Erste Hilfe in Digitaler Forensik

Dr. Wetter, Dirk

Bahrenfelder Chaussee 100a
D-22761 Hamburg
<dirk_bei_drwetter_punkt_de>

1 Motivation

Gehackte Computersysteme sind leider heutzutage keine Seltenheit mehr. In den meisten Fällen ist es von Interesse herauszufinden, wer der Täter war, in welcher Weise dieser das System kompromittiert hat und welche Konsequenzen dies für das betroffene Unternehmen beziehungsweise für die betroffene Behörde hat.

Nicht selten ist der Vorgang Spurensuche, Beweiserfassung plus Analyse zweigeteilt, da Computerforensik-Experten nicht von Anfang an vor Ort bzw. involviert sind. In solchen Fällen obliegt zumindest das *Incident Handling* typischerweise einem lokalen Administrator und seinen Vorgesetzen in der IT. Aber gerade die ersten Momente des Entdeckens der Kompromittierung bis zum digitalen Duplizieren der Systeme sind entscheidend für die Unversehrtheit der digitalen Beweise und manchmal in geschäftlicher Hinsicht auch wichtig für das Unternehmen. Folgender Beitrag soll daher helfen, die wichtigen ersten Schritte in die richtige Richtung zu lenken.

2 Begriffe und Arbeitsweise

Die Wissenschaft der Forensik ist älter als man denken mag. Überlieferungen nach war der Wissenschaftler Archimedes (287-212 vor Christus) wohl der Erste, der von forensischen Methoden Gebrauch machte: Er bekam den Auftrag, die Echtheit einer Goldkrone seines Königs Hiero II. von Syrakrus zu prüfen, ohne diese zu beschädigen. Material abzuschaben, damit den Schmelzpunkt zu analysieren, oder die ganze Krone einzuschmelzen und in einen Quader mit bekannter Dichte von Gold zu gießen, fiel also aus. Nach einiger Überlegung[1] kam er der Fälschung durch ein Wasserverdrängungs-experiment auf die Schliche, was später als Archimedisches Prinzip bekannt wurde.

Historisch gesehen die häufigste und nach Archimedes die früheste[2] Disziplin der Forensik sind die so genannten „gerichtlichen Wissenschaften“ und die Rechtsmedizin, die heute viele verschiedene Teilbereiche umfassen: Pathologische und Anthropologische Forensik, Dentalforensik, Pyschologische Forensik, Toxikologie, Entomologie. Abseits der Medizin zählen beispielsweise die Ballistik, Daktyloskopie (Fingerabdrücke), DNA-Spurenanalyse, Brandursachenforschung zu den forensischen Wissenschaften.

---

[1] Der Gedanke soll im ihm beim Einsteigen in die Badewanne gekommen sein. Der kolportierten Anekdote zur Folge ist er dann voller Begeisterung über seinen Geistesblitz aus der Wanne gesprungen und unbekleidet auf die Straße gelaufen, „(H)Eureka“ schreiend. Was altgriechisch ist und so viel heißt wie „ich hab's gefunden“.

[2] Erste schriftliche Abhandlung in Form eines Bandes mit 53 Kapiteln in 5 Büchern von Song Ci (1247) „Gesammelte Fälle von berichtigter Ungerechtigkeit“.

2.1 Begriff Forensik

Verschiedene Definitionen des Wortes „Forensik“ beschreiben schon recht treffend für Ermittler aller Fachgebiete, die mit forensischen Aufgaben betreut sind, die Aufgabenstellung, die sie erwartet. Wikipedia [1] schrieb Ende 2007:

Unter dem Begriff Forensik werden die Arbeitsgebiete zusammengefasst, in denen systematisch kriminelle Handlungen identifiziert, analysiert oder rekonstruiert werden.

Die wichtigen Schlüsselworte, die ebenso auf die Digitale Forensik zutreffen, lauten: **systematisch**, **kriminell**, **identifizieren**, **analysieren** und **rekonstruieren**.

Besonders die mit dem Incident Handling betreute(n) Person(en) sollte(n) sich diese Begriffe immer vergegenwärtigen und die entsprechende Vorsicht walten lassen im Umgang mit digitalen Beweisen, damit eine etwaige gerichtliche Verwertbarkeit in strafrechtlichen oder zivilrechtlichen Verfahren möglich ist.

2.2 Methodik

Bei kompromittierten Rechnersystemen sollte man, was Täterkreis, Motivation und Schadensausmaß angeht, unbehelligt zu Werke gehen. Weder zum Zeitpunkt des Entdeckens noch später sollte man eine zur gerade ins Bild passende Richtung implizit Schlüsse ziehen, sondern in seiner Vorgehensweise stets neutral sein. Es besteht sonst die Gefahr, dass nach nur nach Beweisen, die eine bestimmte These untermauern, gesucht wird, und ggf. andere, die nicht in das Bild passen, übersehen werden.

Das so genannte Austauschprinzip von Edmond Locard, ein französischer Mediziner (1877-1966), hat auch bei digitalen Spuren (*digital evidence*) in gewisser Weise seine Gültigkeit: **Jeder Kontakt hinterlässt eine Spur** oder ausführlicher: **Jeder und alles am Tatort nimmt etwas mit und lässt etwas zurück**.

Digitale Spuren können ebenso wie physikalische am Tatort leicht verändert werden: Der Täter hinterlässt in beiden Fällen Spuren. Das „Mitnehmen“ digitaler Spuren ist nur zum Beispiel beim Keylogger, Sniffer oder etwa beim Diebstahl von Kundendaten aus einer Datenbank gegeben; im nicht-digitalen Fall sind beim Täter eigentlich immer Spuren vom Tatort nachzuweisen, also zum Beispiel an seiner Kleidung oder seinem Körper.

Umgekehrt gilt für Ermittler, dass deren Anwesenheit am Tatort Beweise verändert. Dies hat in der Digitalen Forensik auch seine Gültigkeit: Der Status eines in Betrieb befindlichen, kompromittierten Rechnersystems beispielsweise lässt sich nie ohne Veränderung sicherstellen und kopieren.

Ab dem Zeitpunkt der Kopie haben jedoch digitale Spuren entscheidende Vorteile: Man muss (und sollte) nicht Originale zur Untersuchung nehmen. Sie sind beliebig kopierbar. Auch lässt sich durch Prüfsummenbildung jederzeit feststellen, ob Kopie und Original immer noch identisch sind.

Ein weiterer Vorteil gegenüber der Rechtsmedizin: Die gegenwärtigen Methoden aller Betriebssysteme zur Löschung von Beweisen (=Daten) sind recht unvollständig[3]. Selbst bei Einsatz von DoD-konformen Löschwerkzeugen[4] lassen sich unverschlüsselte Daten mittels Magnetfeld- oder Elektronenmikroskopie zumindest teilweise rekonstruieren.

Wichtig ist also in jedem Schritt, sorgfältig zu Werke zu gehen. Schematisch kann man den gesamten Vorgang folgendermaßen unterteilen:

1. Beweis erhärten
2. Volatile Daten sichern (*Live Response*)[5]
3. Duplikation von Platteninhalten (*forensic duplication, disk imaging*)
4. Sicheres Aufbewahren von Original und Duplikat(en)
5. Post-Mortem-Analyse
6. gerichtsfähige Aufarbeitung aller erhaltener Daten und Präsentation im forensischen Report

---

[3] Man mag nicht ohne Recht argumentieren, dass der Einsatz von Verschlüsselung wie einer „Full Disk Encryption“ mit nachfolgender Entsorgung des Schlüssels eine nach gegenwärtigem Stand der Technik recht sichere Methode der Datenvernichtung ist.

[4] Die Norm heißt DoD 5220-22.M, siehe [2].

[5] Findet eine Analyse zu diesem Zeitpunkt statt, spricht man von *Live Forensik*.

Bei allen Schritten ist eine umfassende Dokumentation unabdingbar: **Wer** hat **wann was** gemacht, der genaue Ort (**wo**), und das **Wie** gehören ebenso dokumentiert.

Die Dokumentation sollte lückenlos im Sinne einer Beweiskette (*chain of custody*) sein. Man sollte versuchen, sich in die Rolle eines Beschuldigten/Angeklagten zu versetzen, der alles tun wird, um Beweise anzuzweifeln und sich so zu entlasten. Die Dokumentation und die gefunden Beweise sollten soweit wie möglich auch für Laien – Richter und andere Verfahrensbeteiligte sind keine IT-Spezialisten – nachvollziehbar sein. Zeugen – das Vier- oder Mehraugenprinzip – erhöhen die Glaubwürdigkeit.

Digitale Hilfe zur Dokumentation bekommt man z.B. von den Programmen `date` zur Datumsfeststellung – vom kompromittierten System und vom via NTP geeichten Referenzsystem – und fürs Mitschreiben von Befehlen sind `script` und `screen -L` recht brauchbar. Prüfsummen – solche Algorithmen mit einer dem Stand der gegenwärtigen Technik nach hinreichend niedrigen Wahrscheinlichkeit für eine Kollision unter den gegebenen Umständen, also mindestens MD5, zählen zum Handwerkszeug.

So leicht sich wie oben erwähnt, digitale Spuren duplizieren lassen, so leicht ist es leider auch möglich, digitale Beweise, ebenso wie eine lediglich digital vorliegende Dokumentation, nachträglich zu manipulieren. Um daher einer etwaigen mangelnden Gerichtsverwertbarkeit vorzubeugen, kann man eine erhaltene Dokumentation mitsamt Prüfsummen auf nummerierten Seiten (`enscript`, `a2ps`) ausdrucken und handschriftlich, gegebenenfalls mit einem Zeugen, unter Angabe von Datum und Ort unterzeichnen.

2.3 Sicherheitsvorfallsmanagement – Incident Handling

Sowohl auf Unternehmens- beziehungsweise Behördenseite als auch auf der Seite des Admins, der technisch mit der Kompromittierung konfrontiert ist, ist eine geplante, strategische Vorgehensweise bei einem Sicherheitsvorfall angeraten.

Was die Firma[6] angeht, ist es wichtig, dass das Incident Handling in die Unternehmensprozesse eingebunden wird. Sicherheit generell, ein Incident erst recht nicht, ist nie eine ausschließlich technische Angelegenheit, sondern fällt mit in den Verantwortungsbereich des Managements. Man denke nur an den schlimmsten Fall, dass eine Kompromittierung so große Auswirkungen hat, dass das Überleben der Firma auf dem Spiel steht, etwa durch einen Imageschaden oder erhebliche finanzielle Verluste, die nicht mehr durch das Firmenkapital gedeckt werden können.

Der Administrator, der sich technisch mit der Kompromittierung auseinandersetzen muss, sollte die Vorgehensweisen der Digitalen Forensik beherzigen. Je nach Vorbildung wird ein Anfänger im „Junior-Level“ dazu weniger geeignet sein. Der sorgsame Umgang erfordert einen vorausschauenden Umgang mit dem „Corpus Delicti“. Incidents sind in ihrer Art nicht gleich und für eine korrekte Behandlung der digitalen Beweise ist einiges an Erfahrung und Wissen nötig ist.

2.3.1 Prozesse der Firma

Es ist sicherlich ein Unterschied, ob der Desktop des Hausmeisters kompromittiert worden ist, das Webportal mit den Kreditkartendaten von circa 1000 Kunden oder ob auf einer Eingangs-Webseite eines IT-Providers „owned“ oder ähnliches für einige Tage zu lesen steht. Mögen diese Fälle noch halbwegs klar in der Auswirkung, vielleicht weniger von den zu folgenden technischen Maßnahmen, sein, aber wie soll vorgegangen werden, falls ein Admin spät abends feststellt, dass der Desktop der Chefsekretärin wahrscheinlich ein Rootkit hat? Hatte sie Zugang zu sensitiven Daten ihres Chefs oder sind dort andere Firmengeheimnisse gespeichert? Kann und darf der Admin zu dieser Stunde eigenmächtig handeln?

Wichtig für das Unternehmen ist bei Sicherheitsvorfällen generell die entsprechende Meldung an die zuständige Instanz in der Firma, die in solchen Fällen weiter ans Management eskaliert wird, das denn über die weitere Vorgehensweise zu entscheiden hat.

Falls das Unternehmen weder ein komplettes ISMS (Informationssicherheits-Managementsystem) noch Notfallprozeduren etabliert hat, sollte es zumindest gewisse Prozesse im Vorfeld eines Sicherheitsvorfalls bereits aufgebaut haben. Im Rahmen dieses Papers kann der Autor nicht auf Einzelheiten dieser umfassenden Aufgabe eingehen und beschränkt sich daher auf die wichtigsten Tipps, die einen guten Einstieg in die Thematik bieten sollten.

---

[6] Die Begriffe Firma/Unternehmen werden der Einfachheit halber ab hier stellvertretend für Firma und Behörde verwendet.

Für Unternehmen ab mittlerer Größe bieten sich zur Etablierung von Notfallprozeduren für Sicherheitsvorfälle als Einstieg der IT-Grundschutz des BSI (Bundesamt für Sicherheit in der Informationstechnik) an, spezifischer: die BSI-Grundschutzkataloge [3]. Ergänzend dazu aus dem BSI-Standard 100-1 [4] besonders die Kapitel 4 (Management-Prinzipien) und Kapitel 8 (IT-Sicherheitskonzept)[7].

Generell geben die IT-Grundschutz-Kataloge Standardsicherheitsempfehlungen organisatorischer, personeller, infrastruktureller und technischer Art. Die Kataloge haben einen außerordentlichen Umfang, gegenwärtig sind es 3612 Seiten. Man muss sich glücklicherweise nicht komplett damit befassen. Die Kataloge sind strukturiert und nach dem Baukastenprinzip aufgebaut. Aus den relevanten Bausteinen lassen sich für bestimmte Aufgabenstellungen, wie in diesem Fall für Sicherheitsvorfälle, Maßnahmenbündel für eine Institution erstellen. So enthalten die Bausteine (B) der Grundschutzkataloge grundsätzlich eine Kurzbeschreibung der Vorgehensweisen bzgl. der IT-Komponenten. Verwiesen wird in diesen Bausteinen auf die Gefährdungskataloge (G) und Maßnahmenkataloge (M).

Ein guter Einstieg – eine umfassende Diskussion würden den Rahmen des Beitrags sprengen – ist das Notfallvorsorge-Konzept (Baustein B 1.3) plus dem Baustein B 1.8 (Behandlung von Sicherheitsvorfällen). Beide verweisen auf eine Reihe Maßnahmen, bei denen es sich empfiehlt, sie nach Überprüfung ihrer Relevanz für die eigene Firma umzusetzen. Im wesentlichen sind dies die folgenden:

- Notfalldefinition + Notfallverantwortlicher (M 6.2)
- Regelung der Verantwortung im Notfall (M 6.7)
- Festlegen von Verantwortlichkeiten bei Sicherheitsvorfällen (M 6.59)
- Verhaltensregeln und Meldewege bei Sicherheitsvorfälle (M 6.60)
- Notfallhandbuch (M 6.3)
- Etablierung eines Managementsystems zur Behandlung von Sicherheitsvorfällen (M 6.58)
- Eskalationswege, betroffene Stellen (M 6.65)
- Nachbereitung von Vorfällen (M 6.66)

Die wesentlichen Punkte sind, Strukturen in der betreffenden Organisation zu schaffen, mit denen aus Unternehmenssicht eine angemessene Reaktion auf einen Sicherheitsvorfall möglich ist. Dies beinhaltet zunächst eine eindeutige Klärung von Verantwortlichkeiten bei Sicherheitsvorfällen, dann der nötigen Handlungsweisen auf allen Ebenen. Nach Schaffung der Strukturen geht man üblicherweise daran, für adäquate Eskalationspfade bis zur Leitungsebene zu sorgen, und es zu ermöglichen, jeden Notfall in ihrer Bedeutung für die Organisation zu klassifizieren (Incident-Management-Plan). Für die Vorgehensweise bei Sicherheitsvorfällen kann – nach BSI-Grundschutz sollte – man ein Notfallhandbuch haben, die detaillierte Anweisungen enthalten wie zu verfahren ist, um den Schaden einzugrenzen.

2.3.2 Vorgehensweise für den Administrator

Zuerst sollen in diesem Paper die Methoden der Gegenseite erläutert werden, um die Vorgehensweise der späteren Schritte verständlicher erscheinen zu lassen.

2.3.2.1 Kleine Lehre des Versteckens

Rootkits stellen in den meisten Fällen der Kern der Kompromittierung dar. Das erste Rootkit gab es für SunOS – je nachdem, welchen Quellen man traut, bereits 1990 oder 1994 [5,6,7]; also nachdem die ersten Paketfilter verfügbar aber noch nicht überall im Einsatz waren. Und in dem Zeitraum, in dem der Linux-Kernel gerade mal geboren wurde.

Rootkit-Techniken halten wie alles andere auch Schritt mit dem Stand der Entwicklung in der IT und sind sicherlich viel ausgefeilter als zu dessen Geburtsstunde. Generell weisen Rootkits folgende Eigenschaften auf:

---

[7] Als Einstieg bietet sich ebenso im BSI-Grundschutzkatalog [3] der Baustein B 1.0 an.

- Verstecken ihrer Anwesenheit
  - Prozesse
  - Dateien (Binärdateien, Bibliotheken, Log-Beschneidung)
  - Sockets
  - ggf. im RAM und ggf. auf Datenträgern
- Hintertür zur Fernsteuerung (ggf. Integration in Botnetz)
- ggf. Abgreifen von Daten
- Sprungbrett für weitere „Aktivitäten“

Abgesehen von gezielten Kompromittierungen – bei Innentätern, Ex-Beschäftigten oder mit dem Ziel einer Wirtschaftsspionage oder hohen „Gewinnaussichten“ – werden zur Penetration im Regelfall[8] keine Einzelanstrengungen mehr unternommen. Die Kombination *Exploit – Verstecken – Steuerung* geschieht automatisiert per Massenscan und -penetrationsversuchen. Beim Penetrieren von Webanwendungen hat sich der Erfahrung des Autors nach gezeigt, dass nicht selten Suchmaschinen bemüht werden, Verwundbarkeiten einfacher aufzuspüren (Google-Hacking).

Außer vielleicht bei letztgenannter Kategorie von Kompromittierungen sieht man heutzutage selten noch ein reines User-Land-Rootkit. Diese Spezies ist recht einfach aufzuspüren, da sie vordergründige Methoden des Versteckens bietet. Es gibt die Möglichkeit, LD_PRELOAD oder LD_LIBRARY_PATH zu benutzen, Verzeichnisse mit einem Leerzeichen oder Punkt plus Leerzeichen anzulegen oder die Masche, Prozesse mit ähnlichem Namen wie existierende zu starten (wie `crond`, `[kswap0]` von Benutzer `www-data`). Jedoch sollten diese recht alten Techniken spätestens bei einer zweiten gewissenhaften Prüfung mit entsprechenden Werkzeugen oder schon bei einer händischen Inspektion auffallen.

Die häufiger anzutreffenden Kernel-Rootkits haben es da einfacher. Nach einer Rechteausweitung auf Root wird üblicherweise entweder ein Linux-Kernel-Modul (LKM) geladen, dass die Aktivitäten auf User-Ebene durch umbiegen von Systemaufrufen (Syscalls) in der Prozessliste, im Dateisystem und in den Netzverbindungen verbirgt. Alternativ zum diesen LKM-basierten Rootkits bietet sich eine prinzipielle Schwäche von Linux zur Kompromittierung an: `/dev/kmem`. Es ist für Root beschreibbar und kann so ähnlich wie ein LKM Systemaufrufe umbiegen. **Der** Vertreter der Kmem-Rootkits schlechthin ist das sogenannte SucKIT [8], was auf ein „Magic Packet“ hin eine C*onnect-back Shell* aktiviert, einen Sniffer beinhaltet und ein Hand-in-Hand mit einem eigenen `/sbin/init` agiert [9].

Aber selbst Kernel-Rootkits und Eindringlinge, die solche verwenden, sind, was das Verstecken angeht, zum Glück meistens nicht perfekt, so dass man diesen im User- oder auch im Kernel-Land im laufenden System auf die Schliche kommen kann. Die Verbreitung dieser Imperfektionen über Mailinglisten und andere Quellen funktioniert dabei recht gut, sodass die mit der Forensik betreute Person nur selten mit einem „Zero-Day-Rootkit“ konfrontiert ist. Das SucKIT beispielsweise lässt sich recht einfach über `/sbin/init` mit `rkhunter`/`chkrootkit` nachweisen. Umgebogene Syscall-Aufrufe oder -Tabellen kann man entweder durch einen Adressenüberprüfung oder durch beispielsweise eine Laufzeitanalyse durch Werkzeuge wie Patchfinder [10,11] aufspüren. Natürlich hat die „Unsichtbarkeit“ des Kernel-Rootkits generell ein Ende, sobald man das verdächtige System mit einem neuen Kernel beispielsweise von DVD/CD startet.

Wer nun meint, der Kampf wäre dadurch irgendwie gewonnen, verkennt den typischen Mechanismus des Wettrennens von Technologien in diesem Bereich. Eine neue, noch nicht weit verbreitete Spezies haben im Vergleich zu diesen *persistenten Rootkits* gelernt: Sie sind rein speicherbasiert (*Memory-Based Rootkits*), was nicht optimal für häufig neu startende Desktops geeignet ist, wohl aber für Server, die sich durch eine lange Uptime auszeichnen. Ein extremes Beispiel: Bei einem dem Autor bekannten Unix-System betrug diese bis vor kurzem über 800 Tage; für das Überleben eines solchen Rootkits optimal. Ein sehr gutes Beispiel für rein speicherbasierte Rootkits ist die „Zecke“ – ein Proof-of-Concept von Tobias Klein [12,13], in gewissen Grenzen[9] z.B. auch das DKOM-basierte FU-Rootkit [14] und einer der Nachfolger, der Shadow Walker [15].

Das einzige, was bei solchen rein speicherbasierten Rootkits für die spätere Analyse hilft, ist das Ziehen von Speicherabzügen bei laufendem System[10]– und zwar von einzelnen Prozessen plus dem gesamten Hauptspeicher. Aber auch andere Rootkits können in solchen Fällen nützliche Informationen liefern, die man nicht verloren geben sollte.

---

[8] Der häufig diskutierte Bundestrojaner zählt auch zu den Einzelanstrengungen.

[9] In Grenzen, da FU zwar nicht einen Reboot übersteht, aber mit zwei **Dateien** arbeitet, einem Binärprogramm und einem Kernel-Treiber.

[10] Hier gilt zu bedenken, dass das Rootkit dies unterbinden bzw. die Speicherabzüge „fälschen“ kann. Letztendlich lassen sich nur zuverlässig mit Hardware Speicherabzüge erstellen [16].

3 Verdacht erhärten

3.1 Anfangsverdacht

In den allermeisten Fällen einer Kompromittierung hat man bereits erste Hinweise, dass „etwas faul“ ist mit dem betreffenden Knoten. „Man“ kann entweder ein Benutzer sein, der sich glaubhaft wegen eigentümlichen Systemverhaltens beschwert oder ein System- oder Netzverantwortlicher, der Auffälligkeiten bemerkt hat, zum Beispiel durch ein Anti-Rootkit-Software, durch Integritätschecker (HIDS), Log-Meldungen im NIDS bzw. zentralen Syslog oder vielleicht durch auffälligen Netzverkehr.

Je nach etablierten Notfallprozeduren der Organisation muss ein Anfangsverdacht bereits zu diesem Zeitpunkt mitgeteilt werden.

Um einen Anfangsverdacht zu erhärten, kann man grundsätzlich entweder in den Netzverkehr oder den betreffenden Knoten näher inspizieren.

3.2 Netz

3.2.1 Netzverkehr

Anhand des Netzverkehrs seinen Verdacht zu erhärten, ist am wenigsten invasiv, aber unter Umständen auch am wenigsten erfolgreich. Auch kann einige Zeit vergehen, bis man sich so einer Kompromittierung sicher sein kann. Zeit, die eine Organisation unter Umständen aus Verfügbarkeitsgründen oder wegen eines Unternehmensrisikos nicht zu tragen bereit ist.

Den Netzverkehr kann man durch folgende gängige Methoden analysieren:

- Switch mit Mirror-Port konfigurieren
- Einen Hub zwischen Switch und Knoten stellen („Hubbing out“)
- MITM-Werkzeuge: Ettercap, Cain & Abel

Das Problem hierbei ist, dass man je nach Fall recht lange warten kann, bis man wirklich eine relevante Aktivität sieht. Auch Zeitverschiebungen können eine Rolle spielen: In einem Fall des Autors reichte der beschränkte Plattenplatz auf dem gehosteten Server mit 12 virtuellen Hosts via Apache nicht aus, den gesamten Netzverkehr über Nacht mitzuschneiden. Bei einem anderen Fall wurde der Autor nicht richtig schlau aus den Paketinhalten, sodass er hätte gleichzeitig aufs System schauen müssen: Hier fanden die Aktivitäten nur zu deutschen Nachtzeiten statt, weil wie sich später herausstellte, die Fernsteuerung des Rootkits von Japan aus geschah.

Untersuchungen zur Verdachtsverhärtung im Netz sind meistens nur aufschlussreich, solange man in die Pakete reinschauen kann. Bei Verwendung von Verschlüsselung, was mehr in Mode zu kommen scheint, ist eine Untersuchung eines Endpunktes der Verschlüsselung nicht zu vermeiden.

3.2.2 Scan

Auf alle Fälle bietet sich ein Scan des verdächtigen Knotens im LAN an, am einfachsten ein voller Port-Scan (TCP- und UDP-Ports, IP-Protokolle) mit `nmap`. Wünschenswert ist, dass man zu diesem Zeitpunkt so etwas wie eine Baseline aus einer unkompromittierten Zeit hat, damit man bei nicht ganz so offensichtlichen Auffälligkeiten beim Scan eher entscheiden kann, ob die gefundene Abweichung normal ist oder nicht. Einige kommerzielle und halb kommerzielle Werkzeuge wie Retina von eEye und Nessus von Tenable bieten dieses „Baselining“ an, bei `nmap` ist ein wenig Programmierung gefragt, ein guter Start ist `ndiff` [17].

3.3 System

Die Inspektion des betreffenden Systems zur Verdachtserhärtung ist immer invasiv. Um das Maß der Beweisbeschädigungen gering zu halten, ist es wichtig zu wissen, welche Beweise für die spätere Post-Mortem-Analyse relevant sind.

Ein Ziel dieser forensischen Untersuchung ist es, gelöschte Dateien wiederherzustellen – beispielsweise eine Shell-History-, eine System-Log-Datei oder Teile des Rootkits. Ein weiteres Ziel ist es, herauszufinden, wann welche Dateien angelegt, angefasst und modifiziert wurden (Timeline-Analyse)[11].

Als Konsequenz für die Verdachtserhärtungsphase sollte man auf jeden Fall vermeiden, Dateisysteme zu beschreiben – auch der Gefahr wegen, dass Platz gelöschter Dateien überschrieben wird. Wenn nicht unbedingt nötig, sollten man auch keine Dateien zum Lesen öffnen[12]. Als Vorsichtsmaßnahme oder um Beschädigungen des Zugriffszeitstempels gleich aus dem Wege zu gehen, sollte man Dateisysteme, sofern möglich, am laufenden System so neu einhängen, dass der Zugriffszeitstempels nicht modifiziert wird (Linux: `mount -o remount,noatime <dir>`). Dateisysteme, die keine Dateien zum Schreiben geöffnet haben, kann man im ro-Modus neu einhängen.

Anhand der in 2.3.2.1 diskutierten Rootkit-Techniken sollte nachvollziehbar sein, dass man einem kompromittierten System überhaupt nicht trauen kann. Dies betrifft alle Binärdateien (auch die Shell), Systembibliotheken und den Kernel. Unbedingt muss ab der Voruntersuchung zumindest mit vertrauenswürdigen, statisch gelinkten Werkzeugen gearbeitet werden. Auf das noch „lebende System" gelangen diese – physikalischen Zugang vorausgesetzt – per CD/DVD oder USB-Stick bzw. -Platte. (Wobei ersterem aufgrund der fehlenden Möglichkeit zur Manipulation durch Beschreiben der Vorzug zu geben ist.) Falls im größeren Rechenzentrum vorhanden und bereits eingehängt, kann man auch die Forensik-Werkzeuge via NFS, AFS oder Samba auf einen sicheren Server gelegt, zur Verfügung (`ro`-Modus!) stellen. Bei gehosteten Servern ohne physikalischen Zugang sollte man die Werkzeuge in ein dediziertes Verzeichnis kopieren und den Zugriff auf diese wenigstens mit Unix-ACLs auf das nötige Minimum reduzieren.

3.3.1 Woher nehmen?

Es gibt eine Reihe mehr oder weniger guter „Werkzeugkästen" für eine Open-Source-basierte Digitale Forensik, eine gute Übersicht bietet [18]. Die meisten davon beinhalten zwei grundlegende Sammlungen von forensischen, Open-Source-basierten Werkzeugen:

- The Sleuth Kit (kurz; TSK) von Brian Carrier
- The Coroner's Toolkit (kurz TCT) von Dan Farmer und Wietse Venema

Die Helix-CD [19] ist ein solcher Werkzeugkasten, die die Firma e-fense unter GPL zur Verfügung stellt und recht regelmäßig pflegt. Sie basiert auf einer modifizierten Knoppix-Distribution und lässt sich für alle Phasen der digitalen Ermittlung verwenden (Verdachtserhärtung, Datensicherung, Post-Mortem-Analyse). Zwei wichtige Einschränkungen: Seit einiger Zeit sind die Solaris-Binärprogramme von der CD verschwunden[13], und zum Sichern von Prozessspeicher fehlt beispielsweise `pd` von Tobias Klein oder `pcat` aus dem TCT. Wer häufiger mit Sicherheitsvorfällen befasst ist, wird früher oder später die sonst recht brauchbare CD nach eigenem Gusto ergänzen wollen.

3.3.1.1 Wer suchet, der findet

Aus forensischer Sicht wäre eine bereits geöffnete und forensisch „koschere" Shell für den Anfang der Suche am besten, da schon beim Login via Konsole oder per SSH eine ganze Reihe Dateien „angefasst"

---

[11] Manche Dateisysteme wie NTFS oder ext2/3 haben einen vierten Zeitstempel.

[12] Der Autor kann sich gut erinnern, dass vor einigen Jahren ihm stolz jemand berichtete, dass der Rechner ja nicht das XYZ-Rootkit haben kann, da er ein rekursives `find` ausgehend von „`/`" plus `strings` nach der bekannten Zeichenkette vorgenommen hatte.

[13] Sie sollten extra herunterzuladen sein über die Webseite, jedoch funktioniert der Link zur Zeit der Drucklegung nicht. Die Programme waren nicht geeignet für Solaris 10.

werden und man sich schon nicht sicher sein kann, ob die Shell, eine von der Shell dynamisch geladene Bibliothek oder eine der Startup-Dateien komprimiert ist.

Grundsätzlich und egal, was man verwendet, man sollte unmittelbar nach dem Einhängen der CD/des USB-Sticks eine Reihe von Vorsichtsmaßnahmen treffen:

- `$PATH` kontrollieren, als erstes bzw. wenn möglich als einziges im Pfad sollten die Programme der CD stehen, kein Punkt am Ende
- keine History-Datei wegschreiben lassen (`HISTFILE=/dev/null`)
- Das Shell-Environment inspizieren, auf `$LD_LIBRARY_PATH` und `$LD_PRELOAD` achten und ggf. ein `unset` eingeben
- nach dem Anmelden eine statisch gelinkte Bash, beispielsweise von der Helix-CD, benutzen
- `mount -o remount,noatime /`, ggf andere Dateisysteme mittels `ro` neu einhängen

Ein paar Anregungen für Anhaltspunkte:

- Offene Sockets plus zugeordnete Programme anschauen (`lsof -i -Pn, netstat -atupn`)
- Wer ist/war auf dem System und welche Prozesse laufen?
- läuft der Syslog, schreibt er was ins Log, fehlen MARK-Meldungen?
- Shell-History von Root und Benutzern anschauen, Zeitstempel davon
- Ist die Netzschnittstelle im Promiscuous-Modus?
- Bei verdächtigen Aktivitäten `/proc/$PID/{env,cmdline,cwd,exe}` anschauen

Invasiv, zerstört Zugriffszeitstempel (siehe Remount-Option oben):

- `rpm -Va` bzw. `debsums -s` (sinnvoll, solange der Eindringling nicht die lokale DB kompromittiert hat)
- `rkhunter` (`chkrootkit`)

Nachdem sich der Verdacht erhärtet hat, sollten die unternehmensinternen Notfallprozeduren befolgt werden. Im Regelfall bedeutet dies, dass der bisherige Befund eskaliert wird und dass eine übergeordnete Stelle dann die weiteren Maßnahmen vorgibt.

4 Sicherungswerkzeuge

Bevor dieses Paper die weitere Vorgehensweise zur Sicherung erläutert, sollen zunächst ein paar Basiswerkzeuge vorgestellt werden.

Zur Sicherung von volatilen Daten zu sichern und zum Duplizieren von Festplatteninhalten gibt es eine Reihe Möglichkeiten. Dies gilt ebenso für die Auswertung der digitalen Beweise bei der Post-Mortem-Analyse. Abseits der oben erwähnten Open-Source-basierten CDs haben sich gerade bei professionellen Computerforensikern – aus Open-Source-Sicht muss man sagen „leider“ – kommerzielle und bis auf ein bis zwei Ausnahmen nur unter Windows laufende Werkzeuge etabliert, die bei der Festplattenduplizierung und Post-Mortem-Analyse verwendet werden.

4.1 Closed-Source-Werkzeuge

Sowohl deren Analyseprogramme als auch die zugehörigen *Imager* sind nicht zuletzt durch die Verwendung eines GUIs einfach zu bedienen und werden relativ häufig gepflegt. Die recht mächtigen Analyseprogramme können mit einer Reihe Dateisysteme auf Dateisystemebene umgehen, auch die gängigsten von Linux (ext2/2, reiser3fs) zählen dazu. Für kleinere Organisationen lohnt sich in den seltensten Fällen eine Anschaffung. Aufgrund der Tatsache, dass solche Forensik-Programme grundsätzlich BLOBs sind, weiß man nicht, wie sie intern arbeiten, noch kann man sich bei den Produkten sicher sein, dass sie keinerlei Hintertüren enthalten. Für die Verarbeitung von Daten, die von straf- oder zivilrechtlicher Relevanz und nicht selten für die Organisation von heikler Natur sind, eine nicht-optimale

Voraussetzung. Zumal wenn man aufgrund von etwaigen Rahmenbedingungen die Forensiksoftware nicht in abgesicherten Laborumgebungen verwenden kann.

Einige der kommerziellen Forensikprogramme haben einen eigenen Imager. Folgende Closed-Source-Imager kommen generell in Frage:

Produkt	**Preis**	**Hersteller**	**Gehört zu Forensik-Software**	**Ausgabeformat**
LinEn	umsonst[14]	Guidance Software	EnCase	Expert Witness Compression Format[15]
FTK Imager	$89	Access Data	FTK (Forensik Tool Kit)	→ EnCase, SMART, dd
X-Ways Capture	240 €	X-Ways Software Technology AG	X-Ways Forensics	→ EnCase, dd
SMART Acquisition	[16]	SMART (Storage Media Archival Recovery Toolkit)[17]	ASR Data	SMART
Iximager[17]	umsonst[18]	IRS-CI	iLook	IDIF, IRBF, IEIF (proprietäres Format)

4.2 Open-Source-Basiswerkzeuge

Unter Linux wie anderen Unices gibt es eine Reihe offener Werkzeuge für die Kommandozeile, die mindestens genauso leistungsfähig sind. Sie eignen sich für das Sichern volatiler Daten und für das „Imaging“ der Festplatte. Im wesentlichen handelt es sich dabei um zwei Basiswerkzeuge bzw. ihre Ableger:

- `dd`, zum Kopieren von Block- und ggf. Hauptspeicherinhalten
- `netcat`, senden und empfangen von Daten-Streams über das Netz

Üblicherweise werden beide kombiniert in einer Pipe à la:

```
dd if=/dev/hda bs=65535 | netcat <remotehost> 4242
```

Bei einer Festplatte wie hier im Beispiel ist bei fehlerfreien Medien aus Performancegründen eine größere Blockgröße als der Standard von 512 Bytes zu empfehlen. Mittels `gzip` beispielsweise können Daten in einem Rutsch zusätzlich komprimiert werden:

```
dd if=/dev/hda bs=65535 | gzip -c | netcat <remotehost> 4242
```

Wobei auf *remotehost* und Port 4242 ein `netcat`-Listener laufen muss,

```
netcat -l -p 4242 >hda.img
```

der die übers Netz gesendeten Daten wegschreibt. Damit dies gerichtsverwertbar ist, sollte der Name des Quellobjekts, das genaue Datum des Transfers und die Prüfsumme mit auf dem Ziel gespeichert werden. Per Hand angeworfen, wird dies bei der Unmenge von Live-Daten schnell sehr mühselig und

---

[14] Ist auf der Helix-CD zu finden, allerdings sind keine Quellen verfügbar.

[15] Begrenzt auf 2 GB, Dokumentation siehe [20].

[16] Nicht einzeln erhältlich, nur der Vollständigkeit halber aufgeführt.

[17] Läuft ausschließlich unter Linux.

[18] Nur für Angehörige von Strafverfolgungsbehörden.

fehlerträchtig, sodass es aus verschiedenen Quellen Skripte gibt, die dies automatisieren und unter gewissen Rahmenbedingungen als Beweis zugelassen werden. Mehr dazu siehe unten.

Sowohl das „normale“ `netcat`, als auch `dd` sind allerdings nur sehr begrenzt für forensische Aufgaben zu empfehlen, da beide jeweils eine Reihe von Nachteilen besitzen.

4.2.1 Disk Dump

`dd` aus den „GNU fileutils“ ist ohne weiteres nicht empfehlenswert wegen seines Verhaltens bei Hardware-bedingten Lesefehlern, auch zeigt es nur beim Senden eines USR1-Signals[19] den Status des Kopiervorgangs an. Gerade in der Digitalen Forensik, wo Daten im Gigabyte-Bereich und mehr kopiert werden, möchten die meisten Forensiker jedoch wissen, an welcher Stelle des Dupliziervorgangs sie sich befinden. Folgende Alternativen existieren:

- `sdd`: von Jörg Schilling, vor allen Dingen schneller
- GNUs `ddrescue`: etwas toleranter bei Lesefehlern
- `dd_rescue`: von Kurt Garloff – toleranter bei Lesefehlern, Möglichkeit zum Rückspulen, um sich defekten Blöcken von hinten zu nähern
- `rdd`: vom Netherlands Forensic Instituts – soll auch robuster gegenüber Lesefehlern sein. MD5-, SHA1-Hashes, kann implizit übers Netz übertragen und beinhaltet `split`
- `dcfldd`: Status während des Kopierens, verschiedene Hash-Algorithmen (MD5, SHA1, SHA256, SHA512 u.a.), Ausgabe des Hashes in eine Datei, Ausgabe des Error-Logs in eine Datei, implizites Split
- `dccidd`: Version vom Defense Cyber Crime Institut. Der Nachfolger von `dcfldd`, nur per E-Mail auf Anfrage erhältlich
- `dc3dd`: eine brandneue Version von Jesse Kornblum: Gepatchte Version von GNU `dd`, Features vergleichbar mit `dcfldd`, kombiniertes Log für Hash und Fehler

Was die Anzahl der genullten Blöcke bei Lesefehlern angeht, scheinen sich laut einer Präsentation/eines Vortrags auf dem letztjährigen Digital Forensics Research Workshop die Werkzeuge unterschiedlich zu verhalten [21]. `dd` sowie dessen Abkömmlinge überspringen schlechte Sektoren, sofern sie entsprechend instruiert werden[20]. Grundsätzlich sollte bei zu erwartenden Lesefehlern die kleinst mögliche Blockgröße verwendet werden, da sonst folgende, nicht-defekte Blöcke übersprungen werden.

Der Autor hat gute Erfahrungen mit `dcfldd` gemacht, typischerweise mit folgendem Aufruf:

```
dcfldd if=eingabe conv=sync,noerror bs=8192 hashlog=dateiname1  \
       hash=sha256 hashwindow=bytes errlog=dateiname2 ....
```

Für `hashwindow` ist ein Wert zu empfehlen, der es einem je nach persönlicher Geduld erlaubt, einen Fortschritt beim Duplizieren zu erkennen. `errlog` sollte auf jeden Fall nach Ende des Kopiervorgangs untersucht werden. Findet man hier wider Erwarten einen Hinweis auf Lesefehler, ist der gesamte Vorgang mit der Standardblockgröße von 512 Bytes zu wiederholen (weglassen des Parameters `bs`). Vermutet man defekte Sektoren oder möchte einfach auf der sicheren Seite sein, ist dies von Anfang an der beste Weg.

4.2.2 Netcat

Bei `netcat` – je nach Linux-Distribution existiert entweder eine etwas mächtigere GNU-Version[21] oder die Ursprungsversion von „*Hobbit*“[22] – ist zu bemerken, dass es nachteiligerweise weder die Streams

---

[19] Die `info`-Datei auf einem Linux-System Ende 2007 erwähnte hierzu das SIGINFO-Signal, was es unter Linux nicht gibt, eher aber SIGPWR entspricht. Richtig aber ist, hier wie oben erwähnt, SIGUSR1.

[20] Beim GNU-dd und einigen anderen ist `bs=512,conv=noerror,sync` dazu nötig.

[21] Als Autor ist nur dieses Pseudonym angegeben.

[22] Der Parameter -e erlaubt normalerweise einen Listener als Programm zu übergeben. Aus gutem (Sicherheits-)Grund ist dies häufig bei der Ursprungsversion nicht mit einkompiliert, siehe Parameter

übers Netz verschlüsselt, noch eine Authentifizierung vorgesehen ist. In suspekten Umgebungen also nicht das Mittel der Wahl für eine korrekte Datenerfassung. Folgende Alternativen stehen zur Verfügung:

- `cryptcat`: Netcat plus Twofish-Verschlüsselung. Default-Passwort ist `metallica`, zu ändern mit `-k`
- `socat`: sehr mächtig, Blowfish-Verschlüsselung, Authentisierung mittels X.509-Zertifikaten, IPv6, HTTP- und Socks-Proxy-Support, Raw-IP, u.v.m.
- weitere: `aes-netcat`, `sbd`, `ncat`, `netcat6`

5 Daten sichern – Evidence Collection

Die Daten- bzw. Beweissicherung wird folgenderweise vorgenommen, ungefähr in der Reihenfolge der Halbwertszeit der digitalen Beweise:

1. Sicherung volatiler Daten
2. Rechner außer Betrieb nehmen
3. Forensisches Duplikat der ausgebauten Festplatte erstellen
4. Original und Duplikat sowie gedruckte Dokumentation sicher aufbewahren, fern von in Frage kommenden Verdächtigen

Bei dieser Reihenfolge (2,3) handelt es sich um eine *Dead Acquisition*. Besteht keine Möglichkeit, die Festplatte **nach** dem Ausbauen zu Duplizieren – etwa weil der Plattenadapter ein seltenes Exemplar ist[23] – kann man auch eine *Live Acquisition* vornehmen, das heißt, man kopiert die Platte übers Netz **bevor** man den Rechner außer Betrieb nimmt. Letzteres sollte man wirklich nur in diesen Fällen in Erwägung ziehen. Eine kompromittierte Maschine zum Kopieren zu nehmen, ist aufgrund der Möglichkeiten, die ein potenziell gehackter Kernel erlaubt, keine gute Idee. Auch ist man unter Umständen bei der Aufdeckung von HPAs/DCOs etwas eingegrenzt, mehr dazu unten.

Beim Kopieren übers Netz ist anzumerken, dass in den häufigsten heute anzutreffenden Konstellationen – 100 Mbit-Netz plus aktuelle Festplatten – das 100 Mbit-Netz den Flaschenhals darstellt. Bei Nettotransferraten von 10 MB/s benötigt man für 100 GB 2,7 Stunden; lokal mit modernen SATA-Festplatten, die ein „Perpendicular Recording“ haben, würde diese Zeit auf unter eine Stunde schrumpfen.

5.1 Volatile Daten sichern

Mit solchen Daten sind alle Informationen gemeint, die bei der Außerbetriebnahme des Rechners in irgendeiner Weise verloren gehen würde, dies sind:

- der gesamte Hauptspeicherinhalt
- die von einzelnen Prozessen belegten Speicherinhalte
- allgemeine und Statusinformationen (wie `arp`, `ifconfig`, `route`, offene Netzverbindungen, geladene Kernel-Module, auch Infos über eine vorhandene HPA gehört dazu)
- Unter Solaris meistens, unter Linux selten: ein im RAM vorhandenes `/tmp`
- verschlüsselte Dateisysteme
- ggf. Swap

Besonders das Vorhandensein einer Verschlüsselung wird bei vielen Incident-Response-Verantwortlichen vergessen. Dem Autor wurden von einem Kunden Daten übergeben, bei dem sich nach einiger Recherche im Verlaufe der Untersuchungen herausstellte, dass der Innentäter wahrscheinlich wichtige Daten in

-DGAPING_SECURITY_HOLE, bei der GNU-Version leider schon.

[23] Bei RAID-Leveln, mit der die Forensik-Software während der P.M.-Analyse nicht klar kommt, empfiehlt sich dies auch.

einem Truecrypt-Image verbarg. Da der Systemverantwortliche vor Ort dies nicht vor der Außerbetriebnahme überprüft hatte, blieben die Daten dort drin für immer verborgen.

Eins der vielen Skripte zur Erfassung volatiler Daten befindet sich auf der Helix-CD (`linux-ir.sh`). Neben dem bereits o.g. Fehlen von Werkzeugen zur Sicherung von Prozessspeicher, kopiert dies leider ebenso Daten, die später vom Duplikat erhalten werden können (z.B. `/etc/resolv.conf`) und nimmt überflüssigerweise MD5-Prüfsummen einer ganzen Reihe von Systemprogrammen, ohne den Kernel vorher zu instruieren, die „atime" des Dateisystems nicht zu aktualisieren. Besser ist das `ir-linux.sh` -Skript auf der Forensik-CD aus der iX [22], die auch via computer-forensik.org (Alexander Geschonneck) erhältlich ist [23]. Der Autor dieses Papers hält eine fortentwickelte Version eines des Helix-Skriptes unter [24] vorrätig.

Die Ausgabe des verwendeten Skriptes sollte man entweder mit der Lieblings-Netcat-Variante auf einem Rechner im LAN sichern oder lokal auf einem lokalen USB-/Firewire-Medium. Bei letzterem ist zu beachten, dass für die Sicherung des RAMs genug Speicherplatz zur Verfügung stehen sollte.

Die resultierende Textdatei sowie die Speicherabzüge sollten Prüfsummen und Datumsstempel bekommen. Sinnvoll ist es, beides plus Textdatei auszudrucken (beispielsweise mittels `a2ps`) und mit Datum, Ort und am besten mehr als einer Unterschrift zu versehen. Ein unabhängiger Zeuge verleiht dem Dokument unter Umständen mehr Beweiskraft.

5.2 Außerbetriebnahme: Herunterfahren versus Strom wegnehmen

Nachdem alle volatilen Daten gesichert worden sind, ist es wichtig, den Rechner **richtig** außer Betrieb zu nehmen. Dies bedeutet im einfachsten Fall schlicht, den Netzstecker zu ziehen. Sauber herunterzufahren würde den Zugriffszeitstempel unzähliger Dateien modifizieren, einige Dateien löschen, in viele offene Dateien schreiben und einige sogar neu anlegen, was nicht reallozierten Platz gelöschter Dateien überschreibt. Das Ausschalten sollte nie mittels Ausschaltknopf geschehen. Viele Betriebssysteme sind so konfiguriert, dass sie den Rechner dann nicht ausschalten, sondern herunterfahren.

Einen kleinen Haken hat die Sache dennoch: Unter Umständen ist bei der Post-Mortem-Analyse das Einhängen von Dateisystemen schwieriger als beim Ausschalten, da Journal-Logs[24] entweder zurück gespielt oder mit dem Dateisystem-Debugger als ungültig markiert werden müssen.

Ein paar Tipps: Zwei [25] `sync`-Befehle vor dem Steckerziehen halten die Dateisysteme unter Umständen konsistenter. Eine bessere Alternative, die unter Linux immer funktionieren sollte: Falls SysRq [26] konfiguriert ist – anschalten gegebenenfalls per `sysctl -w kernel.sysrq=1` – hilft die SysRq-Sequenz S-S-U-O, was zweimal dem Kernel befiehlt, etwaige im Speicher befindliche Dateisystempuffer wegzuschreiben (Sync), gefolgt von einem ro-Mount (U) aller eingehängten Dateisysteme und einem Poweroff (O)[25].

Falls man eine serielle Konsole hat, kann man die Auffassung des Kernels bezüglich seiner Task-, Liste, seiner Register/Flags und aktuelle Timer-Informationen (SysRq-T-P-Q) vor dem Gnadenstoß noch dort ausgeben und in eine Textdatei (mit Prüfsumme und Datum versehen) sichern lassen.

5.3 Erstellen des forensischen Duplikats

Wie oben erwähnt, empfiehlt sich nach dem Ausschalten das Ausbauen der Festplatte(n), danach die Erstellung der forensischen Kopie (Dead Acquisition), wenn möglich mit Hilfe eines Write-Blockers, der verhindert, dass irgendwelche Daten des Beweises – der Originalfestplatte – nach dem Ausbau verändert werden.

Falls man keinen Write-Blocker zur Hand hat, sollte man zur Duplikation des der Festplatte(n) das System immer von CD – wie zum Beispiel der Helix-CD – zum Leben erwecken. Das System von einer anderen Platte ohne Write-Blocker zu starten, indem man das Beweisstück etwa als ATA-Slave oder mit einer SCSI-ID konfiguriert, die „wahrscheinlich" nicht zum Booten benutzt wird, sind zu riskant für die

---

[24] Ein Aushängen oder `mount -o remount,ro /<dateisystem>` mag helfen.

[25] Falls SysRq-O nicht funktioniert, sollte es ein SysRq-B (Reboot) auf jeden Fall tun, allerdings sollte man danach schnell genug mit dem Netzstecker sein.

Unversehrtheit der Beweise, es sei denn, dies ist ein eingespielter Vorgang, der schon zur Genüge exerziert worden ist.

Das Duplikat kann man entweder direkt auf eine oder mehrere Sicherungsplatten (USB, Firewire, ATA) kopieren oder auf einen entfernten Rechner im Netz mittels der oben vorgestellten Werkzeuge.

5.3.1 Umfang der Kopie

Kurz formuliert: Es muss die gesamte Platte dupliziert werden. Nur die Partitionen zu kopieren, reicht auf keinen Fall. In unbenutzten Plattenbereichen lassen sich Informationen verstecken, eine etwa vorher dort angelegte Partition ließe sich beispielsweise zur (schlechten) Verwischung von Spuren aus der Partitionstabelle löschen.

Zusätzlich existieren je nach Betriebssystem und Partitionierung von der Plattengeometrie vorgegebene Lücken zwischen den Partitionen, wo der Täter unter Umständen Informationen drin versteckt haben kann oder die vorher Teil einer Partition war. Außerdem existieren Rootkits, die Teile ihres Codes im Masterboot-Record verstecken [27].

Zumindest bei der Live Acquisition sollte auch die Partitionstabelle durch ein Forensikwerkzeug wie `mmls` mit ausgegeben werden, sofern dies noch nicht während der Erfassung der volatilen Daten geschehen ist.

Auf der/den Zielfestplatte(n) ist am besten eine große Partitionen vorzubereiten und die Daten dort hinein zu kopieren. 2 GB-Grenzen spielen bei allen nativen Linux-Dateisystemen glücklicherweise keine Rolle mehr. Von der gesamten Platte als auch vom Image und der Ausgabe von `mmls` sind Prüfsummen zu erstellen, diese zu vergleichen und mit Datum der Dokumentation hinzuzufügen.

5.3.1.1 HPA, DCO

Eine weitere Herausforderung beim „Imaging“ stellen mittels ATA-Kommandos am Ende der Festplatte reservierte Sektoren (DCO, HPA) dar. DCO steht für *Device Configuration Overlay* und ist ein Standard der ATA-Version 6, HPA – *Host Protected Area*[26] – existiert seit ATA-Version 4. Ist der Plattenbereich durch HPA oder DCO begrenzt worden, liefert der ATA-Controller nur Blöcke bis zu dem nicht reservierten Bereich Sektoren zurück. Mit HPA versteckte Bereiche erkennt Linux-Kernel 2.6 bei PATA-Disks automatisch beim Start (`dmesg | grep kernel | grep Protected`). Alternativ gelingt dies mittels einer Reihe anderer Werkzeuge, worauf hier kurz eingegangen werden soll.

Mit `hdparm` funktioniert die Erkennung einer HPA bei PATA-Festplatten immer, leider bei den mittlerweile weit verbreiteten SATA-Platten erst ab Version `8.1`[27]. Nötig ist dafür ein 2.6.24-Kernel[28] bzw. eine jüngerer mit Backport-Patch. Der Parameter -N zeigt und modifiziert ggf. die HPA:

```
myhost:/tmp/hdparm-8.1 0# ./hdparm -N /dev/sda

/dev/sda:
 max sectors   = 312581808/312581808, HPA is disabled
myhost:/tmp/hdparm-8.1 0#
```

Beim Booten eines Systems mit der oben erwähnten Helix-CD wird in einigen Fällen automatisch die HPA bei der Originalplatte deaktiviert. Das Vorhandensein der HPA und die Deaktivierung muss auf jeden Fall dokumentiert werden.

Weitere Open-Source-Werkzeuge, die im Umgang mit HPAs nützlich sind:

- `disk_stat` (aus TSK), Autor: Brian Carrier
- `hpafs`, HPA einhängen via FUSE. Autor: Paul Bolle

---

[26] Thinkpads der 40er Serie und manche neuere haben ab Werk die Recovery-Partition versteckt, mit dem sich die mitgelieferte Windows-Partition ggf. reparieren lässt. Im IBM-Jargon heißt der Bereich „Pre Desktop Area“.

[27] Mit dieser Version ist es auch möglich, Sektoren mittels `--make-bad-sector` als Defekt zu definieren [28] und so dem oberflächlich suchenden Auge zu entziehen.

[28] Siehe `kernel_patches/README` im Quellcode von `hdparm 8.1`.

- `fiesta`, Backup des HPA, Autor: „krzakan“
- `hpatools`: Zum Kopieren der HPA, Autor: Christof Böckler

5.4 Abschließende Maßnahmen

Die Originalfestplatte sollte eindeutig in den begleitenden Dokumenten anhand der Aufschrift als Beweis identifiziert werden[29] („Festplatte Seagate *ST3000XXX*, Seriennummer *ABCABC*, ausgebaut aus Computer *CDECDE* am 29.3.2007 um 12:42 Uhr im Betrieb des Kunden.Ort *OOO*, Datum *DDD*, Name1, Name2, Unterschriften“). Wie oben erwähnt, sollte sie fern von möglichen Verdächtigen, also auch Innentäter, aufbewahrt werden. Ebenso sollte mit dem forensischen Duplikat und der gedruckten Dokumentation verfahren werden, die beide als Grundlage für die Post-Mortem-Analyse dienen.

5.5 Nachbereitung, „lesson learned?“

Nach dem abschließenden forensischen Report sollte eine Bewertung des Incidents innerhalb der Firma erfolgen, sowohl in technischer als auch organisatorischer Hinsicht erfolgen, siehe Kapitel 2.3.1.

6 Literaturverzeichnis

[1] http://de.wikipedia.org/wiki/Forensik, Wikipedia zum Wort Forensik

[2] http://www.dtic.mil/whs/directives/corres/html/522022m.htm, National Industrial Security Program Operating Manual, DoD 28.2.2006

[3] http://www.bsi.de/gshb/index.htm, BSI-Grundschutzkataloge

[4] http://www.bsi.de/literat/bsi_standard/standard_1001.pdf, Managementsysteme für Informationssicherheit (ISMS), BSI 100-1

[5] http://en.wikipedia.org/wiki/Rootkit, Wikipedia zum Thema Rootkit

[6] http://www.rootsecure.net/content/downloads/pdf/unix_rootkits_overview.pdf, Anton Chuvakin (iDEFENSE Labs) „An Overview of Unix Rootkits“

[7] http://www.magellan-net.de/honeynet/papers/thesis_sven_mueller.pdf, Sven Müller, Diplomarbeit RWTH Aachen 2005, „Planung und Realisierung eines Honeynet zur Analyse realer Angriffe aus dem Internet“

[8] http://phrack.org/issues.html?issue=58&id=7#article, Phrack #58 (2001) über SucKIT „Linux on-the-fly kernel patching without LKM“

[9] http://www.dfn-cert.de/team/bunten/rootkits_dimva2004.pdf, Andreas Bunten, DFN-Cert Services GmbH „Unix- und Linux-basierte Kernel-Rootkits“

[10] http://www.phrack.org/issues.html?issue=59&id=10#article, »Jan K. Rutkowski« in Phrack #59 (2002) über „Execution path analysis: finding kernel based rootkits“

[11] http://invisiblethings.org/papers/rootkits_detection_with_patchfinder2.pdf, „Detecting Windows Server Compromises with Patchfinder2“, Joanna Rutkowska, 1/2004

[12] Dirk Wetter, iX 3/2005, S. 12, „Monokulturen, Zecken und Kuckuckseier“

[13] http://www.trapkit.de/research/adv_expl/advanced_exploiting_tk_defense2005.pdf, „Advanced Exploiting“, Tobias Klein, IT-Defense 2005

[29] Auch wenn das Incident-Response-Skript einen entsprechenden `hdparm`-Aufruf hatte.

[14] http://www.rootkit.com/project.php?id=12, FU-Rootkit

[15] http://www.phrack.org/issues.html?issue=63&id=8#article, Phrack #63 (2005) Sherri Sparks, Jamie Butler über „Shadow Walker, Raising The Bar For Windows Rootkit Detection“

[16] http://www.digital-evidence.org/papers/tribble-preprint.pdf, Brian D. Carrier Joe Grand „A Hardware-Based Memory Acquisition Procedure for Digital Investigations“

[17] http://www.vinecorp.com/ndiff/, ndiff

[18] http://www.forensicswiki.org/index.php?title=Tools, Forensik-Wiki über Tools

[19] http://www.e-fense.com/helix/, Helix-CD-Downloads und Informationen

[20] http://www.asrdata.com/SMART/whitepaper.html, Expert Witness Compression Format Specification

[21] http://dfrws.org/2007/proceedings/p13-lyle.pdf, „Issues with imaging drives containing faulty sectors“, James R. Lyse, Mark Wozar, DFRWS 2007, Elsevier Ltd.

[22] iX 07/2007, CD in Juli-Ausgabe der iX 2007

[23] http://computer-forensik.org/tools/ix/, Download derselben CD und Updates.

[24] http://software.drwetter.de/ir/, Incident-Response-Skript des Autors

[25] http://www.opengroup.org/onlinepubs/007908799/xsh/sync.html, Online-Hilfeseite der Open Group zu sync

[26] file:///usr/src/linux/Documentation/sysrq.txt, lokale Dokumentation zu Linux' SysRq

[27] http://www.symantec.com/enterprise/security_response/weblog/2008/01/from_bootroot_to_trojanmebroot.html, Windows Rootkit im MBR

[28] http://lkml.org/lkml/2008/2/16/126/ Mark Lords Ankündigung zu `hdparm` 8.1 auf der Linux-Kernel-Mailing-Liste

Über den Autor

Dirk Wetter (`http://drwetter.de`) ist Chemiker und hat in Festkörperphysik promoviert. Seine universitären Unix-Ambitionen machte er danach (1996) zum Beruf, war als Angestellter in großen HPC-Rechenzentren in Hamburg und für mehr als zweieinhalb Jahre an der Ostküste der USA als Linux-Architekt und Solaris-System-Engineer beschäftigt. Gleichzeitig gehörten immer Sicherheitsfragen der Unternehmens- und Behörden-IT in technischer und organisatorischer Hinsicht zu seinem Verantwortungsbereich.

Seit 2003 ist er Berater für internationale Kunden im Bereich IT-Sicherheit und Open-Source-Technologien. Er hat zahlreiche Fachbeiträge in verschiedenen Print- und Onlinemedien veröffentlicht, ist Co-Autor eines Buches, gestaltet den LinuxTag mit und ist in zahlreiche Konferenzen und Events der GUUG bei der Organisation, Gestaltung und in Programmkomitees und vieles mehr involviert.