapname>.

Existenz:
Online Auswertung: (1), (2), (3), (4), (5), (6), (7)
Offline Auswertung: (1), (2), (3), (4), (5), (6)

---

(1) Konfigurationsdatei /etc/nsswitch.conf (Seite 205)

Beschreibung der Quelle:
Die Konfigurationsdatei ist eine Textdatei und wird durch den Aufruf des Kommandos ypmatch gelesen und ausgwertet. Dabei wird das Dateiattribut “letzter Zugriff“ der Konfigurationsdatei auf die aktuelle Systemzeit und Datum gesetzt.

Ergiebigkeit:
Mittel. In der Konfigurationsdatei werden die Mechanismen für die meisten auch über Netzwerk verteilbare Servicedateien (z.B. passwd, group) festgelegt.

- Name der Datenbank: Namen der lokalen Konfigurationsdatei
- Mechanismen: Mechanismen, die in gegebener Reihenfolge, angewandt werden, um eine Information aus der betreffenden Datenbank zu gewinnen. Liefert ein Versuch kein Ergebnis, wird in der Voreinstellung das nächste Verfahren versucht.

Die folgenden Mechanismen sind möglich:

- Files: mittels lokaler Dateien (z.B. /etc/passwd)
- Dns: mittels DNS
- Nis oder yp: mittels NIS
- NOTFOUND=return: bewirkt, daß die Suche gestoppt wird, wenn der vorhergehenden Mechanismus zu keinem Ergebnis kommt.
- Compat: aktiviert die Auswertung einer erweiterten Syntax in den Datenbanken passwd, shadow und group

Die Konfigurationsdatei hat folgendes Format:

`Name der Datenbank: Mechanismen`

Das Dateiattribut “letzter Zugriff” gibt den Zeitpunkt an, an dem der Vorgang: *Benutzer frägt Informationen über NIS ab*, gestartet wurde.

(2) Konfigurationsdatei /etc/sysconfig/ypbind (Seite 205)

Beschreibung der Quelle:
Die Konfigurationsdatei wird von ypbind gelesen und ausgewertet. Dabei wird das Dateiattribut “letzter Zugriff“ der Konfigurationsdatei auf die aktuelle Systemzeit und Datum gesetzt.

Ergiebigkeit:
Schlecht. Die Konfigurationsdatei enthält folgende Parameter:

- YPBIND_BROADCAST [yes, **no**]: ob ypbind die Konfigurationsdatei `/etc/yp.conf` liest oder den NIS- Server im lokalen Subnetz per broadcast sucht
- YPBIND_BROKEN_SERVER [yes, **no**]: “yes” wenn im eigenen Netz ein NIS-Server ist, der nur an Ports >1024 bindet.
- YPBIND_LOCAL_ONLY [yes, **no**]: “yes” ypbind bindet nur an Loopback Interfaces und remote Hosts können diese nicht bedienen.
- YPBIND_OPTIONS: Extra Optionen wie “-upset”, “-ypsetm‘”, “-p port” oder “-no-ping”
- -ypset: Root kann von jedem Remote Rechner die Bindung an eine Domain mit dem Kommando ypset ändern
- -ypsetme: Root kann nur vom lokalen Rechner aus die Bindung an eine Domain mit dem Kommando ypset ändern
- -no-ping: ypbind überprüft nicht, ob die Bindung aktiv ist

Das Dateiattribut “letzter Zugriff” gibt den Zeitpunkt an, an dem der Vorgang: *Benutzer frägt Informationen über NIS ab*, gestartet wurde.

(3) Konfigurationsdatei /etc/yp.conf (Seite 205)

Beschreibung der Quelle:
Die Konfigurationsdatei ist eine Textdatei und wird von ypbind bei der Kontaktaufnahme zu einem NIS- Server einer Domain gelesen, wenn in der Konfigurationsdatei `/etc/sysconfig/ypbind` die Parameter YPBIND_BROADCAST auf “no” gesetzt ist. Dabei wird das Dateiattribut “letzter Zugriff“ der Konfigurationsdatei auf die aktuelle Systemzeit und Datum gesetzt.

Ergiebigkeit:
Mittel. Die Konfigurationsdatei enthält den Domänennamen und den NIS- Server. Die Konfigurationsdatei kann folgende Einträge haben.

```
domain nisdomain server hostname
domain nisdomain broadcast
ypserver hostname
```

Das Dateiattribut “letzter Zugriff” gibt den Zeitpunkt an, an dem der Vorgang: *Benutzer frägt Informationen über NIS ab*, gestartet wurde.

---

(4) Eintrag in /var/yp/binding/<domain>.<version>(Seite 218)

Beschreibung der Quelle:
Die Datei ist eine binding Datei und wird beim Aufruf von ypbind ohne broadcast bzw. der NIS um einen Eintrag ergänzt.

Ergiebigkeit:
Mittel. Die Datei enthält die IP- Adresse des NIS- Servers der spezifizierten Domain und den Port den der NIS- Server verwendet. Das Dateiattribut “letzter Zugriff” gibt den Zeitpunkt an, an dem der Vorgang: *Benutzer frägt Informationen über NIS ab*, gestartet wurde.

---

(5) Konfigurationsdatei /etc/yp/DOMAINNAME (Seite 205)

Beschreibung der Quelle:
Die Datei ist eine Textdatei und wird beim Aufruf von *ypbind* ohne broadcast gelesen. Dabei wird das Dateiattribut “letzter Zugriff“ der Konfigurationsdatei auf die aktuelle Systemzeit und Datum gesetzt.

Ergiebigkeit:
Schlecht. Die Konfigurationsdatei enthält eine Liste von NIS- Servern für die Domain. Das Dateiattribut “letzter Zugriff” gibt den Zeitpunkt an, an dem der Vorgang: *Benutzer frägt Informationen über NIS ab*, gestartet wurde.

---

(6) Eintrag in History Liste des Benutzers (Seite 221)

Beschreibung der Quelle:
vgl. Seite 26

Ergiebigkeit:
Mittel. Die History Liste enthält folgende Kommandos:

- ypbind -broadcast oder ypset -d <domain> -h <hostname> <server>
- ypcat -d <domain> -h <hostname> <mapname> oder ypmatch -d <domain> <key> ... <mapname>

---

(7) UDP- Verbindung (Seite 231)

Beschreibung der Quelle:
Eine UDP- Verbindung wird vom *nis Client* erzeugt.

Ergiebigkeit:
Gut. Die UDP- Verbindung birgt folgende Informationen:

- Local Adress: IP- Adresse des Client
- Local Port: >1024
- Foreign Adress: IP- Adresse des Servers
- Foreign Port: <1024
- State: aktive Verbindung

6.6.5.2.2 NIS- Server auf zu untersuchenden System

Der Benutzer ist auf einem anderen Rechner eingeloggt und will vom untersuchenden System Maps abfragen. Er sucht einen für die NIS- Domain zuständigen Server via broadcast mit dem Kommando `ypbind -broadcast` oder mittels der Konfigurationsdatei `/etc/yp.conf`.

Existenz:
Online Auswertung: (1), (2), (3), (4), (5), (6)
Offline Auswertung: (1), (2), (3), (4), (5)

---

(1) Konfigurationsdatei /etc/ypserv.conf (Seite 205)

Beschreibung der Quelle:
Die Konfigurationsdatei ist eine Textdatei und wird beim Zugriff des Benutzers auf den NIS- Server gelesen. Dabei wird das Dateiattribut "letzter Zugriff" der Konfigurationsdatei auf die aktuelle Systemzeit und Datum gesetzt.

Ergiebigkeit:
Mittel. Die Konfigurationsdatei enthält folgenden Informationen:
Optionen:

- IP- Adresse: IP- Adresse, für die die Regel gilt. Netzwerkadressen und Wildcards sind ebenso zulässig.
- NIS- Domain: Name der NIS- Domain, für die diese Regel gilt. Ein Stern "*" steht für "alle Domains".
- Map- Name: Name der Map, für die die Regel gilt. Ein Stern "*" steht für "alle Maps".
- Sicherheit: Die 3 möglichen Werte sind "none", "port" und "deny".
- "none" erlaubt den Zugriff stets, während "deny" diesen konsequent ablehnt. Mit "port" werden Zugriffe nur gestattet, wenn sie an einem Port mit einer Nummer <1024 erfolgen.

Zugangsregeln:

- Dns [yes, **no**]: der NIS- Server versucht Rechnernamen, die nicht in den hosts*-Maps enthalten sind, durch Befragung des DNS-Servers aufzulösen. Üblich ist, Clients so zu konfigurieren, dass sie den DNS-Server selbst kontaktieren, falls die Befragung des NIS-Servers kein Ergebnis erbrachte.

- Files: Anzahl der Map- Dateihandles, die der Server maximal im Cache zwischenspeichern soll.
- Trusted_Master: Diese Option ist für NIS- Slave- Server relevant und enthält den vollständigen Namen des Master-Servers. Der Slave-Server akzeptiert neue Maps nur vom angegebenen Master-Server. Fehlt diese Option, wird der Slave keine neuen Maps akzeptieren.
- xfr_check_port: der NIS-Server wartet an Ports mit Nummern <1024.

Das Dateiattribut "letzter Zugriff" gibt den Zeitpunkt an, an dem der Vorgang: *Benutzer frägt Informationen über NIS ab*, gestartet wurde.

---

(2) Konfigurationsdatei /etc/sysconfig/ypserv (Seite 205)

Beschreibung der Quelle:
Die Konfigurationsdatei ist eine Textdatei und wird beim Zugriff des Benutzers auf den NIS- Server gelesen. Dabei wird das Dateiattribut "letzter Zugriff" der Konfigurationsdatei auf die aktuelle Systemzeit und Datum gesetzt.

Ergiebigkeit:
Mittel. Die Konfigurationsdatei enthält folgende Parameter:

- YPPWD_SCR [/etc]: YP Source Verzeichnis für passwd, shadow und group
- YPPWD_CHFN [**yes**, no]: Darf der Benutzer sein GECOS Feld mit dem Kommando `ypchfn` ändern
- YPPWD_CHSH [**yes**, no]: Darf der Benutzer seine login Shell mit dem Kommando `ypchsh` ändern

Das Dateiattribut "letzter Zugriff" gibt den Zeitpunkt an, an dem der Vorgang: *Benutzer frägt Informationen über NIS ab*, gestartet wurde.

---

(3) Datei /var/yp/securenets (Seite 217)

Beschreibung der Quelle:
Die Datei ist eine Textdatei und wird gelesen, wenn der Benutzer Maps vom NIS-Server abrufen möchte. Ist die Datei vorhanden wird nur bestimmten Rechnern der Zugriff auf den NIS- Server gestattet. Wenn die Datei nicht vorhanden ist, ist allen Rechner der Zugriff gestattet. Dabei wird das Dateiattribut "letzter Zugriff" der Datei auf die aktuelle Systemzeit und Datum gesetzt.

Ergiebigkeit:
Mittel. Die Datei enthält Netzmaske und Netzadresse der Rechner, denen der Zugriff auf den NIS- Server gestattet ist. Die Datei hat folgendes Format:

```
Netzadresse des Rechners Netzmaske
```

Das Dateiattribut "letzter Zugriff" gibt den Zeitpunkt an, an dem der Vorgang: *Benutzer frägt Informationen über NIS ab*, gestartet wurde.

(4) Datei /var/yp/nicknames (Seite 217)

Beschreibung der Quelle:
Die Datei ist eine Textdatei und wird von `ypcat` gelesen. Dabei wird das Dateiattribut "letzter Zugriff" der Datei auf die aktuelle Systemzeit und Datum gesetzt.

Ergiebigkeit:
Mittel. Die Datei enthält Alias Namen und die zugehörigen Maps. Die Datei hat folgendendes Format:

`Alias Map Name.[option]`
Als Option wird angeben wie die anderen Benutzer an die Mapinformation kommt: per Nummer (`bynumber`), per Name (`byname`), per Alias (**aliases**), per Adresse (**byaddr**). Das Dateiattribut "letzter Zugriff" gibt den Zeitpunkt an, an dem der Vorgang: *Benutzer frägt Informationen über NIS ab*, gestartet wurde.

(5) Map /var/yp/<nisdomain >/<mapname>(Seite 217)

Beschreibung der Quelle:
Die Map wird durch die Anfrage des Benutzers gelesen. Dabei wird das Dateiattribut "letzter Zugriff" der Datei auf die aktuelle Systemzeit und Datum gesetzt.

Ergiebigkeit:
Mittel. Das Dateiattribut "letzter Zugriff" gibt den Zeitpunkt an, an dem der Vorgang: *Benutzer frägt Informationen über NIS ab*, gestartet wurde.

(6) UDP- Verbindung (Seite 231)

Beschreibung der Quelle:
Eine UDP- Verbindung wird vom *nfsd* erzeugt.

Ergiebigkeit:
Gut. Die TCP- Verbindung birgt folgende Informationen:

- Local Adress: IP- Adresse NIS Servers
- Local Port: >1024
- Foreign Adress: IP- Adresse des Clients, der Informationen vom NIS Server abfrägt
- Foreign Port: <1024
- State: aktive Verbindung

6.6.5.3 Vorgang: Benutzer ändert Passwort in der NIS- Datenbank

Der Benutzer ist/ war entweder am zu untersuchenden System eingeloggt und will sich von diesem Rechner aus sein Passwort auf einem anderen Rechner ändern oder ist/ war nicht am zu untersuchenden System eingeloggt und will sein Passwort am NIS- Server ändern.

6.6.5.3.1 NIS- Client auf zu untersuchenden System

Der Benutzer ist auf dem zu untersuchenden System eingeloggt und ruft `yppasswd` auf um das Passwort zu ändern.

Existenz:
Online Auswertung: (1), (2)
Offline Auswertung: (1)

---

(1) Eintrag in History Liste des Benutzers (Seite 221)

Beschreibung der Quelle:
vgl. Seite 26

Ergiebigkeit:
Mittel. Die History Liste enthält folgendes Kommando: `yppasswd`

---

(2) UDP- Verbindung (Seite 231)

Beschreibung der Quelle:
Eine UDP- Verbindung wird vom *nis Client* erzeugt.

Ergiebigkeit:
Gut. Die UDP- Verbindung birgt folgende Informationen:

- Local Adress: IP- Adresse des Client
- Local Port: <1024
- Foreign Adress: IP- Adresse des Servers
- Foreign Port: >1024
- State: aktive Verbindung

6.6.5.3.2 NIS- Server auf zu untersuchenden System

Der Benutzer ist auf einem anderen Rechner eingeloggt und will vom untersuchenden System aus sein Passwort mittels `yppasswd` ändern. Auf dem Server läuft der *rpc.yppasswd Dämon* (NIS password update Dämon).

Existenz:
Online Auswertung:
Mit Shadow Map:(1), (2), (3), (4), (5), (6), (7)
Ohne Shadow Map: (1), (2), (4), (6), (7)
Offline Auswertung:
Mit Shadow Map: (1), (2), (3), (4), (5), (6)
Ohne Shadow Map: (1), (2), (4), (6)

(1) Passwortdatei /etc/passwd (Seite 205)

Beschreibung der Quelle:
Die passwd- Datei ist eine Textdatei und wird durch rpc.yppasswd gelesen und ausgewertet. Die Parameter YPSRCDIR in der Datei `/var/yp/Makefile` gibt das Verzeichnis an, in der sich die passwd- Datei befindet. Dabei wird das Dateiattribut "letzter Zugriff" der Konfigurationsdatei auf die aktuelle Systemzeit und Datum gesetzt.

Ergiebigkeit:
Gut. Die Passwortdatei enthält folgende Informationen über den Benutzer, der sein Passwort über NIS ändern möchte: Passwort (falls keine `/etc/shadow` existiert) die UserID und GroupID aller Benutzer, Info, Namen des Home Directories und die verwendete Shell. Der Eintrag des Benutzers hat folgendes Format:

```
Username:Password:UID:GID:Info:Home:Shell
```

Das Dateiattribut "letzter Zugriff" gibt den Zeitpunkt an, an dem der Vorgang: *Benutzer ändert Passwort in der NIS Datenbank*, gestartet wurde.

(2) Feld in Passwortdatei /etc/passwd (Seite 206)

Beschreibung der Quelle:
Die passwd- Datei ist eine Textdatei und durch *rpc.yppasswd* wird das Passwortfeld geändert. Das Passwortfeld wird nur geändert wenn `rpc.yppasswd` keine Shadow-Passwörter unterstützt, d.h. die Parameter MERGE_PASSWD in der Datei `/var/yp/Makefile` auf "yes" gesetzt ist. Ansonsten steht dort ein "x".

Ergiebigkeit:
Gut. Das Feld in der passwd- Datei enthält das neue verschlüsselte Passwort des Benutzers.

(3) Feld in Shadowdatei /etc/shadow (Seite 206)

Beschreibung der Quelle:
Die Shadow- Datei ist eine Textdatei und durch *rpc.yppasswd* wird das Passwortfeld geändert. Das Passwortfeld wird nur geändert wenn rpc.yppasswd Shadow-Passwörter unterstützt, d.h. die Parameter MERGE_PASSWD in der Datei `/var/yp/Makefile` auf "no" gesetzt ist. Die Parameter YPSRCDIR in der Datei `/var/yp/Makefile` gibt das Verzeichnis an, in der sich die Shadow- Datei befindet.

Ergiebigkeit:
Gut. Das Feld in der Shadow- Datei enthält das neue verschlüsselte Passwort des Benutzers.

(4) /var/yp/<nisdomain>/passwd.<key>(Seite 218)

Beschreibung der Quelle:
Die passwd- Map ist eine Datei, in der die Daten im DBM-Format (Database

Management) organisiert sind und in dem Verzeichnis `/var/yp/<nisdomainname>` gespeichert sind. Rpc.yppasswd führt das Script `pwupdate` aus, um die Map `passwd.key` neu zugenerieren. Das Verzeichnis, in dem sich die Map befindet wird durch die Parameter YPMAPDIR in der Datei `/var/yp/Makefile` angegeben.

Ergiebigkeit:
Gut. Das Passwortfeld (Information) bezüglich des Schlüssels (key) des Benutzers enthält das neue verschlüsselte Passwort, falls `rpc.yppasswd` Shadow-Passwörter nicht unterstützt, d.h. die Parameter MERGE_PASSWD in der Datei `/var/yp/Makefile` auf "yes" gesetzt ist. Ansonsten steht dort ein "x".

(5) /var/yp/<nisdomainname>/shadow.byname(Seite 218)

Beschreibung der Quelle:
Die shadow.byname- Map ist eine Datei, in der die Daten im DBM-Format (Database Management) organisiert sind und in dem Verzeichnis `/var/yp/<nisdomainname>` gespeichert sind. Rpc.yppasswd führt das Script `pwupdate` aus, um die Map `shadow.byname` neu zugenerieren. Das Verzeichnis, in dem sich die Map befindet wird durch die Parameter YPMAPDIR in der Datei `/var/yp/Makefile` angegeben.

Ergiebigkeit:
Gut. Das Passwortfeld (Information) bezüglich des Schlüssels (key) des Benutzers enthält das neue verschlüsselte Passwort.

(6) Eintrag in Log- Datei von /var/log/rpc.yppasswd (Seite 210)

Beschreibung der Quelle:
Die Log- Datei von *rpc.yppasswd* ist eine Textdatei und wird um einen Eintrag erweitert, wenn der Benutzer eine Passwort update Anfrage macht. Ein Eintrag wird nur geschrieben, falls die Protokollierung in der Konfigurationsdatei `/etc/syslog.conf` des Sylogd Dämon aktiviert ist, das heißt ein entsprechender Eintrag existiert.

Ergiebigkeit:
Gut. Der Eintrag enthält die ursprüngliche Host- IP- Adresse, den Benutzernamen und die UID des Benutzers.

(7) UDP- Verbindung (Seite 231)

Beschreibung der Quelle:
Eine UDP- Verbindung wird vom *rpc.yppasswd Dämon* erzeugt.

Ergiebigkeit:
Gut. Die UDP- Verbindung birgt folgende Informationen:

- Local Adress: IP- Adresse des NIS Servers
- Local Port: >1024
- Foreign Adress: IP- Adresse des Clients, der das Passwort in der NIS Datenbank ändert

- Foreign Port: <1024
- State: aktive Verbindung

6.6.6 SSH Secure Shell

SSH gewährleistet verschlüsselte “remote logins”. Auf einem Serversystem ist ein SSH- Server installiert. Es gibt zwei verbreitete Versionen von ssh: einmal das kommerzielle Produkt von SSH Communications Security, dann die OpenSSH, die im Rahmen des OpenBSD- Projekts weiterentwickelt wird. OpenSSH unterstützt Protokoll Version 1 und 2. Anwender, die Dienste des Serversystems nutzen wollen melden sich mit ihrem SSH- Klienten am SSH- Server an.

Version 1:
Protokollversion 1 unterstützt die Authentifikation des Benutzers beim SSH-Server mit **Benutzername und Passwort**, mittels der `/$HOME/.rhosts`, `/$HOME/.rhosts` mit **RhostsRSAAuthentication** oder mit **RSA Challenge-Response Authentifikation**. Bei der Passwortauthentifikation muss in der Konfigurationsdatei `/etc/ssh/sshd_config` die Parameter **PasswortAuthentication** auf “yes” gesetzt sein. Bei der Authentifizierung mittels `/.hosts` muss **IgnoreRhosts** auf “no” und **RhostsAuthentifikation** auf “yes” gesetzt sein. Bei **RhostsRSAAuthentikation** wird zur Identität des Hosts nicht die IP-Adresse sonder der zu jedem Host gehörende öffentliche Schlüssel (PublicKey) verwendet (`/etc/ssh/ssh_host_key.pub`). Nur wenn dieser passt, d.h. in der Datei `/.ssh/knownhosts bzw. /etc/ssh/ssh_known_hosts` ist, wird ein passender Eintrag in der Datei `/$HOME/.rhosts` akzeptiert. Bei der letzten Möglichkeit erzeugt jeder Benutzer für sich mit dem Programm *ssh-keygen* ein Schlüsselpaar, dabei muss der Parameter **ChallengeResponseAuthentication** in der Konfigurationsdatei von sshd auf “yes” gesetzt sein. Beim Aufruf ohne Parameter wird ein Schlüssel erzeugt. Zu dem Schlüssel wird zusätzlich ein Passwort (passphrase) abgefragt, das für die spätere Authentifizierung gebraucht wird. Auf dem SSH-Server wird die Datei `/$HOME/.ssh/authorized_keys` um den betreffenden Public- Key ergänzt. Für jede Verbindung wird dann der Benutzer nach dem Passwort des Schlüssels gefragt, dies kann verhindert werden in dem das Programm *ssh-agent* zur Verwaltung der Benutzerschlüssel verwendet wird. Mit dem Programm *ssh-add* wird ein Schlüssel in den Agenten geladen und dieser wird der Passphrase des Benutzers entsperrt. Danach kann jedes Programm, das Zugriff auf den Agenten hat, die Schlüssel dort auslesen und ohne weitere Abfragen verwenden.

Version 2:
Protokoll Version 2 bietet **Public- Key Authentifikation**, **Hostbasierte Authentifikation**, **passwortbasierte Authentifikation**. Für die Authentifiaktion des Benutzers gegenüber dem Server mittels Public Keys muss in der Konfigurationsdatei `/etc/ssh/sshd_config` die Parameter **PubkeyAuthentication** auf “yes‘” gesetzt sein. Um Hostbasierte Authentifikation zu verwenden muss die Parameter **HostbasedAuthentication** in der Konfigurationsdatei von sshd auf “yes” gesetzt sein. Bei der Passwortauthentifikation muss in der Konfigurationsdatei `/etc/ssh/sshd_config` die Parameter **PasswortAuthentication** auf “yes” gesetzt sein. Bei der letzten Möglichkeit erzeugt jeder Benutzer für sich mit dem Programm *ssh-keygen* ein Schlüsselpaar, dabei muss die Parameter **ChallengeResponseAuthentication** in der Konfigurationsdatei von sshd auf “yes” gesetzt sein. Beim Aufruf mit Parameter `-t rsa oder -t dsa` wird ein Schlüssel erzeugt. Zu dem Schlüssel wird zusätzlich ein Passwort (passphrase) abgefragt, das für die spätere Authentifizierung gebraucht wird. Auf dem SSH- Server wird die Datei `/$HOME/.ssh/authorized_keys` um den betreffenden Public- Key ergänzt. Für jede Verbindung wird dann der Benutzer nach dem Passwort des Schlüssels gefragt, dies kann verhindert werden in dem das Programm ssh-agent zur Verwaltung der Benutzerschlüssel verwendet wird. Mit dem Programm *ssh-add* wird ein Schlüssel

in den Agenten geladen und dieser wird der Passphrase des Benutzers entsperrt. Danach kann jedes Programm, das Zugriff auf den Agenten hat, die Schlüssel dort auslesen und ohne weitere Abfragen verwenden.

6.6.6.1 Vorgang: Benutzer meldet sich an

Der Benutzer ist/war entweder am zu untersuchenden System eingeloggt und will sich von diesem Rechner aus an einen anderen Rechner unter Verwendung von SSH einloggen oder ist/war an einem Rechner eingeloggt und will sich am zu untersuchenden System anmelden.

6.6.6.1.1 SSH- Client auf zu untersuchenden System

Der Benutzer ist auf dem zu untersuchenden System eingeloggt und will sich von diesem Rechner aus an einem anderen Rechner einloggen.

Existenz:
Passwortbasierte Authentifikation:
Online Auswertung: (1), (2), (7), (8)
Offline Auswertung: (1), (2), (7)
RhostsRSAAuthentifikation:
Online Auswertung: (1), (2), (5), (7), (8)
Offline Auswertung: (1), (2), (5), (7)
Public Key Authentifikation:
Online Auswertung: (1), (2), (3), (4), (6), (7), (8)
Offline Auswertung: (1), (2), (3), (4),(6), (7)

---

(1) Environment Variablen (Seite 229)

Beschreibung der Quelle:
Beim Aufruf von ssh werden Environment Variablen gesetzt.

Ergiebigkeit:
Mittel. Folgende Environment Variablen werden beim Aufruf von ssh gesetzt:

- DISPLAY: Ort des X11 Servers
- HOME: Home Directory des Benutzers
- LOGNAME: wie USER
- MAIL: Pfad zur Mailbox des Benutzers
- PATH: Default Pfad, wenn ssh kompiliert wurde
- SSH_ASPPASS: Passphrase
- SSH_AUTH_SOCK: Pfad einer Linux Domänen Socket, um mit dem Agenten zu kommunizieren
- SSH_CLIENT: IP- Adresse des Clients, Portnummer des Clients, Portnummer des Servers
- SSH_ORIGINAL_COMMAND: Originale Kommandozeile

- SSH_TTY: Name des Terminals, an dem ssh aufgerufen wird
- TZ: Gegenwärtige Zeitzone
- USER: Loginname

---

(2) Konfigurationsdatei /etc/ssh/sshconfig (Seite 208)

Beschreibung der Quelle:
Die systemweite Konfigurationsdatei ist eine Textdatei und wird beim Aufruf von `ssh` gelesen und ausgewertet. Dabei wird das Dateiattribut "letzter Zugriff" der Konfigurationsdatei auf die aktuelle Systemzeit und Datum gesetzt.

Ergiebigkeit:
Schlecht. Die Konfigurationsdatei enthält folgende wichtige Parameter:

- ForwardAgent: Definiert ob Verbindungen mit Äuthentification Agentsän einem Remote-Host weitergeleitet werden sollen oder nicht. Hiermit ist es beispielsweise möglich, dass der lokale ssh-agent über eine bereits bestehende SSH Sitzung sich bei einem weiteren Rechner authentifiziert.
- ForwardX11: Dient der automatischen Weiterleitung von X11 Verbindungen. Mit der Kommandozeilenoption "-x" kann man ebenfalls das X11-Forwarding abschalten. X11 Forwarding ermöglicht es, den X Window Datenverkehr durch einen durch SSH gesicherten Tunnel zu schicken.
- RhostsAuthentication: Hiermit kann man den Versuch einer Authentifikation anhand der .rhosts Dateien von Clientseite aus eintragen. Da die Authentifizierung über .rhosts nicht besonders sicher ist, bieten die meisten Server sinnvollerweise in der Regel diese Option ohnehin nicht an.
- RhostsRSAAuthentication: Ermöglicht den Zugriff ohne Passwortauthentifizierung, wenn der Rechner/Benutzer in `hosts.equiv` oder `.rhosts` eingetragen ist und wenn der Schlüssel mit dem gespeicherten Host-Key übereinstimmt.
- RSAAuthentication: Gibt an ob RSA Authentifikation verwendet werden soll oder nicht. Hierzu muss entweder eine RSA-Identitätsdatei `(/$HOME/.ssh/identity)` vorhanden sein oder ein `ssh-agent` laufen.
- PasswordAuthentication: Mit den Werten "yes" und "no" kann eine passwortbasierende Authentifikationsmöglichkeit aktiviert bzw. deaktiviert werden.
- FallBackToRsh: Sollte der ssh-Dämon aus irgendeinem Grund nicht erreichbar sein, kann man dem ssh-Client mit dieser Option mitteilen, dass er in diesem Falle automatisch rsh aus den r-Tools zum Kommunikationsaufbau verwenden soll.
- UseRsh: Diese Option hilft einem, falls man rsh/rlogin anstelle von ssh verwenden möchte.
- CheckHostIP: Zusätzlich verwendet ssh die `known_hosts` Datei und die IP-Adresse des Rechners im Zusammenhang mit dem Rechnernamen zu überprüfen.
- StrictHostKeyChecking: "yes" Die Logins zu Rechnern werden unterbunden, deren Hostkey nicht bekannt ist oder verändert wurde.

- IdentityFile: Standardmässig liest ssh die Datei `/.ssh/identity` als RSA-Identitätsdatei ein.
- Port: Standardport für den SSH Dienst, ist der Port 22.
- GlobalKnownHostsFile: Definiert den Pfad zur gloablen known-hosts Datei. Normalerweise befindet sich diese in `/etc/ssh_known_hosts` oder in `/etc/ssh/ssh_known_hosts` und beinhaltet für das gesamte System gültige bekannte Rechner.
- HostName: Abkürzungen und Spitznamen für Rechner
- Protocol: OpenSSH unterstützt sowohl SSH Protokollversion 1 und 2.
- UsePrivilegedPort: Wenn für ausgehende durch ssh gesicherte Verbindungen priviligierte Ports verwendet werden sollen, so kann man dies hier mit “yes‘” definieren. Schaltet man diese Funktion ab (“no”), so stehen die Funktionen RhostsAuthentification und RhostsRSAAuthentification automatisch nicht mehr zur Verfügung.
- User: Definiert den Benutzer, welcher automatisch zum Einloggen verwendet werden soll.
- UserKnownHostsFile: Standardmässig befindet sich die benutzerspezifische `known_hosts` Konfigurationsdatei in `/$HOME/.ssh/known_hosts`.

Das Dateiattribut “letzter Zugriff” gibt den Zeitpunkt an, an dem der Vorgang: *Benutzer meldet sich an*, gestartet wurde.

---

(3) /$HOME/.ssh/{id_rsa, id_dsa, identity} (Seite 223)

Beschreibung der Quelle:
Bei der Protokoll Version 1 wird nur die Schlüsseldatei identity verwendet. Bei der Protokollversion 2 werden `id_rsa` und `id_dsa` verwendet. Die Parameter `IdentityFile` in der Konfigurationsdatei `/etc/ssh/ssh_config` spezifiziert die Schlüsseldatei abhängig von der Protokollversion. Die Schlüsseldateien sind Textdateien.
Die Schlüsseldatei entsteht beim Aufruf von `ssh-keygen` um Schlüsselpaare zu generieren. Mit dem Kommando `ssh-keygen -t rsa` entsteht eine privater RSA Schlüssel und mit `ssh-keygen -t dsa` ein privater DSA Schlüssel für den Benutzer bei Version 2. Beim Aufruf von `ssh-keygen` ohne Parameter entsteht der RSA-Schlüssel für Version 1.

Ergiebigkeit:
Mittel. Die Schlüsseldatei enthält die Authentifikationsidentität (privater Schlüssel) des Benutzers.

---

(4) /$HOME/.ssh/{id_dsa, id_rsa, identity }.pub (Seite 223)

Beschreibung der Quelle:
Bei der Protokoll Version 1 wird nur die Schlüsseldatei `identity.pub` verwendet. Bei der Protokollversion 2 wird entweder `id_rsa.pub oder id_dsa.pub` verwendet. Die öffentlichen Schlüsseldateien sind Textdateien. Die Schlüsseldatei

entsteht beim Aufruf von `ssh-keygen` um Schlüsselpaare zu generieren. Mit dem Kommando `ssh-keygen -t rsa` entsteht eine öffentlicher RSA Schlüssel und mit `ssh-keygen -t dsa` ein öffentlicher DSA Schlüssel für den Benutzer.

Ergiebigkeit:
Mittel. Die Schlüsseldatei enthält den öffentlichen Schlüssel, der zum privaten Schlüssel des Benutzers passt.

---

(5) Eintrag in /$HOME/.ssh/known_hosts (Seite 223)

Beschreibung der Quelle:
Diese Datei ist eine Textdatei und wird von ssh ausgewertet, wenn ssh rhosts mit RSA Host Authentification oder Protokolversion 2 hostbasierte Authentifikation verwendet, um den öffentlichen Schlüssel des Hosts zu überprüfen. Mit der Parameter `StrictHostKeyChecking` in der Konfigurationsdatei `/etc/ssh/ssh_config` wird geregelt, ob ssh automatisch Hostschlüssel zu der Datei `/$HOME/.ssh/known_hosts` hinzufügen darf. Ist die Parameter auf yes gesetzt, wird dies unterbunden. Es wird nur ein Eintrag hinzugefügt, wenn der Benutzer sich von einem unbekannten Host einloggen will.

Ergiebigkeit:
Mittel. Die Datei enthält den Hostschlüssel, bei denen sich der Benutzer einloggen will. Die Datei hat folgendes Format:

```
<Rechnername des Remote Hosts>, <IP- Adresse des Remote Hosts>,
<Schlüssel>
```

---

(6) Datei /$HOME/.ssh/known_hosts (Seite 222)

Beschreibung der Quelle:
Die Datei ist eine Textdatei und wird von ssh ausgewertet. Die Datei wird nur von ssh überprüft, wenn die Parameter `CheckHostIP` in der Konfigurationsdatei `/etc/ssh/ssh_config` auf "yes" gesetzt ist. Dabei wird das Dateiattribut "letzter Zugriff" der Konfigurationsdatei auf die aktuelle Systemzeit und Datum gesetzt.

Ergiebigkeit:
Mittel. Die Datei enthält alle Schlüssel der Hosts, bei denen sich der Benutzer eingeloggt hat bzw. später wieder einloggen möchte. Das Dateiattribut "letzter Zugriff" gibt den Zeitpunkt an, an dem der Vorgang: *Benutzer meldet sich an*, gestartet wurde.

---

(7) Eintrag in History Liste des Benutzers (Seite 221)

Beschreibung der Quelle:
vgl. Seite 26

Ergiebigkeit:
Mittel. Die History Liste enthält folgendes Kommando:

- gegebenenfalls `ssh-keygen -t rsa oder ssh-keygen -t dsa oder ssh-keygen`

- gegebenenfalls `ssh-agent <options>`
- gegebenenfalls `ssh-add <options>`

---

(8) TCP- Verbindung (Seite 231)

Beschreibung der Quelle:
Eine TCP- Verbindung wird vom *ssh Client* erzeugt.

Ergiebigkeit:
Gut. Die TCP- Verbindung birgt folgende Informationen:

- Local Adress: IP- Adresse des Client
- Local Port: <1024
- Foreign Adress: IP- Adresse des Servers
- Foreign Port: 22
- State: aktive Verbindung

6.6.6.1.2 SSH- Server auf zu untersuchenden System

Der Benutzer loggt sich mit ssh auf dem zu untersuchenden System ein.

Existenz:
Passwortbasierte Authentifikation:
Online Auswertung: (1), (2), (4), (14), (15), (18), (19), (20), (21), (22), (23)
Offline Auswertung: (1), (2), (4), (14), (12), (18), (19), (20), (21), (22)
RhostsRSA Authentifikation:
Online Auswertung: (1), (2), (4), (5), (6), {(9), (10), (11) oder (12) }, (18), (19), (20), (21), (22), (23)
Offline Auswertung: (1), (2), (4), (5), (6), {(9), (10), (11) oder (12) }, (18), (19), (20), (21), (22)
Public Key Authentifikation:
Online Auswertung: (1), (2), (3), (4), (6), (13), (18), (19), (20), (21), (22), (23)
Offline Auswertung: (1), (2), (3), (4), (6), (13), (18), (19), (20), (21), (22)

---

(1) Konfigurationsdatei /etc/nologin (Seite 205)

Beschreibung der Quelle:
Die Konfigurationsdatei ist eine Textdatei und wird von sshd gelesen und ausgewertet. Wenn diese Datei existiert, ist jedes “normale”

Einloggen im System unmöglich. Die Konfigurationsdatei `/etc/nologin` ist nur vorhanden, wenn die Parameter NOLOGIN_FILE in der Konfigurationsdatei `/etc/login.defs` auf diese Konfigurationsdatei gesetzt ist. Dabei wird das Dateiattribut “letzter Zugriff“ der Konfigurationsdatei auf die aktuelle Systemzeit und Datum gesetzt.

Ergiebigkeit:
Schlecht. Die Konfigurationsdatei /etc/nologin enthält Benutzernamen, die sich nicht einloggen dürfen. Das Dateiattribut “letzter Zugriff” gibt den Zeitpunkt an, an dem der Vorgang: *Benutzer meldet sich an*, gestartet wurde.

---

(2) Konfigurationsdatei /etc/ssh/sshd_config (Seite 208)

Beschreibung der Quelle:
Die Konfigurationsdatei ist eine Textdatei und wird von sshd gelesen und ausgewertet. Dabei wird das Dateiattribut “letzter Zugriff“ der Konfigurationsdatei auf die aktuelle Systemzeit und Datum gesetzt.

Ergiebigkeit:
Schlecht. Die Konfigurationsdatei enthält folgende wichtige Parameter:

- DSAAuthentication [yes—no]: DSA Authentifizierung wird genutzt oder nicht. Ist nur bei der Verwendung des SSH-2 Protokoll verfügbar.
- HostKey [Dateiangabe]: Datei, die den privaten Schlüssel für den Server beinhaltet. Normalerweise wird bei einem Debian GNU/Linux System mit OpenSSH der private Schlüssel in `/etc/ssh/ssh_host_key` und der öffentliche Schlüssel in `/etc/ssh/ssh_host_key.pub` gespeichert.
- IgnoreRhosts [yes—no]: Die `.rhosts- und .shosts-Datei` werden bei der Authentifizierung miteinbezogen oder nicht. Die Systemweitgültigen `/etc/hosts.equiv` und `/etc/ssh/shosts.equiv` werden weiterhin in die Authentifizierung miteinbezogen.
- IgnoreUserKnownHosts [yes—no]: sshd greift auf `/$HOME/.ssh/known_hosts` im Falle der RhostsRSAAuthentication zurück oder nicht (“yes”). Standard ist “no”.
- PasswordAuthentication [yes—no]: Passwort Authentifizierung ist aktiviert oder nicht. In der Regel erfordert Passwortauthentifizierung die Angabe des normalen Login-Passwortes.
- PermitEmptyPasswords [yes—no]: Erlaubt bzw. unterbindet das Einloggen von Benutzeraccounts ohne Passwörter, sofern Passwort Authentifikation eingeschaltet ist.
- PermitRootLogin [yes—no—without-password] Spezifiziert ob es dem Superuser (root) erlaubt ist, sich direkt mit einem SSH Client einzuloggen oder nicht. Mit der Angabe “without-password” wird nur das Einloggen über die Passwort Authentifikation unterdrückt.
- PidFile [Pfadangabe]: Speicherort für die Prozeß-ID von sshd (`/var/run/sshd.pid`).
- Protocol [1,2]: OpenSSH erlaubt sowohl die Verwendung von SSH-1 als auch SSH-2. Hier kann man die Unterstützung für beide Protokolle aktivieren oder gezielt die Unterstützung für eines der beiden Protokolle aktivieren.
- RhostsAuthentication [yes—no]: Mit dieser Option kann man einstellen, dass zur Authentifizierung ausschließlich die Dateien `/$HOME/.rhosts oder /etc/hosts.equiv` verwendet werden sollen. Davon sollte man aus Sicherheitsgründen absehen.

- RhostsRSAAuthentication [yes—no]: Zusätzlich zur RhostsAuthentication noch eine RSA basierende Authentifikation durchgeführt.
- RSAAuthentication [yes—no]: RSA (Publik Key) Authentifizierung wird genutzt oder nicht.
- SkeyAuthentication [yes—no] S/Key ist eine Einmal-Passwortverfahren. OpenSSH bietet hier die Möglichkeit S/Key Authentifizierung zu aktivieren ("yes"). Die Aktivierung von PasswordAuthentication ist eine Voraussetzung.
- StrictModes [yes—no]: Es ist für die Gesamtsicherheit rund um OpenSSH wichtig, dass einige bestimmte Dateien die benötigten bzw. nur die notwendigsten Rechte besitzen. Beispielsweise könnten lasche Dateirechte bei essenziellen Dateien wie der `authorized_keys` die Sicherheitsmechanismen aushebeln. Mit StrictModes kann man nun angeben, dass sshd die Rechte einiger wichtiger benutzerspezifischer Dateien und Verzeichnisse überprüfen soll. Fällt die Überprüfung positiv aus, also werden nur Dateien oder Verzeichnisse mit korrekten Rechten gefunden, so wird ein Login- Verbindungsversuch erlaubt.

Das Dateiattribut "letzter Zugriff" gibt den Zeitpunkt an, an dem der Vorgang: *Benutzer meldet sich an*, gestartet wurde.

---

(3) Schlüsseldatei /etc/ssh/ssh_host{, rsa, dsa }_key.pub (Seite 208)

Beschreibung der Quelle:
Bei der Protokollversion 1 wird nur die Schlüsseldatei `ssh_host_key.pub`. Bei der Protokollversion 2 wird entweder `ssh_host_rsa_key.pub oder ssh_host_dsa_key.pub` verwendet. Die Schlüsseldateien des Hosts sind Textdateien. Schon existent oder entsteht einmalig für Protokollversion 1 beim Aufruf von `ssh-keygen -t rsa -f /etc/ssh/ssh_host_key.pub`. Für Protokollversion 2 entsteht die Schlüsseldatei durch den Aufruf von `ssh-keygen -t rsa -f /etc/ssh/ssh_host_rsa_key.pub oder ssh-keygen -t dsa -f /etc/ssh/ssh_host_dsa_key.pub`.
Die Schlüssel werden bei Protokollversion 1 zur RSA Challenge Response Authentifikation und bei 2 zur Public Key Authentifikation verwendet.

Ergiebigkeit:
Mittel. Die Schlüsseldatei enthält den öffentlichen Schlüssel, der zum privaten Schlüssel des Hosts passt.

---

(4) Konfigurationsdatei /etc/ssh/moduli (Seite 208)

Beschreibung der Quelle:
Die Konfigurationsdatei ist eine Textdatei und wird von sshd gelesen und ausgewertet. Zu Beginn der Sitzung werden die Diffie- Hellmann Schlüssel ausgetauscht, dabei wird ein gemeinsam genutzter, geheimer Wert erstellt. Der Wert kann nur von beiden Kommunikationspartnern erstellt werden. Dabei wird das Dateiattribut "letzter Zugriff" der Konfigurationsdatei auf die aktuelle Systemzeit und Datum gesetzt.

Ergiebigkeit:
Schlecht. Enthält Diffie- Hellmann Gruppen, die für den "Diffie- Hellmann Group Exchange" gebraucht werden. Das Dateiattribut "letzter Zugriff" gibt den Zeitpunkt an, an dem der Vorgang: *Benutzer meldet sich an*, gestartet wurde.

---

(5) Schlüsseldatei /etc/ssh/ssh_host{ , dsa, rsa }_key (Seite 208)

Beschreibung der Quelle:
Bei der Protokollversion 1 wird nur die Schlüsseldatei `ssh_host_key` verwendet. Bei der Protokollversion 2 wird entweder `ssh_host_rsa_key oder ssh_host_dsa_key` verwendet. Die öffentlichen Schlüsseldateien des Hosts sind Textdateien und werden zur RhostsRSAAuthentication und HostbasedAuthentification. Die Parameter `HostKey` in der Konfigurationsdatei `/etc/ssh/sshd_config` spezifiziert die Schlüsseldatei abhängig von der Protokollversion. Die Schlüsseldatei für Protokollversion 1 entsteht beim Aufruf von `ssh-keygen -t rsa -f /etc/ssh/ssh_host_key`. Für Protokollversion 2 entsteht die Schlüsseldatei durch den Aufruf von `ssh-keygen -t rsa -f /etc/ssh/ssh_host_rsa_key oder ssh-keygen -t dsa -f /etc/ssh/ssh_host_dsa_key`.

Ergiebigkeit:
Mittel. Die Schlüsseldatei enthält den privaten Teil des Hostschlüssels.

---

(6) Konfigurationsdatei /etc/ssh/ssh_known_hosts (Seite 208)

Beschreibung der Quelle:
Die Konfigurationsdatei ist eine Textdatei und wird beim Anmelden am System mittels ssh gelesen. Der öffentliche Schlüssel `/etc/ssh/ssh_host_key.pub`, `/etc/ssh/ssh_host_dsa_key.pub` oder `/etc/ssh/ssh_rsa_host_key.pub` des Hosts muss mit dem Schlüssel in `/etc/ssh/ssh_known_hosts` oder `/$HOME/ssh_known_hosts` übereinstimmen. Dabei wird das Dateiattribut "letzter Zugriff" der Konfigurationsdatei auf die aktuelle Systemzeit und Datum gesetzt.

Ergiebigkeit:
Mittel. Die Konfigurationsdatei enthält eine Liste von allen bekannten Hostschlüsseln von Remote- Rechnern. Das Dateiattribut "letzter Zugriff" gibt den Zeitpunkt an, an dem der Vorgang: *Benutzer meldet sich an*, gestartet wurde.

---

(7) Konfigurationsdatei /etc/ssh/sshrc (Seite 208)

Beschreibung der Quelle:
Die Datei ist eine Textdatei und wird von ssh beim Einloggen des Benutzers gelesen und ausgewertet. Dabei wird das Dateiattribut "letzter Zugriff" der Konfigurationsdatei auf die aktuelle Systemzeit und Datum gesetzt.

Ergiebigkeit:
Mittel. Die Datei enthält Kommandos, die von ssh ausgeführt werden, wenn der Benutzer sich einloggt, bevor die Benutzershell gestartet wird. Das Dateiattribut "letzter Zugriff" gibt den Zeitpunkt an, an dem der Vorgang: *Benutzer meldet sich an*, gestartet wurde.

(8) Datei /$HOME/.ssh/rc (Seite 222)

Beschreibung der Quelle:
vgl. Seite 181 Informationsquelle (7)

(9) Konfigurationsdatei /etc/hosts.equiv (Seite 205)

Beschreibung der Quelle:
Die Konfigurationsdatei ist eine Textdatei und wird nur bei `.rhosts` Authentifikation (keine Passwortauthentifikation)

gelesen und ausgewertet. Dabei wird das Dateiattribut "letzter Zugriff" der Konfigurationsdatei auf die aktuelle Systemzeit und Datum gesetzt.

Ergiebigkeit:
Mittel. Die Konfigurationsdatei enthält eine Liste von Nutzern, die Zugang zu bestimmten Rechnern haben.

- + erlaubt den Zugang von sämtlichen Rechnern aus
- - Wird ein Rechner oder Nutzer vom Zugang ausgeschlossen (führendes -), bedeutet das ein Anmelden immer die Angabe des Passwortes erfordert Root wird niemals der passwortfreie Zugang gestattet

Die Konfigurationsdatei hat folgendes Format:

```
[+|-] <Rechnername> [<username>]
```

Das Dateiattribut "letzter Zugriff" gibt den Zeitpunkt an, an dem der Vorgang: *Benutzer meldet sich an*, gestartet wurde.

(10) Datei /$HOME/.rhosts (Seite 220)

Beschreibung der Quelle:
Die Datei /$HOME/.rhosts ist eine Textdatei und wird von *sshd* nur gelesen, wenn in der Konfigurationsdatei `/etc/ssh/sshd_config` die Parameter `IgnoreRhosts` auf "no" gesetzt ist. Dabei wird das Dateiattribut "letzter Zugriff" der Datei auf die aktuelle Systemzeit und Datum gesetzt.

Ergiebigkeit:
Mittel. Die Datei /$HOME/.rhosts enthält Rechnernamen/ Loginnamen von den der Benutzer sich auf den Remote- Rechner ohne Passwort einloggen darf. Die Datei hat folgendendes Format:

```
HOST LOGINNAME
```

Das Dateiattribut "letzter Zugriff" gibt den Zeitpunkt an, an dem der Vorgang: *Benutzer meldet sich an*, gestartet wurde.

(11) Datei /$HOME/.shosts (Seite 220)

Beschreibung der Quelle:
vgl. Seite 182 Informationsquellen (10)

(12) Konfigurationsdatei /etc/ssh/shosts.equiv (Seite 208)

Beschreibung der Quelle:
vgl. Seite 182 Informationsquelle (9)

(13) Eintrag in Datei /$HOME/.ssh/authorized_keys (Seite 223)

Beschreibung der Quelle:
In die Datei wird vor dem Anmelden am anderen System vom Benutzer der öffentliche Schlüssel `id_rsa.pub oder id_dsa.pub` für Protollversion 2 und für Protokollversion 1 `identity.pub` kopiert. In der Konfigurationsdatei `/etc/ssh/sshd_config` ist die Parameter `AuthorizedKeysFile` auf die Datei `/$HOME/.ssh/authorized_keys` gesetzt. Der Eintrag wird nur vom Benutzer vom Quellrechner in das Verzeichnis auf dem Remote Host (zu untersuchende System) kopiert mittels `scp /.ssh/id_rsa.pub oder id_dsa_pub oder /.ssh/identity.pub <server>:.ssh/authorized_keys`, wenn er sich mit Public Authentication einloggen möchte. Der Schlüssel wird verwendet damit der Client sicher sein, dass es sich um den wahren Host handelt.

Ergiebigkeit:
Gut. Das Verzeichnis enthält eine Liste von öffentlichen Schlüsseln (RSA oder DSA)

(14) Konfigurationsdatei /etc/passwd (Seite 205)

Beschreibung der Quelle:
Die Passwortdatei ist eine Textdatei und wird vom sshd gelesen und ausgewertet, falls die PAsswortbasierte Authentifikation in der Konfigurationsdatei `/etc/ssh/sshd_config` durch den Parameter `PasswordAuthentication` zugelassen ist. Dabei wird das Dateiattribut "letzter Zugriff" der Konfigurationsdatei auf die aktuelle Systemzeit und Datum gesetzt.

Ergiebigkeit:
Schlecht. Aus der Passwortdatei kann man den Loginnamen, Passwort (falls keine `/etc/shadow` existiert) die UserID und GroupID, Info, Namen des Home Directories und die verwendete Shell des Benutzers bekommen, der sich mit ssh am Server einloggt.

(15) Konfigurationsdatei /etc/shadow (Seite 205)

Beschreibung der Quelle:
Die Shadowdatei ist eine Textdatei und wird vom sshd gelesen und ausgewertet. Die Datei ist nur vorhanden, falls in der Passwortdatei `/etc/passwd` an 2. Stelle ein "x" steht und die Passwortbasiertes Authentifikation zugelassen wird. Dabei wird das Dateiattribut "letzter Zugriff" der Konfigurationsdatei auf die aktuelle Systemzeit und Datum gesetzt.

Ergiebigkeit:
Mittel. Aus der Shadow Datei kann man folgenden Informationen über den Benutzer beziehen, der sich mit ssh auf dem Server einloggt.

- Username: wie in /etc/passwd
- Passwort: Verschlüsseltes Passwort
- DOC: Day of last change, Tag ab dem 1.1.1970, an dem das Passwort zuletzt geändert wurde
- MinD: Minimale Anzahl Tage, die das Passwort gültig ist
- MaxD: Maximale Anzahl Tage, die das Passwort gültig ist
- Warn :Anzahl der Tage vor Ablauf der Lebensdauer des Passwortes, ab der vor dem Verfall zu warnen ist
- Exp: Expire, wieviele Sekunden das Passwort trotz Ablauf der MaxD gilt
- Dis: Bis zu diesem Tag (gezählt ab 1.1.1970) ist dieser Account gesperrt
- Res: Reserve, Feld wird derzeit nicht ausgewertet

Das Dateiattribut "letzter Zugriff" gibt den Zeitpunkt an, an dem der Vorgang: *Benutzer meldet sich an*, gestartet wurde.

---

(16) Konfigurationsdatei /etc/ssh/environment (Seite 208)

Beschreibung der Quelle:
Die Konfigurationsdatei ist eine Textdatei und wird beim Login mit ssh gelesen. Dabei wird das Dateiattribut "letzter Zugriff" der Konfigurationsdatei auf die aktuelle Systemzeit und Datum gesetzt.

Ergiebigkeit:
Mittel. Die Konfigurationsdatei enthält systemweite Definitionen von Environment Parametern mit ihren Werten für jede neue Verbindung. Das Dateiattribut "letzter Zugriff" gibt den Zeitpunkt an, an dem der Vorgang: *Benutzer meldet sich an*, gestartet wurde.

---

(17) Konfigurationsdatei /$HOME/.ssh/environment (Seite 222)

Beschreibung der Quelle:
Die Konfigurationsdatei ist eine Textdatei und wird beim Login gelesen. Dabei wird das Dateiattribut "letzter Zugriff" der Datei auf die aktuelle Systemzeit und Datum gesetzt.

Ergiebigkeit:
Mittel. Die Konfigurationsdatei enthält benutzerspezifische Definitionen von Environment Variablen mit ihren Werten für jede neue Verbindung. Das Dateiattribut "letzter Zugriff" gibt den Zeitpunkt an, an dem der Vorgang: *Benutzer meldet sich an*, gestartet wurde.

---

(18) Eintrag in Log- Datei /var/log/messages (Seite 210)

Beschreibung der Quelle:
vgl. Seite 26

Ergiebigkeit:
Gut. Der Eintrag enthält folgende Informationen:

- Time: Wann sich der Benutzer am Host eingeloggt hat
- Host: Rechnername des Hosts
- pid: Pid vom SSH Dämon
- User: Name des Benutzers
- IP- Address: IP- Adresse des Rechners von dem aus sich der Benutzer eingeloggt hat
- Port: Welcher Port wurde beim Einloggen verwendet

Der Eintrag hat folgendes Format:

```
<time> <host> sshd[<pid>]: Accepted password for <user> from
::ffff:<IP- address>port <number> ssh2
```

(19) Eintrag in Log- Datei /var/log/wtmp (Seite 210)

Beschreibung der Quelle:
vgl. Seite 27

Ergiebigkeit:
Gut. Der Eintrag in der Log- Datei enthält den Benutzernamen, den Terminal, die Loginzeit und von wo aus sich der Benutzer eingeloggt hat.

(20) Eintrag in Log- Datei /var/log/utmp (Seite 210)

Beschreibung der Quelle:
vgl. Seite 27

Ergiebigkeit:
Gut. Der Eintrag enthält folgende Informationen über den eingeloggten Benutzer:

- Ut_type: Typ des Logins (Login_Prozess)
- Ut_pid: Prozess- ID des Login Prozesses
- Ut_line: Gerätename des Terminals
- Ut_id: ID des Init Prozesses oder Name des Terminals
- Ut_user: Name des Benutzers
- Ut_host: Hostname von dem sich der Benutzer aus eingeloggt hat

- Ut_session: Session ID
- Ut_addr_v6: IP Adresse des remote Host

---

(21) Eintrag in Log-Datei /var/log/wtmp (Seite 210)

Beschreibung der Quelle:
vgl. Seite 27

Ergiebigkeit:
Gut. Der Eintrag in der Log- Datei enthält den Benutzernamen, den Terminal, die Loginzeit und von wo aus sich der Benutzer eingeloggt hat.

---

(22) Eintrag in der Log- Datei /var/log/lastlog (Seite 210)

Beschreibung der Quelle:

vgl. Seite 27

Ergiebigkeit:
Gut. Der Eintrag enthält den Benutzernamen, Port, pts/¡number¿ und die Zeit des erfolgreichen Einloggens.

---

(23) TCP- Verbindung (Seite 231)

Beschreibung der Quelle:
Eine TCP- Verbindung wird beim Einloggen via ssh erzeugt.

Ergiebigkeit:
Gut. Die ausgehende TCP- Verbindung birgt folgendene Informationen:

- Local Adress: IP- Adresse des SSH Servers
- Local Port: 22
- Foreign Adress: IP- Adresse des Clients, der sich via ssh auf dem Server einloggt
- Foreign Port: >1024
- State: Verbindung hergestellt

6.6.6.2 Vorgang: Benutzer kopiert Datei remote

Der Benutzer ist/war entweder am zu untersuchenden System eingeloggt und will kopiert von diesem Rechner aus eine Datei auf einen anderen Rechner unter Verwendung von `scp` oder ist/war an einem Rechner eingeloggt und will auf das zu untersuchenden System eine Datei mit `scp` kopieren.

6.6.6.2.1 Client auf dem zu untersuchenden System

Der Benutzer ist auf dem zu untersuchenden System eingeloggt und kopiert eine Datei mit dem Kommando `scp <options> <file1> <user>@<host>:<file2>` auf den remote Host.

Existenz:
Online Auswertung: (1), (2), (3), (4)
Offline Auswertung: (1), (2), (3)

---

(1) Eintrag in History Liste des Benutzers (Seite 221)

Beschreibung der Quelle:
vgl. Seite 26

Ergiebigkeit:
Mittel. Die History Liste enthält folgendes Kommando: `scp <options> <file1> <user>@<host>:<file2>`

---

(2) Dateiattribut "letzter Zugriff" (Seite 230)

Beschreibung der Quelle:
Wenn der Benutzer die Datei remote kopiert, wird das Dateiattribut "letzter Zugriff" auf die aktuelle Systemzeit und Datum gesetzt.

Ergiebigkeit:
Mittel. Wann der Benutzer die Datei remote kopiert hat.

---

(3) Datei <filename>(Seite 233)

Beschreibung der Quelle:
Die Datei wird vom Benutzer remote kopiert.

Ergiebigkeit:
Gut. Die Datei enthält folgende gesetzen Dateiattribute:

- File: Name der zu kopierenden Datei
- Device: Gerätename
- Inode: Nummer des Inodes
- Access: reguläre Datei und Zugriffsart
- Links: Anzahl der Dateinamen (Hardlinks)

- Uid: User ID des Benutzers, der die Datei remote kopiert
- Gid: Group ID des Benutzers, der die Datei remote kopiert
- Device Type: Gerätetyp
- Size: Gesamtgröße in Bytes
- IO Block: Blockgröße vom Dateisystem IO
- Access: aktuelle Datum und aktuelle Zeit des Systems (Informationsquelle (2))
- Modify: Datum des letzten schreibenden Zugriffs
- Change: Datum der letzten Statusänderung

6.6.6.2.2 Remote Host auf zu untersuchenden System

Der Benutzer ist nicht auf dem zu untersuchenden System eingeloggt und kopiert eine Datei auf das zu untersuchende System mittels `scp`.

Existenz:
Online Auswertung: (1), (2), (3)
Offline Auswertung: (1), (2)

---

(1) Datei (Seite 233)

Beschreibung der Quelle:
Die Datei wird vom Benutzer remote kopiert und befindet sich dann auf dem zu untersuchenden System.

Ergiebigkeit:
Gut. Die Datei enthält folgende gesetzen Dateiattribute:

- File: Name der auf den Server kopierten Datei
- Device: Gerätename
- Inode: Nummer des Inodes
- Access: reguläre Datei und Zugriffsart
- Links: Anzahl der Dateinamen (Hardlinks)
- Uid: User ID des Serves
- Gid: Group ID des Servers
- Device Type: Gerätetyp
- Size: Gesamtgröße in Bytes
- IO Block: Blockgröße vom Dateisystem IO
- Access: aktuelle Datum und aktuelle Zeit des Systems
- Modify: Datum des letzten schreibenden Zugriffs

- Change: Datum der letzten Statusänderung

---

(2) Eintrag in Log- Datei /var/log/messages (Seite 210)

Beschreibung der Quelle:
vgl. Seite 26

Ergiebigkeit:
Gut. Der Eintrag enthält folgende Informationen:

- Time: Wann sich der Benutzer am Host eingeloggt hat, um die Datei auf den Remote Host zu kopieren
- Host: Rechnername des Hosts
- pid: Pid vom SSH Dämon
- User: Name des Benutzers
- IP- Address: IP- Adresse des Rechners von dem aus sich der Benutzer eingeloggt hat
- Port: Welcher Port wurde beim Einloggen verwendet

Der Eintrag hat folgendes Format:

```
<time> <host> sshd[<pid>]: Accepted password for <user> from
::ffff:<IP- address> port <number> ssh2
```

---

(3) TCP- Verbindung (Seite 231)

Beschreibung der Quelle:
Eine TCP- Verbindung wird vom *sshd* erzeugt.

Ergiebigkeit:
Gut. Die TCP- Verbindung birgt folgende Informationen:

- Local Adress: IP- Adresse des SSH Server
- Local Port: 22
- Foreign Adress: IP- Adresse des Client, der die Datei kopiert hat
- Foreign Port: >1024
- State: aktive Verbindung

6.7 Softwareverwaltung

Programme unter Linux bestehen aus mehreren Dateien und sind zu einem Paket zusammengefasst. Ein Paket enthält ein Archiv von Dateien, Paket Informationen, die den Namen, die Version und die Beschreibung einschließt. Als Struktur eines solchen Pakets existieren im wesentlichen drei Formate:

1. RPM- Pakete (Red Hat Package Manager) mit der Dateiendung `.rpm`
2. Source Pakete (Tar- Archiv) mit der Dateiendung `.tar.gz`
3. SuSE Linux Source Pakete (Source- RPM) mit der Dateiendung `.spm`

6.7.1 Vorgang: Benutzer installiert RPM- Paket

Ein rpm Paket kann auf zwei verschiedene Weisen installiert werden:

1. Mittels dem Kommando `rpm -i [install options] <package name>.rpm`
2. Mittels der graphischen Oberfläche YAST2 → Software → Software installieren oder löschen

Existenz:
Online und Offline Auswertung: (1), (2), (3)

---

(1) Eintrag in RPM Datenbank /var/lib/rpm (Seite 215)

Beschreibung der Quelle:
Die RPM Datenbank ist das Verzeichnis `/var/lib/rpm` und enthält Dateien, die detaillierte Infromationen über das installierte Paket enthalten. Die RPM- Dateien werden bei der Installation des RPM- Paketes um Einträge erweitert.

Ergiebigkeit:
Mittel. Die RPM Datenbank enthält folgende Informationen über das installierte Paket:

- `conflictsindex.rpm`: Mit welchen Paketen das installierte Paket in Konflikt steht
- `fileindex.rpm`: Falls das installierte Package eine bestimmte Datei enthält, ist es in dieser Datei eingetragen
- `groupindex.rpm`: Alle Pakete, die zu einer bestimmten Gruppe gehören, möglicherweise auch das installierte
- `nameindex.rpm`: Paket mit bestimmten Namen
- `packages.rpm`: Alle installierten Packages, auch das zu installierende wird dort eingetragen
- `providesindex.rpm`: Welches Paket bietet eine bestimmte Fähigkeit, die das zu installierende Paket zusätzlich braucht.
- `requiredby.rpm`: Welche Pakete brauchen eine bestimmte Fähigkeit, um richtig zu arbeiten
- `triggerindex.rpm`: Triggerskripte des zuinstallierenden Paketes

(2) RPM- Paket <package name>.rpm (Seite 233)

Beschreibung der Quelle:
Das RPM Paket wird vom Benutzer durch Methode 1. und 2. installiert.

Ergiebigkeit:
Gut. Das RPM- Paket enthält folgende Informationen:

1. Integrität des Paketes:
   - Dateiname inklusive Pfad des installierten RPM Paketes
   - Dateigröße in Bytes des installierten RPM Paketes
   - Zeit der letzten Modifikation des installierten RPM Paketes
   - MD5-Prüfsumme (fehlt bei symbolischen Links) des installierten RPM Paketes
   - Zugriffsrechte und Dateityp des installierten RPM Paketes
   - Eigentümer des installierten RPM Paketes
   - Besitzende Gruppe des installierten RPM Paketes
   - Handelt es sich um eine Konfigurationsdatei [1, ja] oder [0, nein]
   - Handelt es sich um eine Dokumentationsdatei [1, ja] oder [0, nein]
   - Ist die Datei im Speicher verschiebbar (relocatable) [1, ja] oder [0, nein]
   - Bei einem symbolischen Link der Name der Zieldatei oder “X” sonst
2. Detaillierte Auskunft über das Paket:
   - Name: Name des RPM-Pakets
   - Version: Versionsnummer der enthaltenen Software
   - Vendor: Anbieter der Software
   - Release: Versionsnummer des Pakets
   - Build Date: Wann das Paket erstellt wurde
   - Date: Wann das Paket installiert wurde
   - Group: zugehörige Gruppen
   - Source RPM: Vollständiger Name der RPM- Paketes
   - Size: Größe des RPM- Paketes
   - License: Lizens
   - Packager: Name des Erzeugers des Paketes
   - Summary: Kurze Beschreibung zum Inhalt des Paketes
   - Description: Kurze Beschreibung über das was das Paket enthält bzw. macht
   - Authors: Autoren

(3) Eintrag in History Liste des Benutzers (Seite 221)

Beschreibung der Quelle:
vgl. Seite 26

Ergiebigkeit:
Mittel. Die History Liste enthält folgende Kommandos:

1. `rpm -i [install options] <package name>.rpm`
2. kein Eintrag

6.7.2 Vorgang: Benutzer stellt RPM Anfrage

Der Benutzer stellt eine RPM- Anfrage:

1. Mittels dem Kommando `rpm -q [package selection options] [query-options]`
2. Mittels der graphischen Oberfläche YAST2 → Software installieren oder löschen

Existenz:
Online und Offline Auswertung: (1), (2), (3) oder (1), (2)

(1) RPM Datenbank /var/lib/rpm (Seite 216)

Beschreibung der Quelle:
Die RPM Datenbank ist das Verzeichnis `/var/lib/rpm` und enthält Dateien, die detaillierte Informationen über das angefragte Paket enthalten. Die rpm Dateien werden bei der Anfrage je nach der Package Selection Option gelesen: `conflictsindex.rpm` automatisch, `fileindex.rpm` mit `-f`, `groupindex.rpm` mit `-g`, `nameindex.rpm` mit `<package name>`, `packages.rpm` automatisch, `providesindex.rpm` mit `-whatprovides`, `requiredby.rpm` mit `-whatrequires` und `triggeredby.rpm` mit `-triggeredby`.

Ergiebigkeit:
Gut. Die RPM Datenbank enthält folgende Informationen über das angefragte Paket:

- `conflictsindex.rpm`: Mit welchen Paketen das angefragte Paket in Konflikt steht
- `fileindex.rpm`: Falls das angefragte Paket eine bestimmte Datei enthält, ist es in dieser Datei eingetragen
- `groupindex.rpm`: Alle Pakete, die zu einer bestimmten Gruppe gehören, möglicherweise auch das angefragte Paket.
- `nameindex.rpm`: Angefragte Paket mit bestimmten Namen
- `packages.rpm`: Alle installierten Packages, auch das angefragte Paket ist dort eingetragen

- `providesindex.rpm`: Welches Paket bietet eine bestimmte Fähigkeit, die das zu angefragte Paket zusätzlich braucht.
- `requiredby.rpm`: Welche Pakete brauchen eine bestimmte Fähigkeit, um richtig zu arbeiten
- `triggerindex.rpm`: Triggerskripte des angefragten Paketes

---

(2) RPM Paket (Seite 233)

Beschreibung der Quelle:
Das RPM Paket wird vom Benutzer durch Methode 1. und 2. angefragt.

Ergiebigkeit:
Mittel. Das angefragte RPM- Paket enthält folgende Informationen:

- Name: Name des RPM- Pakets
- Version: Versionsnummer der enthaltenen Software
- Vendor: Anbieter der Software
- Release: Versionsnummer des Pakets
- Build Date: Wann das Paket erstellt wurde
- Install date: Wann das Paket installiert wurde
- Group: zugehörige Gruppen
- Source RPM: Vollständiger Name der RPM- Paketes
- Size: Größe des RPM- Paketes
- License: Lizens
- Packager: Name des Erzeugers des Paketes
- Summary: Kurze Beschreibung zum Inhalt des Paketes
- Description: Kurze Beschreibung über das was das Paket enthält bzw. macht
- Authors: Autoren

---

(3) Eintrag in History Liste des Benutzers (Seite 221)

Beschreibung der Quelle:
vgl. Seite 26

Ergiebigkeit:
Mittel. Die History Liste enthält folgende Kommandos:

1. `rpm -q [package selection options] [query- options]`
2. kein Eintrag

6.7.3 Vorgang: Benutzer verifiziert RPM Paket

Der Benutzer verifiziert das RPM Paket folgendermaßen:

1. Mittels dem Kommando `rpm -V [verify options]<package name>.rpm`
2. Mittels der graphischen Oberfläche YAST2 → Software installieren oder löschen

Existenz:
Online und Offline Auswertung: (1), (2), (3) oder (1), (2)

---

(1) RPM Datenbank /var/lib/rpm (Seite 216)

Beschreibung der Quelle:
Die RPM Datenbank ist das Verzeichnis `/var/lib/rpm` und enthält Dateien, die detaillierte Informationen über das angefragte Paket enthalten. Bei der Verifikation werden alle zu einem Paket gehörenden Dateien im Dateisystem mit den Daten, die in der RPM Datenbank gespeichert sind, gelesen und verglichen.

Ergiebigkeit:
Mittel. Die RPM Datenbank enthält folgende Informationen über das verifizierte Paket:

- `conflictsindex.rpm`: Mit welchen Paketen das verifizierte Paket in Konflikt steht
- `fileindex.rpm`: Falls das verifizierte Paket eine bestimmte Datei besitzt, ist es in dieser Datei eingetragen
- `groupindex.rpm`: Das verifizierte Paket gehört zu einer bestimmten Gruppe
- `nameindex.rpm`: Verifizierte Paket mit bestimmten Namen
- `packages.rpm`: Das verifizierte Pakete ist dort eingetragen, da es bereits installiert ist.
- `providesindex.rpm`: Welches Paket bietet eine bestimmte Fähigkeit, die das verifizierte Paket zusätzlich braucht.
- `requiredby.rpm`: Welche Pakete brauchen eine bestimmte Fähigkeit, um richtig zu arbeiten
- `triggerindex.rpm`: Triggerskripte des verifizierten Paketes
- MD5 Prüfsumme jeder einzelnen Datei, die RPM- Paket enthalten ist
- Dateigröße jeder enthaltenen Datei
- symbolischer Link jeder enthaltenen Datei
- Gerät
- Dateibearbeitungszeit jeder enthaltenen Datei
- Benutzer jeder enthaltenen Datei
- Gruppe jeder enthaltenen Datei

- Modus (einschließlich Berechtigungen und Dateityp) der enthaltenen Datei

---

(2) Dateien des RPM Paketes (Seite 233)

Beschreibung der Quelle:
Die Dateien des RPM Paketes werden mit den Daten, die in der RPM Datenbank gespeichert sind verglichen.

Ergiebigkeit:
Mittel. Jede einzelne Datei im Dateisystem, die zum RPM Paket gehört, trägt folgende Informationen:

- MD5 Prüfsumme der Datei
- Dateigröße
- symbolischer Link der Datei
- Gerät
- Dateibearbeitungszeit
- Benutzer der Datei
- Gruppe der Datei
- Modus (einschließlich Berechtigungen und Dateityp) der Datei

---

(3) Eintrag in History Liste des Benutzers (Seite 221)

Beschreibung der Quelle:
vgl. Seite 26

Ergiebigkeit:
Mittel. Die History Liste enthält folgendes Kommando:

1. `rpm -V [verify options]<package name>.rpm`
2. kein Eintrag

6.7.4 Vorgang: Benutzer deinstalliert RPM Paket

Der Benutzer kann ein RPM Paket folgendermaßen deinstallieren:

1. Mittels dem Kommando `rpm -e [deinstall options] <package name>`
2. Mittels der graphischer Oberfläche YAST2 → Software installieren oder löschen

Existenz:
Online und Offline Auswertung: (1)

Nicht- Existenz:
Die Informationsquellen (1), (2) dürfen nicht mehr existent sein.

(1) Eintrag in der RPM Datenbank /var/lib/rpm (Seite 215)

Einträge in den Dateien `conflictindex.rpm`, `conflictindex.rpm`, `groupindex.rpm`, `packages.rpm`, `providesindex.rpm`, `requiredby.rpm` und der `triggerindex.rpm` bzgl. des zu deinstallierenden Paketes werden durch Methode 1. und 2. entfernt.

(2) RPM Paket <filename>.rpm (Seite 233)

Dieses RPM Paket wird entfernt.

(3) Eintrag in History Liste des Benutzers (Seite 221)

Beschreibung der Quelle:
vgl. Seite 26

Ergiebigkeit:
Mittel. Die History Liste enthält folgendes Kommando:

1. `rpm -e [deinstall options] <package name>`
2. kein Eintrag

6.7.5 Vorgang: Benutzer installiert Source Paket

Der Benutzer installiert das Source Paket folgendermaßen:

- Entpacken des Paketes mittels dem Kommando `tar xzf <package name>`
- Erzeugen des Makefiles, falls nicht im Stammverzeichnisses des Source Paketes vorhanden, mit dem Kommando `./configure`
- Installation mit dem Kommando make install oder gegebenenfalls mit `sh install.sh`

Existenz:
Online und Offline Auswertung: (1), (2), (3) oder (1), (2)

(1) Verzeichnis /<installation path>/<source package>(Seite 233)

Beschreibung der Quelle:
Das Tar- Archiv ist ein Verzeichnis und wird beim Entpacken mit dem Kommando `tar xzf <package name>` des Source Paketes erstellt. Für absolute Pfadangaben wird die `-C /<installation path>` aufgerufen.

Ergiebigkeit:
Mittel. Das Verzeichnis enthält die Dateien des Tar Archives, wie die Installationsdatei `install.sh`, die Readme Datei, den Ordner man mit den Man Pages und andere Dateien und Ordner, die abhängig vom Programm sind.

(2) Makefile des Source Paketes (Seite 233)

Beschreibung der Quelle:
Die Makefile, falls sie nicht im Stammverzeichnisses des Source Paketes vorhanden ist, wird mit dem Kommando `./configure` erzeugt.

Ergiebigkeit:
Mittel. Die Makefile enthält Informationen zur Installation.

(3) Einträge in History Liste des Benutzers (Seite 221)

Beschreibung der Quelle:
vgl. Seite 26

Ergiebigkeit:
Mittel. Die History Liste enthält folgende Kommandos:

- `tar xzf <package name>`
- gegebenenfalls `./configure`
- `make install oder sh install.sh`

6.7.6 Vorgang: Benutzer deinstalliert Source Paket

Der Benutzer kann das Source Paket folgendermaßen deinstallieren:

1. Aufruf von `uninstall bzw. uninstall.sh`
2. Aufruf von `make uninstall`, falls das Makefile ein “Ziel” zur Deinstallation beinhaltet

Existenz:
Online und Offline Auswertung: (2)

Nicht- Existenz:
Die Informationsquelle (1) darf nicht existent sein.

(1) Tar Archiv (Seite 233)

Der Benutzer entfernt das Tar- Archiv des Source Paketes mit dem Kommando `rm -d <package name>`.

(2) Einträge in History Liste des Benutzers (Seite 221)

Beschreibung der Quelle:
vgl. Seite 27

Ergiebigkeit:
Mittel. Die History Liste enthält folgende Kommandos:

- `uninstall bzw. uninstall.sh` oder `make uninstall`
- `rm -d <package name>`

6.7.7 Vorgang: Benutzer installiert Source Paket von SuSE Linux

Der Benutzer installiert das Paket `<source name>.spm` mit der graphischen Oberfläche YAST2.

Existenz:
Online und Offline Auswertung: (1), (2), (3), (4)

---

(1) Datei /usr/src/SPECS/<source name>.spec (Seite 228)

Beschreibung der Quelle:
Die Datei ist eine Meta- Makefile und wird bei der Installation des Source Paketes im Verzeichnis `/usr/src/packages/SPECS` erzeugt.

Ergiebigkeit:
Mittel. Die Datei enthält Informationen für den Build Prozess.

---

(2) /usr/src/packages/SOURCES/<source name>.dif (Seite 228)

Beschreibung der Quelle:
Die Datei wird bei der Installation des Source Paketes im Verzeichnis `/usr/src/packages/SOURCES` erzeugt.

Ergiebigkeit:
Mittel. Die Datei enthält distributionsspezifische Anpassungen.

---

(3) /usr/src/packages/SOURCES/<source name>.tar.gz (Seite 228)

Beschreibung der Quelle:
Die Datei ist eine Binärdatei und wird bei der Installation des Source Paketes im Verzeichnis `/usr/src/packages/SOURCES` erzeugt.

Ergiebigkeit:
Mittel. Die Datei ist die originale Quelle.

---

(4) Source Paket <source name>.spm (Seite 233)

Beschreibung der Quelle:
Dieses Source Paket wird vom Benutzer installiert.

vgl. Seite 199 Informationsquelle (2)

6.7.8 Vorgang: Benutzer erzeugt RPM Paket

Der normale Benutzer und root können RPM Pakete erzeugen.

- Der Benutzer editiert die SPEC- Datei `<package name>.spec`.
- Mit dem Kommando `rpm -bStadium [Optionen] <package name>.spec` wird das RPM Paket erzeugt.

6.7.8.1 Benutzer root

Existenz:
Online und Offline Auswertung: (1), (2), (3), (4), (5), (6), (7), (8), (9)

---

(1) /usr/src/packages /SPECS/<package name>.spec (Seite 228)

Beschreibung der Quelle:
Die Paketanleitung ist eine Textdatei und wird vom Benutzer mittels einem Texteditor erstellt und dann von *rpm* gelesen und geparst. Das Verzeichnis wird durch den Parameter %_specdir in der Konfigurationsdatei `/usr/lib/rpm/macros` spezifiziert.

Ergiebigkeit:
Mittel. Die Paketanleitung enthält allgemeine Informationen und eine Art Bauanleitung zum Paket. Die Datei besteht aus mehreren Sektionen:

- `Präambel:`Die Präambel enthält allgemeine Informationen zum RPM-Paket und zum Bau.
- `%description- Sektion:` Die %description- Sektion enthält ein aussagekräftige Beschreibung zum Inhalt des Pakets.
- `%prep- Sektion:` In der %prep- Sektion findet die Vorbereitung zum Bau des Pakets statt. Letztlich beinhaltet die Sektion verschiedene Shellbefehle, die die Quellen für den anschließenden Kompiliervorgang - sofern ein solcher erforderlich ist - aufbereiten.
- `%build- Sektion:` Im Falle von Quellpaketen muss die zu installierende Software erst erzeugt werden. Genügt hierfür ein einfacher Aufruf von make, kann die build- Sektion entfallen. Im anderen Fall werden die auszuführenden Schritte angegeben.
- `%changelog- Sektion:` Die %changelog- Sektion beinhaltet Informationen über die wesentlichen Änderungen der enthaltenen Software.
- `%clean:` RPM entsorgt die "Überreste" der Installation selbst, mit der Ausnahme der Verwendung eines buildroot- Verzeichnisses. Ein solches sollte im clean-Eintrag entfernt werden.
- `%files- Sektion:` In der %files- Sektion folgen die Dateien, die das RPM-Paket enthalten soll.
- `%install- Sektion:` %post, %postun: Die Sektionen beinhalten Bashskripte oder die Namen der dem RPM-Paket beiliegenden Skriptdateien, die nach der Installation (%post) bzw. nach dem Löschen (%postun) des Pakets ausgeführt werden sollen. %pre, %preun: Die Sektionen beinhalten Bashskripte oder die Namen von dem RPM-Paket beiliegenden Skriptdateien, die vor der Installation (%pre) bzw. vor dem Löschen (%preun) des Pakets ausgeführt werden sollen.

Verweis auf andere Vorgänge:

- *Benutzer erstellt neue Datei* (vgl. Seite 86)
- *Benutzer ändert Dateiinhalt* (vgl. Seite 88)

(2) Konfigurationsdatei /usr/lib/rpm/rpmrc (Seite 227)

Beschreibung der Quelle:
Die Konfigurationsdatei ist eine Textdatei und wird von rpm gelesen. Die Konfigurationsdatei wird nur ausgwertet, falls die beiden Konfigurationsdateien `/etc/rpmrc` und `/.rpmrc` nicht existieren. Dabei wird das Dateiattribut "letzter Zugriff" der Konfigurationsdatei auf die aktuelle Systemzeit und Datum gesetzt.

Ergiebigkeit:
Schlecht. Die Konfigurationsdatei enthält die folgenden relevanten Einträge enthalten:

- arch_canon: Eintrag für Architektur
- os_canon: Eintrag für Betriebsystem
- buildarchtranslate: Ausgewählte Architektur für den BUILD Prozess
- buildostranslate: Ausgewähltes Betriebsystem für den BUILD Prozess
- arch_compat: Architekturen, die miteinander kompatibel sind
- os_compat: Betriebsysteme, die miteinander kompatibel sind
- builddir: Verzeichnis, in dem das RPM Paket erzeugt wird
- buildroot: Root Verzeichnis
- dbpath: Verzeichnis, in dem sich die RPM Datenbank befindet (`/var/lib/rpm`)
- optflags: Optionen, die während des BUILD Prozess standardmäßig verwendet werden können
- `%_topdir:` Verzeichnisbaum (`/home/user/myrpms`)
- `%_rpmdir:` Verzeichnis mit den fertigen RPM Paketen
- `%_sourcedir:` Verzeichnis mit den Quellen und Patches
- `%_specdir:` Verzeichnis mit den SPEC- Dateien, d.h. die Anleitungen zu jedem Paket
- `%_srcrpmdir:` Verzeichnis mit den fertigen SRPM Paketen

Das Dateiattribut "letzter Zugriff" gibt den Zeitpunkt an, an dem der Vorgang: *Benutzer erzeugt RPM Paket*, gestartet wurde.

(3) Konfigurationsdatei /etc/rpmrc (Seite 205)

Beschreibung der Quelle:
Die Konfigurationsdatei ist eine Textdatei und wird von rpm gelesen. Die Konfigurationsdatei wird nur ausgwertet, falls die Konfigurationsdatei `/.rpmrc` nicht vorhanden ist. Dabei wird das Dateiattribut "letzter Zugriff" der Konfigurationsdatei auf die aktuelle Systemzeit und Datum gesetzt.

Ergiebigkeit:
vgl. Seite 199 Informationsquelle (2)

(4) Datei /$HOME/.rpmrc (Seite 220)

Beschreibung der Quelle:
Die Datei ist eine Textdatei und wird von rpm gelesen und ausgewertet. Dabei wird das Dateiattribut "letzter Zugriff" der Konfigurationsdatei auf die aktuelle Systemzeit und Datum gesetzt.

Ergiebigkeit:
vgl. Seite 199 Informationsquelle (2)

(5) Datei /usr/lib/rpm/macros (Seite 227)

Beschreibung der Quelle:
Die Konfigurationsdatei ist eine Textdatei und wird nur von rpm gelesen und ausgewertet. Dabei wird das Dateiattribut "letzter Zugriff" der Konfigurationsdatei auf die aktuelle Systemzeit und Datum gesetzt.

Ergiebigkeit:
Schlecht. Die Konfigurationsdatei `/usr/lib/rpm/macros` legt die Umgebung fest, um die Pakete zu erzeugen. Die Konfigurationsdatei enthält folgende Umgebung:

- `%_topdir`: Verzeichnisbaum (`/usr/scr/packages`)
- `%_rpmdir`: Verzeichnis mit den fertigen RPM Paketen
- `%_sourcedir`: Verzeichnis mit den Quellen und Patches
- `%_specdir`: Verzeichnis mit den SPEC- Dateien, d.h. die Anleitungen zu jedem Paket
- `%_srcrpmdir`: Verzeichnis mit den fertigen SRPM Paketen

Das Dateiattribut "letzter Zugriff" gibt den Zeitpunkt an, an dem der Vorgang: *Benutzer erzeugt RPM Paket*, gestartet wurde.

(6) Sources im Verzeichnis /usr/src/packages/BUILD (Seite 228)

Beschreibung der Quelle:
In das Verzeichnis `/usr/src/packages/BUILD` werden Dateien des Source Paketes entpackt und dann kompiliert. Das Verzeichnis wird durch den Parameter %_topdir in der Konfigurationsdatei `/usr/lib/rpm/macros` spezifiziert.

Ergiebigkeit:
Mittel. Das Verzeichnis enthält die Source Dateien die in der %prep- Sektion der SPEC- Datei angegeben sind.

(7) /usr/src/packages/RPMS/<package name>.rpm (Seite 228)

Beschreibung der Quelle:
Das RPM Paket wird im Verzeichnis `/usr/src/packages/RPMS` erzeugt,. Das Verzeichnis wird durch den Parameter %_rpmdir in der Konfigurationsdatei `/usr/lib/rpm/macros` spezifiziert.

Ergiebigkeit:
Mittel. Das RPM Paket enthält die aufgeführten Dateien und Verzeichnisse, die in der %files- Sektion der SPEC- Datei angegeben sind.

(8) /usr/src/packages /SRPMS/<package name>.spm (Seite 228)

Beschreibung der Quelle:
Das SRPM Paket wird im Verzeichnis `/usr/scr/packages/SRPMS` erzeugt. Das Verzeichnis wird durch den Parameter %_srcrpmdir in der Konfigurationsdatei `/usr/lib/rpm/macros` spezifiziert.

Ergiebigkeit:
Mittel. Das SRPM Paket enthält die aufgeführten Dateien und Verzeichnisse, die in der %files- Sektion der SPEC- Datei angegeben sind.

(9) Eintrag in History Liste des Benutzers (Seite 221)

Beschreibung der Quelle:
vgl. Seite 26

Ergiebigkeit:
Mittel. Die History Liste enthält folgendes Kommando:

- `vi /<path to spec>/<package name>.spec`
- `rpm -bStadium [Optionen] <package name>.spec`

6.7.8.2 Normaler Benutzer

Existenz:
Online und Offline Auswertung: (1), (2), (3), (4), (5), (6), (7), (8), (9), (10)

---

(1) Konfigurationsdatei /usr/lib/rpm/rpmrc (Seite 227)

---

vgl. Seite 199 Informationsquelle (2)

---

(2) Konfigurationsdatei /etc/rpmrc (Seite 205)

---

vgl. Seite 201 Informationsquelle (3)

---

(3) Datei /$HOME/.rpmrc (Seite 220)

---

vgl. Seite 201 Informationsquelle (4)

---

(4) Datei /$HOME/.rpmmacros (Seite 220)

Beschreibung der Quelle:
Die Konfigurationsdatei ist eine Textdatei und wird nur von rpm gelesen und ausgewertet. Dabei wird das Dateiattribut “letzter Zugriff“ der Konfigurationsdatei auf die aktuelle Systemzeit und Datum gesetzt.

Ergiebigkeit:
Schlecht. Die Konfigurationsdatei `/.rpmmacros` legt die Umgebung fest, um die Pakete zu erzeugen. Die Konfigurationsdatei enthält folgende Umgebung:

- `%_topdir`: Verzeichnisbaum (`/home/user/myrpms`)
- `%_rpmdir`: Verzeichnis mit den fertigen RPM Paketen
- `%_sourcedir`: Verzeichnis mit den Quellen und Patches
- `%_specdir`: Verzeichnis mit den SPEC- Dateien, d.h. die Anleitungen zu jedem Paket
- `%_srcrpmdir`: Verzeichnis mit den fertigen SRPM Paketen

Das Dateiattribut “letzter Zugriff” gibt den Zeitpunkt an, an dem der Vorgang: *Benutzer erzeugt RPM Paket*, gestartet wurde.

---

(5) Sources in /$HOME/<user>/myrpms/BUILD (Seite 221)

Beschreibung der Quelle:
In das Verzeichnis `/$HOME/user/myrpms/BUILD` werden Dateien des Source Paketes entpackt und dann kompiliert. Das Verzeichnis wird durch den Parameter %_topdir in der Konfigurationsdatei `/.rpmmacros` spezifiziert.

Ergiebigkeit:
Mittel. Das Verzeichnis enthält die Source Dateien die in der %prep- Sektion der SPEC- Datei angegeben sind.

---

(6) /$HOME/<user>/myrpms/RPMS/<name>.rpm (Seite 221)

Beschreibung der Quelle:
Das RPM Paket wird im Verzeichnis `/$HOME/user/myrpms/RPMS` erzeugt. Das Verzeichnis wird durch den Parameter %_rpmdir in der Konfigurationsdatei `/.rpmmacros` spezifiziert.

Ergiebigkeit:
Mittel. Das RPM Paket enthält die aufgeführten Dateien und Verzeichnisse, die in der %files- Sektion der SPEC- Datei angegeben sind.

(7) /$HOME/<user>/myrpms/SPECS/<name>.spec (Seite 221)

Beschreibung der Quelle:
Die Paketanleitung ist eine Textdatei und wird vom Benutzer mittels einem Texteditor erstellt und dann von rpm gelesen und geparst. Das Verzeichnis wird durch den Parameter %Mittel. Das SRPM Paket enthält die aufgeführten Dateien und Verzeichnisse, die in der %files- Sektion der SPEC- Datei angegeben sind.
specdir in der Konfigurationsdatei `/.rpmmacros` spezifiziert.

Ergiebigkeit:
vgl. Seite 199 Informationsquelle (1)

(8) /$HOME/<user>/myrpms/SRPMS/<name>.spm (Seite 221)

Beschreibung der Quelle:
Das SRPM Paket wird im Verzeichnis `/$HOME/user/myrpms/SRPMS` erzeugt. Das Verzeichnis wird durch den Parameter %_srcrpmdir in der Konfigurationsdatei `/.rpmmacros` spezifiziert.

Ergiebigkeit:
Mittel. Das SRPM Paket enthält die aufgeführten Dateien und Verzeichnisse, die in der %files- Sektion der SPEC- Datei angegeben sind.

(9) Eintrag in History Liste des Benutzers (Seite 221)

Beschreibung der Quelle:
vgl. Seite 26

Ergiebigkeit:
Mittel. Die History Liste enthält folgendes Kommando: `rpm -bStadium [Optionen] <package name>.spec`

Kapitel 7

Bottom Up Sicht

Die Bottom Up Sicht enthält die Einteilung in Verzeichnisse.

7.1 Einteilung in Unterverzeichnisse

7.1.1 Konfigurationsdateien im Unterverzeichnis /etc

1. Konfigurationsdateien: statisch lesbar

Nr.	Informationsquelle	Seite	Nr.	Informationsquelle	Seite
(1)	`at.allow`	113	(17)	`nsswitch.conf`	138, 163
(2)	`at.deny`	113	(18)	`passwd`	56, 77, 78 120, 135 169, 183
(3)	`cron.allow`	115	(19)	`profile`	57
(4)	`cron.deny`	115	(20)	`protocols`	137
(5)	`filesystems`	105	(21)	`resolv.conf`	139, 151
(6)	`fstab`	103	(22)	`rpmrc`	201, 203
(7)	`hosts`	138	(23)	`securetty`	55, 134
(8)	`host.conf`	138	(24)	`services`	137
(9)	`hosts.allow`	148	(25)	`shadow`	56, 120 135, 183
(10)	`hosts.deny`	148	(26)	`shells`	56, 70, 72
(11)	`hosts.equiv`	182	(27)	`ttytype`	57
(12)	`iptos`	133	(28)	`usertty`	57
(13)	`login.defs`	54, 63, 67 76, 78, 80	(29)	`vsftpd.chroot_list`	120
(14)	`motd`	58	(30)	`vsftpd.conf`	125, 130 119
(15)	`named.conf`	152	(31)	`yp.conf`	164
(16)	`nologin`	55, 178	(32)	`/yp/DOMAINNAME`	165
(17)	`/sysconfig/ypserv`	167	(32)	`/sysconfig/ypbind`	164
(18)	`/etc/group`	79, 80	(33)	`/etc/ypserv.conf`	166

Modifikationsmöglichkeit:
Der Eigentümer der Konfigurationsdateien ist root. Deshalb kann unter normalen

Umständen nur der Superuser mit der Kennung root die Konfigurationsdatei lesen und modifizieren. Der normale Benutzer kann alle Konfigurationsdateien bis auf (25) und (30) lesen.

Auswertung:
Die statisch lesbare Konfigurationsdatei bzw. deren Informationen ist sowohl offline und online vefügbar.

Auswertungsaufwand:
Niedrig. Bei der statisch lesbaren Konfigurationsdatei wird durch die Ausführung des Vorganges nicht der Inhalt sondern das Dateiattribut "letzter Zugriff" geändert. Um an die Informationen in der Konfigurationsdatei zu kommen genügt ein einfaches Kommando, wie `cat` um den Inhalt der Konfigurationsdatei auszugeben und mit dem Kommando `stat <Konfigurationsdatei>` können, die MAC Zeiten betrachtet werden. Dabei muss die a- Time einen Zeitpunkt in der Zeitspanne haben und die beiden anderen Zeiten müssen einen Zeitpunkt vor dem Vorfall haben.

Lebensdauer:
Das Dateiattribut "letzter Zugriff" der Konfigurationsdatei kann eine sehr kurze Lebensdauer haben, falls sie von anderen Vorgängen in der Zeitspanne gelesen wird. Das Dateiattribut kann eine lange Lebensdauer haben, wenn die Konfigurationsdatei nur genau einem Vorgang gelesen wird, der aber nicht so häufig stattfindet.

Möglichkeit der Fehlinterpretation:
Möglich. Die statisch lesbaren Konfigurationsdateien können von mehreren Vorgängen gelesen und ausgewertet werden. Eine eindeutige Zuordnung der Informationsquelle mit deren Ergiebigkeit zu nur einem Vorgang ist nicht möglich.

Fälschungsmöglichkeit:
Mittel. Die Konfigurationsdatei kann nur mit root Rechten modifiziert werden. Ein Angreifer der die Konfigurationsdatei modifizieren möchte muss zuerst root Rechte erlangen.

2. Konfigurationsdateien: statisch modifizierbar

Systemdateien: statisch modifizierbar

Nr.	Informationsquelle	Seite
(1)	`/etc/auto.master`	146
(2)	`/etc/auto.<mounpoint>`	146
(3)	`/etc/crontab`	115
(4)	`/etc/exports`	142
(5)	`/etc/fstab`	61, 101, 145
(6)	`/etc/group`	73, 75, 82, 83
(7)	`/etc/gshadow`	68, 73, 75, 79, 81, 83
(8)	`/etc/mtab`	62, 102, 105, 107, 145,
(9)	`/etc/passwd`	64, 70, 72, 82, 170,
(10)	`/etc/shadow`	64, 70, 77, 78, 82, 170,
(11)	`/dev/pts/<Zahl>`	58

Modifikationsmöglichkeit:
Der Eigentümer der Konfigurationsdatei ist root. Deshalb kann unter normalen

Umständen nur der Superuser mit der Kennung root die Konfigurationsdatei modifizieren. Normale Umstände bedeuten, dass die Konfigurationsdatei die Zugriffsrechte besitzt, die das System nach der Installation hat. Ein Eintrag kann entweder durch ein privilegiertes Kommando, durch Editieren der Konfigurationsdatei oder durch Verwendung der graphischen Oberfläche YAST2, hinzugefügt bzw. entfernt werden.

Auswertung:
Die statisch modifizierbaren Konfigurationsdateien (1) - (10) bzw. deren Informationen sind sowohl offline und online vefügbar. Die Datei `/dev/pts/<Nummer der geöffneten Konsole>` ist nur online verfügbar, da der Benutzer Eigentümer der Datei wird, wenn er sich auf dem System einloggt.

Auswertungsaufwand:
Bei der statisch modifizierbaren Konfigurationsdatei muss der neue Eintrag, die neuen Einträge, das geänderte Feld oder die geänderten Felder ermittelt werden. Der Auswertungsaufwand ist dabei niedirg, falls dem Forensiker ein Backup des Systems vor dem Vorfall zur Verfügung steht, kann er die alte Konfigurationsdatei mit der aktuellen Konfigurationsdatei mit dem Kommando `diff` vergleichen und die Änderungen somit erkennen. Der Auswertungsaufwand ist dabei hoch, falls kein Back Up dem Forensiker zur Verfügung steht. Er muss dann andere Informationsquellen heranziehen und sie untereinander kombinieren, um die neu hinzugekommene Informationen in Form eines Eintrages, von Einträgen, von Felder oder eines Feldes zu erkennen. Ist beispielsweise die History Liste des Benutzers, der den Vorgang ausgeführt hat bekannt, kann der Forensiker anhand der übergebenen Parameter eines Kommandos möglicherweise die neu hinzugekommenen Informationen in der Konfigurationsdatei erkennen.

Lebensdauer:
Statisch modifizierbare Konfigurationsdateien werden durch Vorgänge modifiziert, d.h. es werden Zeilen hinzugefügt bzw. entfernt. Die Lebensdauer des Eintrages in der Konfigurationsdatei spielt nur eine Rolle, wenn das zu untersuchende System vor der Entdeckung des Vorfalls noch online war. Die Lebensdauer ist dabei kurz, falls mehrere Vorgänge, die in der Regel sehr häufig ausgeführt werden, in der Zeispanne stattgefunden haben und den Eintrag löschen bzw. ihn überschreiben. Der Eintrag in der Konfigurationsdatei hat dabei eine lange Lebensdauer, wenn das System offline ist, d.h. nach dem Vorfall sofort abgeschaltet wurde.

Möglichkeit der Fehlinterpretation:
Möglich. Mehrere Vorgänge ändern die gleiche Konfigurationsdatei ab. Es kann dadurch vorkommen, dass in der Zeitspanne des Vorfalls mehrere Vorgänge die Konfigurationsdatei modifizieren.
Folgende Fälle sind möglich:

1. Der eine Vorgang fügt einen Eintrag hinzu und der andere Vorgang löscht diesen Eintrag wieder. Die Konfigurationsdatei hat dann andere MAC Zeiten, aber es können keine Informationen aus der Konfigurationsdatei gewonnen werden.

2. Der eine Vorgang fügt einen Eintrag hinzu und der andere Vorgang überschreibt diesen mit einem neuen Eintrag. Der Eintrag kann nicht eindeutig einem Vorgang in der Zeitspanne zu geordnet werden. Die gewonnene Information ist deshalb bei der Top Down Analyse unbrauchbar. Da der Eintrag in der Konfigurationsdatei mit der dazugehörigen Ergiebigkeit unbedingt exi-

stent sein muss, um die Aussage “Der Vorgang hat mit 100 % Wahrscheinlichkeit stattgefunden”, treffen zu können.

3. Der eine Vorgang fügt einen Eintrag hinzu und der andere Vorgang fügt auch einen Eintrag hinzu. In der Zeitspanne haben mehrere Vorgänge stattgefunden.

Fälschungsmöglichkeit:
Mittel. Die Konfigurationsdatei kann nur mit root Rechten modifiziert werden. Ein Angreifer der die Konfigurationsdatei modifizieren möchte, muss zuerst root Rechte erlangen. Der Angreifer editiert dann die Konfigurationsdatei und fügt eine Zeile hinzu oder entfernt eine Zeile. Da die viele von diesen Konfigurationsdateien Parameter mit Werten enthalten, könnte ein Angreifer beispielsweise die Parameter von bestimmten Konfigurationsdateien ändern, so dass das System nicht mehr ordnungsgemäß arbeitet bzw. abstürtzt. Ein Angreifer wird beispielsweise die Crontab Datei so abändern, falls sie Logdateien verwaltet, dass kein Back Up von den Log Dateien gemacht wird, um so seine Spuren zu verwischen.

7.1.1.1 Konfigurationsdateien und Schlüsseldateien von ssh und sshd im Unterverzeichnis /ssh

1. Konfigurationsdateien: statisch lesbar

Nr.	Informationsquelle	Seite
(1)	`environment`	184
(2)	`moduli`	180
(3)	`shosts.equiv`	183
(4)	`ssh_config`	175
(5)	`ssh_known_hosts`	181
(6)	`sshd_config`	179
(7)	`sshrc`	181
(8)	`ssh_host_key.pub`	180
(9)	`ssh_host_rsa_key.pub`	180
(10)	`ssh_host_dsa_key.pub`	180
(11)	`ssh_host_key`	181
(12)	`ssh_host_rsa_key`	181
(13)	`ssh_host_dsa_key`	181

Modifikationsmöglichkeit:
Der Eigentümer der Schlüsseldateien (8) - (13) und der Konfigurationsdateien (1) - (7) ist root. Deshalb kann unter normalen Umständen nur der Superuser mit der Kennung root die Schlüsseldatei löschen. Der normale Benutzer ist nicht dazuberechtigt die Dateien (2), (11), (12) und (13) zu lesen. Die anderen aufgeführten Dateien können von ihm gelesen werden.

Auswertung:
Der Schlüsseldateien und die Konfigurationsdateien von ssh und sshd sind online und offline verfügbar.

Auswertungsaufwand:
Der Inhalt der Schlüsseldatei kann mit dem Kommando `cat` angezeigt werden.

Der Forensiker muss feststellen, ob auf die Schlüsseldateien in der Zeitspanne des Vorfalls nur gelesen wurden. Er kann dies mit dem Kommando `stat <Schlüsseldatei>`, wobei der Zeitstempel atime einer der öffentlichen Schlüsseldateien (.pub) auf einen Zeitpunkt in der Zeitspanne des Vorfalls gesetzt sein muss, herrausfinden.

Lebensdauer:
Das Dateiattribut "letzter Zugriff" der Konfigurationsdateien kann eine sehr kurze Lebensdauer haben, falls die Dateien von anderen Vorgängen, die mit ssh zu tun haben, in der Zeitspanne gelesen werden. Das Dateiattribut der Schlüsseldateien hat eine lange Lebensdauer, da die Schlüsseldateien nur von einem Vorgang gelesen werden. Die Konfigurationsdateien haben nur dann eine Lebensdauer, wenn sich in der Zeitspanne des Vorfalls sich keine weiteren Benutzer mit ssh am Rechner angemeldet haben.

Möglichkeit der Fehlinterpretation:
Eine Fehlinterpretation ist nicht möglich, da die Schlüsseldateien nur von dem Vorgang: *Benutzer meldet sich an* gelesen und ausgewertet werden. Bei den Konfigurationsdateien ist eine Fehlinterpretation schon möglich, da die Konfigurationsdateien von sshd und sshd von zwei Vorgängen gelesen werden und so das Dateiattribut "letzter Zugriff" ändern.

Fälschungsmöglichkeit:
Mittel. Die Konfigurationsdateien und Schlüsseldateien können nur mit root Rechten modifiziert werden. Ein Modifizieren dieser Dateien ist für Angreifer untypisch. Ein Angreifer entfernt höchstens die Schlüssel und fügt seine eigenen Schlüssel hinzu, um sich ein Backdoor für einen späteren Zugriff auf dem Rechner zu verschaffen. Die Datei `moduli` enthält die Diffie- Hellmann Gruppen für den Austausch des Diffie- Hellmann Schlüssels. Der Austausch erfolgt am Anfang der SSH Sitzung und es wird ein gemeinsam genutzter, geheimer Wert erstellt, der von keiner Seite allein erstellt werden kann. Dieser Wert wird zur Host Authentifizierung verwendet. Ein Angreifer könnte dementsprechend den Austausch des Schlüssels zweier ihm bekannter Rechner abwarten und versuchen dann an diesen geheimen Wert zukommen und sich gegenüber eines der beiden Rechner als falschen Host auszugeben. Der Angreifer muss dabei schon auf einem anderen Rechner eingebrochen sein, um so dann an den Wert zu kommen.

7.1.2 Dateien im Verzeichnis /var

7.1.2.1 Log- Dateien im Unterverzeichnis /log

1. Konfigurationsdateien: statisch modifizierbar

Nr.	Informationsquelle	Seite
(1)	`lastlog`	59, 140, 186
(2)	`messages`	66, 68, 71, 74, 75, 77, 80, 81, 82, 84, 123, 140, 147, 185, 189, 150
(3)	`rpc.yppasswd`	171
(4)	`utmp`	60, 140, 185
(5)	`vsftpd.log`	122, 126, 131
(6)	`wtmp`	59, 140, 185, 186

Modifikationsmöglichkeit:
Der Eigentümer der Log- Datei ist root. Deshalb kann unter normalen Umständen nur der Superuser mit der Kennung root die Konfigurationsdatei modifizieren. Die `/var/log/messages` kann von einem normalen Benutzer nicht gelesen werden. Der normale Benutzer ist aber dazuberechtigt einige Informationen, die in drei Log- Dateien `/var/log/lastlog, /var/log/utmp und /var/log/wtmp` mit den Kommandos `lastlog, w, und last` auszugeben.

Auswertung:

- Die Log- Dateien (2), (3) und (5) bzw. deren Informationen sind sowohl offline und online verfügbar.
- Die Log- Dateien (1), (4) und (6) bzw. deren Informationen sind nur online verfügbar. Da durch das Herunterfahren des Systems mit `shutdown` die Strukturelemente der wtmp Datei `ut_name` den Wert shutdown und `ut_line` den Wert annehmen, ist die wtmp Datei nur online verfügbar. Wird jedoch das System mit der Option `-d` heruntergefahren, bleiben die Informationen der wtmp Datei erhalten.

Lebensdauer:

- Der Eintrag der Log- Dateien (2), (3) und (5) bleibt solange bestehen bis die Datei eine maximale Größe erreicht hat. Die maximale Größe wird in `/etc/login.defs` durch die Environment Parameter ULIMIT beschränkt. Die maximale Größe ist 2 GB. Das entspricht 4194303 Blöcke zu 512 Byte. Die Lebensdauer kann auch abhängig von der Konfiguration vom cron Dämon und logrotate sein (siehe Seite 239).
- Wird das System ausgeschaltet, sind die Informationen aus den Log- Dateien `/var/log/wtmp und /var/log/utmp` nicht mehr da. Je nachdem wann das System ausgeschaltet wird, ergibt sich eine Lebensdauer zwischen den Werten kurz und lang.

Auswertungsaufwand:

- Bei den Log- Dateien (2), (3)und (5) ist der Aufwertungsaufwand niedrig, da immer ein Eintrag am Ende der Datei angefügt wird. Dadurch, dass der genaue Zeitpunkt der Ausführung des Vorganges, auch in der Log- Datei protokolliert wird, kann der jeweilige Eintrag durch die Zeitangabe erkannt werden. Unter normalen Umständen, d.h. die Log- Datei wurde nicht durch Dritte modifiziert, wird der Eintrag gewählt, deren Zeitangabe in der Zeitspanne des Vorfalls liegt.
  Mit den folgenden Kommandos können jeweils die Informationen der Log-Datei ausgegeben werden und dann der richtige Eintrag ausgewählt werden:
  (1) `lastlog`
  (2) `tail -f /var/log/messages`
  (3) `tail -f /var/log/rpc.yppasswd`
  (4) `w, who`
  (5) `tail -f /var/log/vsftpd.log`
  (6) `last`

- Bei den anderen Log- Dateien (1), (4) und (6) kann der jeweilige Eintrag nicht einfach ermittelt werden, da es sich bei den Log- Dateien um Binärdateien handelt. Der Aufwertungsaufwand ist dabei niedrig, falls ein Back Up der Log- Dateien vor dem Vorfall vorliegt. Der Forensiker kann dann die neuen Einträge jeweils ermitteln, in dem er jeweils das Backup der Log- Datei mit dem jeweiligen Kommando in eine Textdatei umleitet, d.h. die Ausgabe auf den Bildschirm wird in eine Datei umgeleitet. Und dann die sichergestellte Log- Datei in das Verzeichnis kopiert, wo sie sich normalerweise befindet und wiederum das Kommando aufruft, um den Inhalt der Log- Datei anzuzeigen und leitet diese wiederum in eine Textdatei, um dann mit dem Kommando `diff` die neuen Einträge zu ermitteln. Folgende Kommandos werden verwendet:
  - Der Inhalt der lastlog Datei eines offline sichergestellten Systems kann mit dem Kommando lastlog -f /<sichergestellte lastlog> angezeigt werden. Da das `lastlog`, `w`, `who` und das `last` Kommando auch auf die Passwortdatei zugreift, muss auf dem Rechner, der das Dateisystem des sichergestellten Systems untersucht, die Passwortdatei durch die Passwortdatei des zu untersuchenden Dateisystems ausgetauscht werden.
    * lastlog -f <sichergestellte lastlog Datei> > file1.txt
    * lastlog <Back Up der lastlog Datei> > file2.txt
    * `diff file1.txt file2.txt`
  - Für die utmp und wtmp Datei:
    * w und who, last /var/log/<sichergestellte utmp Datei>, /<sichergestellte wtmp Datei> > file1.txt
    * w und who, last /var/log/<Back Up der utmp Datei>, <Back Up der wtmp Datei> > file2.txt
    * `diff file1.txt file2.txt`

Möglichkeit der Fehlinterpretation:

- Die Log- Dateien (2), (3) und (5) werden von mehreren Vorgängen modifiziert. Jedoch ist die jeweilige Ergiebigkeit der Einträge der verschiedenen Vorgänge unterschiedlich. Deshalb ist eine Fehlinterpretation hier unmöglich. Das Ausführen von priveligierten Kommandos und die Verwendung der graphischen Oberfläche YAST2 hinterlassen die gleichen Eintrag in Log- Datei.

- Bei den Log- Dateien (4), und (6) ist es möglich die Informationen aus der Informationsquelle (Eintrag) falsch zu deuten, da andere Vorgänge die Log- Dateien auch modifizieren bzw. die neuen Einträge die gleiche Ergiebigkeit haben.
  Durch ein `reboot` oder das Ändern der Systemzeit mit `date` können spezielle Einträge in der wtmp Datei geändert werden. Beim Aufruf von Telnetd, xterm, getty oder init können die Einträge in der utmp Datei verändern.
  Telnetd schreibt einen LOGIN_PROCESS Eintrag für die Parameter ut_type und überlässt das den Rest dem Login Programm. `Xterm` und andere Terminal Emulatoren erzeugen einen USER_PROCESS Eintrag für die Parameter ut_type und einen Eintrag für die Parameter ut_id mit den letzten beiden Buchstaben von `/dev/ttyp%c` oder mit `/dev/pts/%d`. Init findet einen Prozess der bald beendet wird, lokalisiert seinen utmp Eintrag mittels der ut_id, setzt die Parameter `ut_type` auf DEAD_PROCESS und die Parametern ut_user, ut_time und ut_host auf Null Bytes. `Getty` lokalisiert den Eintrag in der utmp Datei mittels der pid und setzt die Parameter ut_type auf LOGIN_PROCESS, verändert ut_time und setzt ut_line neu. `Login` setzt nach erfolgreicher Authentifikation des Benutzers den Wert der Parametern ut_type auf USER_PROCESS , ändert ut_time und setzt ut_host und ut_addr neu.

Fälschungsmöglichkeit:
Die Einträge in den Log- Dateien (1), (4) und (6) können mit root Rechten entfernt werden, d.h. der Benutzer öffnet die Datei mit einem Texteditor und entfernt die entsprechenden Zeilen. Der Aufwertungsaufwand wird somit als mittelhoch eingeschätzt. Die Log- Datei wird dabei direkt abgeändert. Der Aufwertungsaufwand diese drei Log- Dateien zu ändern ist hoch, wenn der syslogd Dämon so abgeändert wird, dass er bestimmte Protokollereignisse nicht mehr meldet. Änderungen am syslog Quellcode können nur mit Kenntnissen eines erfahrenen Systemprogrammierers vorgenommen werden. Die Informationen (Strukturelemente) der Log- Dateien (2), (3) und (5) können nur mit Kenntnissen eines erfahrenen Programmieres im Bereich der Systemprogrammierung geändert werden, d.h. es müssen die Kommandos, die die Anmeldungen verzeichnen, wie login, in.telnetd, sshd, manipuliert werden. Entweder erstellen sie dann keine Einträge in der Log- Datei oder ignorieren die von ihnen vorgegebenen Kriterien. Außerdem können die einzelnen Struktureelemente mit Kommandos aus dem Bereich der Systemprogrammierung explizit gesetzt werden. Im Internet sind Programme verfügbar, die Einträge aus den Log- Dateien löschen können. Um die Informationen aus der wtmp Datei zu entfernen gibt es beispielsweise das Programm `wted` und um die Einträge in der lastlog Datei zu entfernen gibt es beispielsweise das Programm `zz`. Diese Programme können auch von einem normalen Benutzer ohne tiefgreifende Systemkenntnisse verwendet werden.

7.1.2.2 Zonen- Dateien im Unterverzeichnis /named

1. Zonen- Dateien: statisch lesbar

Nr.	Informationsquelle	Seite
(1)	`<domain>.zone`	155
(2)	`<umgekehrter Netzanteil>.zone`	156

Modifikationsmöglichkeit:
Der Eigentümer der beiden Zonen- Dateien ist root. Dieser kann die Zonen-

Dateien direkt durch Editieren modifizieren. Der normale Benutzer kann beide Zonendateien auch lesen.

Auswertung:
Die Zonen- Dateien sind offline und online verfügbar. Sie liegen auf dem DNS Server.

Auswertungsuafwand:
Niedrig. Bei der statisch lesbaren Zonendatei wird durch die Ausführung des Vorganges nicht der Inhalt sondern das Dateiattribut "letzter Zugriff" geändert. Um an die Informationen in der Zonendatei zu kommen genügt ein einfaches Kommando, wie `cat` um den Inhalt auszugeben. Mit dem Kommando `stat <Konfigurationsdatei>` können, die MAC Zeiten betrachtet werden. Dabei muss die a- Time einen Zeitpunkt in der Zeitspanne haben und die beiden anderen Zeiten müssen einen Zeitpunkt vor dem Vorfall haben.

Lebensdauer:
Die Zonendateien können eine sehr kurze Lebensdauer haben, falls sie von anderen Vorgängen in der Zeitspanne gelesen oder modifiziert werden. Die Zonendateien können eine lange Lebensdauer haben, wenn sie nur genau von einem Vorgang gelesen werden, der aber nicht so häufig stattfindet.

Möglichkeit der Fehlinterpretation:
Eine Fehlinterpretation ist nicht möglich, da das Dateiattribut "letzter Zugriff" nur von dem Vorgang: *Benutzer stellt DNS Anfrage* geändert wird.

Fälschungsmöglichkeit:
Mittel. Die Zonendatei kann nur mit root Rechten modifiziert werden. Ein Angreifer wird für den Angriff auf einen DNS Server seine Zonen- Dateien modifizieren, sondern dem DNS Cache des DNS Servers mit verfälschten Einträgen füllen. Die Clients werden dann den Rechner des Angreifers verwiesen. Der Angreifer versucht dann alle Passwörter oder andere sensible Daten der angeschlossenen Clients abzugreifen.

2. Zonen- Dateien: statisch modifizierbar

Die zwei Dateien sind auch modifizierbar. In der Arbeit wurde jedoch kein Vorgang bei DNS angegeben, der die beiden Dateien ändert. Ein möglicher Vorgang wäre: "Root ändert Address-Mapping".

7.1.2.3 Benutzerspezifische Dateien im Unterverzeichnis /spool

1. Dateien: statisch modifizierbar

Nr.	Informationsquelle	Seite
(1)	`atjobs`	113
(2)	`/mail/<user>`	114
(3)	`/cron/tabs/<user>`	116

Modifikationsmöglichkeit:
Eigentümer der Datei `/var/mail/<user>` ist der jenige Benutzer den jeweiligen Benutzernamen hat und nur er hat unter normalen Umständen Lese- und Schreibrecht. Nur der Eigentümer selbst und der Superuser mit der Kennung root können deshalb die Datei modifizieren und sie löschen. Das Verzeichnis `/var/spool/atjobs`

gehört dem at Dämon. Das Verzeichnis enthält die At- Jobs von jedem Benutzer. Jeder At- Job in `/var/spool/atjobs` kann nur von dem Benutzer modifiziert und gelöscht werden, der in auch initiiert hat. Da der Superuser volle Zugriffsberechtigungen auf das System hat, kann er alle At- Jobs modifizieren und löschen mittels dem Kommando `atrm <job number>`. Der Eigentümer von dem Verzeichnis `/var/spool/cron/tabs` ist root. In diesem Verzeichnis hat dann jeder Benutzer eine eigene Crontab, von der nur er der Eigentümer ist und kann sie mit `crontab -e` editieren und löschen. Eine direkte Änderung der Crontab ist nicht möglich. Nur root und der Eigentümer, der Crontab können den Inhalt der Crontab einsehen.

Auswertung:
Alle drei Dateien sind sowohl online und offline verfügbar.

Auswertungsaufwand:
Die At- Job Datei entsteht nach dem Aufruf des at Dämon neu. Deshalb muss kein neu hinzugekommener Eintrag ermittelt werden. Mit dem Kommando `cat` kann der Inhalt der Datei ausgegeben werden. Der Aufwertungsaufwand ist somit gering. Der Inhalt der Warteschlange kann mit `atq` angezeigt werden. Die neue Mail (Einträge) in der Datei `/var/spool/mail/<user>` kann durch die Zeile `received: from <anderer Rechner> by <lokaler Rechner> for <e- mail Adresse des Eigentümers der Mail Datei>, Zeitpunkt` erkannt werden. Der Zeitpunkt als die Mail eintraf muss in der Zeitspanne des Vorfalls liegen. Der Aufwertungsaufwand bei beiden Dateien ist somit gering. Steht dem Forensiker ein Backup der Crontab vor dem Vorfall zur Verfügung, kann er die alte Crontab mit der aktuellen Crontab mit dem Kommando `diff` vergleichen und die Änderungen somit erkennen. Der Aufwand ist somit niedrig. Der Aufwertungsaufwand ist hoch, falls kein Back Up dem Forensiker zur Verfügung steht. Er muss dann andere Informationsquellen heranziehen und sie untereinander kombinieren, um die neu hinzugekommene Informationen in Form eines Eintrages, von Einträgen, von Felder oder eines Feldes zu erkennen. Mit dem Kommando `crontab -u <user> -l` kann der Forensiker die crontab von jedem Benutzer ansehen.

Lebensdauer:
Die At- Job Datei ist nur solange vorhanden bis der Zeitpunkt eintritt, an dem der Job ausgeführt werden soll. Je nach Zeitangabe und aktueller Zeit kann die sich die Lebensdauer zwischen den Werten kurz und lang bewegen. Die Crontab bzw. der Eintrag ist solange existent, bis er entweder mit `crontab -r` gelöscht wird oder mit `crontab -r <filename>` durch ein neue Tabelle ersetzt wird. Die Mail Datei `/var/spool/mail/<user>` wird immer um Einträge ergänzt, die am Dateiende angefügt werden. Der Eintrag (Mail) bleibt solange bestehen bis die Datei eine maximale Größe erreicht hat. Die maximale Größe wird in `/etc/login.defs` durch die Environment Parameter ULIMIT beschränkt. Die maximale Größe ist 2GB. Das entspricht 4194303 Blöcke zu 512 Byte. Die Lebensdauer kann auch abhängig von der Konfiguration vom *cron Dämon* und *logrotate* sein.

Möglichkeit der Fehlinterpretation:
Eine Fehlinterpretation ist bei diesen Dateien nicht möglich. Die Dateien werden jeweils nur durch einen Vorgang modifiziert.

Fälschungsmöglichkeit:
Mittel. Um die Datei modifizieren zu können, sind root Rechte nötig. Es können dann Einträge entfernt bzw. hinzugefügt werden, um beispielsweise Spuren eines Angriffes zu verwischen. Durch Änderung der Systemzeit mit dem Kommando `date` oder mit Datum & Zeit des KDE- Kontrollbereiches ist ein Vordatieren der At- Jobs

möglich. Es ist dadurch möglich einen Job, der nur zu einem bestimmten Zeitpunkt eines Benutzers starten soll, früher zu starten. Um das Kommando `date` ausführen zu können, muss man Root Rechte besitzen.

7.1.2.4 Dateien im Unterverzeichnis /lib

1. Konfigurationsdateien: statisch modifizierbar

Nr.	Informationsquelle	Seite
(1)	`/nfs/etab`	143, 148
(2)	`/nfs/rmtab`	148
(3)	`/nfs/xtab`	143, 148
(4)	`/rpm/conflictsindex.rpm`	190, 196
(5)	`/rpm/fileindex.rpm`	190, 196
(6)	`/rpm/groupindex.rpm`	190, 196
(7)	`/rpm/nameindex.rpm`	190, 196
(8)	`/rpm/packages.rpm`	190, 196
(9)	`/rpm/providesindex.rpm`	190, 196
(10)	`/rpm/requiredby.rpm`	190, 196

Modifikationsmöglichkeit:
Der Eigentümer der Dateien ist root. Deshalb kann unter normalen Umständen nur der Superuser mit der Kennung root die Datei modifizieren. Ein Eintrag kann entweder durch ein privilegiertes Kommando, durch Editieren der Datei oder durch Verwendung der graphischen Oberfläche YAST2, hinzugefügt bzw. entfernt werden. Die RPM Datenbank, die sich aus den Dateien (4) - (10) zusammensetzt gehört root. Die RPM Datenbank kann nicht direkt durch Editieren mit vi modifiziert werden, da die RPM Dateien keine Textdateien sind.

Auswertung:
Die Dateien (2) - (3) sind nur online verfügbar, da sie Informationen über gerade remote gemountete Dateisysteme von Clients enthalten. Die Datei (1) ist online und offline verfügbar, das sie nur Informationen über alle exportierenden Dateisystemen des Servers enthalten. Die RPM Datenbank ist online und offline verfügbar.

Auswertungsaufwand:
Der Inhalt der Dateien (2) - (3) kann online mit dem Kommando `vi <filename>` angezeigt werden. Der richtige Eintrag bei (1) kann durch Abgleich mit der Datei `/etc/exports` ermittelt werden. In der RPM Datenbank wird der Zeitpunkt der Installation, Namen der im Paket enthaltenen Dateien, die Größe jeder Datei, Zeitstempel, MD5 Prüfsumme zum Zeitpunkt der Installation abgespeichert. Mit dem Kommando `rpm -verify <package name> oder -all` können diese Informationen zu einem späteren Zeitpunkt überprüft werden. Das Kommando zeigt dann alle Dateien an, die seit ihrer Installation modifiziert wurden. Mit dem Kommando `rpm -qa` können alle installierten Paket angezeigt werden.

Lebensdauer:
Statisch modifizierbar Dateien werden durch Vorgänge modifiziert, d.h. es werden Einträge hinzugefügt bzw. entfernt. Die Lebensdauer des Eintrages in der Datei spielt nur eine Rolle, wenn das zu untersuchende System vor der Entdeckung des Vorfalls noch online war. Die Lebensdauer ist dabei kurz, falls mehrere Vorgänge, die in der Regel sehr häufig ausgeführt werden, in der Zeispanne stattgefunden

haben und den Eintrag löschen bzw. ihn überschreiben. Der Eintrag in der Datei hat dabei einelange Lebensdauer, wenn das System offline ist, d.h. nach dem Vorfall sofort abgeschaltet wurde.

Möglichkeit der Fehlinterpretation:
Eine Fehlinterpretation ist bei der RPM Datenbank nicht möglich, da sie nur durch Vorgänge die sich auf RPM beziehen, modifiziert wird.
Bei den Dateien (1) - (3) ist es wiederum auch nicht möglich die Informationen falsch zu deuten, da sie nur durch Vorgänge, die nur NFS betreffen, modifiziert werden.

Fälschungsmöglichkeit:
Die Dateien (1) - (3) können mit root Rechten gefälscht werden, d.h. Einträge gelöscht bzw. hinzugefügt werden. Der Aufwand diese Dateien zu ändern ist mittelhoch, da es sich um Textdateien handelt und durch Editieren gefälscht werden können. Ein Angreifer kann beispielsweise durch das Kommando `showmount -e <NFS Server>` einen NFS- Server abfragen, um zu erfahren, welche Dateisysteme mit welchen Optionen exportiert werden und welche Computer die Dateisysteme momentan gemountet haben. Wo der Host dann das Dateisystem gemountet hat wird nicht angezeigt, oft handelt es sich um das gleiche Verzeichnis. Der Angreifer kann dadurch an folgendeInformationen gelangen: Hostnamen, Benutzernamen aus dem Namen der Partitionen `/home/username`. Die Benutzernamen können dann späterverwendet werden, wenn es darum geht die Passwörter für das Netzwerk zu knacken. Softwarepakete, die am NFS Server installiert sind und vom Client gemountet werden, können trojanisiert sein. Dann ist nicht nur der Server kompromittiert, sondern alle Clients, die die installierten Softwarepakete mounten. Dieses Beispiel zeigt, dass der Angreifer keine Informationen aus den Dateien fälscht, sondern einfach nur durch diese Dateien an sensible Informationen gelangt, um einen Angriff zu starten.
Die RPM Datenbank wird selbst nicht gefälscht, sondern die Pakete deren Informationen in RPM Datenbank stehen. Zum Beispiel tauscht ein Angreifer wichtige Systemdateien `/bin/ls` oder `/usr/sbin/sshd` durch seine eigene Variante aus.

2. Konfigurationsdateien: statisch lesbar

Nr.	Informationsquelle	Seite
(1)	`/rpm/conflictsindex.rpm`	192, 194
(2)	`/rpm/fileindex.rpm`	192, 194
(3)	`/rpm/groupindex.rpm`	192, 194
(4)	`/rpm/nameindex.rpm`	192, 194
(5)	`/rpm/packages.rpm`	192, 194
(6)	`/rpm/providesindex.rpm`	192, 194
(7)	`/rpm/requiredby.rpm`	192, 194

Modifikationsmöglichkeit:
Die RPM Datenbank, die sich aus den Dateien (1) - (7) zusammensetzt gehört root. Die RPM Datenbank kann nicht direkt durch Editieren mit `vi` modifiziert werden, da die RPM Dateien keine Textdateien sind. Der normale Benutzer hat das Leserecht auf die RPM Datenbank, da er das Kommando `rpm` aufrufen darf.

Auswertung:
Die RPM Datenbank ist online und offline verfügbar.

Auswertungsaufwand:
In der RPM Datenbank wird der Zeitpunkt der Installation, Namen der im Paket enthaltenen Dateien, die Größe jeder Datei, Zeitstempel, MD5 Prüfsumme zum Zeitpunkt der Installation abgespeichert. Mit dem Kommando `rpm -verify <package name> oder -all` können diese Informationen zu einem späteren Zeitpunkt überprüft werden. Das Kommando zeigt dann alle Dateien an, die seit ihrer Installation modifiziertwurden. Mit dem Kommando `rpm -qa` können alle installierten Paket angezeigt werden. Mit dem Kommando `stat <RPM Datenbankdatei>` können die MAC Zeiten betrachtet werden. Dabei muss die a-Time einen Zeitpunkt in der Zeitspanne haben und die beiden anderen Zeiten müssen einen Zeitpunkt vor dem Vorfall haben.

Lebensdauer:
Die Lebensdauer der Informationen der RPM Datenbank hängen davon ab, wann ein Paket installiert, deinstalliert oder ein RPM Paket verifiziert wird. Das Dateiattribut "letzter Zugriff" hat eine lange Lebensdauer, wenn das zu untersuchende System nach dem Vorfall sofort heruntergefahren wurde.

Möglichkeit der Fehlinterpretation:
Eine Fehlinterpretation ist bei der RPM Datenbank nicht möglich, da sie nur gelesen und ausgewertet wird.

Fälschungsmöglichkeit:
Die RPM Datenbank wird selbst nicht gefälscht, sondern die Pakete deren Informationen in RPM Datenbank stehen. Zum Beispiel tauscht ein Angreifer wichtige Systemdateien `/bin/ls` oder `/usr/sbin/sshd` durch seine eigene Variante aus. Das Dateiattribut der RPM Datenbank wird durch das austauschen der Systemdateien nicht geändert.

7.1.2.5 NIS- Dateien im Unterverzeichnis /yp

1. Konfigurationsdateien und NIS Maps: statisch lesbar

Nr.	Informationsquelle	Seite
(1)	`/<nisdomain>/passwd.byname`	136
(2)	`/<nisdomain>/passwd.uid`	136
(3)	`/<nisdomain>/shadow.byname`	136
(4)	`nicknames`	167
(5)	`securenets`	167
(5)	`ypservers`	160
(6)	`Makefile`	160
(7)	`<mapname>`	168

Modifikationsmöglichkeit:
Der Eigentümer der NIS Maps und der Konfigurationsdateien von NIS ist root. Die Konfigurationsdateien (4) - (5) können direkt mit root Rechten modifiziert werden. Die NIS Maps (1) - (3) können nur mit root Rechten indirekt geändert werden, d.h. die lokale Datei (`z.B. /etc/passwd`) aus der die Map generiert wird, muss modifiziert werden. Bei den lokalen Dateien werden entweder Einträge gelöscht oder hinzugefügt. Das Modifizieren der lokalen Datei verlangt auch root Rechte. Danach muss das Script `/var/yp/Makefile -C` aufgerufen werden, um die Map (Passwd Map) neu zu generieren. Der normale Benutzer kann die Konfigurationsdateien von

NIS und die NIS Maps lesen.

Auswertung:
Die NIS- Dateien (4), (5) und (6) ist online und offline verfügbar. Die NIS Maps werden auf dem Client, der sie anfordert nicht gespeichert, sondern nur auf der Standardausgabe ausgegeben. Die NIS Maps sind deshalb nur auf dem NIS Server verfügbar und dort online und offline.

Auswertungsaufwand:
Bei den statisch lesbaren NIS Dateien müssen keine neu kinzugekommenen Einträge ermittelt werden. Es wird nur dasDateiattribut "letzter Zugriff" geändert. Deshalb ist der Aufwertungsaufwand niedrig. Mit dem Kommando `ypcat -k -d <nisdomainname> -h <Rechner> Mapname` kann eine bestimmte Map der NIS Domäne abgefragt werden. Das Kommando `ypmatch -x oder ypcat -x` geben alle verfügbaren NIS Maps des NIS Servers aus. Mit dem Kommando `stat <NIS Mapp oder Konfigurationsdatei von NIS>` können, die MAC Zeiten betrachtet werden. Dabei muss die a- Time einen Zeitpunkt in der Zeitspanne haben und die beiden anderen Zeiten müssen einen Zeitpunkt vor dem Vorfall haben.

Lebensdauer:
Die NIS- Dateien können eine sehr kurze Lebensdauer haben, falls sie von anderen Vorgängen in der Zeitspanne gelesen oder modifiziert werden. Die NIS-Dateien können eine lange Lebensdauer haben, wenn sie nur genau von einem Vorgang gelesen werden, der aber nicht so häufig stattfindet. Wird das System nach dem Vorfall sofort heruntergefahren, haben die NIS Dateien eine lange Lebensdauer.

Möglichkeit der Fehlinterpretation:
Möglich. NIS Map könnnen jenach Konfiguration von mehreren Vorgängen gelesen, unteranderem von den Vorgängen `Benutzer loggt sich remote beim Host` und *Benutzer meldet sich mittels ssh an* und so das Dateiattribut "letzter Zugriff" geändert werden.

Fälschungsmöglichkeit:
Mittel. Die NIS spezifischen Dateien können nur mit root Rechten modifiziert werden. Ein Angreifer, der diese modifizieren möchte, muss zuerst root Rechte erlangen.

2. Konfigurationsdateien: statisch modifizierbar

Nr.	Informationsquelle	Seite
(1)	`/<nisdomainname>/<mapname>`	158
(2)	`/<nisdomainname>/passwd.<key>`	170
(3)	`/<nisdomainname>/shadow.byname`	171
(4)	`/binding/<domainname>.<version>`	165
(5)	`nicknames`	158
(6)	`Makefile`	161

Modifikationsmöglichkeit:
Der Eigentümer der NIS Maps und der Konfigurationsdateien von NIS ist root. Die Konfigurationsdateien (4) - (6) können direkt mit root Rechten modifiziert werden. Die NIS Maps (1) - (3) können nur mit root Rechten indirekt geändert werden, d.h. die lokale Datei (`z.B. /etc/passwd`) aus der die Map generiert wird, muss

modifiziert werden. Bei den lokalen Dateien werden entweder Einträge gelöscht oder hinzugefügt. Das Modifizieren der lokalen Datei verlangt auch root Rechte. Danach muss das Script /var/yp/Makefile -C aufgerufen werden, um die Map (Passwd Map) neu zu generieren.

Auswertung:
Die NIS- Dateien (4), (5) und (6) und somit das Dateiattribut "letzter Zugriff" sind online und offline verfügbar. Die NIS Maps werden auf dem Client, der sie anfordert nicht gespeichert, sondern nur auf der Standardausgabe ausgegeben. Die NIS Maps sind deshalb nur auf dem NIS Server verfügbar und dort online und offline.

Auswertungsaufwand:
Bei den statisch modifizierbaren NIS Dateien müssen die neu hinzugekommenen Einträge ermittelt werden. Neu hinzugekommene Einträge in der NIS Map können mit niedrigem Aufwand ermittelt werden. Dabei muss ein Back Up von den NIS Maps vor dem Vorfall vorliegen. Die Informationen der aktuellen Map werden dann mit dem Kommando ypcat <mapname> | current.map in die Datei current map geschrieben. Damit man die Informationen aus dem Back Up Map bekommt, muss die Map in den Ordner /var/yp/<nisdomainname>/ an die Stelle, wo sich die aktuelle Map befindet kopiert werden, um ypcat verwenden zu können. Genauso wird dann auch das Back Up der Map mit ypcat <mapname> | backup.map behandelt. Zuletzt kann dann der neu hinzugekommene Eintrag mit dem Kommando diff current.map backup.map ermittelt werden.

Lebensdauer:
Die Lebensdauer des Eintrages in der NIS Map spielt nur eine Rolle, wenn das zu untersuchende System vor der Entdeckung des Vorfalls noch online war und wann das System ohne einer Online Analyse abgeschalten wurde bzw. vom Netz entfernt wurde. Die Lebensdauer ist dabei kurz, falls die Maps sehr häufig neu generiert werden. Die Map hat dabei eine lange Lebensdauer, falls das System nach der entdeckung des Vorfalls sofort abgeschaltet wurde.

Möglichkeit der Fehlinterpretation:
Unmöglich. Die sechs Dateien werden nur durch den Vorgang: *Benutzer erzeugt Map* modifiziert.

Fälschungsmöglichkeit:
Mittel. Die NIS spezifischen Dateien können nur mit root Rechten modifiziert werden. Ein Angreifer, der diese modifizieren möchte, muss zuerst root Rechte erlangen. Der Angreifer kann nach einem erfolgereichen DoS auf einen NIS-Master- Server dann gefälschte Benutzerkonten (passwd Map) verteilen oder neue Vertrauenseinstellungen aufbauen, um sich weitere Zugangsmöglichkeiten zu verschaffen. Er könnte beispielsweise alle vorhandenen Benutzerkonten mit ypcat passwd.byname > /var/yp/<nisdomainname>/passwd.byname kopieren und ein root äquivalentes Konto ohne Passwort (keine Shadow Map) zu /var/yp/<nisdomainname>/passwd.byname hinzufügen. Fazit ist, dass alle NIS Clients im Netzwerk die Anmeldung ohne Passwort zulassen.

7.1.3 Benutzerspezifische Dateien im Unterverzeichnis /home/<user>

1. Datei: statisch lesbar

Nr.	Informationsquelle	Seite
(1)	/$HOME/.rhosts	182
(2)	/$HOME/.rpmmacros	203
(3)	/$HOME/.rpmrc	201, 203
(4)	/$HOME/.shosts	182
(5)	/$HOME/.profile	58
(6)	/$HOME/.cshrc	58
(7)	/$HOME/.login	58

Modifikationsmöglichkeit:
Der Eigentümer der Datei ist der Benutzer der im Pfad /home/<user> angegegeben ist. Die Datei kann deshalb vom Eigentümer und root modifiziert oder gelöscht werden. Jenach dem wie die Zugriffsrechte der Datei für die Gruppe und Andere gesetzt, können diese mit vergebenen Schreibrecht die Datei auch ändern. Eine Schreibrecht impliziert auch unter normalen Umständen ein Leserecht.Es kann explizit das Schreibrecht ohne eine Leserecht vergeben werden.

Auswertung:
Die Dateien sind online und offline verfügbar.

Auswertungsaufwand:
Bei den statisch lesbaren Dateien im Home- Verzeichnis müssen keine neu hinzugekommenen bzw. entfernten Einträge ermittelt werden. Es wird nur das Dateiattribut "letzter Zugriff" geändert. Der Inhalt der Datei kann mit `cat` angezeigt werden mit dem Kommando stat <Datei> können, die MAC Zeiten betrachtet werden. Dabei muss die a- Time einen Zeitpunkt in der Zeitspanne haben und die beiden anderen Zeiten müssen einen Zeitpunkt vor dem Vorfall haben.

Lebensdauer:
Die benutzerspezifischen Dateienkönnen eine sehr kurze Lebensdauer haben, falls sie von anderen Vorgängen in der Zeitspanne gelesen oder modifiziert werden. Die benutzerspezifsichen Dateien können eine lange Lebensdauer haben, wenn sie nur genau von einem Vorgang gelesen werden, der aber nicht sohäufig stattfindet.

Möglichkeit der Fehlinterpretation:
Unmöglich. Die benutzerspezifischen Dateien (2) und (3) werden nur vom Vorgang: *Benutzer erstellt RPM- Paket* gelesen. (1), (4), (5), (6) und (7) werden nur beim Einloggen gelesen.

Fälschungsmöglichkeit:
Mittel. Die Datei kann nur gefälscht werden, falls der Angreifer die gleiche UID wie der Eigentümer hat oder an root Rechte gelangt ist. Die Datei `.rhosts` bzw. `shosts`, die Rechnernamen/ Loginnamen enthalten, von denen sich der Benutzer auf den Remote- Rechner mit dem spezifizierten Loginnamen ohne Passwort einloggen darf, hat die Rechte 644. Ist also nur vom Eigentümer und root schreibbar, aber von allen anderen lesbar und befindet sich auf dem Rechner auf dem man sich

ohne Passwort einloggen darf. Der Angreifer kann auf dem Remote System, falls er root Rechte hat, die Datei um seinen Rechnernamen und ein bereits existierenden Username, der in der Datei steht erweitern, um sich später ohne Passwort einloggen zu können. Der Angreifer schafft hiermit ein Backdoor für spätere Zugriffe. Wenn der Angreifer das gleiche User/Rechnernamen Paar vorweisen kann, kann er eine andere Identität annehmen.

2. Datei: modifizierbar bzw. neu erstellt

Nr.	Informationsquelle	Seite
(1)	`/myrpms/RPMS/<package name>.rpm`	203
(2)	`/myrpms/SPECS/<package name>.spec`	204
(3)	`/myrpms/SRPMS/<package name>.spm`	204
(4)	`/myrpms/BUILD/<sources>`	204
(5)	`/$HISTFILE`	63, 66, 69, 71, 72, 74, 75, 77, 78, 80, 81, 83, 84, 86, 88, 90, 93, 94, 96, 97, 98, 101, 103, 106, 107, 111, 112, 114, 116, 116, 118, 129, 134, 144, 147, 162, 165, 169, 177, 187, 192,193, 195, 196, 197, 197, 202,204, 152,

Modifikationsmöglichkeit:
Der Eigentümer der Datei ist der Benutzer der im Pfad `/home/<user>` angegegeben ist. Die Datei kann deshalb vom Eigentümer und root modifiziert oder gelöscht werden. Jenach dem wie die Zugriffsrechte der Datei für die Gruppe und Andere gesetzt, können diese mit vergebenen Schreibrecht die Datei auch ändern. Die History Liste ist nur vom Eigentümer selbst lesbar.

Auswertung:
Die Dateien sind sowohl offline und online verfügbar. Die History Liste selbst ist nicht offline verfügbar, da sie beim Start der Benutzersitzung aus der History Datei aufgebaut wird. Die History Datei kann z.B. bei der bash- Shell die Datei `bash_history` im Home- Verzeichnis des Benutzers sein.

Auswertungsaufwand:
Bei History Liste ((5)) muss der neue Eintrag, die neuen Einträge ermittelt werden. Der Auswertungsaufwand ist niedrig, falls dem Forensiker ein Back Up der History Datei, aus der die History Liste aufgebaut wird, vor dem Vorfall zur Verfügung steht. Er kann dann die alte History Datei mit der aktuellen History Datei mit dem Kommando `diff` vergleichen und die Änderungen somit erkennen. Ein Back Up steht dem Forensiker nicht zur Verfügung. Er muss dann andere Informationsquellen heranziehen und sie untereinander kombinieren, um die neu hinzugekommene Informationen in Form eines Eintrages oder von Einträgen (z.B. aus /var/log/messages falls vorhanden).
Um die neu erstellten Dateien (1) - (4) zu finden, muss der Forensiker in den Ordnern nachschauen und dann die Dateien auswählen, deren drei MAC Zeiten innerhalb der Zeitspanne liegen. Die MAC Zeiten müssen alle aber den gleichen Zeitpunkt haben, damit es sich nicht um modifizierte Pakete nach der Installation handelt. Der Aufwertungsaufwand ist dabei gering, da der Forensiker nur in den einzelnen Ordner mit dem Kommando `ls -la` nachschauen muss. Da das Kommando von jeder Datei den letzten Zugriff anzeigt, kann er schon eine Vorauswahl

von den Dateien, deren "letzter Zugriff" in der Zeitspanne des Vorfalls liegt, treffen. Und danach die ausgewählten Dateien mit dem Kommando `stat <filename>` genauer unter die Lupe nehmen und die beiden anderen Zeitstempel überprüfen.

Lebensdauer:
Der Eintrag in der History Liste ((5)) bleibt solange bestehen bis die History- Liste durch das Kommando `history -c` gelöscht wird oder der Eintrag überschrieben wird, falls die Anzahl der aufgerufenen Kommandos die Größe der History Datei in der Environment Parametern HISTFILESIZE übersteigt. Der Eintrag in der History Datei bleibt solange bestehen bis die Datei eine maximale Größe erreicht hat. Die maximale Größe wird in `/etc/login.defs` durch die Environment Variable ULIMIT beschränkt. Die maximale Größe ist 2 GB. Das entspricht 4194303 Blöcke zu 512 Byte. Die Lebensdauer kann auch abhängig von der Konfiguration vom *cron Dämon* und *logrotate* sein.

Möglichkeit der Fehlinterpretation:
Fehlinterpretation ist bei der History Liste und den anderen Dateien nicht möglich. Die History Liste wird bei den meisten Vorgängen die mit Kommandos ausgeführt werden modifiziert, aber die Ergiebigkeit der einzelnen Einträge ist immer unterschiedlich.

Fälschungsmöglichkeit:
Mittel. Ein Angreifer, der die History Datei eines anderen Benutzers lesen möchte, muss root Rechte haben. History Dateien sind für Angreifer interessant, da sie unter Umständen Passwörter und andere sicherheitsrelevante Informationen enthalten können. Die History Liste kann nur mit root Rechten oder vom Eigentümer selbst modifiziert werden. Möchte ein Angreifer sein Spuren verwischen, löscht er mit Kommando `history -c` einfach seine Befehlsfolgen aus der History Datei, die in der Environment Variablen HISTFILE angegeben ist.
Die Dateien in den verschiedenen Ordnern können mit root Rechten oder falls sich der Angreifer als der Eigentümer einloggen konnte, gelöscht werden. Ein Angreifer modifiziert die Dateien nicht, sondern tauscht sie durch seine eigenen trojanisierten Versionen mit dem gleichen Namen aus. Um seine Spuren zuverwischen wird er noch den Eigentümer ändern, um einen lokalen Benutzer für spätere Schäden auf dem System, verantwortlich zu machen.

7.1.3.1 Benutzerspezifische Dateien im Unterverzeichnis /.ssh

1. Datei: statisch lesbar

Nr.	Informationsquelle	Seite
(1)	`environment`	184
(2)	`known_hosts`	177
(3)	`rc`	182
(4)	`authorized_keys`	183
(5)	`id_dsa`	176
(6)	`identiy`	176
(7)	`id_rsa`	176
(8)	`id_dsa.pub`	176
(9)	`id_rsa.pub`	176
(10)	`identity.pub`	176

Die benutzerspezifsichen Dateien von ssh im Home Verzeichnis des Benutzers habe gleiche Kriterien, wie die Dateien von SSH im Unterverzeichnis `/etc`, nur dass sie vom Eigentümer (normaler Benutzer) und von root modifiziert werden können. Die Dateien (1), (2), (3), (4), (8), (9) und (10) können von root, von Mitgliedern der spezifizierten Gruppe und von Anderen gelesen werden. Die restlichen Dateien (5), (6) und (7), die die privaten Schluessel enthalten, sind nur vom Eigentümer und root lesbar.

2. Datei: statisch modifizierbar bzw. neu erstellt

Nr.	Informationsquelle	Seite
(1)	`authorized_keys`	183
(2)	`id_dsa`	176
(3)	`identiy`	176
(4)	`id_rsa`	176
(5)	`id_dsa.pub`	176
(6)	`id_rsa.pub`	176
(7)	`identity.pub`	176
(8)	`known_hosts`	177

Modifikationsmöglichkeit:
Der Eigentümer der Datei ist der Benutzer der im Pfad `/home/<user>` angegegeben ist. Die Datei kann deshalb vom Eigentümer und root modifiziert oder gelöscht werden. Jenach dem wie die Zugriffsrechte der Datei für die Gruppe und Andere gesetzt sind, können diese mit vergebenen Schreibrecht die Datei auch ändern.

Auswertung:
Die Datei ist online und offline verfügbar.

Auswertungsaufwand:
Der Auswertungsaufwand der Schlüsseldatei (2) - (7) ist niedrig, da der Forensiker nur auf die MAC Zeiten der Schlüsseldatei mit dem Kommando `stat <Schlüsseldatei>` schauen muss, um zu erkennen, ob sie innerhalb der Zeitspanne des Vorfalls erzeugt wurden, d.h. die drei MAC Zeiten haben alle den gleichen Zeitpunkt innerhalb der Zeitspanne.
Die beiden Dateien (1) und (8) enthalten mehrere Schlüssel. Ein neu hinzugekommener Schlüssel in Form eines Eintrages muss ermittelt werden. Der Auswertungsaufwand ist niedrig, falls dem Forensiker ein Back Up der Schlüsseldateien zur Verfügung steht und er mit mit dem Kommando `diff <backup der Schlüesseldatei> <aktuelle Schlüsseldatei>` den neu hinzugekommenen Eintrag bestimmen kann. Der Inhalt der Datei kann mit dem Kommando `cat` angezeigt werden. Der Forensiker muss feststellen, ob auf die Schlüsseldateien in der Zeitspanne des Vorfalls nur gelesen wurden. Er kann dies mit dem Kommando `stat <Schlüsseldatei>`, wobei der Zeitstempel atime einer der öffentlichen Schlüsseldateien (.pub) auf einen Zeitpunkt in der Zeitspanne des Vorfalls gesetzt sein muss, herrausfinden.

Lebensdauer:
Die Schlüsseldateien haben eine und die Konfigurationsdateien habe eine lange Lebensdauer, da sie durch die Vorgänge nur gelesen werden.

Möglichkeit der Fehlinterpretation:
Eine Fehlinterpretation ist bei den Konfigurationsdateien und den Schlüsseldateien

möglich, sie werden nur neu erstellt bzw. modifiziert, wenn sich der Benutzer mit ssh auf einen entfernten Rechner einloggen möchte.

Fälschungsmöglichkeit:
Mittel. Die Konfigurationsdateien und Schlüsseldateien können nur mit root Rechten modifiziert werden. Ein Modifizieren dieser Dateien ist für Angreifer untypisch. Vielmehr verübt ein Angreifer einen Man in the middle Attack um eine Sitzung zwischen einem Client und Server so zu übernehmen, dass die sensiblen Daten an den Angreifer gehen. Der Angreifer spielt in die Datei `/$HOME/known_hosts` des Clients seinen Host Schlüssel ein. Wenn sich der Client dann anmeldet und eine Überprüfung des Hostschlüssels nicht eingeschaltet ist, loggt sich der Client beim Angreifer ein.

7.1.4 Prozessdateisystem im Verzeichnis /proc

1. Virtuelle Datei: statisch lesbar

Nr.	Informationsquelle	Seite
(1)	`filesystems`	100, 101, 105, 144
(2)	`partitions`	104

Modifikationsmöglichkeit:
Der Eigentümer der virtuellen Datei ist root. Root kann die virtuelle Datei nicht direkt modifizieren. Die virtuelle Datei kann nur indirekt durch ein Programm bzw. Kommando gelesen werden, das der normale Benutzer oder root aufruft.

Auswertungsaufwand:
Niedrig. Der Inhalt der statisch lesbare virtuelle Datei wird durch die Ausführung des Vorganges nicht verändert, sondern nur das Dateiattribut "letzter Zugriff" der virtuellen Datei. Um an die Informationen in der Konfigurationsdatei zu kommen genügt ein einfaches Kommando, wie `cat` um den Inhalt der Konfigurationsdatei auszugeben. Mit dem Kommando `stat <virtuelle Datei>` können, die MAC Zeiten betrachtet werden. Dabei muss die a- Time einen Zeitpunkt in der Zeitspanne haben und die beiden anderen Zeiten müssen einen Zeitpunkt vor dem Vorfall haben.

Auswertung:
Die virtuelle Dateien und somit ihre Informationen in Form von Einträgen sind nur online verfügbar, da das Proc- Dateisystem nur im Hauptspeicher existent ist.

Lebensdauer:
Die Lebensdauer der virtuellen Dateien hängt davon ab, wann das System ausgeschaltet wird.

Möglichkeit der Fehlinterpretation:
Möglich. Die Datei (1) wird von mehreren Vorgängen gelesen und so das Dateiattribu geändert. Bei Datei (2) ist eine Fehlinterpretation nicht möglich, da sie nur von einem Vorgang gelesen wird.

Fälschungsmöglichkeit:
Da die virtuellen Dateien nicht direkt modifiziert werden können, kann ein Angreifer keinen Eintrag durch Editieren löschen bzw. entfernen. Er muss deshalb Vorgänge

aufrufen, die beispielsweise einen Eintrag oder Einträge entfernen. Er läßt die Einträge entfernen, die andere Programme brauchen, um ordnungsgemäß zu arbeiten. Es kann dann vorkommen, dass die Programme, die eine bestimmte Zeile aus der virtuellen Datei beim Start lesen, abstürzen oder nicht mehr korrekt arbeiten können.

2. Virtuelle Datei: statisch modifizierbar

Nr.	**Informationsquelle**	**Seite**
(1)	`/fs/nfs/exports`	149, 143
(2)	`mounts`	62, 103, 106, 107
(3)	`partitions`	100

Modifikationsmöglichkeit:
Der Eigentümer der virtuellen Datei ist root. Root kann die virtuelle Datei nicht direkt modifizieren. Die virtuelle Datei kann nur indirekt durch ein Programm bzw. Kommando geändert werden, das der normale Benutzer oder root aufruft modifiziert werden.

Auswertung:
Die virtuellen Dateien (1), (2) und (3) sind nur online verfügbar, da das `proc` Dateisystem nur im Hauptspeicher des Rechners verfügbar ist.

Auswertungsaufwand:
Bei der virtuellen Dateien muss der neue Eintrag ermittelt werden. Da die virtuelle Datei `partitions` (Partitionstabelle) nicht direkt modifiziert werden kann, wird ein Eintrag immer am Ende der Datei angefügt. Der Forensiker erkennt ihn daran, dass er der letzte ist. Die Dateien (1) und (2) werden nur online mit dem Kommando `showmount -e` und `mount` ausgewertet, wobei jeweils der die letzte Zeile der Ausgabe, der neu hizugefügte Eintrag durch den Vorgang ist. `mount` gibt dabei alle gemounteten Dateisysteme, auch die remote über NFS gemounteten Dateisysteme an. Mit `showmount -e` kann angezeigt werden, welche Dateisysteme exportiert werden.

Lebensdauer:
Die virtuellen Dateien werden durch Vorgänge modifiziert, d.h. es werden Zeilen hinzugefügt bzw. entfernt. Die Lebensdauer des Eintrages in der Konfigurationsdatei spielt nur eine Rolle, wenn das zu untersuchende System vor der Entdeckung des Vorfalls noch online war. Die Lebensdauer ist dabei kurz, falls mehrere Vorgänge, die in der Regel sehr häufig ausgeführt werden, in der Zeispanne stattgefunden haben und den Eintrag löschen bzw. ihn überschreiben. Der Eintrag in der Konfigurationsdatei hat dabei eine lange Lebensdauer, wenn das System offline ist, d.h. nach dem Vorfall sofort abgeschaltet wurde.

Möglichkeit der Fehlinterpretation:
Möglich. Mehrere Vorgänge ändern die gleichen virtuellen Dateien (1) und (2) ab. Es kann dadurch vorkommen, dass in der Zeitspanne des Vorfalls mehrere Vorgänge die virtuelle Datei modifizieren. Bei Datei (3) ist eine Fehlinterpretation nicht möglich, da sie nur von einem Vorgang modifiziert wird.

Folgende Fälle sind möglich:

1. Der eine Vorgang fügt einen Eintrag hinzu und der andere Vorgang löscht diesen Eintrag wieder. Die virtuelle Datei hat dann andere MAC Zeiten, aber es können keine Informationen aus der virtuellen Datei gewonnen werden.

2. Der eine Vorgang fügt einen Eintrag hinzu und der andere Vorgang überschreibt diesen mit einem neuen Eintrag. Der Eintrag kann nicht eindeutig einem Vorgang in der Zeitspanne zu geordnet werden. Die gewonnene Information ist deshalb bei der Top Down Analyse unbrauchbar. Da der Eintrag in der Konfigurationsdatei mit der dazugehörigen Ergiebigkeit unbedingt existent sein muss, um die Aussage "Der Vorgang hat mit 100 % Wahrscheinlichkeit stattgefunden", treffen zu können.

3. Der eine Vorgang fügt einen Eintrag hinzu und der andere Vorgang fügt auch einen Eintrag hinzu. In der Zeitspanne haben mehrere Vorgänge stattgefunden.

Fälschungsmöglichkeit:
Da die virtuellen Dateien nicht direkt modifiziert werden können, kann ein Angreifer keinen Eintrag durch Editieren löschen bzw. entfernen. Es muss deshalb Vorgänge aufrufen, die beispielsweise seine Zeile, die er in der virtuellen Datei erzeugt hat, löschen und gegebenenfalls überschreiben, um seine Spuren zu verwischen.

7.1.4.1 Prozessdateien im Unterverzeichnis /proc/<pid>

1. Virtuelle Datei: neu erstellt

Nr.	Informationsquelle	Seite
(1)	`cmdline`	108
(2)	`cwd`	108
(3)	`environment`	108 , 109
(4)	`exe`	108
(5)	`fd`	111
(6)	`maps`	108
(7)	`mem`	108
(8)	`mounts`	108
(9)	`root`	108
(10)	`stat`	108
(11)	`statm`	108

Modifikationsmöglichkeit:
Die Prozessdateien können nur indirekt durch ein Programm bzw. Kommando geändert werden, das der Benutzer aufruft. Der Eigentümer des jeweilige Verzeichnisses `/proc/<pid>` und den Prozessdateien ist der Benutzer, der den Prozess gestartet hat. Der Benutzer kann den Inhalt aller Prozessdateien des gestarteten Prozesses ansehen.

Auswertung:
Die Prozessdateien sind nur online verfügbar. Wird das System ausgeschaltet, werden alle laufenden Prozess beendet und die dazugehörigen Prozessdateien gelöscht.

Auswertungsaufwand:
Der Ordner, mit der Prozess ID entsteht beim Starten des Prozesses. Der INIT Prozess, der Vater aller Prozesse, hat die Prozess ID 0. Das `top` Kommando zeigt die aus der Prozesstabelle aktivsten Prozesse an, so dass man feststellen kann, welche zurzeit aktiven Prozesse die CPU und den Speicher belegen. Das

Kommando `pstree` gibt die Prozesshierarchie aus. Für jeden Prozess speichert der Kernel, von welchem Prozess dieser erzeugt wurde. Dies ist der Vater- Prozess. Die neugestarteten Prozesse heißen Kind- oder Tochter- Prozesse. Die Prozess ID kann aus der Prozesstabelle mittels dem Kommando `ps -aux | grep <Kommando> oder <dämon>` ermittelt werden, die eine Zuordnung von Prozess IDs zu Prozessnamen gibt. Mit der Option `w`, die auch mehrfach angegeben werden kann und mehr Text der Kommandozeilen der laufenden Prozesse ausgeben. Um die einzelnen Prozessdateien zu betrachten, muss der Forensiker in den Ordner `/proc/<pid>` wechseln und kann dann den Inhalt der einzelnen Dateien (2), (4), (9) mit `ls -la`. Es handelt sich hier bei um Links. (1), (3), (5), (6), (7) und (8) können mit dem Kommando `less oder cat` betrachtet werden. Mit dem Kommando `fuser` können Prozesse identifiziert werden, die eine bestimmte Datei oder Dateisystem gerade benutzen. Mit `lsof -i` können die Prozesse, die mit einem bestimmten Port zusammenhängen angezeigt werden. Der Aufwertungsaufwand den laufenden Prozess zu finden und die zum Prozess gehörenden Informationen auszuwerten sind dabei niedrig.

Lebensdauer:
Wenn der Prozess beendet wird, werden auch die dazugehörigen Dateien gelöscht. Die Lebensdauer ist also davon abhängig wann der Prozess bei einer Online Analyse beendet wird oder wann das System heruntergefahren wird.

Möglichkeit der Fehlinterpretation:
Fehlinterpretation ist nicht möglich, da in der Prozesstabelle genau angegeben wird wer den Prozess gestartet hat.Eine eindeutige Zuordnung von gestarteten Prozess zu Benutzer findet statt.

Fälschungsmöglichkeit:
Die Prozessdateien können nicht direkt geändert werden. Ein Angreifer ersetzt das Programm ps durch ein trojanisiertes ps aus einem Root Kit, um laufende Prozesse zu verstecken. Das trojanisierte ps Programm gibt dann nicht mehr alle Informationen über einen Prozess aus der Prozesstabelle aus.

7.1.5 Dateien im Unterverzeichnis /usr

7.1.5.1 Dateien im Unterverzeichnis /lib

1. Dateien: statisch lesbar

Nr.	Informationsquelle	Seite
(1)	`/rpm/macros`	201
(2)	`/rpm/rpmrc`	203, 199

Modifikationsmöglichkeit:
Der Eigentümer beider Dateien ist root. Die Gruppe und Andere habe nur Leserecht. Nur root kann die Datei direkt durch Editieren modifizieren.

Auswertung:
Die Dateien sind offline und online verfügbar.

Auswertungsaufwand:
Niedrig. Der Inhalt der statisch lesbaren Konfigurationsdatei wird durch die Ausführung des Vorganges nicht verändert, sondern nur das Dateiattribut “letzter

Zugriff” wird geändert. Um an die Informationen in der Konfigurationsdatei zu kommen genügt ein einfaches Kommando, wie `cat` um den Inhalt der Konfigurationsdatei auszugeben. Mit dem Kommando`stat <Datei>` können, die MAC Zeiten betrachtet werden. Dabei muss die a- Time einen Zeitpunkt in der Zeitspanne haben und die beiden anderen Zeiten müssen einen Zeitpunkt vor dem Vorfall haben.

Lebensdauer:
Die Dateien können eine sehr kurze Lebensdauer haben, falls sie von anderen Vorgängen in der Zeitspanne gelesen oder modifiziert werden. Die Dateien können eine lange Lebensdauer haben, wenn sie nur genau von einem Vorgang gelesen werden, der aber nicht so häufig stattfindet.

Möglichkeit der Fehlinterpretation:
Eine Fehlinterpretation ist bei (2), da sie von zwei Vorgängen gelesen wird, möglich. Bei (1) ist keine Fehlinterpretation möglich, das sie nur von einem Vorgang gelesen wird.

Fälschungsmöglichkeit:
Mittel. Die beiden Dateien können nur mit root Rechten modifiziert werden. Ein Angreifer der die Datei modifizieren möchte muss zuerst root Rechte erlangen. Ein Angreifer ändert die Parameter in der Datei `/usr/lib/macros`, so ab dass die Pakete am falschen Ort im Dateisystem erzeugt werden oder Parameter werden entfernt, dass Pakete nicht mehr ordnungsgemäß erstellt werden können.

7.1.6 Dateien im Unterverzeichnis /src/packages

1. Dateien: statisch modifizierbar bzw. neu erstellt

Nr.	Informationsquelle	Seite
(1)	`/RPMS/<package name>.rpm`	202
(2)	`/SOURCES/<source name>.dif`	198
(3)	`/SOURCES/<source name>.tar.gz`	198
(4)	`/SPECS/<source name>.spec`	198, 199
(5)	`/SRPMS/<package name>.spm`	202
(6)	`/BUILD/<sources>`	202

Modifikationsmöglichkeit:
Der Eigentümer der Ordner `RPMS, SOURCES, SPECS, SRPMS, BUILD` ist root. Folgenden Zugriffsrechte besitzen die Ordner: Der Eigentümer, die angebene Gruppe und Andere haben das Lese- und Schreibrecht und falls die Datei ausführbar ist, auch das Recht sie auszuführen. Das Sticky Bit ist auch gesetzt damit ein Benutzer keine Datei aus dem Verzeichnis löschen kann, von denen er auch Eigentümer ist. Damit kann der Benutzer keine Dateien entfernen, die anderen Benutzern gehören.

Auswertung:
Die Dateien sind offline und online verfügbar.

Auswertungsaufwand:
Um die neu erstellten Dateien zu finden, muss der Forensiker in den Ordnern nachschauen und dann die Dateien auswählen, deren drei MAC Zeiten innerhalb der Zeitspanne liegen. Die MAC Zeiten müssen alle aber den gleichen Zeitpunkt

haben, damit es sich nicht um modifizierte Pakete nach der Installation handelt. Der Aufwertungsuaufwand ist dabei gering, da der Forensiker nur in den einzelnen Ordner mit dem Kommando `ls -la` nachschauen muss. Da das Kommando von jeder Datei den letzten Zugriff anzeigt, kann er schon eine Vorauswahl von den Dateien, deren "letzter Zugriff" in der Zeitspanne des Vorfalls liegt, treffen. Und danach die ausgewählten Dateien mit dem Kommando `stat <filename>` genauer unter die Lupe nehmen und die beiden anderen Zeitstempel überprüfen.

Möglichkeit der Fehlinterpretation:
In dem Verzeichnis `/usr/src/packages` werden nur die Dateien des Quellpaketes `.spm` von SuSE Linux angelegt. Eine Fehlinterpretation ist deshalb nicht möglich.

Fälschungsmöglichkeit:
Die Dateien in den verschiedenen Ordnern können mit root Rechten gelöscht werden. Ein Angreifer modifiziert die Dateien nicht, sondern tauscht sie durch seine eigenen trojanisierten Versionen mit dem gleichen Namen aus. Um seine Spuren zuverwischen wird er noch den Eigentümer ändern, um einen lokalen Benutzer für spätere Schäden auf dem System, verantwortlich zu machen.

7.1.7 Environment Variablen

7.1.7.1 Benutzerumgebung

HOME , SHELL, USER/ LOGNAME, PATH, TERM

vgl. Seite 59

7.1.7.2 Netzwerkumgebung

Bei SSH:
DISPLAY, HOME, LOGNAME, MAIL, PATH, SSH_ASKPASS, SSH_AUTH_SOCK, SSH_CLIENT, SSH_ORIGINAL_COMMAND, SSH_TTY, TZ, USER

vgl. Seite 174

Modifikationsmöglichkeit:

Die Environment Variablen des Benutzers können nur während der Benutzer eingeloggt ist von ihm selbst oder von root neu gesetzt werden. Der Benutzer kann dabei alle oben angegebenen Variablen, ausser LOGNAME, neu setzen. Environment Variablen, die von Programme, gesetzt werden, können nur mit root Rechten neu gesetzt werden. Sie werden dann direkt mit Kommando `setenv oder export` gesetzt.

Auswertung:
Die Environment Variablen können nur online ausgewertet werden.

Auswertungsaufwand:
Niedrig. Mit `echo $VAR` kann der Wert jeder Environment Variablen angezeigt werden. Mit `printenv` kann man alle Environment Variablen anzeigen.

Lebensdauer:
Bleibt solange unverändert, bis der Vorgang der die Environment Variable setzt

beendet wird oder ein anderer Vorgang setzt sie neu.

Möglichkeit der Fehlinterpretation:
Fehlinterpretation ist möglich, da manche Environment Variablen auch von anderen Vorgängen gesetzt werden, aber dann unterschiedliche Werte den Variablen geben.

Fälschungsmöglichkeit:
Mit root Rechten kann das Kommando `setenv oder export` aufgerufen werden, um den Wert der Environment Variablen neu zusetzen.
Ein Angreifer kann beispielsweise die Environment Variablen eines Programmes, die es beim Start setzt, verändern, so dass andere Programme, die die Environment Variablen lesen, möglicherweise nicht mehr korrekt arbeiten können bzw. abstürzen.

7.1.8 Dateiattribute

Die folgenden vier Dateiattribute sind für die Forensiker am relevantesten. Die anderen Dateiattribute werden bei der Ergiebigkeit von Dateien, die als Paramter bei Kommandoaufrufen übergeben werden, angegeben. Ein Beispiel ist das Kommando `vi <filename>`.

Nr.	Informationsquelle	Seite
(1)	`letzter Zugriff`	85, 85, 89, 91, 187, 95, 96, 98
(2)	`letzter schreibender Zugriff`	87, 88, 91
(3)	`letzte Statusänderung`	87, 89,91, 93, 96, 98, 95
(4)	`UID und GID`	58

Modifikationsmöglichkeit:
Die Dateiberechtigungen lesen, schreiben und ausführen einer Datei bzw. Verzeichnisses kann der Eigentümer der Datei bzw. des Verzeichnisses für die Gruppe und Andere (Others) festlegen. Root kann Dateiberechtigungen aller Dateien des Systems festlegen. Dateien bzw. Verzeichnisse können nur gelöscht werden, wenn der Benutzer das Schreibrecht hat. Die Göße der Datei bzw. Verzeichnisses wird indirekt geändert, wenn die Datei modifiziert wird. Eine Modifikation ist nur möglich, wenn es sich um den Eigentümer, root oder um einen Benutzer handelt der das Schreibrecht auf die Datei bzw. Verzeichnis besitzt. Die Inodenummer wird nur vom System geändert und kann deshalb von keinem Benutzer direkt modifiziert werden. Nur der Eigentümer der Datei bzw. des Verzeichnisses kann mit dem Kommando `chown` den Gruppeneigentümer wechseln, aber nur für eine Gruppe, in der der Eingetümer Mitglied ist. Die GID der Datei bzw. Verzeichnisses wird dabei auf den neuen Gruppeneigentümer gesetzt. Nur root kann den Eigentümer einer Datei bzw. Verzeichnisses ändern. Die UID der Datei bzw. Verzeichnisses wird dann auf die UID des neuen Eigentümers gesetzt.

Auswertung:
Die Dateiattribute sind offline und online verfügbar.

Auswertungsaufwand:
Alle Dateiattribute können mit dem Kommando `stat` angezeigt werden. Um Änderungen von Dateiattributen mit niedrigem Aufwand zu erkennen, muss ein Back Up des Systems vorliegen. Der Forensiker kann dann folgende Kommandofolge anwenden, um geänderte Dateiattribute zuerkennen:

- `stat <aktuelle Datei> > file1.txt`

- `stat <Back Up der Datei> > file2.txt`
- `diff file1.txt file2.txt`

Dabei wird jeweils die Ausgabe auf den Bildschirm in eine Datei umgeleitet und dann die beiden mit dem Kommando `diff` verglichen.

Lebensdauer:
Die Dateiattribute haben eine kurze Lebensdauer, falls auf die Datei oder Verzeichnis sehr häufig von verschiedenen Vorgängen zugegriffen wird. Eine lange Lebensdauer der Dateiattribute der Datei oder des Verzeichnisses ist nur möglich, falls nur wenige Vorgänge auf die Datei zu greifen und diese nicht sehr häufig in einem Zeitraum stattfinden.

Fehlinterpretation:
Die meisten dokumentierten Vorgänge ändern die Dateiattribute ab und die sich daraus ergebenden Informationen sind die Gleichen. Eine Zuordnung der geänderten Dateiattribute zu einem Vorgang ist nicht möglich. Deshalb können die geänderten Dateiattribute falsch gedeutet werden.

Fälschungsmöglichkeit:
Durch Änderung der Systemzeit mit dem Kommando `date` oder mit Datum & Zeit des KDE- Kontrollbereiches ist ein Rückdatieren der Datei möglich. Es ist dadurch möglich eine Aktion, die auf die Datei ausgeführt wurde zu verschleiern. Um das Kommando `date` ausführen zu können, muss man Root Rechte haben. Nur bei ext2 kann mit Hilfe von dem Kommando `chattr +A <filename>` verhindert werden, dass der `atime` Record des Inodes der Datei auf die aktuelle Systemzeit und aktuelle Datum gesetzt wird. Die Vorgehensweise ist leicht, da `chattr` von normalen Benutzern ausgeführt werden kann. Aber einige Attribute können nur mit root Rechten geändert werden (siehe `man chattr`). Mit dem Kommando `utime` ist es das Dateiattribut "letzter Zugriff" (atime) zu ändern. Das Kommando kann nur mit root Rechten aufgerufen werden.

7.1.9 Netzwerkverbindungen

Nr.	Informationsquelle	Seite
(1)	`eingehende TCP Verbindung`	147, 139, 149, 178, 186, 189, 134 , 122, 124, 127, 128, 130
(2)	`ausgehende TCP Verbindung`	147, 139, 149, 134, 178, 186, 189 131, 126, 129, 125
(3)	`UDP Verbindung`	152 , 157, 165, 168, 169, 171

Modifikationsmöglichkeit:
TCP Verbindungen können von demjenigen Benutzer erzeugt werden, der gerade eingeloggt ist.

Auswertung:
TCP Verbindungen sind nur existent, wenn das System online ist und an das Netz angebunden ist.

Auswertungsaufwand:
Der Auswertungsaufwand ist niedrig. Mit dem Kommando `netstat -ra` kann

die Routingtabelle des Hosts, eine Liste der offenen Netzwerk Ports, das Remote System (falls vorhanden) und der Status der Verbindung angezeigt werden. Mit `netstat-p` werden alle Prozesse angezeigt, die mit einer bestimmten Netzwerkverbindung assoziert sind. Eine aktive Verbindung ist durch den Status ESTABLISHED gekennzeichnet. Eine eingehende Verbindung wird daran erkannt, dass die Spalte lokale Adresse den Wert `<local IP address>:<port>` hat, wobei die Portnummer kleiner als 1023 ist und die Spalte Foreign Adresss den Wert `<foreign IP address>:<port des Dienstes>` hat, wobei die Portnummer größer als 1023 ist. Eine ausgehende Verbindung hat dementsprechend die Werte `0:0:0:0:Portnummer >1023` und `<local IP address>:<port>`, wobei die Portnummner einen Wert kleiner 1023 hat. Dienste, wie ssh, telnet etc. die nur einen Kanal zur Datenübertragung verwenden, haben eine aktive Verbindung, die in beiden Richtungen Datenübertragungen zulassen. Im Gegensatz zu diesen hat FTP zwei Kanäle. Bei FTP wird deshalb unterschieden, ob die Verbindung eingehend oder ausgehend ist. Ports <1023 sind privilegierte Ports, die das System für bestimmte Dienste verwendet. Welcher Dienst nun welchen Port verwendet, wird in der Konfigurationsdatei `/etc/services` festgelegt. Der Forensiker kann den Sniffer `tcpdump` auf dem zu untersuchenden System einsetzen, um die Netzwerkdaten, die das System weiterleitet, abzugreifen und zu analysieren. Tcpdump kann beispielswiese den gesamten Header (einschließlich der IP- und TCP- Header) anzeigen.

Lebensdauer:
Die eingehende TCP Verbindung bleibt solange bestehen bis der Zielrechner die Verbindung auflöst, das Netzwerkkabel des lokalen Rechners gezogen wird oder der lokale Rechner die Verbindung normal abbaut.

Fehlinterpretation:
Unmöglich.

Fälschungsmöglichkeit:
TCP Verbindungen selbst können nicht gefälscht werden. Das Programm netstat wird Ziel eines Trojaner Angriffes, wenn ein Angreifer das System übernimmt. Hat der Angreifer schon root Zugriff auf dem System ersetzt er netstat und andere Programme, um seine Präsenz oder das Vorhandensein von versteckten Backdoors am System zu verheimlichen. Bei diesem Angriff wird eine manipulierte Netstat-Binärdatei einfach auf das System kopiert (dazu wird FTP oder RPC) benutzt); anschließend werden die alten Versionen durch die neue überschrieben. Um alle Netzwerkdaten von einem Hostsystem abzugreifen, die der Host weiterleitet, installiert ein Angreifer auf diesem Host einen Sniffer. Der Angreifer kann somit an sensible Informationen, die bei vielen Protokollen im Klartext übertragen werden, gelangen. Dienste, wie Telnet, FTP und HTTP sind Beispiele für Protokolle, die Passwörter im Klartext über das Netzwerk übertragen.

7.1.10 Sonstige Dateien

In diesem Abschnitt befinden sich Dateien, deren Aufenthaltsort variiert, d.h. vom Benutzer selbst gewählt werden kann. Der Aufenthaltsort dieser Dateien im Dateisystem ist nicht abhangig von Parametern in bestimmten Konfigurationsdateien, die den Pfad zu Datei festlegen und vom FHS.

Nr.	Informationsquelle	Seite
(1)	/<mount point>/quota.user	61
(2)	/<mount point>/quota.group	61
(3)	Datei <filename>	87, 89, 92, 92, 94, 95 97, 98, 124, 127, 128, 130, 187, 188, 195
(4)	At Datei <filename>	113
(5)	<filename>.rpm	191, 193, 196
(6)	Tar Archiv /<installation path>/<source package>	196, 197
(7)	<source name>.spm	198
(8)	Makefile	197

Modifikationsmöglichkeit:
Die Dateien (1) und (2) können nur von root geändert werden. Die beiden Dateien werden nicht direkt editiert sondern mit dem Kommando edquota -u <user> und edquota -g <group name> indirekt um Einträge erweitert. Die anderen Dateien können auf jeden Fall von root modifiziert werden und jenachdem wer der Eigentümer der Datei ist und die Zugriffsrechte für die Gruppe, in der der Eigentümer Mitglied ist, und für die Anderen vergeben sind. Die Gruppe und die anderen müssen mindestens das Schreibrecht besitzen, um die Datei zu modifizieren.

Auswertung:
Die Dateien sind offline und online verfügbar.

Auswertungsaufwand:
Bei den Dateien (1) und (2) muss ermittelt werden, ob die Quotas für die Gruppen und/oder den Benutzern sich geändert haben. Der Auswertungsaufwand ist niedrig, falls der Forensiker ein Back Up beider Dateien zur Verfügung hat und die Änderung der Quotas aus dem Vergleich der aktuellen Quotadatei mit dem Back Up erkennen kann. Da die beiden Dateien nicht mit vi geöffnet werden können, können sie nicht mit `diff` miteinander verglichen werden, sondern die Ausgaben des Kommandos `repquota -a` werden mit `diff` untereinander vergichen.
Die Online Auswertung erfolgt dabei folgendermaßen:

- `repquota -a > file1.txt`, wobei sich das Original auf dem laufenden System befindet
- `repquota -a > file2.txt`, wobei das Back Up auf das laufende System kopiert wird
- `diff file1.txt file2.txt`

Die Offline Auswertung ist im Gegensatz zur Online Auswertung komplizierter, da der Forensiker auf seinem Rechner, der das Dateisystem des zu untersuchenden Systems eingehängt hat, die gleiche Quota Umgebung schaffen muss, um an den Inhalt der Quotadateien zu gelangen. Er muss seine /etc/fstab, um die Partiton erweitern, die die Quotas unterstützt und diese mounten. Anschließend muss er noch die Passwdortdatei an die Stelle seiner kopieren, da diese von repquota -a ausgelesen wird. Ist die gleiche Umgebung bereitgestellt, können Änderungen mit den Kommandos wie obene beschrieben erkannt werden.
Die Quotas eines Benutzers kann online mit Kommando quota -u <user> angezeigt werden.
Bei den anderen Dateien (3) - (8) muss zuerst einmal der Aufenthaltsort und der genaue Name der Datei bestimmt werden und dann zu schauen, ob diese dann auch im Dateisystem existiert. Bei vorhandener History Liste kann der Forensiker den Namen der Datei herrausfinden, dieser steht nach dem aufgerufenen Kommando, das mit der Datei arbeitet. Wurde die Datei auf dem System gefunden kann sie mit dem Kommando stat <filename> genauer unter die Lupe genommen werden. Dieses Kommando zeigt alle Dateiattribute der Datei an. Für die Datei (6) können detailliert Informationen mit dem Kommando rpm -q -i <package name> abgefragt werden. Updates, Kofiguration und Änderungen des Paketes kann man zu diesem Paket mittels rpm -q -changelog <package name> ausgegeben werden. Es stehen leider nur 5 Einträge zur Verfügung. Im Paket selbst sind alle Einträge der letzten zwei Jahre enthalten. Die Abfrage rpm -qp -changelog <cd> funktioniert nur, wenn die richtige SuSE Linux CD, auf der sich das Paket befindet, eingelegt ist.

Lebensdauer:
Die Quotadateien habe eine kurze Lebensdauer, wenn sehr häufig neue Benutzerkonten am System angelegt werden. Das zu untersuchende System ist beispielsweise ein Rechner im CIP Pool. Dementsprechend ist die Lebensdauer lang, wenn es sich um einen Firmenrechner handelt. Die Lebensdauer ist auch davon abhängig wie schnell das Plattenkontingent belegt wird, und dann Änderungen der Quotas notwendig sind. Bei den anderen Dateien kann man nichts genaueres über die Lebensdauer aussagen.

Fehlinterpretation:
Fehlinterpretation ist bei den Quotadateien nicht möglich.

Fälschungsmöglichkeit:
Die Quotadateien nur mit root Rechten indirekt mit dem Kommando edquota -u <username> modifiziert werden. Ein Angreifer könnte demnach, falls er root Rechte hat, seinen Quotaeintrag ändern, um sein Kontingent an Plattenplatz zu vergrößern und es mit Datenschrott zu füllen. Ziel des Angreifers ist es das System durch eine volles Dateisystem in die Knie zu zwingen.

7.1.11 Sonstige Verzeichnisse

Verzeichnis (1), wird neuerstellt, in das Verzeichnis (2) wird bei einem anonymen FTP Zugang gewechselt, das Verzeichnis (3) wird durch einen Vorgang gelöscht und (4) ist der Mülleimer des Benutzers, der mit zu löschenden Dateien und Verzeichnissen gefüllt wird.

Nr.	**Informationsquelle**	**Seite**
(1)	`/$HOME/<newuser>`	65
(2)	`/srv/ftp`	122
(3)	`/$HOME/<user>`	121
(4)	`/$HOME/<user>/Desktop/Trash`	95

Für die Verzeichnisse (1) - (4) gelten die gleichen Kriterien wie in Abschnitt 7.1.2. Unterabschnitt 2. Datei: statisch modifizierbar.

Kapitel 8

Ausblick und Zusammenfassung

Die vorliegende Arbeit soll den Forensiker bei der forensischen Analyse unterstützen bzw. entlasten. Einerseits soll geklärt werden ob ein Vorfall, der zu einem bestimmten Zeitpunkt oder in einer Zeitspanne stattgefunden haben soll, auf dem zu untersuchenden System wirklich stattgefunden hat und gegebenenfalls zu welcher Wahrscheinlichkeitder er stattgefunden hat. Der Forensiker bedient sich dabei der Top Down Sicht dieser Arbeit.ein Auf deren anderen Seite verwendet der Forensiker die Bottom Up Sicht, um anhand von sichergestellten Informationsquellen herrauszufinden, was für ein Vorgang auf dem System stattgefunden hat. In diesem Abschnitt werden kurz die Vorteile und Nachteile der Darstellungsmöglichkeiten und Implementierungen der Top Down und Bottom Up Sicht erläutert.

8.1 Top Down Sicht

8.1.1 Top Down Sicht im vorliegenden Dokument

Vorteile

- Die Top Down Sicht ist sehr einfach anzuwenden. Der Forensiker sucht aus der Tabelle 6.1. die einzelnen Vorgänge, die den Vorfall zusammen bilden und wird dann auf die Seiten der einzelnen Vorgänge verwiesen, wo sich die durch den Vorgang geänderten Informationsquellen befinden.

Nachteile

- Die Anwendung des Dokumentes für eine Top Down Analyse ist für den Forensiker aufwändiger als die Implementierung, da er die jeweiligen Informationsquellen auf dem zu untersuchenden System suchen muss und deren Ergiebigkeit mit den dokumentierten Ergiebigkeiten vergleichen muss.
- Der Forensiker muss selbst die Wahrscheinlichkeit des Stattfindens eines jeden Vorgangs, anhand der gefundenen Informationsquellen mit den entsprechenden Ergiebigkeiten und der Anzahl an unbedingt existenten (Kriterium Existenz) Informationsquellen, bestimmen und dann die Gesamtwahrscheinlichkeit errechnen. Die betreffenden Aussage wird dann geäß der Wahrscheinlichkeit getroffen (siehe Abschnitt 4.2.).

8.2 Bottom Up Sicht

8.2.1 Einteilung in Verzeichnisse

Vorteile

- Die Informationsquellen können nach Namen in der Bottom Up Sicht leicht aufgefunden werden.
- Durch diese Darstellung werden Informationsquellen die gleiche Bewertungen haben zusammengefasst und so Wiederholungen vermieden.

Nachteile

- Um eine Bottom Up Analyse durch zuführen muss der Forensiker durch die Darstellung der Bottom Up Sicht als Einteilung in Verzeichnisse von der Bottom Up Sicht in die Top Down Sicht und umgekehrt hin und her blättern. Der Forensiker geht dabei nach dem Algorithmus in Abschnitt 4.2.3. vor.

8.2.2 Darstellung als Graph

Vorteile

- Der Graph kann sehr einfach mit dem Algorithmus in Abschnitt 4.2.3. zusammen für eine Bottom Up Analyse verwendet werden.

Nachteile

- Das Hinzufügen eines neuen Vorganges in den Graphen ist sehr aufwendig, da sehr viele Kanten zwischen den Knoten, die die Informationsquellen darstellen, gezeichnet werden müssen. Kanten, beschriftet mit der Ergiebigkeit, zwischen den Informationsquellen werden dabei nur gezeichnet, falls der Vorgang diese Informationsquellen abändert. Gegebenfalls müssen auch neue Knoten auf jeder Ebene hinzugefügt werden, falls der neue Vorgang eine Informationsquelle abändert, die kein anderer Vorgang abändert.
- Der Graph wird für sehr viele Vorgänge sehr groß und daher sehr unübersichtlich.

8.2.3 Implementierung

Vorteile

- Die Implementierung erleichtert die Arbeit des Forensikers. Dieser muss nur den Zeitpunkt des Vorfalls und die einzelnen Vorgänge, die denn Vorfall zusammensetzen, in das Programm eingeben.
- Der Forensiker bekommt eine konkrete Aussage (siehe Abschnitt 4.1.1.) über den Vorfall zurück.

Nachteile

- Um einen weiteren Vorgang hinzuzufügen, muss für den Vorgang und die dazugehörigen Informationsquellen ein ganz neues Perl Script geschrieben werden.

8.2.4 Implementierung des Graphen

Vorteile:

- Die Implementierung des Graphen erleichtert die Bottom Up Analyse auf dem zu untersuchenden System. Der Forensiker muss dabei nur eine auf dem zu untersuchenden System sichergestellte Informationsquelle und deren dazugehörige Ergiebigkeit in das Programm eingeben und dieses zu starten.

- Der Forensiker bekommt dann eine der drei Aussagen wie in Abschnitt 4.1.2. geschildert.

- Ein Vorgang und die dazugehörigen Informationsquellen kann sehr leicht zu dem Programm hinzugefügt werden, da jeder Vorgang unabhängig von den anderen Vorgängen mit einem dreidimensionalen Array dargestellt wird. Der dreidimensionale Array muss dann gemaß der in Abschnitt 5.1.2. genannten Datenstruktur gefüllt werden.

In dieser Arbeit wurden nur die wichtigsten Vorgänge untersucht und könnten in weiteren Arbeiten um weitere relevante Vorgänge erweitert werden, beispielsweise HTTP, Mail etc. Für die einzelnen schon dokumentierten Vorgängen könnten die Perl Scripte im Detail entworfen werden oder eine andere Möglichkeit gefunden werden, die Top Down Sicht zu implementieren.

Kapitel 9

Erklärungen

Cron	Der Cron Dämon führt beliebige Kommandos automatisch zu vorbestimmten Zeitpunkten aus. Der cron Dämon kann beispielsweise kann die Logdateien überwachen, damit sie nicht über eine bestimmte Größe anwachsen. Alle zu prüfenden Log- Dateien werden in der Konfigurationsdatei `/etc/logfiles` eingetragen. Je Datei wird eine Zeile angegeben, Name der Datei; maximale Größe, die die Datei erreichen darf; Zugriffsmodus (permissions); Eigentümer der Datei (owner und group).
Domains und Zonen	Bezüglich des Domain Name Services beinhaltet alle unter ihm liegenden Domains, Zonen und Systeme (Hosts), d.h. alle baumabwärtsliegenden Verwaltungseinheiten gehören zu dieser Domain. Zonen sind an verschiedene Einrichtungen und Unternehmen abgegeben, die die Verwaltung übernehmen und neue Knoten zeitnaher und schneller einbinden können. Jede Zone ist in sich abgeschlossen und beinhaltet nur die Knoten, die direkt zu ihr gehören.
FTP	*aktives FTP:* 1. Client- Ports 1024 - 65535 (TCP) intern Server-Port 21 (TCP) Nach erfolgreichem Verbindungsaufbau teilt der Client dem Server mit auf welchem (unpriviligiertem) Port x er einen Verbindungsaufbau für die Datenübertragung erwartet. Daraufhin initiiert der Server, ausgehend von Port 20 eine Verbindung zu Port 2. Client- Ports 1024 - 65535 (TCP) intern ← Server-Port 20 (TCP) *passives FTP:* 1. Client Ports 1024 - 65535 (TCP) intern → Server-Port 21 (TCP) Nach erfolgreichem Verbindungsaufbau teilt der Server dem Client mit, welcher (unpriviligierte) Port x für die Datenübertragung bereitsteht. 2. Client-Ports 1024 - 65535 (TCP) intern → Server-Ports 1024 - 65535 (TCP).

Logrotate	Das Programm logrotate rotiert die Protokolldateien. Typisch ist der Aufruf von logrotate über einen Cronjob; die Frequenz hängt vom Datenaufkommen ab und sollte zwischen einmal im Monat und täglich gewählt werden. Darüber hinaus vermag logrotate eine Protokolldatei erst zu bearbeiten, wenn sie eine bestimmte Dateigröße überschritten hat.Das Verhalten von logrotate wird üblicherweise durch mehrere Dateien beschrieben. Dabei ist `/etc/logrotate.conf` der ursprüngliche Anlaufpunkt für Konfigurationen.
NIS- Map	Ein spezielles Werkzeug (makedbm) transformiert eine ASCII-Datei (z.B. `/etc/passwd`) in eine Map im DBM- Format (z.B. `/var/yp/passwd.byname`). Da diese Dateien einfach indiziert sind, kann auf sie nur über einen Schlüssel zugegriffen werden. Jeder Map- Eintrag ist daher ein Tupel (key, information). Zu einigen Dateien gibt es daher mehrere Maps, die die gleiche Information enthalten aber unterschiedlich indiziert sind. Zur Datei `/etc/passwd` existieren beispielsweise die Maps `passwd.byname` mit dem Schlüssel Benutzername und `passwd.byuid` mit dem Schlüssel User-ID.
Prozesstabelle	Das Betriebssystem besitzt eine Prozesstabelle (Feld von Datenstrukturen), die für jeden Prozess einen Eintrag enthält. Dieser Eintrag enthält Informationen über den Zustand des Prozesses, seinen Befehlszähler, seinen Kellerzeiger, seine Speicherbelegung, den Zustand seiner geöffneten Dateien, Verwaltungs- und Schedulinginformationen sowie alle anderen Informationen über den Prozess, die gespeichert werden müssen, wenn der Prozess von Zustand rechnend in die Zustände rechenbereit oder blockiert übergeht.
RPM- Paket	Die rpm Datenbank verwaltet alle installierten Binärpakete. Die Datenbank enthält folgende in einem rpm- spezifischen Format: `Conflictsindex.rpm`, `fileindex.rpm`, `groupindex.rpm`, `nameindex.rpm`, `packages.rpm`, `providesindex.rpm`, `requiredby.rpm`, `triggerindex.rpm`.
Shadow Suite	In der /etc/passwd steht für das Kennwort nur noch ein “x”. Die Kennwörter stehen in der nur für root zugänglichen /etc/shadow. Die Datei `/etc/passwd` ist für alle Benutzer lesbar.
Syslogd	Der Syslog-Dämon syslogd fängt Fehler- und/oder Protokollausgaben von Programmen ab und verwaltet diese nach konkreten Regeln. Programme umfasst sowohl Prozesse, als auch Hardwaretreiber oder den Benutzer, der mit Hilfe des Kommandos logger selbst Meldungen einbringen kann oder Meldungen aus dem Netz. Abhängig von der Konfiguration können Meldungen in eine bestimmte Datei geschrieben, auf das Terminal ausgewählter (oder aller) Benutzer geschrieben, insofern sie am System angemeldet sind oder zur Behandlung an den syslogd eines anderen Rechners weitergeleitet werden.Was der Syslogd genau protokollieren soll wird in der Konfigurationsdatei `/etc/syslogd.conf` festgelegt. Eine Zeile der Datei `/etc/syslog.conf`, besteht aus zwei, durch Leerzeichen oder Tabulatoren voneinander getrennten Teilen.Der zweite, rechts stehende Teil beschreibt das Ziel der Nachricht. Das Ziel kann eine Datei (Protokolldatei, Device-Datei, Pipe) oder der Name eines Rechners sein, falls die Protokollierung von einem entfernten syslogd erledigt werden soll.Der erste Teil besteht aus Paaren von Herkunft.Priorität (durch den Punkt getrennt), mehrere solcher Paare können, jeweils durch ein Semikolon getrennt.

Literaturverzeichnis

[1] Jochen Hein, *Linux Systemadministration: Einrichtung, Verwaltung, Netzwerkbetrieb*. Addison Wesley, München, 1. Auflage, 2002.

[2] Frank Bodammer, *SuSE Linux 8.1: Administrationshandbuch*. SuSE Linux AG, 4. Auflage, 2002.

[3] Chris Prosise, Kevin Mandia, *Incident Response: Investigating Computer Crime*. McGraw-Hill Companies, 1. Auflage, 2001.

[4] Warren G. Kruse, Jay G. Heiser, *Computer Forensics*. Addison Wesley Professional, 1. Auflage, 2002.

[5] Simson Garfinkel, Gene Spafford, *Practical UNIX & Internet Security*. O'Reilly and Associates, 2. Auflage, 1996

[6] Brian Hatch, James Lee, George Kurtz *Das Anti- Hackerbuch für Linux*. Verlag Moderne Industrie Buch AG & CO. KG, Landsberg 1. Auflage, 2001

[7] *Die Linuxfibel*, www.linuxfibel.de

[8] *Log-Files auf Remote-Host*, http://www.tecchannel.de/betriebssysteme/715/1.html

[9] tecChannel, *Hackern auf der Spur - Beweissicherung im Netz*, http://www1.computerwoche.de/heftarchiv/2002/20020920/a80109456.html

[10] *Linux man pages*, http://www.die.net/doc/linux/man/

[11] *Sicherheit unter Linux 16 Schritte zu einem sicheren Linux-System*, http://admin.zid.tuwien.ac.at/zidline/zl05/linuxsectips.html

[12] *Sicherheit unter Linux*, http://www.fh-wedel.de/ si/seminare/ws99/Ausarbeitung/secure/secure0.html

[13] SANS Top 20, *The Twenty Most Critical Internet Security Vulnerabilities (Updated) The Experts Consensus*, http://www.sans.org/top20/

[14] *Das Linux Anwenderhandbuch und Leitfaden für die Systemverwaltung*, http://www.xinux.de/docs/linux/anwenderhandbuch/index.html

[15] Operating System, *Gundlagen*, http://www.operating-system.org/betriebssystem/_german/w-wissen.htm

[16] Red Hat Linux 8.0: Das Offizielle Red Hat Linux Referenzhandbuch, *Kapitel 12. Tripwire*, http://www.europe.redhat.com/documentation/rhl8.0/rhl-rg-de-8.0/ch-tripwire.php3

[17] Computer Forensic Analysis Class, *Class Handouts*, http://www.porcupine.org/forensics/handouts.html

[18] C. Eckert, *IT- Sicherheit: Konzepte, Verfahren, Protokolle*, Oldenbourg- Verlag, München, 2. Auflage

[19] *Operating Systems: Grundlagen*, http://www.operating-system.org/betriebssystem/_german/w-wissen.htm

Tabellenverzeichnis

Abbildungsverzeichnis