Prof. Dr. Stefan Wohlfeil

Kurs 01868

Sicherheit im Internet I - Ergänzungen

LESEPROBE

Fakultät für
Mathematik und Informatik

FernUniversität in Hagen

Inhaltsverzeichnis

Kapitel 1

Benutzersicherheit

Der Autor: Prof. Dr. Stefan Wohlfeil, geb. 12.12.1964

- Studium der Informatik mit Nebenfach Elektrotechnik an der Universität Kaiserslautern (1984–1991)
- Wissenschaftlicher Mitarbeiter am Lehrgebiet Praktische Informatik VI der FernUniversität in Hagen (1991–1998)
- Promotion zum Dr. rer. nat. (1997)
- Mitarbeiter in der Deutsche Bank AG, Abteilung TEC — The Advanced Technology Group (1998–2002)
- Professor an der Fachhochschule Hannover, Fakultät IV, Abteilung Informatik; Arbeitsgebiet: Sichere Informationssysteme (seit 2002)

Liebe Fernstudentin, lieber Fernstudent,
herzlich willkommen beim Kurs über Sicherheit im Internet!

Diese Einführung soll Ihnen einen Überblick darüber geben, worum es im vorliegenden Kurs geht.

Inhalt des Kurses und Vorkenntnisse: Dieser Kurs richtet sich an Informatik-Studierende[1] und setzt die Kenntnis einiger Inhalte aus einem Informatik-Grundstudium voraus. Konkret sollten Sie bereits wissen, wie ein Computer prinzipiell aufgebaut ist, was ein Betriebssystem typischerweise macht und welche Möglichkeiten sich durch die Vernetzung, wie beispielsweise im Internet, für Anwender bieten. Diese Themen werden im Kurs *(01801) Betriebssysteme und Rechnernetze* behandelt. Weiterhin gehen wir davon aus, dass Sie bereits den Kurs *(01866) Sicherheit im Internet 1* bearbeitet haben.

Die Kurseinheit 1 beschäftigt sich mit den Themenbereichen:

- Anonymität
- Aktive Inhalte

[1] Hierzu gehören alle Studierenden, deren Curriculum einen Informatikbestandteil enthält wie beispielsweise auch Studierende der Wirtschaftsinformatik.

- Computer-Forensik

In Kurseinheit 2 werden dann die Themenbereiche Zugriffskontrolle und Benutzerauthentisierung behandelt. Im Bereich Zugriffskontrolle erfahren Sie, welche Mechanismen auf den Ebenen (1) Hardware, (2) Betriebssystem und (3) Software existieren. Bei der Benutzerauthentisierung werden die Bereiche (1) Passwörter und dabei dann das One-Time-Passwort-Verfahren, (2) Kerberos und (3) Biometrische Verfahren besprochen.

In Kurseinheit 3 geht es dann allgemein um Sicherheit in Kommunikationsnetzen. Konkret wird auf die weitverbreiteten Wireless LANs (WLANs) eingegangen und die speziellen Sicherheitsanforderungen dort werden vorgestellt. Weiterhin wird auf eine beliebte Anwendung, das Telefonieren über das Internet (Voice over IP, VoIP) eingegangen.

1.1 Anonymität im Internet

Die großen Datenmengen (eigene Homepage, Einträge bei sozialen Netzen wie XING oder Facebook, Blog-Beiträge, News-Beiträge usw.), die über einzelne Personen im Internet verfügbar sind, lassen sich vielfältig nutzen. Leider auch zum Nachteil eines Einzelnen. Daraus folgt der Bedarf, auch anonym im Internet kommunizieren zu können.

1.1.1 Begriffsbestimmungen

Köhntopp und Pfitzmann [KP04] schlagen eine Terminologie aus dem Bereich *Identity Management* vor. Sie bildet die Basis für die Begriffsbestimmungen hier im Kurs.

Definition

Anonymität (engl. anonymity): Im täglichen Leben kommt es immer wieder vor, dass man anonym auftritt. Dabei bedeutet anonym, dass man ohne Namen bzw. dem Namen nach unbekannt auftritt. Der Autor des Buches „Maximum Security“ hat es beispielsweise vorgezogen, nicht mit dem eigenen Namen als Autor aufzutreten, und hat stattdessen den Namen *Anonymous* gewählt. Man kann aus der Menge der potentiellen Autoren von Informatikbüchern also nicht das Individuum identifizieren, das das Buch geschrieben hat. Man kann allgemein sagen, dass man anonym im Rahmen einer „Anonymitätsmenge“ (engl. **anonymity set**) ist, wenn man aus dieser Menge heraus nicht eindeutig identifiziert werden kann.

Bei der Kommunikation zwischen zwei Parteien können die folgenden Formen der Anonymität auftreten:

Client-Anonymität

Senderanonymität: Hierbei tritt der Absender einer Nachricht anonym auf. Ein klassisches Beispiel hierfür ist der Anruf bei einer Beratungsstelle. In der IT-Welt nennt man diese Form auch **Client-Anonymität**.

Empfängeranonymität: In dieser Form bleibt der Empfänger einer Nachricht namenlos. Chiffreanzeigen in Zeitungen sind die „klassische“ Ausprä-

gung dieser Anonymität. In der IT-Welt spricht man auch von **Server-Anonymität**.

Server-Anonymität

Komplette Anonymität: Hier bleiben Sender und Empfänger ohne Namen.

Verkettbarkeit (engl. unlinkability): In der oben vorgestellten Definition des Begriffes Anonymität steckt noch eine Unschärfe. Der Autor bzw. die Autorin des Buches *Maximum Security* ist ja nicht wirklich namenlos. Die Person hat eine Identität, die allerdings nicht in einen Zusammenhang mit dem Buch gebracht werden kann. Der Begriff der Anonymität verbindet also letztlich Personen mit bestimmten Handlungen. Lässt sich dazwischen keine Verbindung herstellen, dann spricht man von Anonymität.

Verkettbarkeit

Das Konzept der Verkettbarkeit betrifft nicht nur Personen und Handlungen, sondern es kann auch auf Nachrichten, Ereignisse o. Ä. angewendet werden. Eine Nachricht und eine Person sind nicht verkettbar, wenn man nicht sicher wissen kann, ob die Nachricht an diese Person gerichtet ist oder von dieser Person stammt. Zwei Nachrichten sind nicht verkettbar, wenn man nicht sicher sagen kann, ob beide Nachrichten denselben Absender (oder Adressaten) haben. Das zweite Beispiel zeigt die zeitliche Dimension der Verkettbarkeit. Zukünftige Ereignisse können eine bisher unbekannte Verkettung zwischen zwei bereits existierenden Objekten herstellen.

Im E-Commerce stellt sich das Problem der Verkettbarkeit bei Verkaufssystemen die auf Webservern basieren. Da HTTP ein zustandsloses Protokoll ist, kann ein Webserver zunächst keine Verkettung zwischen zwei verschiedenen HTTP-Anfragen herstellen. Hat ein Benutzer in einer Anfrage ein Produkt ausgewählt und will es auf der nächsten Seite bezahlen, dann muss der Webserver den Zusammenhang herstellen können. Dies geschieht häufig mit Hilfe von Cookies (vergleiche z. B. Kurs *(01866) Sicherheit im Internet 1*).

Pseudonymität: Eine „abgeschwächte“ Form, unbekannt zu bleiben, ist die sogenannte **Pseudonymität**. Hierbei wählt sich eine Person einen neuen Namen, das Pseudonym. Unter diesem Namen tritt die Person dann für einen gewissen Zeitraum einer bestimmten Gruppe anderer Personen gegenüber auf. Prominente Beispiele für Pseudonyme sind Künstlernamen. Ein Künstler tritt unter diesem Namen normalerweise für die Dauer seiner Karriere seinem Publikum gegenüber auf. Pseudonyme müssen nicht Namen im eigentliche Sinne sein. Sie können auch Zahlen oder beliebige Symbole sein. Der amerikanische Sänger *Prince* hatte nach einer Auseinandersetzung mit seiner Plattenfirma beispielsweise sein altes Pseudonym gegenüber der Öffentlichkeit annulliert und ein Symbol als neues Pseudonym gewählt. Alternativ war er auch als „The Symbol“ oder „The Artist formerly known as Prince“ aufgetreten.

Pseudonymität

Diese Beispiele nennt man auch **Personenpseudonyme**. Das Pseudonym ist fest einer bestimmten Person zugeordnet. Die Person benutzt es im Prinzip mit allen Kommunikationspartnern. In Abhängigkeit von der Betrachtungsweise (Person vs. Kommunikationspartner) kann man weitere Pseudonym-Typen unterscheiden. Ein **Rollenpseudonym** wird von einer Person immer dann benutzt, wenn sie sich in der zugehörigen Rolle befindet. Dabei ist es egal, wer die

Personenpseudonyme

Rollenpseudonym

Kommunikationspartner sind. Viele Leute, die gerne an Fantasy-Rollenspielen teilnehmen, benutzen in dieser Rolle stets dasselbe Pseudonym.

Beziehungspseudonym

Ein **Beziehungspseudonym** wäre ein Name, den eine Person immer einem bestimmten Kommunikationspartner gegenüber benutzt. In diesem Fall ist es egal, in welcher Rolle die Person sich befindet. Die Kombination in Form eines **Rollenbeziehungspseudonyms** wäre der Fall, wenn eine Person je nach Kommunikationspartner und eigener Rolle ein anderes Pseudonym benutzen würde.

Rollenbeziehungspseudonyms

Je nach Typ eines Pseudonyms kann man unterschiedlich viele Verkettungen erstellen. Ein „konstantes“ Personenpseudonym erlaubt es nicht nur *alle* Aktionen unter diesem Pseudonym der Person zuzuordnen, sondern umgekehrt auch (fast) alle Aktionen der Person zu verfolgen. Bei einem Rollenpseudonym kann man nur die Aktionen der Person in der jeweiligen Rolle miteinander verknüpfen. Ähnliches gilt bei Beziehungspseudonymen und den Aktionen gegenüber einem Kommunikationspartner. Noch weniger Verkettungen lassen sich erstellen, wenn Rollenbeziehungspseudonyme benutzt werden. Die kleinste Verknüpfungsmenge würde entstehen, wenn man bei jeder (Trans-)Aktion ein neues **Transaktionspseudonym** benutzen würde.

Transaktionspseudonym

Identitätsmanagement (engl. identity management): Benutzer von Computern oder von Netzen haben unterschiedliche Anforderungen an Anonymität. Konkret ist es verschieden wichtig, eine Verkettung zwischen Aktionen und Personen herstellen zu können. Bei der Anmeldung an einem Computer möchte man i. d. R. genau wissen, wer sich anmeldet. Beim Surfen im Netz möchte man jedoch nicht immer namentlich auftreten. Identitätsmanagement soll Benutzer darin unterstützen, diese verschiedenen Abstufungen der Verkettbarkeit zu realisieren.

Außerdem sollte Identitätsmanagement dabei helfen, bestimmte Eigenschaften einer Person preiszugeben, während andere Eigenschaften nicht preisgegeben werden. Ein Beispiel hierfür sind Dienste, die nur volljährigen Personen gestattet sind. Man braucht nun nicht die komplette Identität eines Benutzers (Name, Adresse usw.), sondern nur die Bestätigung über das Alter. Technisch muss man eine Verkettung zwischen Eigenschaft und einem Pseudonym herstellen.

1.1.2 Gründe und Gefährdungen der Anonymität

Es gibt viele gute Gründe, warum man in bestimmten Situationen bzw. bei bestimmten Handlungen den eigenen Namen nicht preisgeben möchte. Ein typisches Beispiel sind Beratungsgespräche, bei denen der Ratsuchende unbekannt bleiben soll (anonyme Alkoholiker). Ein weiterer Grund namenlos aufzutreten ist der Schutz der eigenen Privatsphäre (privat = persönlich, häuslich, vertraulich, nicht öffentlich).

Allgemein ist das Recht auf eine Privatsphäre ein sehr hohes Gut. Es ist in Deutschland bereits im Grundgesetz festgelegt.

Welche Gefahren ergeben sich, wenn die Anonymität des Einzelnen nicht ausreichend geschützt ist? Zunächst können Datensammlungen über eine Per-

son erstellt werden. Wenn man einmal zusammenstellt, in welchen Bereichen Daten über Personen erfasst werden, ergibt sich bereits eine lange Liste:

Finanzdaten: Bei welchen Banken und Finanzinstituten hat man Konten? Wie viel verdient jemand? Wohin und für was wird Geld überwiesen? Wo hebt man Geld ab, d. h. wo hält man sich auf? Wo kauft man ein (und bezahlt mit Karte)?

Konsumdaten: Mit Hilfe von Rabattkartensystemen erfassen Geschäfte die Einkäufe ihrer Kunden. Was wird wann und wo gekauft? Daraus lassen sich beispielsweise Ernährungsgewohnheiten ableiten, an denen Krankenkassen möglicherweise Interesse hätten.

Außerdem gewinnen die Geschäfte dadurch wichtige Daten über die Vorlieben und Interessen ihrer Kunden. Erst durch diese Kenntnis ist zielgruppenorientiertes Marketing möglich. Werbung aus dem Bereich „Golf“ wird also nur sportinteressierten, etwas älteren und einkommensstarken Personen zugestellt.

Aber nicht nur Werbung lässt sich durch diese Informationssammlung besser steuern. Auch die Kaufkraft und das bisherige Zahlungsverhalten können ausgewertet werden. Schon heute besitzen Versandhändler Informationen über bessere und schlechtere Wohngegenden. In Abhängigkeit von der Lieferanschrift werden dann verschiedene Bezahlmöglichkeiten angeboten. Aus einer „guten“ Gegend kann man auf Rechnung oder sogar auf Kredit kaufen, während Kunden, die in der „falschen“ Straße wohnen, nur gegen Vorkasse beliefert werden. Das heutige grobe Raster Wohnort kann zukünftig durch weitere Informationen verfeinert werden.

Kommunikationsdaten: Mit wem telefoniert jemand, wer bekommt E-Mails von wem? Telefongespräche und E-Mails können abgehört werden (und werden es auch). In den aktuellen Telekommunikationsgesetzen, konkret in der Telekommunikationsüberwachungsverordnung (TKÜV), werden die Betreiber von Telekommunikationsdiensten dazu verpflichtet, technische Überwachungsmöglichkeiten vorzusehen. Diese Überwachungsmöglichkeiten können dann beispielsweise von Strafverfolgungsbehörden benutzt werden. Zur Zeit (März 2010) gibt es eine intensive Debatte darüber, wie lange Verbindungsdaten gespeichert werden sollen/dürfen. Außerdem wird diskutiert, inwieweit Daten auf Vorrat gespeichert werden dürfen.

Bei Mobiltelefonen kann man zusätzlich orten, wo sich der Teilnehmer aufhält. Auf dem Land, wo Funkzellen eine große Fläche abdecken, ist eine grobe Lokalisierung möglich, in Städten mit kleineren Funkzellen kann der Standort genauer bestimmt werden. Wohin surfen Benutzer im Internet? Welche Seiten schauen sie an und welche Interessen kann man daraus ableiten? In welchen Diskussionsgruppen beteiligen sich Personen und welche Meinungen vertreten sie dort?

Aufenthaltsdaten: Die Zahl von Überwachungskameras (Flughafen, Kaufhaus, öffentliche Plätze, Hotels, Banken usw.) steigt ständig und gibt Auskunft über den Aufenthaltsort von Personen. Bei Flugreisen muss

man sich auch ausweisen und in den USA u. U. auch biometrische Daten (Fingerabdrücke) von sich selbst abgeben.

Verbunden mit neuen Techniken der Bild-Erkennung können die Informationen der Überwachungskameras nicht nur dazu dienen, einzelne Personen aus großen Personengruppen zu identifizieren. Die Erkennung von gesuchten Verbrechern ist hier das gerne genannten Einsatzbeispiel. Man kann[2] zusätzlich auch die anderen Personen identifizieren und die Datensammlung jeder dieser Personen um die Informationen über Aufenthaltsort und Aufenthaltszeit ergänzen.

Radio Frequency Identification (RFID)

Eine neue Technik vereinfacht das Erkennen/Finden von Objekten ungemein: **Radio Frequency Identification (RFID)**. Dahinter verbergen sich kleine integrierte Schaltungen (Chips), die zusammen mit einer Funkantenne beispielsweise in Etiketten untergebracht werden. Auch ohne eigene Stromversorgung kann man den Inhalt der Chips mit Hilfe von Antennen auslesen. Man nennt die Kombination aus Antenne und Chip dann **RFID-Tag** oder auch **Smart Label**.

RFID-Tag
Smart Label

Diese Technik wird heute beispielsweise bei der Identifikation von Haustieren eingesetzt. Das Tier bekommt ein RFID-Tag in einer kleinen Glaskapsel unter die Haut injiziert. In der Industrie besteht großes Interesse an RFID zur Lokalisierung von Waren oder anderen Gegenständen. Da RFID-Tags billig[3] herzustellen sind, können sie im Prinzip als Aufkleber auf jedem Artikel in einem Warenhaus eingesetzt werden. Sind dann auch Lesegeräte an allen Ein- und Ausgängen installiert, kann man den Weg jedes Artikels automatisch verfolgen. Die aufwendige Kontrolle einer Lieferung eines Lieferanten würde wesentlich vereinfacht. Auch die automatisierte Nachbestellung von Waren wäre möglich. Sind Lesegeräte zusätzlich an den Regalen montiert, dann könnte das System selbstständig erkennen, welche Regale aufgefüllt werden müssen.

Das System kann dann aber auch erkennen, wenn gekaufte Waren das Kaufhaus erneut betreten. RFID-Tags in Kleidungsstücken würden es also erlauben, den Käufer später erneut zu identifizieren. Dazu müsste der Käufer beim späteren Besuch nur das vorher gekaufte Kleidungsstück mit dem RFID-Tag anziehen. Der Betreiber hat dann technisch die Möglichkeit, die Interessen einzelner Kunden zu erfassen. Welche Artikel sind im Einkaufswagen vor dem Kunden? Vor welchen Regalen bleibt der Kunde wie lange stehen?

Diese Liste ist nicht vollständig. Zur Zeit kann man sich damit trösten, dass viele dieser Informationen zwar anfallen, aber auf verschiedene Institutionen verteilt sind. Die Aggregation *aller* Informationen ist also noch nicht möglich. Neben technischen Problemen (Vernetzung, Datenformate, Datenmengen usw.)

[2] Erst wenn die automatische Gesicht-Erkennung deutliche Fortschritte gemacht hat, wird dieses Verfahren tatsächlich möglich.

[3] Im März 2005 kann man RFID-Tags für etwa 50 Euro-Cent herstellen. Es wird erwartet, dass die Kosten durch technische Fortschritte und Massenproduktion in Zukunft sinken werden.

verhindern organisatorische Maßnahmen (wer gibt wem warum welche Daten preis? Wer darf Daten überhaupt weiter geben? usw.) die totale Überwachung. Für den Einzelnen ist es aber trotzdem wichtig, die Kontrolle über seine Privatsphäre am besten selbst zu behalten. Insbesondere im Internet sollte man wissen, wann man welche Informationen über sich preis gibt und wie man das technisch verhindern bzw. kontrollieren kann.

Identitätsdaten im Internet

Im Internet gibt man an verschiedenen Stellen seine Identität preis. Wenn man eine E-Mail versendet, dann steht der eigene Name normalerweise im Absenderfeld (engl. **from field**) des Kopfes der E-Mail. Wenn man eine Antwort auf seine Nachricht bekommen möchte, dann ist das auch sinnvoll. Da der Inhalt dieses Feldes beliebig verfälscht werden darf, kann man den Informationen darin nicht wirklich vertrauen. Spammer nutzen dieses Mittel. Der Betreiber eines E-Mail-Servers kennt jedoch die IP-Adresse des einliefernden Rechners. In SMTP ist auch festgelegt, dass der einliefernde Benutzer auch seine E-Mail-Adresse angeben muss. Aber auch an dieser Stelle findet keine Überprüfung dieser Informationen statt, so dass Spammer auch hier falsche Angaben machen können.

Auch beim Surfen im Internet hinterlässt jeder Benutzer Spuren. In den Protokolldateien der Webserver stehen Informationen über den Aufrufer der einzelnen Seiten. In diesem Protokoll finden sich folgende Informationen:

- IP-Adresse des aufrufenden Rechners
- Falls eine HTTP-Authentisierung stattgefunden hatte, dann auch die Benutzerkennung des Aufrufers
- Datum und Uhrzeit des Aufrufs
- Aufgerufene Seite (URL)
- Informationen über die Seite, von der aus man auf die aktuelle Seite verwiesen wurde. Auf Englisch heißt dieses Feld *referrer*.
- Informationen über den Browser des Benutzers, wie Name der Software, eingestellte Sprache, Betriebssystem, Versionsnummer der Software usw.

Aus diesen Informationen lässt sich zwar der Rechner genau identifizieren, nicht jedoch der Benutzer. Viele Anbieter von Seiten im Internet benutzen hierfür die sog. *Cookies*. Das sind kleine Datensätze, die vom Server auf dem Client gespeichert werden und vom Client dann bei Bedarf an den Server zurück geschickt werden. Setzt der Server eines Internetbuchhändlers bei einem Benutzer ein Cookie, dann kann der Benutzer von diesem Buchhändler wiedererkannt werden. Andere Händler im Netz (mit anderen Webservern) würden das identifizierende Cookie jedoch nicht zugeschickt bekommen.

Cookies

Hat jedoch eine Firma auf den Webseiten von vielen verschiedenen Händlern einen eigenen Bestandteil, z. B. eine kleine Werbegrafik, dann können Kunden auch händlerübergreifend identifiziert werden. Das funktioniert dann wie folgt:

1. Der Händler plaziert eine Werbegrafik auf seiner Seite.

2. Beim Aufruf der Seite wird die Werbegrafik nachgeladen. Das geschieht dann vom Server der Werbefirma.

3. Der Werbeserver erkennt anhand der Referrer-Information, von welcher Händlerseite der Aufruf stammt.

4. Der Werbeserver setzt ein Cookie.

Besucht derselbe Kunde nun einen anderen Händler, der auch eine Werbegrafik dieser Firma auf seiner Seite hat, dann kann der Werbeserver den Kunden wiedererkennen. Auf dem Werbeserver stehen dann also Informationen über die verschiedenen Händler bereit, die der Kunde im Internet besucht hat.

Als Benutzer sollte man also kontrollieren, welche Server auf dem eigenen Rechner Cookies speichern dürfen. Außerdem ist es sinnvoll, die gespeicherten Cookies hin und wieder zu löschen. Möchte man die Datensammler in die Irre führen, dann kann man seine Cookies auch mit anderen tauschen, so dass keine persönlichen Nutzungsprofile mehr entstehen. Für weitergehenden und wirksameren Schutz sind jedoch die folgenden Techniken erforderlich.

1.1.3 Allgemeine Anonymisierungstechniken

Broadcasts und implizite Adressierung: Eine einfache Möglichkeit den Empfänger einer Nachricht zu anonymisieren besteht darin, die Nachricht einfach an *alle* potentiellen Empfänger zu schicken. Ein Angreifer kann nun nicht mehr erkennen, für wen genau die Nachricht gedacht war. So wie man beim Fernsehempfang oder bei Rundbriefen nicht mehr erkennen kann, wer von den Inhalten Kenntnis nimmt, so kann man das bei Broadcast auch nicht mehr. Schließlich entscheidet jeder der Empfänger in seiner Privatsphäre, ob er den Fernseher einschaltet, bzw. die Nachricht liest.

Man kann nun auf zwei verschiedenen Wegen den tatsächlichen bzw. geplanten Empfänger der Nachricht adressieren:

Explizite Adressierung: Hierbei wird der Empfänger der Nachricht explizit durch seinen Namen, den Ort an dem er bzw. sein Computer sich befindet o. ä. angesprochen. Auch alle anderen Empfänger können die Adresse mitlesen.

Implizite Adressierung: Eine implizite Adresse bezeichnet weder eine Person noch einen Computer direkt. Sie ist vergleichbar mit Pseudonymen, die Absender und Adressat vorab vereinbart haben müssen.

Das Broadcast-Verfahren schützt zwar den Empfänger der Nachricht, nicht aber den Nachrichteninhalt. Dieser wird an alle potentiellen Empfänger verteilt und kann von allen gelesen werden. Möchte man das vermeiden, dann kann man Verschlüsselungstechniken benutzen. Außerdem geht dieses Verfahren nicht sehr ökonomisch mit der Bandbreite in einem Verbindungsnetz um. Im Internet würde dies zu großer Netzlast führen, die überwiegend unerwünschte Informationen enthält. Auch der Schutz des Absenders ist bei diesem Verfahren nicht gegeben.

Dummy Traffic: Die Idee dieses Verfahrens ist vergleichbar mit der Steganographie. Der Absender verschickt ständig Nachrichten, auch wenn eigentlich keine Informationen übertragen werden sollen. Letzteres sind also überflüssige, sog. Dummy-Nachrichten. Die tatsächliche Nachricht fällt in der Menge der Nachrichten nicht mehr auf.

Proxies: Wie bereits in Kurs *(01866) Sicherheit im Internet 1* vorgestellt, sind Proxies Stellvertretersysteme. Ein HTTP-Proxy, der das Intranet einer Firma mit dem Internet verbindet, verschleiert, welcher Mitarbeiter der Firma eine Anfrage stellt. Der Proxy ersetzt die echten Absenderdaten durch seine eigene Adresse und leitet die so veränderte Nachricht weiter. Kommt eine Antwort zurück, dann trägt der Proxy dort wieder die ursprüngliche Adresse (also die Adresse des Mitarbeiters) ein. Somit trennt ein Proxy eine Verbindung zwischen Absender und Empfänger in zwei neue Verbindungen, (1) zwischen Absender und Proxy und (2) zwischen Proxy und Empfänger.

Der Betreiber des Proxys kann jedoch Absender und Empfänger identifizieren. Als Benutzer muss man also dem Betreiber des Proxys insoweit trauen, dass er diese Informationen nicht preis gibt. Aber auch ein „externer“ Angreifer, der nur die Kommunikationsverbindungen des Proxys beobachten kann, kann eine Verkettung zwischen Absender und Empfänger herstellen.

Wenn die Nachricht nicht wesentlich verändert wird, dann kann man innerhalb der Ausgabenachrichten einfach die Nachricht suchen, die einer bestimmten Eingabenachricht am ähnlichsten ist. Ein HTTP-Proxy tauscht beispielsweise nur die Absenderadresse im Nachrichtenkopf aus. Der Rest der Nachricht bleibt unverändert.

Aber selbst wenn die Nachrichten durch Verschlüsselung verändert würden, gibt es Möglichkeiten der Verkettung. Hierzu können Informationen über die zeitliche Abfolge der Nachrichten zum Proxy hin und vom Proxy weg sowie Informationen über die Größe der Nachrichten herangezogen werden. Wird die dritte Eingabenachricht in den Proxy auch immer als dritte Ausgabenachricht weiter versendet, dann wäre die Verkettung trivial.

Mix-Netz: Das Konzept der **Mixe** geht auf David Chaum [Cha81] zurück. Mixe
Die Grundidee eines Mixes besteht darin, die Verkettung von eingehenden und ausgehenden Nachrichten anhand der zeitlichen Abfolge oder der Nachrichtengröße zu verhindern. Um das zu erreichen bearbeitet ein Mix immer nur eine bestimmte Menge von Nachrichten derselben Größe auf einmal. Nachdem alle Eingabenachrichten im Mix angekommen sind, werden sie erst verarbeitet (siehe unten) und dann in eine neue Reihenfolge gebracht. In dieser neuen Reihenfolge verlassen die Nachrichten dann den Mix.

Zum Schutz der Nachrichteninhalte und damit die Eingabe- und Ausgabenachrichten nicht zu ähnlich aussehen, wird zusätzlich asymmetrische Verschlüsselung benutzt. Anhand der Situation aus Abbildung 1.1 soll das Prinzip erklärt werden. Gegeben seien:

Absender A: Er besitzt ein Schlüsselpaar aus Private Key G_A und Public Key P_A.

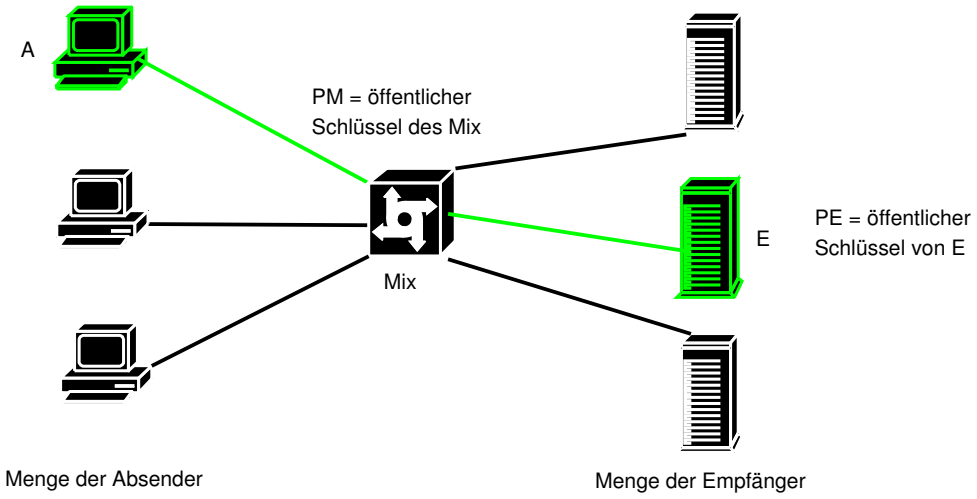

Abbildung 1.1: Nachrichtenversand mit einem Mix

Mix M: Er besitzt ein Schlüsselpaar aus Private Key G_M und Public Key P_M.

Empfänger E: Er besitzt ein Schlüsselpaar aus Private Key G_E und Public Key P_E.

Im Folgenden bezeichnet $P(X)$ das Ergebnis der Verschlüsselung der Nachricht X mit dem Schlüssel P. Das Symbol + bezeichne die Konkatenation von Nachrichten. Der Ablauf beim Versand einer Nachricht N vom Absender A zum Empfänger E über den Mix M ist nun wie folgt:

1. Der Absender verschlüsselt die Nachricht mit dem öffentlichen Schlüssel des Empfängers. Aus N wird also $P_E(N)$.

2. Der Absender erstellt eine neue Nachricht für den Mix. In ihr steht (1) der Empfänger E und (2) die für E verschlüsselte Nachricht $P_E(N)$. Diese Nachricht verschlüsselt der Absender nun mit dem öffentlichen Schlüssel des Mixes und sendet sie an den Mix. Der Mix bekommt also $P_M(E + P_E(N))$ zugeschickt.

3. Der Mix entschlüsselt die empfangene Nachricht und erfährt somit, wohin diese Nachricht weitergeleitet werden soll.

4. Der Mix sendet die entschlüsselte Nachricht (also $P_E(N)$) an den Empfänger E.

5. Der Empfänger entschlüsselt die Nachricht.

Ein Angreifer, der die Eingänge und Ausgänge des Mixes überwachen kann, erfährt somit nur, dass der Absender eine Nachricht an den Mix sendet. Da das aber nicht nur der eine Absender ist, sondern eine größere Gruppe, generiert der Mix auch wieder eine Gruppe Ausgangsnachrichten. Ein Angreifer kann auch diese Nachrichten sehen, aber er kann die Inhalte nicht verstehen, da sie ja für den Empfänger verschlüsselt sind. Der Angreifer kann also keine Verkettung zwischen einer Eingangs- und einer Ausgangsnachricht herstellen.

Zu diesem Grundprinzip kommen jedoch noch weitere Sicherheitsmaßnahmen hinzu:

- Damit mögliche Nachrichten von einem Angreifer nicht einfach geraten werden können, wird vor jede Nachricht noch eine Zufallszahl gesetzt. Sonst könnte ein Angreifer einfach die geratene Nachricht selbst verschlüsseln (die öffentlichen Schlüssel sind ja bekannt) und prüfen, ob diese Nachricht dann tatsächlich über eine Eingangs- oder Ausgangsleitung des Mixes übertragen wurde.

  Außerdem muß der Absender einer Nachricht dafür sorgen, dass die vorgegebene gleiche Größe aller Nachrichten erreicht wird. Dazu muss der Absender entweder Füllbytes einfügen oder bei zu großen Nachrichten diese in kleinere Nachrichten zerlegen.

- Der Mix merkt sich, wenn eine Nachricht von ihm übertragen wurde. Wird dieselbe Nachricht noch einmal eingespielt, dann wird sie vom Mix nicht weitergeleitet. Sonst könnte ein Angreifer einfach eine Nachricht ein zweites Mal an den Mix senden und alle Ausgabenachrichten aufzeichnen. Dann besitzt der Angreifer zwei Mengen von Ausgabenachrichten, die von der ersten Übertragung der Nachricht und die von der zweiten Übertragung. Die Nachricht, die in beiden Ausgabenachrichtenmengen enthalten ist muss dann zu der wiederholt eingespielten Eingabenachricht gehören. Dem Angreifer wäre eine Verkettung zwischen einer Eingabe- und einer Ausgabenachricht gelungen.

  Damit der Speicherbedarf im Mix beherrschbar bleibt, speichert der Mix nicht die Nachrichten selbst, sondern einen Hashwert. Mit Hilfe dieser Hashwerte wird überprüft, ob eine empfangene Nachricht möglicherweise schon einmal eingespielt wurde.

- Als Absender der Nachricht benutzt man nicht nur einen Mix, sondern eine Kette von hintereinander geschalteten Mixe, eine sogenannte **Mix-Kaskade** wie in Abbildung 1.2. Das oben genannte Grundprinzip wird also wiederholt angewendet. Der Absender verschlüsselt die Nachricht zuerst für den endgültigen Empfänger, dann für den letzten Mix in der Kette, dann für den vorletzten usw. bis die Nachricht zuletzt für den ersten Mix verschlüsselt wird. Jeder Mix auf dem Weg kennt also immer nur die nächste Station der Nachricht. Erst der letzte Mix kennt den tatsächlichen Empfänger. Mix-Kaskade

  Sollte ein Angreifer die Kontrolle über einen Mix übernehmen und alle Eingangs- und Ausgangsnachrichten dieses Mixes verketten können, dann kann der Angreifer immer noch keine Verkettung zwischen Absender und Empfänger herstellen. Würde ein Angreifer den Mix in der Mitte von Abbildung 1.2 übernehmen, dann erfährt er nur, dass verschlüsselte Nachrichten vom linken Mix kommen und an den rechten Mix weitergeleitet werden. Selbst wenn ein Mix am Anfang oder Ende der Kette übernommen würde, wäre keine Verkettung möglich. Ein Angreifer am Anfang erfährt nur, welche Absender Nachrichten schicken. Da die Nachrichten verschlüsselt sind, weiß der Angreifer nicht mehr über den Inhalt

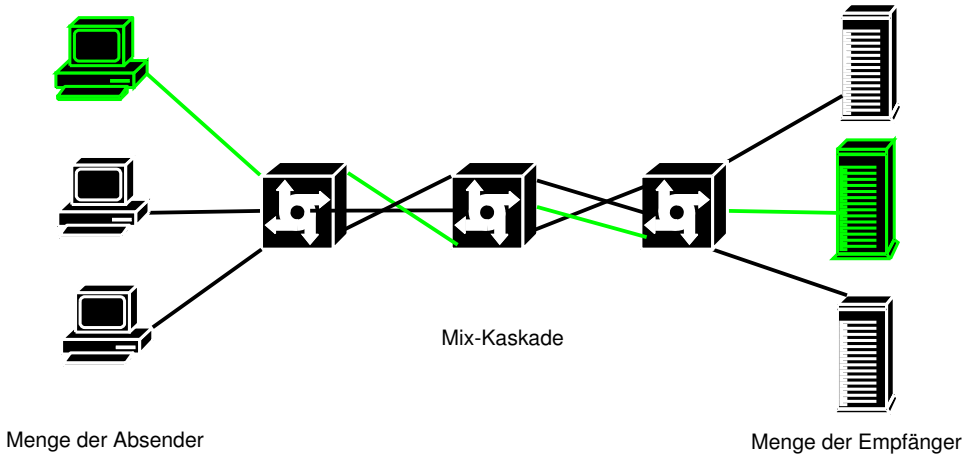

Abbildung 1.2: Nachrichtenversand mit einer Mix-Kaskade

oder den endgültigen Empfänger. Die gesamte Kette der Mixe ist sicher, solange mindestens einer der Mixe nicht vom Angreifer kontrolliert wird.

- Damit der Angreifer nicht anhand des Beginns und des Endes von Sende- oder Empfangsaktivitäten doch eine Zuordnung von Nachrichten herstellen kann, sollten Absender auch dann Nachrichten verschicken, wenn sie eigentlich keine Inhalte zu verschicken haben. Dann sollten Dummy-Nachrichten versendet werden.

Übungsaufgabe 1.1 Kann ein Benutzer die Mixe in der Kette auch in einer anderen Reihenfolge benutzen, beispielsweise erst den mittleren Mix aus Abbildung 1.2, dann den linken und danach den rechten?

Für den Einsatz in der Praxis braucht ein Benutzer ein Client-Programm. Es ist für die Verschlüsselung der Nachrichten, das Generieren und Einfügen von Zufallszahlen sowie das „Zuschneiden“ auf die passende Nachrichtengröße zuständig. Außerdem muss der Client die Kette der Mixe festlegen, die die Nachricht durchlaufen soll. Zur Zeit werden von verschiedenen Institutionen Mixe betrieben. Als Benutzer muss man darauf vertrauen, dass mindestens einer dieser Betreiber tatsächlich keine Protokolldateien speichert und Informationen über seine Arbeit preisgibt.

An der Universität Dresden befasst sich die Arbeitsgruppe von Professor Pfitzmann mit den Themen Datensicherheit und Datenschutz. Dort wurde ein Java-Client (JAP, siehe Abbildung 1.3) als Schnittstelle zu einem Netz von Mixen entworfen und implementiert. Außerdem wird dort ein Mix betrieben.

Möchte man diesen Client einsetzen, dann installiert man den Client und kann dann einen lokalen Proxy-Server starten. Dieser Proxy bekommt über einen Info-Service immer aktuelle Informationen über laufende Mixe und ihre Auslastung. Der Proxy kennt somit eine Mix-Kaskade, die er benutzen kann. Nun muss der Anwender noch seinen Webbrowser so konfigurieren, dass alle HTTP-Requests an den lokalen Proxy-Server geleitet werden. Der lokale Proxy-Server benutzt normalerweise die Portnummer 4001. Der lokale Proxy

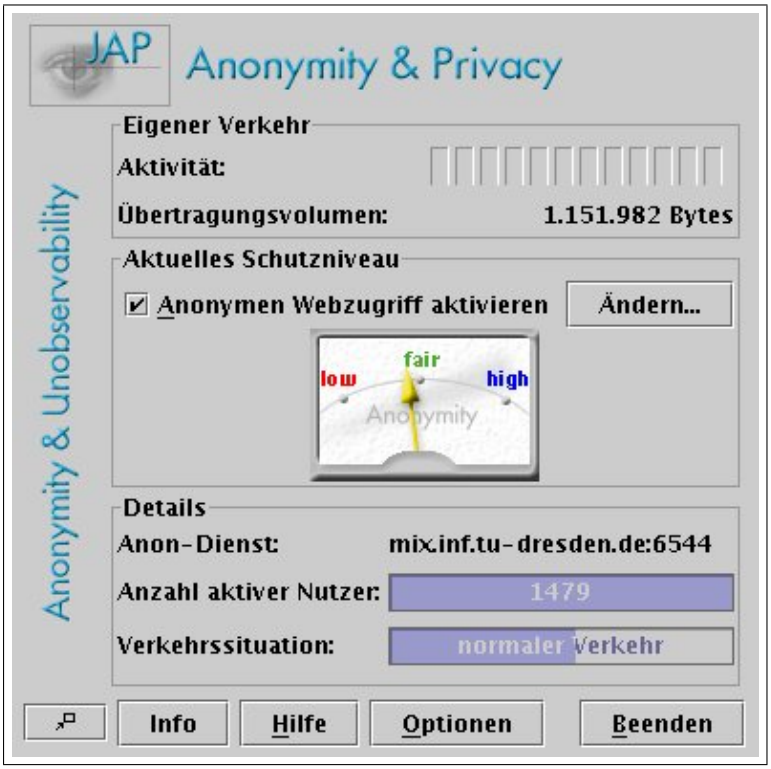

Abbildung 1.3: Benutzerschnittstelle des Programmes *JAP*

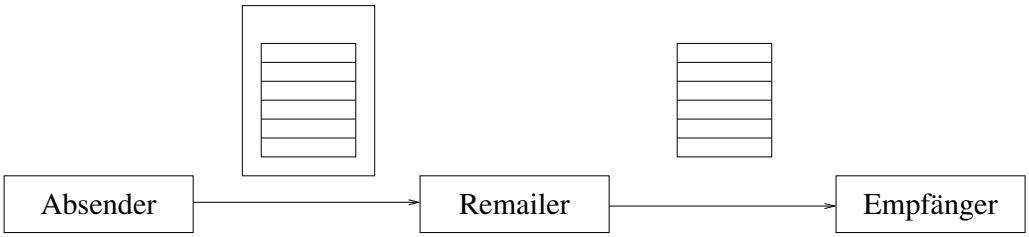

Abbildung 1.4: Prinzip eines anonymen Remail-Dienstes

verschlüsselt nun den Request für den Empfänger bzw. für die Mixe und leitet den Request an den ersten Mix.

Weitere Hinweise zu diesem Werkzeug und die installierbaren Java-Dateien finden Sie im Internet unter folgender URL: URL

`http://anon.inf.tu-dresden.de/`

1.1.4 Anonyme E-Mail

Anonyme Remailer sind Dienste, die eingehende E-Mails anonymisieren und erst danach an den eigentlichen Empfänger weiterleiten. Empfänger kann dabei eine einzelne Person oder eine Newsgruppe sein. Für den Empfänger sieht es so aus, als ob die Nachricht vom Remailer stammt. Der eigentliche Absender bleibt unsichtbar. Abbildung 1.4 zeigt das Prinzip. anonyme Remailer

Wichtig für die korrekte Funktion des Dienstes sind zwei Punkte: Voraussetzungen

1. Der Remailer arbeitet technisch korrekt, d. h. er filtert tatsächlich alle

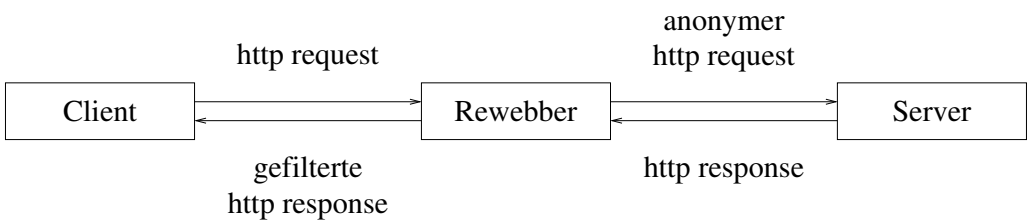

Abbildung 1.5: Prinzip eines Rewebber

Daten aus der Nachricht, die den Absender identifizieren könnten. Das bedeutet jedoch nicht, dass der Nachrichtentext verändert wird, sondern nur, dass die Verwaltungsinformationen (Envelope, Header-Zeilen etc.) entsprechend gefiltert werden.

2. Der Remailer arbeitet organisatorisch korrekt. Dies bedeutet, dass die im Remailer vorliegenden Informationen über den Absender auf keinen Fall preisgegeben werden. Als Anwender muss man also darauf vertrauen, dass der Remailer auf Nachfrage des Empfängers nicht doch den Namen des Absenders verrät.

Dabei muss man auch beachten, dass der Betreiber eines Remailers evtl. von Gerichten dazu gezwungen werden kann, den Namen eines Absenders preiszugeben. Man sollte sich also *nicht* darauf verlassen, dass man bestimmte Straftaten (üble Nachrede, Verleumdung etc.) über einen Remailer ungestraft verüben könnte. In Finnland wurde ein anonymer Remailer unter dem DNS-Namen `anon.penet.fi` betrieben. Nachdem der Betreiber auf gerichtliche Anordnungen mehrmals gezwungen war die Identität hinter einer anonymen E-Mail-Adresse preis zu geben, wurde der Dienst eingestellt.

Auch in Deutschland sind Anbieter von Kommunikationsdiensten gesetzlich verpflichtet die Identität der Teilnehmer zu kennen. Das wollen die Anbieter auch schon deshalb, um sich ihre Dienste auch bezahlen zu lassen. In den zugehörigen Datenschutzregelungen ist festgehalten, wie lange Verbindungsdaten gespeichert werden dürfen und unter welchen Voraussetzungen sie weitergegeben werden dürfen. Im Ausland gibt es allerdings Betreiber von anonymen Remailern. Informationen zu diesem Systemen findet man am besten in der Newsgruppe `alt.privacy.anon-server` auf einem Newsserver.

1.1.5 Anonym Surfen

„Surf-Anonymität“

Dieser Dienst bietet Anonymität beim Surfen durchs Web. Dabei kann Anonymität für Surfer *und* Web-Anbieter realisiert werden. Ein Client surft dabei anonym, indem er seine Anfragen über den Rewebber leitet (siehe Abbildung 1.5).

rewebber

Der Rewebber entfernt aus dem HTTP-Request alle Informationen, die auf den Absender schließen lassen. Dann schickt er den Request an den Server weiter. Der Server sieht die Anfrage vom Rewebber und schickt die Antwort an den Rewebber. Der leitet die Antwort an den Client weiter. Zuvor werden Inhalte, die die Anonymität des Client gefährden könnten (JavaScript oder Java), entfernt. Da die HTTP-Response i. d. R. eine HTML-Seite ist, enthält

sie normalerweise auch Hyperlinks. Diese werden vom Rewebber so verändert, dass der Client beim Klick auf einen Link wieder über den Rewebber geleitet wird.

Das bereits vorgestellte Konzept der Mix-Kaskade bietet auch die Möglichkeit, anonym zu surfen. Als „Rewebber“ fungiert dabei der eigene Rechner, auf dem JAP als Proxy eingesetzt wird. Nun muss der Benutzer seinen Webbrowser noch anweisen für alle Requests den Proxy zu benutzen.

1.1.6 Zusammenfassung: Anonymität

Im täglichen Leben hat man sich daran gewöhnt, dass man letztlich immer selbst entscheiden kann, welche Informationen man über sich preis gibt. Dieses Recht auf informationelle Selbstbestimmung wurde im Volkszählungsurteil vom Bundesverfassungsgericht ausdrücklich bestätigt. Man kann also im Wesentlichen frei entscheiden, ob man seinen Namen nennt oder ob man anonym auftritt. Anbieter von Telediensten müssen ihren Kunden auch die anonyme oder pseudonyme Nutzung und Bezahlung der Dienste möglich machen.

Andererseits besteht z. B. bei der Verfolgung von Straftaten ein berechtigtes Interesse, dass Personen bei der Ermittlung ihre Identität gegenüber den Ermittlungsbehörden preis geben müssen. Dies ist durch Gesetze geregelt.

Bei der Nutzung des Internets gibt man einiges an Informationen automatisch über sich preis. Insbesondere der Internet-Service-Provider kann die gesamte Kommunikation eines Kunden beobachten. Gesetze schränken die Speicherung und Nutzung dieser Informationen durch den ISP einerseits ein (Datenschutz) und verlangen andererseits, dass Strafverfolgungsbehörden bei Bedarf Zugang zu diesen Informationen bekommen können.

Daneben können auch Anbieter von Webseiten einiges an Informationen über ihre Benutzer erlangen. Häufig passiert das, ohne dass die Benutzer davon Kenntnis nehmen. Mit Hilfe von Identitätsmanagement sollen Benutzer selbst steuern können, welche Informationen sie über sich beim Surfen im Netz preis geben wollen. Mit Hilfe von Mixen kann man Nachrichten anonym oder pseudonym austauschen und dadurch anonym im Internet surfen.

1.2 Aktive Inhalte

Immer mehr Seiten im World Wide Web enthalten Programmcode. Dieser wird zusammen mit dem Inhalt (Text und Bilder) der Seite geladen und dann auf dem Client-Rechner ausgeführt. Als Benutzer im World Wide Web kennt man den Ersteller dieses Codes i. d. R. nicht. In den folgenden Abschnitten werden die meistgenutzen Techniken für aktive Inhalte vorgestellt. Neben ihrer Funktionsweise wird insbesondere auf die Sicherheitsrisiken der einzelnen Techniken eingegangen.

1.2.1 JavaScript

Skriptsprachen wurden ursprünglich entworfen, um bestimmte wiederkehrende Aktionen zu automatisieren. Besonders Systemadministratoren benutzen sie

gerne für die verschiedenen Administrationsaufgaben. *Perl* und *TCL/TK* sind zwei Beispiele für Skriptsprachen. Programme in einer Skriptsprache bezeichnet man auch als **Skript**. Sie werden i. d. R. *nicht* in Maschinensprache übersetzt (engl. **to compile**), sondern von einem Interpreterprogramm interpretiert und ausgeführt.

Skript

JavaScript wurde von der Firma *Netscape* entwickelt. Sie ist eine kompakte, objektbasierte Sprache zur Erstellung von Programmen im World Wide Web (WWW). Moderne Webbrowser (z. B. *Firefox*, *Safari*, *Chrome* oder *Internet Explorer*) enthalten einen Interpreter für JavaScript. Ein JavaScript-Programm wird also auf dem Computer des Benutzers und Websurfers ausgeführt. JavaScript-Programme sind in eine HTML-Seite eingebunden.

JavaScript Beispiel

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 3.2//EN">
<HTML><HEAD><TITLE>JavaScript Beispiel</TITLE></HEAD>
<BODY>
  <SCRIPT language="JavaScript">
    var fenster;
    fenster = window.open();
    fenster.document.open();
    fenster.document.write("<!DOCTYPE HTML PUBLIC ...");
    fenster.document.write("<BODY>Diese Seite ist mit");
    fenster.document.write("<EM>JavaScript</EM> erstellt.");
    fenster.document.write("</BODY></HTML>");
  </SCRIPT>
</BODY></HTML>
```

Es muss nun nicht immer das komplette Skriptprogramm in eine HTML-Seite eingefügt werden. Stattdessen kann man auch nur einen Verweis (engl. **link**) auf das Skript in die HTML-Seite schreiben. Das `<SCRIPT>`-Element enthält hierfür das Attribut `src`.

```
...
<SCRIPT language="JavaScript" src="www.feu.de/BeispielScript">
...
```

Ein JavaScript-Programm hat Zugriff auf verschiedene Daten des Webbrowsers, z. B.

- die Cookies, die der Browser gespeichert hat;
- die History-Liste, in der die Adressen der zuvor besuchten Webseiten stehen.

Die Sprachmittel von JavaScript erlauben allerdings keinen Aufbau von Netzverbindungen zu anderen Computern. Man kann jedoch weitere HTML-Seiten von anderen Computern laden. Die Sprache erlaubt auch die Definition von Funktionen, wie man es aus prozeduralen Programmiersprachen kennt.

Meistens werden JavaScript-Funktionen als Reaktion auf ein Ereignis (engl. **event**) gestartet. Ein Mausklick, eine Eingabe in einem Formularfeld oder eine Mausbewegung über eine bestimmte Stelle der Seite sind Beispiele für Ereignisse. Tritt ein Ereignis ein, dann startet der JavaScript-Interpreter die zugehörige Funktion.

Risiken: Obwohl ein JavaScript-Programm *keine* direkte Verbindung zu einem entfernten Computer aufbauen kann, kann es trotzdem Daten nach außen weitergeben. Das geht über den „Umweg“ der Anforderung einer Webseite. Ein JavaScript-Programm kann jede beliebige URL anfordern, auch eine URL, die auf ein CGI-Skript zeigt. CGI-Skripte laufen auf einem Webserver und erzeugen i. d. R. dynamisch HTML-Seiten. Einem CGI-Skript können Parameter übergeben werden. Ein JavaScript-Programm kann also Daten aus dem Webbrowser lesen, ein CGI-Skript auf einem fremden Webserver aufrufen und die gelesenen Daten als Parameter übergeben. Das CGI-Skript speichert dann diese Daten. Gefahren

Für die Übergabe von Parametern an CGI-Skripte gibt es prinzipiell zwei Möglichkeiten:

1. Die Parameter werden an die URL des CGI-Skripts angehängt.
2. Die Parameter werden separat mit der HTTP-Methode POST übertragen.

Werden die Parameter an die URL angehängt, so hat die URL folgendes Aussehen:

```
<A HREF="www.xy.z/cgi-bin/script?name=ihr+name&action=machwas">
```

Das Fragezeichen (?) leitet die Parameterliste ein. Die Parameterliste enthält Paare von Parameternamen (z. B. name) und Parameterwert (z. B. ihr+name). Da einige Zeichen, wie z. B. das Leerzeichen, nicht in einer URL vorkommen dürfen, werden sie durch andere Zeichen ersetzt. Parametername und Parameterwert werden durch das Gleichheitszeichen (=) getrennt, verschiedene Parameter durch das Und-Zeichen (&).

Bei der Parameterübergabe durch die HTTP-Methode POST werden die Parameter für den Benutzer unsichtbar übertragen. Der *Netscape Communicator* weist den Benutzer allerdings mit einer Warnmeldung darauf hin, dass Daten übertragen werden. Weiterhin wird auch gemeldet, dass diese Übertragung möglicherweise ungesichert ist und die Daten unterwegs verfälscht werden könnten.

Die *Same Origin Policy* beschränkt die Möglichkeiten von JavaScript. Sie besagt, dass JavaScript-Programme, die nicht aus derselben Quelle wie die HTML-Seite, auf der sie stehen, stammen, keinen Zugriff auf bestimmte Browserdaten (Cookies, History) haben. Der Interpreter verhindert in diesem Fall den Zugriff auf die Daten. Das JavaScript-Programm kann allerdings vom Benutzer weitergehende Rechte anfordern. Voraussetzung hierfür ist, dass das JavaScript-Programm digital signiert wurde. Der Benutzer kann nun selbst entscheiden, ob das JavaScript-Programm die Rechte erhalten soll oder nicht. Same Origin Policy

Zusammenfassung: Die Sprache JavaScript ist so definiert, dass bestimmte sicherheitskritische Funktionen (z. B. Aufbau von Netzverbindungen, Zugriff auf lokale Festplatten) gar nicht erst möglich sind. Trotzdem können von JavaScript-Programmen Gefahren für die Vertraulichkeit ausgehen. Der Benutzer sollte JavaScript im Browser möglichst deaktivieren oder die Möglichkeiten eines JavaScript-Programmes einschränken (siehe Abbildung 1.6).

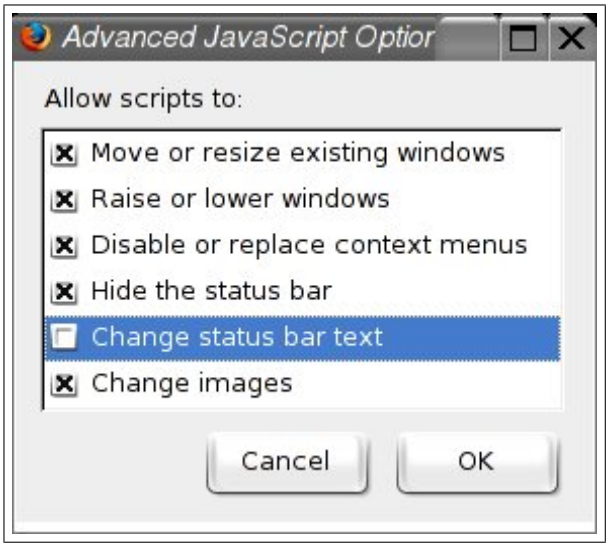

Abbildung 1.6: Rechteeinstellungen für JavaScript in *Firefox*

Übungsaufgabe 1.2 Welche konkrete Gefahr liegt vor, wenn ein JavaScript-Programm die vom Browser angezeigte URL verändern kann?

1.2.2 Java

Java ist eine objektorientierte Programmiersprache, die von der Firma *Sun Microsystems* entwickelt wurde. Detaillierte Informationen zu objektorientierter Programmierung und zu Java finden Sie im Kurs *(01814) Objektorientierte Programmierung*. Große Teile der Java-Dokumentation stehen im Internet unter der Adresse

`http://java.sun.com/docs/`

Java-Byte-Code

Java-Virtual-Machine (JVM)

Programme in Java werden von einem Java-Compiler in einen Zwischencode, den sogenannten **Java-Byte-Code** übersetzt. Dieser Code wird dann von einem Interpreter ausgeführt. Den Interpreter nennt man in diesem Fall **Java-Virtual-Machine (JVM)**. Der Interpreter abstrahiert von einer konkreten Hardware-Architektur und von einem konkreten Betriebssystem. In Java geschriebene Programme können somit auf verschiedenen Rechnertypen unverändert ablaufen.

Moderne Webbrowser enthalten auch eine Java-Virtual-Machine, bzw. ein solcher Interpreter kann nachträglich hinzugefügt werden. Damit können Java-Programme direkt in einem Webbrowser ausgeführt werden. Solche Java-Programme nennt man **Java-Applet**. Sie werden in HTML-Seiten mit Hilfe des Elementes `<APPLET>` eingefügt.

Java-Applet

Java-Sandbox

Während Java-Programme normalerweise alle Operationen ausführen können, laufen Java-Applets in einer speziellen Laufzeitumgebung. Sie wird **Java-Sandbox** genannt und verhindert, dass Java-Applets sicherheitskritische Funktionen ausführen können. So ist es Java-Applets beispielsweise nicht gestattet,

- Dateien des lokalen Systems zu lesen, zu schreiben, zu löschen oder umzubenennen;
- andere Programme auf dem lokalen System zu starten oder
- Netzverbindungen zu anderen Computern zu öffnen, außer zu dem Computer von dem das Applet geladen wurde.

Java-Applets können, ähnlich wie JavaScripts, weitergehende Rechte beim Benutzer anfordern. Dazu müssen sie digital signiert sein, damit der Benutzer zumindest weiß, von wem das Applet stammt. Weiterhin kann der Benutzer sicher sein, dass das Applet auf dem Übertragungsweg nicht manipuliert wurde.

Gefahren

Risiken: Ein Java-Compiler prüft bei der Übersetzung bereits die Einhaltung bestimmter Vorgaben. Dazu gehört die Überprüfung von Zugriffen auf Felder (engl. **arrays**), d. h. offensichtliche Bereichsverletzungen oder ob Zugriffsrechte (private vs. public) eingehalten werden. Außerdem überprüft der Compiler, ob die Parameter bei Funktionsaufrufen mit denen der Funktionsdefinition übereinstimmen. Das bedeutet, dass Anzahl und Datentypen der Parameter stimmen müssen.

Erst wenn diese Prüfungen bestanden wurden, erzeugt der Compiler den Byte-Code. Der Byte-Code ist eine universelle Sprache, die im Prinzip mit Assembler-Sprachen vergleichbar ist. Insbesondere ist es möglich beliebige Byte-Codes (also auch fehlerhafte) zu erstellen. Ein böswillig veränderter Java-Compiler könnte also Byte-Code erzeugen, der die in Java definierten (und vom Compiler garantierten) Schutzeigenschaften umgeht. Außerdem besteht die Gefahr, dass ein Angreifer den Byte-Code auf dem Übertragungsweg durch das Internet manipuliert.

Byte-Code-Verifier

Eine Java-Virtual-Machine enthält daher einen **Byte-Code-Verifier**. Dieser prüft vor der Ausführung, ob der Byte-Code „vernünftig“ aussieht. Konkret wird beispielsweise geprüft, ob kein Stapelüberlauf (engl. **stack overflow**) auftritt und ob die Funktionen mit den richtigen Parametern aufgerufen werden. Die weiteren Überprüfungen und die Details dazu finden Sie in der Java-Dokumentation.

Allerdings kann ein Byte-Code-Verifier niemals 100-prozentig korrekt arbeiten. Dazu müsste er nämlich verstehen, was das Byte-Code-Programm genau machen will. Das kann im Allgemeinen aber nicht gehen, denn dann wäre ja auch das Halte-Problem entscheidbar.

Java-Class-Loader

Objektorientierte Programmiersprachen erlauben die Wiederverwendung von bereits geschriebenem Code, indem neue Klassen von existierenden Klassen abgeleitet werden. Dabei können die Funktionen[4] in der abgeleiteten Klasse überschrieben werden. Ein Angreifer kann also versuchen, wichtige Basisklassen eines Java Systems zu manipulieren. Der **Java-Class-Loader** hat die Aufgabe, das zu verhindern. Details hierzu finden Sie im Kurs *(01814) Objektorientierte Programmierung* oder der Java-Dokumentation.

Security-Manager

Ein **Security-Manager** ist dann die dritte Sicherheitsstufe. Zugriffe auf

[4] In der objektorientierten Programmierung spricht man statt von Funktionen auch oft von Methoden, meint aber im Prinzip dasselbe.

sicherheitskritische Ressourcen aus einem Java-Programm oder einem Applet müssen von ihm erlaubt werden. Dazu bedient sich der Security-Manager der Herkunft des Java-Programms. Über das Internet geladene Applets haben standardmäßig weniger Rechte als Java-Programme, die Sie selbst lokal installiert haben. Der Benutzer kann allerdings über das Netz geladene und digital signierte Applets auf dieselbe Stufe wie lokale Java-Programme (bezüglich der Rechte) stellen.

Zusammenfassung: Java-Applets erlauben die einfache, von Computertyp und Betriebssystem unabhängige und für den Benutzer transparente Ausführung von Programmen auf dem lokalen Computer des Benutzers. Die dabei auftretenden Sicherheitsprobleme werden durch die Sicherheitsstufen *Byte-Code-Verifier*, *Class-Loader* und *Security-Manager* adressiert. Aufgrund der Komplexität der Aufgaben können immer wieder Lücken oder Fehler in den Implementierungen gefunden werden. Da es keine formale Spezifikation der Aufgaben der einzelnen Sicherheitsstufen gibt, ist es auch schwierig die Korrektheit einer Implementierung nachzuweisen.

Der Benutzer sollte den Java-Interpreter in seinem Browser normalerweise deaktivieren. Nur wenn bestimmte Situationen es erfordern, sollte der Java-Interpreter kurzfristig aktiviert werden.

1.2.3 ActiveX

Mit *ActiveX* bezeichnet die Firma *Microsoft* eine Technologie, mit der ein ausführbarer Programmcode auf einen PC gelangen kann und dort dann ausgeführt wird. ActiveX basiert auf dem **Component-Object-Model (COM)**, das die Architektur wiederverwendbarer Softwarekomponenten beschreibt. Konkret werden darin Kommunikationsmechanismen und Schnittstellen definiert, die solch eine Komponente anbieten muss. Diese Softwarekomponenten liegen in Maschinencode vor und werden **ActiveX-Control** genannt. ActiveX-Controls können mit dem Element `<OBJECT>` in eine HTML-Seite eingebunden werden.

Component-Object-Model (COM)

ActiveX-Control

Ein ActiveX-Control kann nicht alleine gestartet werden. Dazu wird ein sogenannter **ActiveX-Container** benötigt. Microsofts Webbrowser *Internet Explorer* kann die Funktion eines ActiveX-Containers übernehmen. Das auf einer HTML-Seite befindliche ActiveX-Control kann also vom *Internet Explorer* geladen und ausgeführt werden. Ein ActiveX-Control kann auch auf dem lokalen Rechner installiert werden. Dann verkürzt sich die Ladezeit der HTML-Seite.

ActiveX-Container

Gefahren

Risiken: Wenn ein ActiveX-Control einmal läuft, dann stehen ihm *alle* Funktionen des Betriebssystems zur Verfügung. Es kann den Inhalt der lokalen Festplatte lesen, Netzverbindungen aufbauen, Daten versenden, jede Ihrer Tastatureingaben (auch Passwörter!) mitschreiben, Systemeinstellungen verändern usw. Das Sicherheitskonzept von *Microsoft* sieht vor, dass ActiveX-Controls zertifiziert werden können. Bei einem zertifizierten, d. h. einem digital signierten ActiveX-Control kann der Benutzer davon ausgehen, dass

- das ActiveX-Control unverändert übertragen wurde und

- der Ersteller des Controls auf Grund der digitalen Signatur bekannt ist.

Der Benutzer kann nun selbst entscheiden, ob er dem Ersteller genug Vertrauen entgegen bringt und er das ActiveX-Control startet. Man kann als Benutzer den Internet Explorer so konfigurieren, dass nur noch signierte ActiveX-Controls geladen und ausgeführt werden.

Als Benutzer sollte man sich allerdings darüber im Klaren sein, dass ein ActiveX-Control beispielsweise die Sicherheitseinstellungen des Internet Explorers verändern kann. Obwohl der Benutzer eigentlich „hohe Sicherheit“ eingestellt hat, kann ein ActiveX-Control das rückgängig machen. Dann werden in Zukunft auch nicht signierte ActiveX-Controls geladen und ausgeführt.

Beispiel: Bereits 1997 hat der Chaos Computer Club (CCC) die Gefahren von ActiveX demonstriert. Ein Benutzer hat zusammen mit einer „Lockseite“ auch ein ActiveX-Control geladen. Dieses Programm hat dann im Hintergrund und vom Benutzer unbemerkt das Programm *Quicken* gestartet. Dann wurde dort automatisch eine Überweisung generiert und abgespeichert. Anschließend wurde Quicken wieder beendet. Hat der Benutzer dann später die Überweisungen an seine Bank geschickt, wurde (falls der Benutzer nicht aufmerksam genug war) die vom ActiveX-Control generierte Überweisung mit verschickt.

Zusammenfassung: Mit Hilfe von ActiveX kann ein Benutzer beliebige Programme aus dem Internet einfach und transparent lokal ausführen. Die Zertifizierung eines ActiveX-Controls sagt dem Benutzer, dass das Control auf dem Übertragungsweg nicht verändert wurde und wer der Autor des Controls ist. Es sagt nichts über die Funktionen, die das Control später einmal ausführen wird.

ActiveX-Controls liegen in Maschinencode vor und haben dieselben Rechte und Zugriffsmöglichkeiten wie jedes lokal installierte Programm auch. Aus Sicherheitsgründen sollten ActiveX-Controls deaktiviert werden.

1.2.4 Sonstige aktive Inhalte

Nicht nur die explizit als aktive Inhalte erkennbaren Techniken wie JavaScript, Java oder ActiveX können aktive Inhalte beherbergen. Auch in den eigentlich als Seitenbeschreibungssprachen gedachten Dateiformaten **Portable-Document-Format (PDF)** und **PostScript** der Firma *Adobe* können aktive Inhalte stehen. In beiden Sprachen sind Zugriffe auf Dateien des lokalen Systems möglich. Weiterhin können aus beiden Sprachen heraus weitere Programme gestartet werden. Ein Angreifer muss in diesen Fällen allerdings vorab wissen, wie die Dateien heißen, wo sie stehen und welche Programme auf dem Computer des Benutzers installiert sind.

Portable-Document-Format (PDF)

PostScript

PDF-Dateien können mit dem Programm *Acrobat Reader* interpretiert und angezeigt werden. Der Acrobat Reader kann auch als *Plug-In* in einem Webbrowser installiert sein. Klickt der Benutzer dann auf eine URL, die auf ein Dokument im PDF-Format zeigt, wird automatisch der Reader gestartet und das Ergebnis im Browserfenster angezeigt. In diesem Fall startet der Reader

allerdings keine Programme, die möglicherweise aus der PDF-Datei heraus gestartet werden sollen.

Verglichen mit den Gefahren beim Einsatz von ActiveX sind die potentiellen Probleme durch aktive Inhalte in PDF- oder PostScript-Dateien sehr gering. Ein Angreifer braucht vorab schon genaue Kenntnisse über ein System und bei einem Angriff sind dann auch nur (vergleichsweise) kleine Schäden möglich. Als Benutzer sollten Sie jedoch wissen, dass auch solche Gefahren existieren.

1.2.5 Zusammenfassung: Aktive Inhalte

Als Benutzer im World Wide Web muss man darauf achten, welche Programme man (evtl. auch unbewusst) auf seinem Rechner ausführt. Man sollte daher wenn möglich die Ausführung solcher Programme nicht gestatten. Auf *keinen Fall* sollten solche Programme automatisch vom Webbrowser oder E-Mail-Programm gestartet werden können. Es sollte auf jeden Fall dem Benutzer vorbehalten bleiben solche Programme durch eigene Aktionen (z. B. anklicken) zu starten.

Aktive Inhalte nie automatisch starten

Man muss als Benutzer letztlich zwischen Bequemlichkeit und Sicherheit abwägen. Es ist einfach und bequem, wenn aktive Inhalte automatisch auf dem lokalen Rechner ablaufen können und dabei für den Benutzer sinnvolle und hilfreiche Aktionen ausführen. Allerdings ist es bisher leider so, dass sinnvolle und hilfreiche Aktionen auch Zugriff auf sicherheitskritische Ressourcen erfordern. Man kann als Benutzer nie sicher sein, dass das geladene Programm nicht auch eine Hintertür (Trojanisches Pferd; siehe auch Kurs *(01866) Sicherheit im Internet 1*) enthält. Sicherheitsbewusste Benutzer kontrollieren daher die Ausführung solcher Programme.

In Betriebssystemen, die mehrere Benutzer unterstützen (z. B. Linux), bietet sich folgendes Vorgehen an: Sie erstellen sich einen „Dummy“-Benutzer und entziehen diesem alle sicherheitskritischen Rechte. Wenn Sie dann im Internet surfen wollen, melden Sie sich unter dieser Benutzerkennung an. Sollten nun aktive Inhalte versuchen, Daten aus Ihrem Rechner zu kopieren, dann kann das nicht mehr gehen. Diese Daten sind für den „Dummy“-Benutzer nicht lesbar. Da der aktive Inhalt höchstens die Rechte des Benutzers erhalten kann, in dessen Namen der Inhalt gestartet wird, kann kaum noch Unheil angerichtet werden. Dieser Benutzer darf dann natürlich auch nicht das Recht haben, Änderungen an der Installation vorzunehmen, z. B. neue Programme zu installieren oder vorhandene Programme zu de-installieren.

Speziellen, rechtlosen Benutzer anlegen

Leider geht das bei Betriebssystemen wie Windows 98 nicht. Dort hat jeder Benutzer die volle Kontrolle über das System und alle Administrationsrechte. Erst mit Windows NT und allen seinen Nachfolgern kann und sollte diese Methode eingesetzt werden.

1.3 Computer-Forensik

Selbst wenn man seinen PC und seine Server möglichst sicher konfiguriert, muss man damit rechnen, dass es einem Angreifer einmal gelingt einen Rechner erfolgreich anzugreifen. In diesem Fall muss man versuchen,

1. den Angriff überhaupt zu erkennen,
2. Beweise zu sichern, um den Angreifer zur Verantwortung ziehen zu können,
3. den Angriff zu analysieren, um die Ursachen zu ergründen und zukünftige Angriffe dieser Art damit unmöglich zu machen,
4. die Angriffsspuren restlos zu beseitigen, und die zugrunde liegenden Schwachstellen des Systems beseitigen.

Der erste Punkt (Erkennen eines Angriffs) ist die wesentliche Aufgabe von Intrusion-Detection-Systemen (IDS, vergleiche Kurs *(01867) Sicherheit im Internet 2*). Die größte Schwierigkeit hier besteht darin, echte Angriffe von der legitimen Benutzung des Systems zu unterscheiden. Da die Absichten eines Angreifers i. d. R. nicht legal sind, möchte man gegen Angreifer möglicherweise gerichtlich vorgehen. Gerichtsverhandlungen sind normalerweise öffentlich und früher fanden diese Verhandlungen auch auf Marktplätzen statt. Das Wort *Forensik* stammt vom lateinischen Wort *forum*, auf Deutsch *Markt* ab. Man meint heute mit dem Wort *forensisch* also „gerichtlich, im Dienste der Rechtspflege stehend“ [Bro02]. Kriminalpolizeiliche Untersuchungen werden auch als Forensik bezeichnet. Mit Computer-Forensik meint man also die vergleichbaren Tätigkeiten im Zusammenhang mit Computern. Computer-Forensik untersucht Vorfälle mit Computern, die möglicherweise vor Gericht verhandelt werden sollen. Sie bezieht sich also auf die Punkte 2 (Beweise sichern) und 3 (Angriff analysieren) aus obiger Liste.

IDS

Begriffsbestimmung

In den folgenden Unterabschnitten werden diese beiden Punkte nun genauer vorgestellt. Anschließend geht es darum, was man nach Abschluss der forensischen Untersuchungen als Nächstes machen sollte, nämlich den Computer wieder in einen vertrauenswürdigen, integeren Zustand versetzen (vergleiche Punkt 4 aus obiger Liste).

1.3.1 Beweise sichern

Der erste Schritt einer forensischen Analyse besteht immer darin, Beweise zu sichern. Damit die Beweise anschließend vor Gericht verwendet werden können, müssen bei der Beweissicherung bestimmte Regeln eingehalten werden. Besteht beispielsweise später vor Gericht der Verdacht, dass ein Beweismittel möglicherweise verändert (manipuliert) wurde, dann verliert es seine Beweiskraft. Man muss also den Zustand des angegriffenen Computers exakt, persistent und unveränderbar sichern.

Anforderungen

Daneben möchte man aber auch, dass das System möglichst schnell wieder regulär betrieben werden kann. Das kann man am schnellsten erreichen, indem man das angegriffene System abschaltet, eine Sicherungskopie von vor dem Angriff installiert und das System wieder einschaltet. Hierbei werden allerdings alle Beweise des Angriffs zerstört. Außerdem befindet sich das System in einem Zustand, der einem erneuten Angriff nicht standhalten kann. Dieses Vorgehen ist also nicht sinnvoll. Man muss also grundsätzlich abwägen, ob man sein System möglichst schnell wieder im Einsatz haben will oder ob man sich

Zielkonflikt

die Zeit für eine Beweissicherung und Analyse des Angriffs nehmen möchte. In den folgenden Abschnitten werden daher Möglichkeiten vorgestellt, die Sie bei Bedarf mit Ihren persönlichen Priorisierungen bzgl. Geschwindigkeit und Gründlichkeit in Einklang bringen müssen.

Systemzustand

Zum Zustand des Computers gehören alle Informationen über das System. Die Informationen können persistent (dauerhaft) oder auch nicht persistent (flüchtig) vorliegen. Nicht persistente Informationen gehen verloren, wenn man das System ausschaltet oder herunterfährt. Persistente Informationen stehen nach einem Neustart des Systems immer noch zur Verfügung. Zum Systemzustand gehören:

- Der Inhalt des Hauptspeichers,
- der Inhalt der Festplatte(n),
- die Liste der angemeldeten Benutzer,
- die Liste der laufenden Prozesse,
- die geöffneten Netzverbindungen und
- die aktuelle Systemzeit.

Bis auf den Festplatteninhalt sind die genannten Zustandsinformationen alle nicht persistent.

Womit sichern?

Sichern nicht persistenter Informationen: An dieser Stelle befindet man sich in einem ersten Dilemma. Um den Zustand des Rechners nicht zu verändern, muss man ihn so benutzen, wie er ist, d. h. man muss mit dem kompromittierten System arbeiten. Jedes Kommando, das man nun eingibt, könnte vom Angreifer bereits manipuliert sein und fehlerhafte Ausgaben liefern. Man muss also dafür sorgen, dass man einen Satz von Kommandos zur Verfügung hat, die vom Angreifer nicht manipuliert werden konnten.

Vertrauenswürdige Kommandos benutzen

Um definitiv vertrauenswürdige Kommandos auszuführen, müsste man den Zustand des Systems ändern. Man könnte einerseits ein Speichermedium (CD-ROM, externe Festplatte, USB-Stick usw.) anschließen, auf dem vertrauenswürdige Programme liegen. Diese Programme müssten dann natürlich statisch gebunden sein, damit sie nicht Bibliotheksfunktionen aufrufen, die vom Angreifer evtl. bereits manipuliert wurden. Dann müsste man aber auch sicherstellen, dass bei der Sicherung des Systemzustands tatsächlich die Programme des externen Mediums benutzt werden.

Andererseits könnte man den Computer von einem vertrauenswürdigen Medium neu starten. Dann könnte man wieder vertrauenswürdige Werkzeuge benutzen. Allerdings verliert man in diesem Fall den Systemzustand, da der Speicher gelöscht und alle Prozesse beendet und neu gestartet wurden. Auch der Festplatteninhalt wird dabei verändert. Neben den eigentlichen Inhalten speichert das Betriebssystem auch Meta-Informationen zu Dateien. Dazu gehören Informationen über den Eigentümer der Datei, Zugriffsrechte, Datum und Uhrzeit der Erstellung, des letzten Lesezugriffs und der letzten Änderung usw. Bei

einem Neustart wird auf viele Systemdateien lesend zugegriffen, so dass sich bei allen diesen Dateien die Meta-Informationen ändern.

Unter UNIX bzw. Linux kann man mit den folgenden Kommandos Informationen über des Systemzustand sammeln: UNIX

Hauptspeicher: Im Kernel erhält man über das Verzeichnis `/proc/meminfo` Informationen zum Hauptspeicher allgemein, also die Größe des physikalischen Speichers, Größe des logischen Speichers, Größe des Swap-Bereiches usw.

Jeder Prozess läuft in seinem eigenen virtuellen Speicherbereich. Auf diesen Bereich kann man über `/proc/PID/mem` zugreifen. Dabei bezeichnet PID die Prozessnummer.

Festplatte: Allgemeine Informationen über das Dateisystem und welche Geräte wo im Dateibaum eingehängt sind, liefert das Kommando `df`. Informationen über die Festplatten, ihre Parameter und die darauf eingerichteten Partitionen liefert das Kommando `fdisk -l`.

Wie man den Inhalt der Festplatte Bit für Bit sichert, wird im folgenden Unterabschnitt über das Sichern persistenter Informationen beschrieben.

Benutzer: Das Kommando `who` zeigt an, welche Benutzer zur Zeit an einem Rechner angemeldet sind.

Prozesse: Die laufenden Prozesse werden vom Kommando `ps -elf` angezeigt. Weitere Informationen zu jedem einzelnen Prozess findet man im Verzeichnis `/proc/PID/`. Hierbei bezeichnet PID wieder eine aktuelle Prozessnummer.

Netzverbindungen: Bevor man die Verbindungen betrachtet, sollte man sich die Konfiguration der Netzhardware anschauen. Hierzu dienen die Kommandos `ifconfig` und `arp`. Sie zeigen die Konfiguration eines Interfaces, also z. B. einer Ethernetkarte an sowie die aktuelle Tabelle mit den Zuordnungen von Hardware-Adressen zu IP-Adressen.

Mit dem Kommando `netstat` kann man nun die aktiven Netzverbindungen ansehen. Das Kommando `lsof` zeigt an, welche Dateien oder Ports von welchen Prozessen geöffnet sind.

Systemzeit: Das Kommando `date` liefert das Datum und die aktuelle Systemzeit.

In den Handbuchseiten (engl. **manual pages**) können sie die genaue Benutzung dieser Kommandos nachschlagen. Dort steht auch, welche Optionen diese Kommandos kennen und wie sie diese benutzen können. Aber auch in Büchern über Computer-Forensik [Ges04, MAC+03] finden Sie Details zum Einsatz der Kommandos.

Unter Windows ist es nicht ganz so einfach, an die gesuchten Informationen zu gelangen. Es existieren spezielle Programme, die diese Informationen liefern können. Diese Programme gehören aber nicht direkt zum Betriebssystem, sondern müssen separat beschafft werden. Unter der URL Windows URL

`http://www.sysinternals.com/`

können einige sehr gute Hilfsprogramme kostenlos geladen werden. Man sollte dies tun, *bevor* es zu einem Zwischenfall gekommen ist. Diese Programme werden nicht installiert, sondern man braucht „nur“ die ausführbare Datei (die `.exe`-Datei). Damit werden die o. g. Bereiche wie folgt abgedeckt:

Benutzer: Von Sysinternals stammt das Paket *PsTools*. Zu diesem Paket gehört auch ein Kommandozeilenprogramm mit dem Namen `psloggedon`. Man startet es aus einer Kommando-Shell und bekommt dann eine Liste der lokal angemeldeten Benutzer.

Prozesse: Zur Anzeige der Prozesse dient einerseits der Task-Manager des Betriebssystems. Andererseits kann man mit dem Kommando `pslist` aus dem Paket *PsTools* auch eine Liste der Prozesse mit weitergehenden Prozesseigenschaften ansehen.

Netzverbindungen: Auch unter Windows existieren die Kommandos `arp` und `netstat`. Zusätzlich zeigt das Kommando `ipconfig` den Status der Netzwerkkarte an.

Das Programm `fport /p` zeigt die geöffneten Ports an. Von Sysinternals gibt es das Programm *tcpview*. Er zeigt nicht nur an, von welchen Rechnern zu welchem Port auf dem eigenen Rechner eine Verbindung besteht, sondern auch welcher Prozess auf dem eigenen Rechner der Endpunkt einer Verbindung ist.

Wohin sichern?

Das nächste Problem besteht darin, diese Zustandsinformationen dauerhaft zu sichern. Wo soll man sie also speichern? Viele Zustandsinformationen werden von den Programmen auf dem Bildschirm angezeigt. Man kann daher den Bildschirminhalt mit einer Digitalkamera einfach fotografieren. Die digitale Speicherung der Bilder ist kostengünstig und man kann die Bilder sehr einfach wieder löschen. Leider sind digitale Bilder auch sehr einfach manipulierbar. Möchte man sie als Beweis vor Gericht verwenden, dann sind wieder umfangreiche Sicherungsmaßnahmen erforderlich.

Bildschirm festhalten

In Dateien umlenken

Speichermedium

Leitet man die Ausgabe der Sicherungskommandos stattdessen in Dateien um, so ändert man den Zustand des Systems, denn man verändert den Festplatteninhalt. Also braucht man ein anderes Speichermedium. Es bieten sich Disketten, CDs oder USB-Sticks an. Jedes der Medien hat seine Vor- und Nachteile bezüglich der Größe des Speichers, der Dauerhaftigkeit oder der Möglichkeiten zur nachträglichen Manipulation. In jedem Fall ändert sich der Systemzustand, wenn man eines dieser Medien anschließt.

Damit man nicht für jeden Befehl die Ausgabe in die Datei umlenken muss gibt es unter Linux das Kommando `script`. Es startet eine neue Shell und protokolliert alle Aktionen darin, also sämtliche Benutzereingaben und Systemausgaben. Beendet man die neue Shell, dann endet auch die Protokollierung. `script` protokolliert in eine Datei.

Netz

Alternativ hierzu kann man die Zustandsinformationen auch über ein möglicherweise vorhandenes Netz sichern. Dabei muss man natürlich sicherstellen, dass die Informationen bei der Übertragung über das Netz nicht verändert

werden können. Am einfachsten geht das, wenn man den Empfängercomputer direkt an den zu sichernden Computer anschließt, also die Netzwerkkabel direkt miteinander verbindet oder einen eigenen kleinen *Switch* benutzt. Man muss außerdem darauf vertrauen, dass die Systemsoftware des angegriffenen Rechners, zumindest was den Netzwerkteil betrifft, noch vertrauenswürdig arbeitet. Mit dem Programm *netcat* kann man Daten direkt über das Netz versenden und solche Daten vom Netz kommend auch wieder entgegennehmen.

Egal mit welchen Kommandos bzw. Programmen man arbeitet und egal wo man die Ausgaben sichert, man muss diese Schritte dokumentieren und die Ausgaben gegen nachträgliche Veränderungen schützen. Deshalb sollte man ein schriftliches Protokoll der ausgeführten Kommandos erstellen. Es enthält Angaben der Art: Wer hat wann welches Kommando ausgeführt und dabei welches Ergebnis mit welcher Prüfsumme erstellt? Am besten hat man hierfür bereits ein Formular vorbereitet.

dokumentieren

Protokoll-Formular

Ermittler: *Name der Person*
Datum: *Aktuelles Datum*
Uhrzeit: *Uhrzeit beim Beginn*

Nr.	Kommando	Ausgabe(datei)	SHA-Prüfsumme
1	ps -elf	proc-list	394ec3204546af8b77141dc768ef89b6681bffe2
2	...	...	...

Für jede Ausgabedatei sollte sofort eine kryptografische Prüfsumme berechnet und protokolliert werden. Dazu ist ein sicherer Algorithmus wie z. B. SHA1 oder einer seiner Nachfolger zu benutzen. Die Prüfsummen sollten dabei nicht nur in einer Datei stehen, sondern auch in einem schriftlichen Protokoll (siehe oben) der Aktivitäten. Es kann auch nützlich sein, das Protokoll um weitere Spalten zu ergänzen, beispielsweise um die Uhrzeit der jeweiligen Kommandoeingabe.

Sichern persistenter Informationen: Angreifer möchten ein einmal erfolgreich angegriffenes System später gerne wieder benutzen können. Damit das auch dann geht, wenn der angegriffene Rechner zwischendurch ausgeschaltet wird, muss der Angreifer zwangsläufig auch den Inhalt der persistenten Speichermedien (also der Festplatten) manipulieren. Diese Manipulationen gilt es zu entdecken. Damit die Untersuchung der Festplatten gefahrlos erfolgen kann und damit man den aktuellen Zustand beweisen kann, muss man den Inhalt der Festplatten zuerst sichern.

Hierzu kann man entweder die betroffenen Festplatten ausbauen und in einem speziellen, garantiert virenfreien Computer kopieren oder man startet den betroffenen Computer von einer CD o. ä. mit einem garantiert integeren Betriebssystem. Hierfür bietet sich beispielsweise die freie *KNOPPIX*-Distribution von Linux an.

Bevor man den Inhalt der Festplatten sichert, sollte man noch einmal die Partitionstabelle untersuchen. Die Kommandos hierzu wurden oben bereits vorgestellt. Anschließend sichert man den eigentlichen Inhalt der Festplatten. Dabei bedeutet „Inhalt“ nicht nur die Dateien und die Verzeichnisse, sondern *jeden* einzelnen Sektor der Platte. Für forensische Zwecke reicht es also nicht, eine

„normale“ Sicherungskopie der Platte zu erstellen. Stattdessen muss man jeden Sektor, Bit für Bit, kopieren.

UNIX Unter UNIX gibt es hierfür das Programm *dd*. Es kopiert Daten aus Dateien oder aus Geräten (engl. **devices**). Dabei kann das Gerät eine komplette Festplatte oder eine Partition der Festplatte sein. Der allgemeine Aufbau des Kommandos ist:

```
dd if=/pfad/zum/input of=/pfad/zum/output
```

Die Abkürzung *if* steht für *input file*, *of* entsprechend für *output file*. Ist das Input-File `/dev/hda`, so wird der gesamte Inhalt der zugehörigen Festplatte kopiert, bei `/dev/hda2` der Inhalt der zweiten Partition. Fehlt eine dieser beiden Angaben, so wird stattdessen die Standardeingabe bzw. -ausgabe benutzt. Man kann die Ausgabe also wieder über das Netz auf einen anderen Rechner kopieren.

Mit diesem Kommando kann man auch sicherstellen, dass die Zielfestplatte, also dort wo die Daten gesichert werden sollen, garantiert leer ist. Als Eingabedatei nimmt man einfach `/dev/zero` und schreibt die Ausgabe auf die Zielfestplatte. Sinnvollerweise ist die Zielfestplatte mindestens so groß wie die Eingabefestplatte.

Speichert man die Ausgabe in einer Datei, so muss man die Größenbeschränkungen des Dateisystems beachten. Häufig darf eine Datei nicht größer als 2 GB werden. Dann muss man dafür sorgen, dass die Ausgaben von `dd` diesen Wert nicht übersteigen. Hierzu besitzt `dd` die Option `count=x`. Es bedeutet, dass `dd` nur die ersten x Blöcke der Eingabedatei kopiert. Mit der Option `bs=x` kann man die Größe x eines Blockes (engl. **block size**) vorgeben. Bei Festplatten beträgt die Blockgröße typischerweise 512 Byte bis 4 KB.

Damit man auch den Rest der Eingabefestplatte kopieren kann, besitzt das Kommando `dd` noch die Option `skip=x`. Damit legt man fest, dass die ersten x Blöcke ignoriert werden sollen. `dd` beginnt erst danach zu kopieren. Möchte man also eine 3 GB große Partition in „CD-kompatible Häppchen“ zerlegen, benötigt man diese drei `dd`-Kommandos:

```
dd if=/dev/hda2 of=/pfad/zu/img1 bs=1M count=700
dd if=/dev/hda2 of=/pfad/zu/img2 bs=1M skip=700 count=700
dd if=/dev/hda2 of=/pfad/zu/img3 bs=1M skip=1400 count=700
```

Das erste Kommando kopiert 700 Blöcke der Größe 1 Megabyte. Das zweite überspringt 700 Blöcke und kopiert die nächsten 700 usw.

Alternativ kann man mit dem Kommando `split` die Ausgabe aufteilen. Das entsprechende Kommando lautet:

```
dd if=/dev/hdc | split -b 700m - /pfad/zu/dateien.
```

Dann werden im Verzeichnis `/pfad/zu` die Dateien `dateien.aa`, `dateien.ab`, `dateien.ac`, usw. angelegt. Sie können dann einzeln auf eine CD gebrannt werden. Das Kommando `cat dateien.*` fügt sie wieder zusammen.

Von allen gesicherten Festplatten(partitionen) muss man natürlich wieder die kryptografischen Prüfsummen berechnen und protokollieren.

Windows

Für das Betriebssystem Windows existieren verschiedene Programme, die den Inhalt kompletter Festplatten bzw. kompletter Partitionen in eine Image-Datei kopieren. Sie werden überwiegend kommerziell vertrieben und müssen extra gekauft werden. Für den Privatanwender ist es daher günstiger, eine „boot-fähige“ Linux-CD zu besitzen und mit den gerade vorgestellten UNIX-Kommandos die Sicherung der persistenten Daten vorzunehmen.

1.3.2 Angriff analysieren

Um zu verstehen, was der Angreifer mit dem System gemacht hat, muss man sich das System genauer ansehen. Konkret untersucht man (1) das Dateisystem und (2) die Protokolldateien (engl. **log files**).

Übungsaufgabe 1.3 Handelt es sich bei diesen beiden Punkten um persistente oder um nicht persistente Information?

Dateisystem untersuchen: Nachdem man die Festplatteninhalte gesichert hat, beginnt man mit der Untersuchung. Dazu erstellt man sich eine weitere Untersuchungskopie der Daten. Hat man eine weitere Festplatte zur Verfügung, dann kann man die Sicherungskopie auf diese Festplatte kopieren und sie in den Computer einbauen. In UNIX muss man den Inhalt einer Platte in den Verzeichnisbaum einbinden (engl. **to mount**). Dazu muss man nur den Typ des Dateisystems kennen und wissen, welche Gerätedatei die Platte vom Betriebssystem zugewiesen bekommt. Das Kommando

```
mount -t vfat -o ro,noexec /dev/fd0 /mnt/analysis
```

bindet den Inhalt einer Floppy-Disk (Gerät: `/dev/fd0`) an der Stelle `/mnt/analysis` in den Verzeichnisbaum ein. Das Dateisystem (vfat) ist das klassische DOS-Dateisystem und die Optionen *read only (ro)* und *no execute (noexec)* werden gesetzt. Das bedeutet, man kann im Verzeichnis `/mnt/analysis` nichts verändern und dort gespeicherte Programmdateien können nicht ausgeführt werden.

Man kann auch die oben erstellten Sicherungskopien der Partitionen direkt einbinden. In UNIX gibt es das sogenannte *Loop-back-Device*. Das Betriebssystem kann dann den Inhalt einer Datei als Partition mit eigenem Dateisystem interpretieren. Ist der Inhalt der Floppy-Disk beispielsweise in einer Datei mit dem Namen `floppy.image` gespeichert, dann ist das Kommando

```
mount -t vfat2 -o ro,noexec,loop floppy.image /mnt/analysis
```

äquivalent zum oben genannten Mount-Kommando.

Man kann nun mit den normalen UNIX-Kommandos durch das Dateisystem navigieren und die Dateien untersuchen. Man sucht hierbei nach „interessanten“ Dateien. Besonders interessant sind Dateien, die

Welche Dateien untersuchen?

1. in irgendeiner Form versteckt wurden,
2. zu denen Systemdateien gehören und *nicht* mehr im ursprünglichen Zustand sind oder

3. vor kurzem erst geändert wurden.

Versteckte Dateien

Zu 1. Eine Datei kann man in UNIX beispielsweise dadurch verbergen, indem man den Dateinamen mit einem Punkt beginnen lässt. Solche Dateien werden vom Kommando `ls` normalerweise nicht angezeigt, können aber durch die Option `-a` sichtbar gemacht werden. Weiterhin sind ausführbare Dateien in ungewöhnlichen Verzeichnissen interessant. Ist dann der Dateiname noch so gewählt, dass er auf einen anderen Inhaltstyp hinweist, dann hat man vermutlich eine Manipulation gefunden. Unter dem Namen `datei.jpg` erwartet man eine Grafik und kein Programm.

Unter UNIX findet man solche Dateien mit Hilfe des Kommandos `file`. Möchte man es auf alle Dateien im aktuellen Verzeichnis und in allen Unterverzeichnissen ausführen, so benötigt man auch noch das Kommando `find`.

```
find . -type f -exec file {} \; > filetypelist
```

Dieses Kommando erzeugt die Datei `filetypelist`, in der dann zu jeder Datei der zugehörige Typ steht. Man kann diese Datei nun selbst durchsehen oder mit Hilfe des Kommandos `grep` nach interessanten Mustern automatisch suchen.

Manipulierte Systemdateien

Zu 2. Um Veränderungen an Systemdateien zu erkennen, muss man natürlich die Originaldateien kennen. Ersatzweise reichen auch kryptografische Prüfsummen, wie die oben bereits erwähnte SHA1-Prüfsumme. Wenn man vor dem Angriff solche Prüfsummen erstellt hat, dann kann man hinterher die veränderten Dateien anhand der nun veränderten Prüfsumme einfach identifizieren. Das Intrusion-Detection-System *tripwire* macht genau das. Es wird in Kurs *(01867) Sicherheit im Internet 2* genauer vorgestellt.

An dieser Stelle sollte man auch einen Blick in `/etc/passwd` und `/etc/shadow` werfen. Angreifer legen für sich gerne neue Benutzer an, damit sie sich später einfach wieder anmelden können. Deshalb sind neue Benutzer oder Benutzer, deren Privilegien verändert wurden, besonders interessant. Veränderte Privilegien könnten eine neue numerische User-ID sein oder eine veränderte Gruppenzugehörigkeit.

Zuletzt veränderte Dateien

Zu 3. Die zuletzt geänderten Dateien erkennt man an den Meta-Informationen der Dateien. In modernen Dateisystemen (wie NTFS, ext2, ext3, ReiserFS usw.) wird zu jeder Datei der Zeitpunkt der Erstellung, der letzten Änderung und des letzten Zugriffs gespeichert. In `UNIX` zeigt das Kommando `ls -ltc` die Dateien eines Verzeichnisses nach dem Datum der letzten Änderung der Datei an. Mit `ls -ltu` wird die Zeit des letzten Zugriffs angezeigt und nach ihr sortiert. Der letzte Zugriff kann dabei ein Schreibzugriff oder ein Lesezugriff sein.

Wie untersuchen?

Nachdem nun interessante Dateien gefunden wurden, müssen diese als Nächstes genauer untersucht werden. Oft handelt es sich um ein compiliertes Programm. Normalerweise startet man sie einfach, d. h. man führt die Datei in der es steht aus. Da man befürchten muss, dass das Programm etwas Böses macht, wird man es aber nicht einfach starten wollen. Stattdessen kann man mit dem Kommando `strings` nach Zeichenketten in der Datei suchen, die ASCII-Strings entsprechen. Dadurch kann man dann oft herausfinden, um welches Programm es sich handelt und wie man es aufruft. Solche Informationen stehen i. d. R. in

einer sog. „Usage“-Nachricht. Sie wird vom Programm auch ausgegeben, wenn man es mit falschen Parametern aufruft.

Man kann dieses Kommando auch auf die gesicherten Festplatten-Images anwenden. Dann werden alle Sektoren, also auch zu keiner Datei mehr gehörende Sektoren durchsucht. So kann man dann auch die Inhalte gelöschter Dateien rekonstruieren. Wonach man nun genau suchen muss, lässt sich nur schwer im Voraus definieren. Strings wie „backdoor“, „/bin/sh“ oder „anonym“ usw. könnten in manipulierten Programmen vorkommen. Intelligente Angreifer können solche Strings aber auch einfach verbergen.

Neben den Inhalten der Dateien sind aber auch andere Bereiche der Festplatte wichtig. Ein Angreifer könnte einige Dateien schließlich auch wieder gelöscht haben. Normalerweise löschen Betriebssysteme Dateien, indem sie die Blöcke/Sektoren der Festplatte als frei markieren. Die alten Daten dort werden nicht gelöscht. Man kann sie also so lange wiederfinden, wie keine neuen Daten auf diese Blöcke der Festplatte geschrieben wurden.

Gelöschte Dateien

Protokolldateien analysieren: Neben den Änderungen an Dateien könnte ein Angreifer auch Spuren in den Protokolldateien (engl. **log files**) hinterlassen haben. Das setzt allerdings voraus, dass (1) die Protokollierung aktiviert war und (2) der Angreifer keine Gelegenheit hatte die Protokolldateien zu verändern. In diesem Fall sollte man die Protokolldateien genauer untersuchen und ungewöhnliche Aktivitäten identifizieren.

Das setzt wiederum voraus, dass man die gewöhnlichen Aktivitäten des Systems kennt. Der Administrator muss also wissen, wie sein System normalerweise benutzt wird und welche Einträge das in den Protokolldateien zur Folge hat.

In den Protokolldateien sucht man mindestens nach:

- erfolglosen Anmeldeversuchen, insbesondere bei Administratorkennungen,
- erfolgreiche Anmeldungen kurz vor dem vermuteten Angriffsbeginn bzw. die letzten Anmeldungen vor Beginn der Analyse,
- neu gestarteten oder neu installierten Diensten,
- Informationen, wann das System vom wem heruntergefahren oder neu gestartet wurde (In UNIX mit dem Kommando `last -f /var/log/wtmp`.),
- Änderungen an der Hardwarekonfiguration.

Weitere Hinweise, in welchen Dateien diese Protokolle stehen, wonach man noch suchen sollte usw. findet man in weiterführender Literatur [Ges04, MAC+03]. Spezielle Software kann die forensischen Untersuchungen unterstützen. Hierzu gehören Programme wie:

Programme

EnCase: Hierbei handelt es sich um ein kommerzielles Programm der Firma *Guidance Software*. Es läuft unter Windows und kann aber auch UNIX-Dateisysteme lesen und bearbeiten.

`http://www.guidancesoftware.com/`

The Sleuth Kit: Dies ist eine Sammlung von UNIX-Hilfprogrammen zur Analyse von Festplatten bzw. Partitionen. Dabei können nicht nur UNIX-Dateisysteme, sondern auch FAT- und NTFS-Dateisysteme aus der Windows-Welt untersucht werden.

`http://www.sleuthkit.org/sleuthkit/`

Autopsy: Hierbei handelt es sich um eine grafische Benutzerschnittstelle für das *Sleuth Kit*.

`http://www.sleuthkit.org/autopsy/`

Die Programme suchen dann automatisch nach verdächtigen Inhalten der Partitionen oder der Protokolldateien. Diese Liste ist natürlich unvollständig und ständigen Erweiterungen unterworfen.

Gegenmaßnahmen: Nachdem man nun erfahren hat, was der Angreifer mit dem System gemacht hat, möchte man solche erfolgreichen Angriffe zukünftig natürlich verhindern. Man muss also die Schwachstelle, die zum Einbruch geführt hat, identifizieren. Irgendwie muss es der Angreifer ja geschafft haben, sich am System anzumelden und Kommandos auszuführen. Das könnte diese Ursachen gehabt haben:

Programmfehler

- Ausnutzen einer bekannten Schwachstelle in einem Dienst, der auf dem System angeboten wird. Das könnte beispielsweise der Webserver sein, in dem ein Buffer-Overflow ausgenutzt wurde.

  Man muss dann dafür sorgen, dass entweder eine neue Version der Software für diesen Dienst installiert wird - der Fehler muss in dieser Version natürlich behoben sein - oder man muss den Dienst vorübergehend sperren.

Passwort ausgespäht

- Kompromittierung einer lokalen Benutzerkennung, inklusive des Passworts. Das kann durch Unachtsamkeit eines Benutzers passieren oder aufgrund einer Passwortübertragung im Klartext. Außerdem könnte ein Virus solche Informationen gesammelt und nach außen übertragen haben.

  Diese Benutzerkennungen müssen auf jeden Fall durch ein neues und sicheres Passwort geschützt werden. Falls die Kompromittierung vermutlich auf Klartextübertragung des Passworts beruht, dann darf das neue Passwort selbstverständlich nicht wieder im Klartext übertragen werden.

Viren, Würmer, Trojaner

- Kompromittierung bestimmter Software. Arbeitet man unter Windows, beispielsweise unter einer Benutzerkennung, die zur Gruppe der Administratoren gehört, und startet unbekannte Programme, dann könnten diese Programme beliebige Änderungen am System vornehmen. Man sollte meinen, dass niemand so leichtsinnig ist. Leider zeigt die Erfahrung, dass dem nicht so ist.

  Als Gegenmaßnahme kann man sich hier „nur“ vornehmen, zukünftig vorsichtiger mit neuen Programmen zu sein.

1.3.3 System aktualisieren

Durch die Aktualisierung des Systems möchte man alle Angriffsspuren entfernen und das System wieder in einen vertrauenswürdigen und benutzbaren Zustand versetzen. Das bedeutet konkret:

- Rückgängig machen aller Veränderungen durch den Angreifer.
- Verhinderung zukünftiger Angriffe dieser Art.

Die Veränderungen des Angreifers kann man dadurch rückgängig machen, indem man eine Sicherungskopie von vor dem Angriff zurück auf das System kopiert. Zur Verhinderung zukünftiger Angriffe muss man dann weitere Änderungen vornehmen. Hierzu gehören die Aktualisierung unsicherer Sofwarekomponenten ebenso wie Änderungen an der Systemkonfiguration (Dienste sperren, Passwörter ändern usw.).

Unter Windows bleibt einem oft keine andere Wahl, als das System komplett neu zu installieren. Man kann dazu ein altes Image einspielen, das man sich einmal als Sicherungskopie erstellt hatte oder die Festplatte formatieren und das System neu installieren. Obwohl Letzteres natürlich sehr zeitaufwendig ist, hat es verschiedene Vorteile:

- Die Änderungen des Angreifers sind garantiert weg, wenn man die Festplatte formatiert hat.
- Das System ist wieder in einem definierten Anfangszustand. Hatte man beispielsweise zu Testzwecken Programme installiert und sie dann wieder entfernt, dann bleiben häufig bestimmte Reste im System, z. B. in der Registry. Auch diese sind nun komplett entfernt.
- Die Dateien liegen wieder defragmentiert auf der Festplatte.
- Man hat die Gelegenheit, auf eine neuere Version des Betriebssystems umzusteigen.

Nach der Neuinstallation sollte man auch sofort alle Sicherheits-Updates der Systemsoftware installieren. Danach kann man eine Sicherungskopie des neuen, frisch installierten und virenfreien Systems erstellen.

Letztlich kann man sich nun auch noch Gedanken machen, ob man gegen den Angreifer gerichtlich vorgehen will. Dazu muss der Angreifer natürlich zweifelsfrei identifiziert worden sein. Da man sich weltweit ins Internet einwählen kann und dann Verbindungen über mehrere Rechner hinweg aufbauen kann, ist es normalerweise sehr schwer jemanden eindeutig zu identifizieren.

Außerdem muss man alle Beweise korrekt erhoben haben, sie gegen nachträgliche Manipulationen sichern und auch beweisen können, dass die Analysewerkzeuge (Hardware und Software) in Ordnung waren und ihre Funktion korrekt erfüllt haben. Für den „normalen“ Privatanwender sind das sehr hohe Hürden und da man sich von einer Gerichtsverhandlung auch nicht unbedingt große Vorteile versprechen kann, ist es am Ende häufig sinnvoller auf ein Gerichtsverfahren zu verzichten.

Für kommerzielle Anwender mag das anders aussehen. Sie sollten sich rechtzeitig (lange vor Angriffen) überlegen, wie sie sich und ihre Systeme schützen. Außerdem sollten sie sich vorab überlegen, was bei einem Angriff zu tun ist. Man kann in diesen Fällen beispielsweise auf die Hilfe professioneller Experten zurückgreifen.

1.4 Zusammenfassung

Nach dem Durcharbeiten dieser Kurseinheit sollten Sie die folgenden Fragen beantworten können:

- Was bedeuten die Begriffe Anonymität und Pseudonymität?
- Welchen Gefährdungen ist Ihre Anonymität bzw. Pseudonymität im Internet ausgesetzt?
- Mit welchen technischen Mitteln können Sie Ihre Privatsphäre im Internet schützen?
- Welche Vorteile haben aktive Inhalte?
- Welchen Gefahren ist man durch aktive Inhalte ausgesetzt?
- Was ist bei der Beweissicherung und der Analyse im Rahmen der Computer-Forensik zu beachten?

Lösungen der Übungsaufgaben

Übungsaufgabe 1.1 Ja, der Benutzer muss nur die Reihenfolge der öffentlichen Schlüssel beim Verschlüsseln der Nachricht an die Reihenfolge der Mixe anpassen.

Übungsaufgabe 1.2 In diesem Fall kann ein sogenannter Phisher einen Webserver aufsetzen, der neben dem HTML-Code, der das Aussehen einer Internet-Banking-Seite nachahmt, auch ein JavaScript-Programm einbauen. Das JavaScript-Programm zeigt dem Benutzer im Browser nun an, dass er vermeintlich auf der Seite der Bank sei und dass die Verbindung natürlich mit SSL verschlüsselt wird.

Übungsaufgabe 1.3 Da es sich bei beiden Punkten um Daten auf der Festplatte handelt, sind es persistente Informationen. Sie würden einen Neustart des Systems überdauern.