Kapitel 3
Künftige Entwicklung von Identitätsmissbrauch und Identitätsdiebstahl

Identitätsmissbrauch wird in Zukunft eine Vielzahl von Formen annehmen, die heute noch nicht komplett überblickt werden können, da auch eine Fülle von neuen Techniken, Paradigmen und Plattformen immer neue Angriffsszenarien ermöglicht.

Gleichzeitig werden die Werkzeuge, die Identitätsmissbrauch ermöglichen, immer ausgefeilter und technisch ausgereifter sein. Hier ist vor allem eine nicht kriminelle Hackerszene („White Hat“) zu nennen, die in einer von der akademischen Welt (vielleicht mit Ausnahme der USA) weitgehend unbeachteten Community immer neue Wege findet, Schutzmechanismen zu umgehen. Ohne diese „White Hat“-Community gäbe es keine Forschung auf diesem wichtigen Gebiet. Zudem geht sie sehr verantwortungsvoll mit ihren Entdeckungen um.

Das generelle Problem kann man ökonomisch umschreiben: Es ist billiger und effizienter, ein System zu hacken, als es gegen diese Hackerangriffe abzusichern. Hier fehlen generelle Konzepte, die auch die Kryptografie in Schutzparadigmen mit einbeziehen sollten.

I. Prognose: Angriffsszenarien

Prognose: Zukünftige Angriffsszenarien werden deutlich vielfältiger werden: Jeder Dienst, für den ein Angreifer einen lohnenden Business Case aufstellen kann, wird früher oder später angegriffen werden. Jeder Schutzmechanismus, der nicht ein beweisbares Sicherheitsniveau besitzt, wird umgangen werden. Und jede neue Technologie, die mit neuen Features die Welt des (mobilen) Internets bereichert, kann zu einer Basis für zukünftige Angreifer werden.

1. Business Cases zukünftiger Angreifer

Prognose: Alle Dienste, für die der Angreifer einen „Business Case“ berechnen kann, bei dem nach Abzug aller Investitionen ein Gewinn erzielt wird, werden

G. Borges et al., *Identitätsdiebstahl und Identitätsmissbrauch im Internet,*
DOI 10.1007/978-3-642-15833-9_3, © Springer-Verlag Berlin Heidelberg 2011

früher oder später Gegenstand eines Angriffs werden. Angriffe auf Onlinebanking und verwandte Dienste (Giropay, Sofortueberweisung.de) werden hier auch weiterhin im Fokus stehen, da der dahinter liegende Business Case sehr gut bekannt ist und ständig weiterentwickelt wird.

Für zukünftige Dienste kann man den erwarteten Umsatz nicht im Vorhinein einschätzen, hier ist zeitnahes Handeln erforderlich. Abschätzen kann man heute aber schon die zu tätigenden Investitionen. Durch Arbeitsteilung werden diese Investitionen minimiert, sodass herkömmliche Kriterien wie z. B. die Angreiferklassifikation der Common Criteria überarbeitet werden müssen.

Folgende Investitionen können abgeschätzt werden:

- Umgehung von Antivirensoftware, die auf AV-Signaturen beruht: Malware wird heute überwiegend in Hochsprachen geschrieben. Funktionsidentische Varianten, die unterschiedliche MD5-Prüfsummen aufweisen, können daher sehr leicht durch Veränderung der Compilereinstellungen verändert werden. Dies führt zu einem explosionsartigen Anwachsen der Zahl der Malwaresignaturen. Hier sind daher für einen Angreifer nur geringe Investitionen zu kalkulieren (vgl. auch [GP09]). Einige Zitate dazu:

  In den ersten acht Monaten 2008 hat PandaLabs mehr Malware identifiziert als in den vergangenen 17 Jahren zusammen – 22.000 täglich. [PandaLabs08-AR]
  2008 entdeckten die Sophoslabs durchschnittlich 16.173 neu mit Schadcode infizierte Websites pro Tag. [Sophos08-STR]

- Umgehung von AV-Software, die auf verhaltens- oder strukturbasierter Analyse beruht: Bei diesen neuen Analyseansätzen wird Schadsoftware entweder anhand ihres Verhaltens (Welche Funktionen des Betriebssystems ruft sie auf, wie verläuft ihre Kommunikation mit dem Internet?) oder anhand ihrer Struktur (Aus welchen Unterprogrammen besteht sie, wie rufen diese sich gegenseitig auf?) erkannt. Da hier noch keine marktfähigen Produkte vorliegen, können die Investitionen in die Umgehung dieser Mechanismen noch nicht abgeschätzt werden.
- Umgehung von Personal Firewalls: Diese sind heute schon nutzlos, da sich die Black-Hat-Community sehr schnell von der Ausnutzung von Schwachstellen in Netzwerkdiensten im Betriebssystem auf sogenannte Drive-by-Downloads [Google08] umgestellt hat.
- Umgehung chipkartenbasierter Sicherheitsmechanismen: Chipkarten als alleiniges Allheilmittel für Sicherheitsprobleme im Internet sind in [Langweg06] und [LS07] entzaubert worden. Bei vernünftiger Aufteilung der Aufgaben zwischen PC und Chipkarte/Kartenleser (vgl. [GLS07]) sind die realisierten Systeme aber technisch unknackbar. (Es bleibt ein Risiko für Social Engineering-Angriffe, dies kann aber durch entsprechende Gestaltung der Nutzerschnittstelle minimiert werden.) Nicht alle Dienste können aber über einen korrekt eingebundenen Chipkartenleser abgesichert werden.
- Arbeitsteilung: Die Umgehung von neu eingeführten Schutzmaßnahmen ist in der arbeitsteiligen Black-Hat-Community nicht mehr Aufgabe des eigentlichen Angreifers, sondern wird von Spezialisten für eine Vielzahl von Kunden erle-

digt. Dies schlägt sich in den Preisen nieder, die für vollautomatische Angriffstools verlangt werden:

- mPack: ca 700–1.000 USD
- IcePack: ca 400 USD
- FirePack: ca 3.000 USD

Zur Abschätzung der zukünftigen Entwicklung wird also der betriebswirtschaftliche Aspekt immer wichtiger: Eine Applikation sollte so abgesichert sein, dass sich ein kommerzieller Angriff nicht lohnt. Hier besteht allerdings noch erheblicher Forschungsbedarf.

2. Umgehung von Standardsicherheitsmaßnahmen

Prognose: Standardsicherheitsmaßnahmen bieten aktuell nur noch einen rudimentären Basisschutz und werden in Zukunft immer mehr an Bedeutung verlieren. Keinesfalls dürfen sie daher als alleinige Lösung für die Probleme des Identitätsdiebstahls gesehen werden. Wichtig ist vielmehr der verantwortungsbewusste Umgang mit dem Medium Internet. Allerdings muss einschränkend erwähnt werden, dass es dem normalen Nutzer nicht abzuverlangen ist, eine korrekte Entscheidung über die Vertrauenswürdigkeit von Websites zu treffen. Wichtig sind daher weitere Entwicklungen, die den Nutzer dabei unterstützen.

Zur Absicherung der Clientcomputer setzt man – wie bereits auf S. 56 ff. beschrieben – häufig auf die Kombination von Antivirensoftware und Personal Firewalls, eventuell unter Zuhilfenahme von Browsererweiterungen. Ziel dieses Abschnittes ist es, nun zu zeigen, mit welchen Methoden bzw. mit welchem Aufwand ein Angreifer diese standardmäßigen Schutzmaßnahmen umgehen kann. Die Angreifer bedienen sich dabei auch immer mehr der Methoden des Social Engineering und nutzen dabei zum einen die „Gier" der Anwender, zum anderen aber auch immer mehr explizit das Gefahrenbewusstsein des Anwenders aus.

a) Umgehen von Antivirenprogrammen

Prognose: Antivirenprogramme lassen sich bereits heute vom Angreifer mit relativ geringem Aufwand umgehen. In Zukunft werden die Aufklärung der Anwender und der daraus resultierende bewusste und vorsichtige Umgang mit dem Medium Internet immer wichtiger werden. Zugleich sind die Hersteller von Antivirenprogrammen gefordert, ihre Erkennungsmechanismen vor allem in Richtung einer proaktiven Erkennung von Malware weiter zu verbessern.

Häufig werden Antivirenprogramme als Standardsicherheitsmaßnahme empfohlen. So spricht sich beispielsweise auch das Bundesamt für Sicherheit in der Informationstechnik (BSI) explizit für den Einsatz von Antivirenprogrammen aus.[171]

[171] Vgl. dazu auch die Websites des BSI: https://www.bsi-fuer-buerger.de/cln_164/BSIFB/DE/ITSicherheit/SchuetzenAberWie/schuetzenaberwie_node.html.

Leider wird im Zuge dieser Diskussion aber gerne verschwiegen, dass Antivirenprogramme nur einen *Basisschutz* bieten können. Wird dieser Aspekt nicht ausreichend kommuniziert, wägt sich der Anwender in scheinbarer Sicherheit, denn gerade Antivirenprogramme können von Angreifern relativ einfach umgangen werden.

Auf S. 57 ff. wurde bereits ausführlich diskutiert, dass die Angreifer sich immer mehr auch der Methoden des Social Engineering bedienen und das Gefahrenbewusstsein der Anwender ausnutzen. Sogenannte Rogue-AV-Programme müssen heute als Standardangriffsweg angesehen werden – eine entsprechende Schulung der Nutzer ist somit unbedingt erforderlich. Angreifer können die heute überwiegend reaktiven Erkennungsmechanismen bei Antivirenprogrammen aber auch ohne den Einsatz von Social Engineering relativ einfach umgehen. Dies soll nun an zwei konkreten Beispielen verdeutlicht werden.

Zum einen sind in entsprechenden Kreisen zahlreiche Programme im Umlauf, die den Angreifer dabei unterstützen, dass seine Malware von aktuellen Antivirenprogrammen nicht erkannt wird.[172] Tools wie Multi AVs Fixer, Scanlix, KIMS[173] und weitere dieser Art bieten dem Angreifer dabei professionelle Unterstützung.[174] Dabei kann zum einen der parallele Scan mit verschiedenen Antivirenprogrammen durchgeführt werden. Zum Teil haben die Tools zudem auch die Option, die erstellte Malware automatisch so zu modifizieren, dass kein Antivirenprogramm die Malware erkennt. Diese Tools sind im Internet frei erhältlich, zum Teil existieren sogar Videoanleitungen[175] für den richtigen Gebrauch. Es muss allerdings betont werden, dass ein Angreifer auch leicht ohne diese automatisierten Tools auskommen kann. Möglicherweise sind die Tools vor allem deswegen ungefährlich, weil sie hauptsächlich von den technisch nicht versierten Angreifern genutzt werden. Ein Verbot solcher Tools kann daher durchaus kritisch diskutiert werden: Technisch versierte Angreifer werden im Zweifel auch ohne diese Tools auskommen, die Massenangreifer lassen sich aber durch diese Tools besser beobachten. Die Abbildungen 17 und 18 zeigen die Tools Multi AVs Fixer[176] und KIMS[177]. Beide bieten dem Angreifer die Möglichkeit, seine Malware auf einfache Art und Weise auf die Erkennung durch Antivirenprogramme zu testen und anschließend so zu modifizieren, dass sie von aktuellen Antivirenprogrammen nicht mehr erkannt wird.

Greveler und Puls konnten zudem in einem Beitrag[178] zum 11. Deutschen IT-Sicherheitskongress des BSI im Jahre 2009 zeigen, dass sich der Aufwand, eine

---

[172] Siehe dazu auch folgenden Bericht: http://www.itweb.co.za/index.php?option=com_content&view=article&id=2519.

[173] Zur Diskussion bzgl. KIMS siehe auch: http://foro.portalhacker.net/index.php/topic,65709.0.html.

[174] Ein detaillierter Bericht zu den verschiedenen Tools findet sich beispielsweise unter: http://pandalabs.pandasecurity.com/archive/Multi-AVs-Scanners.aspx.

[175] Siehe beispielsweise unter: http://www.4shared.com/file/39095752/e615d1ea/MultiAVFixer-VideoTutorial.html.

[176] Quelle: http://pandalabs.pandasecurity.com/.

[177] Zur weiteren Diskussion siehe auch: http://foro.portalhacker.net.

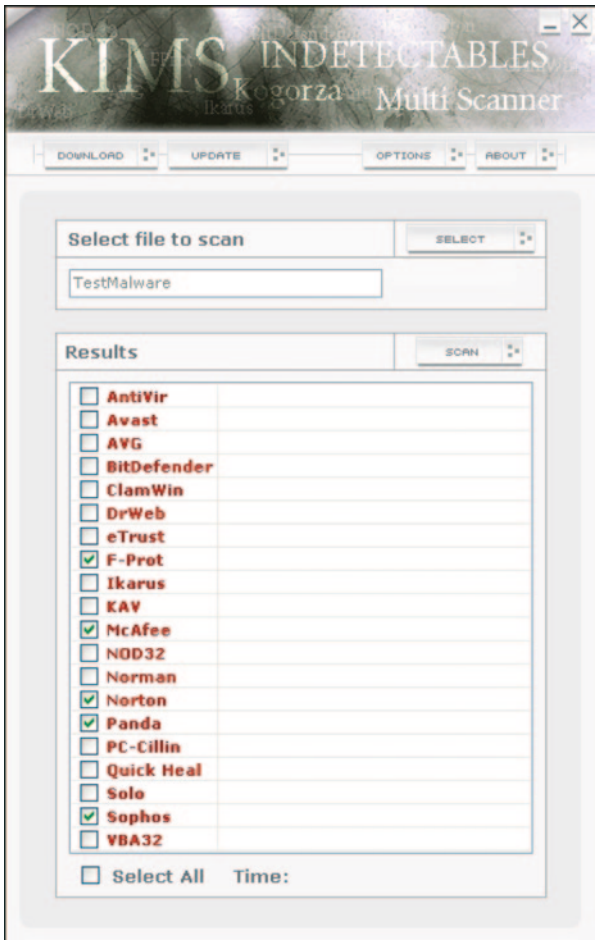
[178] Siehe dazu den Beitrag „Über den Aufwand, Malware auf einem privaten PC zu installieren – Wie einfach lassen sich Virenscanner und Personal Firewalls umgehen?“ von *Greveler* und *Puls* im Tagungsband zum 11. Deutschen IT-Sicherheitskongress „Sichere Wege in einer vernetzten Welt“.

Abb. 17 Tool Multi AVs Fixer. (Quelle: http://pandalabs.pandasecurity.com/multi-avs-scanners/)

Malware zu entwickeln, die von keinem aktuellen Antivirenprogramm gefunden wird, in Grenzen hält. Auf Basis des bekannten Remote-Administrationtools *Back Orifice 2000*[179] entwickelten sie dazu eine Malware, die von keinem der 36 verschiedenen und zum Zeitpunkt der Untersuchung mit aktuellen Virensignaturdateien ausgestatteten Antivirenprogrammen erkannt wurde. Dazu modifizierten sie die Malware in verschiedenen Schritten: Als erster Schritt stand die Neukompilierung der Binärdatei aus dem Quellcode. Bereits dies führte dazu, dass die Malware nur noch von 16 Antivirenprogrammen erkannt wurde. Durch eine weitere Modifikation mittels eines HEX-Editors wurde das Programm schließlich so umgebaut, dass sowohl die Erkennung auf Basis der Virensignaturdateien als auch mittels der Heuristisk vollständig ausgeschaltet werden konnte. Dies ist bemerkenswert, da es sich bei Back Orifice 2000 um eine der bekanntesten Malware handelt. Greveler und

[179] Zu den Details von Back Orifice 2000 siehe beispielsweise: http://www.bo2k.com/whatis.html.

Abb. 18 Tool KIMS



Puls stellen in ihrem Beitrag zudem fest, dass der Einsatz von den oft empfohlenen Standardsicherheitsmaßnahmen durchaus kontrovers diskutiert werden sollte. Sie werfen dabei auch die Frage auf, ob die von Antivirenprogrammen und Personal Firewalls suggerierte, aber oft nicht wirklich erzielte Sicherheit kontraproduktiv wirkt, da sich der Anwender blind auf die Funktionalität verlässt und deswegen unvorsichtig wird. Diese Beispiele zeigen eindrucksvoll, dass Angreifer keine großen Kenntnisse benötigen und ein Verzicht auf die im ersten Absatz dieses Abschnitts diskutierten Tools leicht möglich ist.

b) Umgehen von Personal Firewalls

Prognose: Schutzmaßnahmen wie Paketfilter-orientierte Personal Firewalls werden in Zukunft weiter an Bedeutung verlieren. Moderne Malwareverbreitungsmechanismen nutzen immer mehr freigegebene Kanäle, um nach außen zu kommunizieren.

Erforderlich sind daher mehr und mehr Maßnahmen, die eine vollständige Analyse des Internetverkehrs ermöglichen. Schwierig bleibt dabei aber die Erstellung der Referenzdatenbasis. Zudem muss beachtet werden, dass die Personal Firewalls der Zukunft aufgrund der steigenden Komplexität dann auch zu dem entscheidenden Angriffsvektor im System werden können.

Personal Firewalls werden als Basisschutz empfohlen, bieten gegen moderne Infektionsroutinen aber definitiv auch nicht mehr als einen solchen. Vor allem moderne Drive-by-Malware[180,181] ist in der Lage, die Sicherheitsfunktionen der Personal Firewalls auszuhebeln. Interessant sind in diesem Zusammenhang die sogenannten Web Exploit Toolkits (kurz: WET), die seit Ende 2006 immer mehr an Relevanz gewinnen und eine sehr effektive Art von Angriffssoftware darstellen. WETs spielen im Geschäftsprozess rund um die Malware eine bedeutende Rolle, weil sie maßgeblich für die Infektion neuer Clients und damit den wirtschaftlichen Erfolg des gesamten Malwarekreislaufs verantwortlich sind. WETs integrieren betroffene Clientsysteme – häufig handelt es sich um Systeme, die Windows als Betriebssystem verwenden – beispielsweise in Bot-Netze. Alternativ kann der Betreiber des WET-Servers infizierte Clientsysteme auch an Interessierte weiterverkaufen, die diese dann beispielsweise für Denial-of-Service(DoS)-Angriffe oder SPAM-Aktionen nutzen können. Bekannte Vertreter der WET sind beispielsweise *IcePack*[182], *MPack*[183] oder auch *Neosploit*[184].

Die Infektion eines Clients vollzieht sich dabei in mehreren Schritten: Zunächst muss der Client bzw. dessen Anwender auf eine Website gelockt werden, auf der der entsprechende Schadcode vorhanden ist. Gerne werden dazu Websites verwendet, denen der Nutzer ein gewisses Grundvertrauen entgegenbringt. In der Vergangenheit waren so auch die Websites von News.com, USA Today und auch Unicef betroffen[185]. Ein weiteres Kriterium für die Auswahl der Website kann auch deren Frequentierung durch die Nutzer sein. Erfolgt der Angriff in zeitlicher Hinsicht geschickt, lässt sich das Ausmaß der Infektion entscheidend beeinflussen. Dies wurde beispielsweise beim Angriff auf die Website der Miami Dolphins ausgenutzt, die kurz vor dem entscheidenden Spiel im amerikanischen Football, dem Superbowl, entsprechend manipuliert wurde.[186] Die Abbildung 19 zeigt das Ergebnis spezieller

---

[180] Zur Thematik der Drive-by-Malware siehe auch: http://www.heise.de/security/meldung/Drive-By-Malware-und-Wegwerfviren-im-Trend-206078.html.

[181] Eine umfassende Untersuchung zur Drive-by-Malware von *Niels Provos*, *Dean McNamee*, *Panayiotis Mavrommatis*, *Ke Wang* und *Nagendra Modadugu*, „The ghost in the browser: analysis of web-based malware“, (veröffentlicht auf der HotBots'07-Konferenz) ist online verfügbar: http://www.usenix.org/events/hotbots07/tech/full_papers/provos/provos.pdf.

[182] Für eine detaillierte Analyse und Beschreibung von IcePack siehe auch: http://pandalabs.pandasecurity.com/blogs/images/PandaLabs/2007/12/18/Icepack.pdf.

[183] Für eine detaillierte Analyse und Beschreibung von MPack siehe auch: http://pandalabs.pandasecurity.com/blogs/images/PandaLabs/2007/05/11/MPack.pdf.

[184] Siehe hierzu auch: http://dxp2532.blogspot.com/2007/12/neosploit-exploit-toolkit.html.

[185] In diesem Zusammenhang soll auch auf die Möglichkeit hingewiesen werden, mittels spezieller Google-Suchanfragen IFrame-verseuchte Websites zu lokalisieren.

[186] Siehe dazu auch: http://www.webappsec.org/projects/whid/byid_id_2007-10.shtml.

Abb. 19 Ergebnis spezieller Google-Suchanfragen im Zusammenhang mit der IFrame-Verseuchung diverser Websites

Google-Suchanfragen im Zusammenhang mit der IFrame-Verseuchung diverser Websites.[187]

Darüber hinaus lassen sich auch Methoden der Search Engine Optimization[188] (kurz: SEO) dazu verwenden, den Nutzer mittels Google-Suchergebnissen auf verseuchte Websites zu lenken. Eine weitere Möglichkeit für den Angreifer ist der Kauf von Domains mit ähnlichem Namen in der Hoffnung, dass sich die Anwender vertippen oder den Unterschied bei einem publizierten Link einfach nicht bemerken. Zudem kann der Angreifer Anzeigen bei Google schalten, die dann bei der Suche nach bestimmten Begriffen eingeblendet werden.

Damit die weiteren Schritte des Angriffs Erfolg haben, muss aber zunächst der Schadcode auf dem Webserver platziert werden. Dazu kann etwa die Methode der Remote File Inclusion[189] (kurz: RFI) verwendet werden, bei der meist fehlerhafte PHP-Skripte dazu genutzt werden, eine Shell auf den Webserver zu bringen. Mithilfe dieser Shell kann der Angreifer dann im Kontext des Webserverprozesses beliebige Befehle ausführen und so beispielsweise den Inhalt von auf diesem Server gehosteten Websites manipulieren. Dabei schleust der Angreifer etwa einen (unsichtbaren) *IFrame*[190] ein, der einen Verweis auf die Schadfunktion enthält.

Surft ein Nutzer nun auf eine solche präparierte Website und ist sein Browser anfällig für den dort abgelegten Exploit, so erfolgt die Übernahme des PCs. Der Angriff bzw. das Einbringen des Schadcodes auf den Client erfolgt dann in mehreren Schritten:

1. Initialer Schritt ist, dass der Client auf die manipulierte Website hereinfällt.
2. Das auf dem Webserver platzierte PHP-Skript analysiert den Client – insbesondere die verwendete Browserversion –, indem etwa der User-Agent ausgewertet

[187] Quelle: eigene Arbeiten.

[188] Zur Erklärung von SEO siehe beispielsweise: http://de.wikipedia.org/wiki/Search_Engine_Optimization.

[189] Zur Erklärung der Remote File Inclusion siehe: http://de.wikipedia.org/wiki/Remote_File_Inclusion.

[190] Zur Diskussion bzgl. IFrames siehe beispielsweise: http://de.wikipedia.org/wiki/Inlineframe.

wird. Damit lassen sich unter Umständen auch Rückschlüsse auf das verwendete Betriebssystem ziehen. Die IP-Adresse des Clients kann zudem Hinweise zum Standort und damit auch zur verwendeten Sprachversion liefern.

3. Auf Basis der im vorherigen Schritt gesammelten Daten wird ein passender Exploit für den Client ausgewählt, an diesen übertragen und dann dort ausgeführt. Zugleich wird der PC des Opfers – je nach verwendetem WET – auf dem Server „registriert", um zu verhindern, dass ein und derselbe Rechner mehrfach attackiert wird.
4. Nun lädt der vom WET-Server auf den Client übertragene Exploit eine weitere Datei vom WET-Server oder einem weiteren Server nach, den sogenannten Loader. Durch die Verwendung eines weiteren Servers kann der Angreifer die Verfolgung und Bekämpfung zusätzlich erschweren.
5. Anschließend sorgt dieser Loader dafür, dass die eigentliche Malware auf den Client geladen und dann dort ausgeführt wird.

Die Abbildung 20 zeigt diese Schritte noch einmal in grafischer Form.[191]

Durch die Aufteilung der Infektion in mehrere Schritte – wobei auch mehrere unterschiedliche Server an der Verteilung der einzelnen Komponenten beteiligt sein können – wird die Bekämpfung und Rückverfolgung der Angreifer erheblich erschwert. Zusätzlich lässt sich anhand von dedizierten Logging- und Auswertungskomponenten auf den WET-Servern feststellen, dass mittlerweile eine erhebliche Professionalisierung der Malwareszene stattgefunden hat: Die Verbreitung und auch die anschließende Nutzung der malwareverseuchten PCs folgt einem strikten Geschäftsmodell, das auf Gewinnmaximierung ausgelegt ist und nichts mehr mit der „spielerischen" Komponente vergangener Tage zu tun hat.

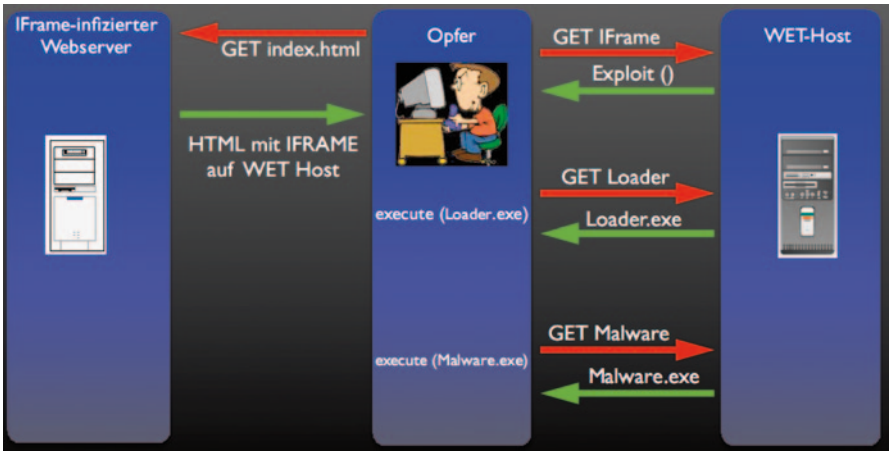


Abb. 20 Schrittfolge eines WET-Angriffs

[191] Quelle: eigene Arbeiten.

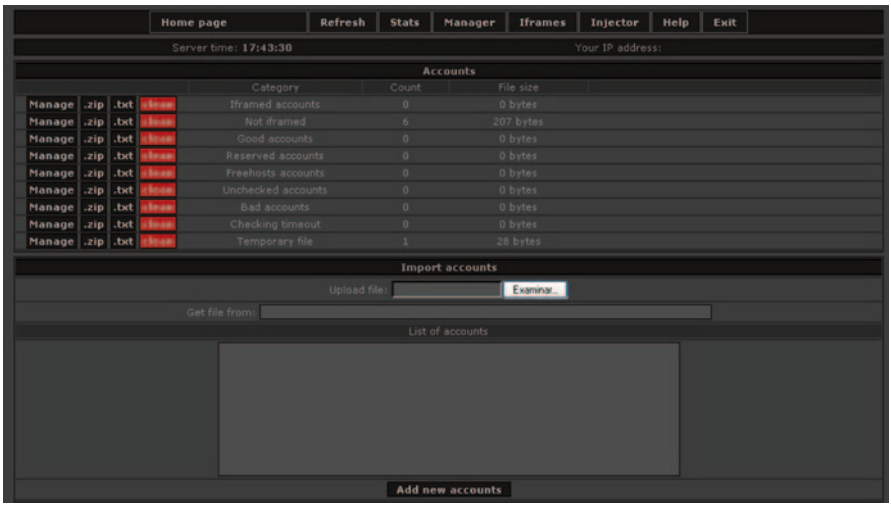


Abb. 21 Statistiken eines T-IFRAMER-WETs. (Quelle: http://mipistus.blogspot.com/2009/11/t-iframer-kit-para-la-inyeccion-de.html)

Wie in anderen Softwaresparten versucht man beispielsweise auch hier, auf Nutzerfreundlichkeit für den Angreifer zu setzen. Grundlegend sind neuartige WETs in vier verschiedene Teilbereiche zu gliedern:

- Statistiken:
  Über sämtliche versuchte wie erfolgreiche Infektionen werden Statistiken für den Betreiber des WETs geführt. So kann sich der Angreifer zu jeder Zeit einen Überblick über das Ausmaß seines WETs verschaffen.
  Zusätzlich lassen sich unter diesem Punkt auch Dateien mit kompromittierten FTP-Accounts hochladen, was exemplarisch in der Abb. 21 aufgezeigt wird. Die WET-Skripte versuchen anschließend, die hochgeladenen Accounts sukzessive auf ihre Validität zu überprüfen. Mit dieser Methode ist es für Angreifer einfach geworden, Malware oder sogar ganze WETs schnell und problemlos weiterzuverbreiten.
- Manager:
  Bei diesem Menüpunkt handelt es sich um die Managementoberfläche zur allgemeinen Administration des WETs. Über diesen Punkt werden auch die getesteten FTP-Accounts verwaltet. Als Administrator des WETs kann man hier festlegen, was mit den FTP-Accounts passieren soll.
- IFrames:
  In diesem Unterpunkt spezifiziert der Betreiber des WETs, wie die IFrame-Infektionen stattfinden sollen. Oftmals referenzieren bösartige IFrames nicht direkt ein WET, sondern leiten den Nutzer zunächst auf ein oder mehrere weitere IFrames um. Somit soll letztlich die Verfolgung des WETs erschwert werden.
- Injector:
  Das Injectormodul ist für die eigentliche Infektion des Clientsystems verantwortlich. Hier können der Exploitcode ausgewählt, Länder- oder IP-Adress-Bereiche

von der Infektion ausgenommen und Schlüsselwörter für die Durchsuchung von kompromittierten FTP-Servern angegeben werden.

Im Falle eines neuartigen WETs mit dem Namen *T-IFRAMER* haben die Entwickler auch auf vielfältige Exploitmöglichkeiten Wert gelegt. Im Folgenden werden einige davon aufgezählt:

- CVE-2009-0927 (Adobe getIcon),
- CVE-2008-2463 (Office Snapshot Viewer),
- CVE-2008-2992 (Adobe util.printf overflow),
- CVE-2008-0015 (MsVidCtl Overflow) und
- CVE-2007-5659 (Adobe Collab overflow).

Die Malware, die letztlich auf das Clientsystem geladen wird, steht den Exploits in Sachen Vielfältigkeit in nichts nach.

- *ehkruz1.exe*
  Hierbei handelt es sich um einen Trojaner, der bereits mit vielen verschiedenen und zufälligen Dateinamen aufgetaucht ist. Es wird vermutet, dass der Trojaner Accountdaten für den WebMoney-Service abgreifen soll.
- *egiz.pdf*
  Das bösartige PDF enthält Exploits wie CVE-2007-5659, CVE-2008-2992 und CVE-2009-0927, um PDF-Reader zu attackieren.
- *manual.swf*
  Diese Datei beinhaltet einen Exploit für den Adobe Flash Player.
- sdfg.jar
  Hierbei handelt es sich um einen Trojaner, der mit einem Exploit als Downloader fungiert.
- *ghknpxds.jpg*
  Diese Datei beinhaltet einen als Bild getarnten Exploit.

Die Infektion eines PCs durch ein WET ist leider nur schwer zu verhindern. Vor allem Standardschutzmechanismen – wie die oft empfohlenen Antivirenprogramme und Personal Firewalls – können hier nur einen sehr geringen Schutz bieten. In jedem Fall ist aber ein regelmäßiges Updaten des Betriebssystems und vor allem auch der verwendeten Internetbrowser zu empfehlen. Auch Maßnahmen wie die Verwendung der Firefox-Erweiterung „NoScript!“ können in der Regel Angriffe mittels WET nicht immer verhindern. Der Angreifer hat in vielen Fällen gerade eine vom Nutzer als vertrauenswürdig eingestufte Website unter seine Kontrolle gebracht, sodass der Whitelisting-Ansatz hier völlig fehlschlägt.

c) Umgehen von weiteren Schutzmaßnahmen

Wie bereits aus den vorherigen Kapiteln zu entnehmen ist, ist ein 100 %iger Schutz vor Malware durch Virenscanner, Firewalls etc. praktisch nicht möglich. Gerade WETs werden häufig in weiteren Releases mit neuen Exploits aufgestockt, sodass Angreifer den Antivirenherstellern meist ein Stück voraus sind. Handelt es sich

bei diesen Schwachstellen auch noch um sogenannte *Zero Day-Schwachstellen*[192] (auch *O-Days* oder *0days* genannt), ist die Infektionsgefahr sehr groß. Bei O-Days handelt es sich um nicht veröffentlichte Schwachstellen, für die aus diesem Grund auch kein Patch existiert. In der Untergrundszene können sie dem Finder sowohl finanziellen als auch reputativen Vorteil bescheren.

Im Jahre 2007 machte eine Auktionsplattform für O-Days namens *WabiSabiLabi*[193] auf sich aufmerksam, die nach dem gleichen Prinzip wie eBay organisiert ist, nur dass dort eben keine herkömmlichen Waren, sondern nicht publizierte Exploits gehandelt werden. Nach kontroversen Diskussionen in der IT-Sicherheitsszene ist die Plattform mittlerweile wieder vom Netz genommen worden.

Da es sich bei der genannten Auktionsplattform eher um eine moralisch anfechtbare Lösung für die *Veröffentlichung* (engl.: Disclosure) *von Sicherheitslücken* handelt, wurde hierfür die Zero Day-Initiative (kurz: ZDI)[194] ins Leben gerufen. Diese Initiative setzt sich für die Belohnung von Sicherheitsforschern ein, die eine Schwachstelle auf eine verantwortungsvolle Art und Weise veröffentlicht haben. Denn oftmals ist die aggressive Haltung der betroffenen Unternehmen mit Schuld daran, dass Sicherheitslücken auf dem Schwarzmarkt gehandelt werden. Sicherheitsexperten, die Lücken in der Software kommerzieller Unternehmen finden und melden, haben oftmals Angst vor juristischen Schritten.

Es ist in der Szene nach wie vor umstritten, wie mit gefundenen Sicherheitslücken umgegangen werden soll. Einerseits ist man überzeugt, dass die betroffenen Unternehmen ein Recht auf das Wissen über eine Lücke haben. Andererseits will man sich nicht dem Risiko aussetzen, nach der Aufklärung über die Lücke mit einem Verfahren konfrontiert zu werden. Viele Sicherheitsforscher behalten O-Days daher für sich oder teilen das Wissen nur mit einem vertrauten Kreis.

3. Umgehung spezieller softwarebasierter Schutzmechanismen

Bei vielen Softwareprodukten, die den heutigen Markt bestimmen, handelt es sich um proprietäre Software bzw. Closed Source-Software. Der Begriff *Closed Source* (engl.: *geschlossene Quellen*) bezeichnet ein Paradigma für die Geschlossenheit von Quelltexten einer Software. Bekannte Beispiele für proprietäre Software sind Microsoft Windows, Adobe Flash Player, iTunes, Adobe Photoshop, Google Earth, Mac OS X, Skype und WinZip.

Die Vor- und Nachteile quelloffener und -geschlossener Software werden in der IT-Sicherheitsszene kontrovers diskutiert, wobei wirtschaftliche, gesellschaftliche und sicherheitstechnische Aspekte im Vordergrund stehen.

Die größte Gefahr liegt hierbei in der sogenannten Security-by-Obscurity-Methodik, die besagt, dass beispielsweise ein benutztes Verschlüsselungsverfahren

---

[192] http://de.wikipedia.org/wiki/Exploit.

[193] http://www.wslabi.com/wabisabilabi/home.do?.

[194] http://www.zerodayinitiative.com/.

einer proprietären Software zwar nicht veröffentlicht wurde, aber deshalb als sicher klassifiziert wird. Dies bedeutet somit gleichzeitig, dass die weltweite Gemeinschaft von Sicherheitsexperten das genutzte Verfahren nicht weiter untersuchen konnte und somit keinerlei objektive Aussage über die Sicherheit des Verfahrens gemacht werden kann. Oftmals erweist sich das Verfahren nach der Security-by-Obscurity-Methodik daher als falsch, denn das Kerckhoffsche Prinzip[195] besagt, dass die Sicherheit eines Verschlüsselungsverfahrens auf der Geheimhaltung des Schlüssels beruhen sollte und nicht auf der Geheimhaltung des Verschlüsselungsalgorithmus. Die Vergangenheit hat zudem gezeigt, dass nahezu jede proprietäre Software früher oder später Sicherheitslücken aufzeigt.

4. Umgehung spezieller hardwarebasierter Schutzmechanismen (Chipkarten, HSM)

Prognose: Der Einsatz von chipkartenbasierten Identifizierungssystemen (z. B. neuer Personalausweis) kann den Identitätsschutz im Internet zwar verbessern, die Gefährdung aber nicht grundsätzlich beseitigen, da diese Identifizierungssysteme mithilfe von Malware umgangen werden können.

Der Einsatz von Chipkarten zur Absicherung von Webanwendungen ist noch ein recht junges technisches Gebiet mit einer geringen Marktdurchdringung, daher liegen noch keine Daten zu realen Angriffen vor. Beispielhaft wurden solche Angriffe jedoch in den Publikationen von Hanno Langweg untersucht, zuletzt in den Jahren 2006 und 2007.

In [Langweg06] standen die Softwareprodukte im Vordergrund, die zur Erzeugung qualifizierter elektronischer Signaturen nach dem deutschen Signaturgesetz zugelassen waren. Eine vorangegangene Untersuchung im Jahre 2001 [CrSL01] hatte hier erhebliche Mängel zu Tage gefördert. Die wichtigste Kritik in dieser ersten Studie bezog sich auf die Tatsache, dass Klasse-1-Leser eingesetzt wurden und somit die PIN des Nutzers über einfache Keylogger mitgelesen werden konnte.

Fünf Jahre später standen diese Produkte erneut auf dem Prüfstand, diesmal aber durchwegs mit Klasse-2-Lesern ausgestattet. Die Eingabe der PIN am Kartenleser genügte indes nicht, die Produkte sicherer zu machen: Mit bekannten Angriffsmethodiken war Herr Langweg in der Lage, Dokumente signieren zu lassen, ohne dass der legitime Nutzer diese jemals zu Gesicht bekommen, geschweige denn bewusst signiert hat. Ähnliche Ergebnisse lieferte [LS07]. Keine der untersuchten HBCI-Implementierungen hielt den Standardangriffen stand.

Es ist hier zu betonen, dass die von Herrn Langweg eingesetzten Angriffstechniken nicht speziell für chipkartenbasierte Systeme entwickelt wurden, sondern zum Stand der Technik in der White-Hat-Community gehörten. Dies impliziert, dass diese Angriffe leicht mit geringen Kosten von der Black-Hat-Community in Angriffstools integriert werden können. Da dies arbeitsteilig geschieht, ist hier von

[195] http://de.wikipedia.org/wiki/Kerckhoffs%E2%80%99_Prinzip.

einem Angreifer auszugehen, der die geringsten Fähigkeiten laut Common Criteria aufweisen muss.

Als Fazit ist hier festzuhalten, dass chipkartenbasierte Systeme in Zukunft nur dann eine hohe Sicherheit bieten können, wenn Chipkarte und Kartenleser fest in die Gesamtapplikation integriert werden. Ein Beispiel hierfür bietet [GLS07].

5. Umgehung von Sicherheitsmechanismen auf Serverseite

Zu den Hauptzielen von Angreifern gehören nach wie vor auch Serversysteme, denn die Kompromittierung von häufig besuchten Websites ist eine gern benutzte Möglichkeit, um beispielsweise Masseninfektionen von Clients durch Web Exploit Toolkits (kurz: WETs) auszuführen. Meist sind hierbei nicht die Informationen auf dem Server selbst das Ziel, sondern langfristig gesehen lediglich die Infektion von Clientsystemen.

Problematisch für Serverbetreiber ist hierbei die Tatsache, dass die Absicherung der Server nicht einfach ist. Denn der Angreifer verfügt über einen entscheidenden Vorteil: Er muss lediglich eine Schwachstelle im System ausnutzen, wohingegen der Verteidiger das System gegen sämtliche Gefährdungen absichern muss. Selbst wenn das System soweit gegen bekannte Angriffe abgesichert werden konnte, besteht die Gefahr durch sogenannte Zero Day-Exploits[196], für die noch kein Sicherheitspatch zur Verfügung steht.

Hinzu kommt, dass sich die Art der Angriffe auf Serversysteme in den letzten Jahren gewandelt hat. Früher spielten sich die meisten Angriffe auf TCP/IP-Ebene des ISO/OSI-Modells ab. Dies hat sich mit der steigenden Popularität von Webanwendungen geändert. Viele Massenkompromittierungen wurden in der Vergangenheit durch Schwachstellen in Webanwendungen ausgeführt – und dieser Trend hält an. Daher ist es nicht verwunderlich, dass Angriffe auf Webanwendungen die TOP-10 der Sicherheitsschwachstellen anführen[197].

Zu den beiden populärsten und weitverbreitetsten Schwachstellen für Webserver zählen die *SQL-Injektion* und die Remote File Inclusion-Schwachstelle. Auf beide Schwachstellen wird im Folgenden genauer eingegangen.

a) SQL-Injektion-Schwachstelle

Im Jahre 2008 waren ungefähr 20 % von 5.600 Schwachstellen, die in die National Vulnerability Database des NIST[198] eingetragen wurden, SQL-Injektionen. SQL-Injektionen erfreuen sich einer großen Beliebtheit unter Angreifern, da sie weitverbreitet sind und sehr einfach automatisiert ausgenutzt werden können.

---

[196] http://de.wikipedia.org/wiki/Exploit.

[197] http://www.sans.org/top20/.

[198] http://nvd.nist.gov.

Ein weiterer Punkt ist, dass sich Entwickler von Webanwendungen nicht bewusst sind, wie Angreifer die SQL-Abfragen von serverseitigen Skripten verändern und für ihre bösartigen Absichten missbrauchen können. SQL-Injektionen existieren aufgrund *mangelnder Überprüfung* (engl.: Insufficient Input Validation) von Benutzereingaben. Der Angreifer versucht dabei, über die Anwendung, die den Zugriff auf die Datenbank bereitstellt, eigene Datenbankbefehle einzuschleusen. Sein Ziel ist es, Daten in seinem Sinne zu verändern oder Kontrolle über den Server zu erhalten. Mit SQL-Injektionen können Daten erstellt, gelesen, verändert oder gelöscht werden. Auch das Ausführen von Systemkommandos kann durch eine SQL-Injektion ermöglicht werden.

In der Regel werden SQL-Injektion-Schwachstellen mit zwei Zielsetzungen ausgenutzt:

- Der Angreifer erhofft sich durch die Ausnutzung einer SQL-Injektion den Zugriff auf sensible Datensätze wie Kontodaten oder Passwörter. In diesem Fall wird der Webserver nicht automatisiert per Skript, sondern meist individuell attackiert. Im Frühjahr 2009 stand eine rumänische Crackergruppe in den Schlagzeilen, die gezielt versuchte, Websites von Sicherheitsfirmen wie Kasperky, Symantec, F-Secure und BitDefender anzugreifen.[199]
- Durch automatisierte Skripte und Programme werden SQL-Injektion-Schwachstellen in populären Webanwendungen wie Typo 3, Joomla und Wordpress im großen Stil ausgenutzt. Durch die SQL-Injektion werden anschließend Einträge in der Datenbank geändert, um beispielsweise auf der Indexseite Referenzen auf bösartige Websites anzulegen. Beim Besuch einer infizierten Indexseite lädt der Browser die bösartige Referenz nach. Über diesen Weg wird dann versucht, das Clientsystem z. B. mit einem Trojaner zu infizieren (siehe hierfür auch I.2.a)).

Bei den Untersuchungen von großen SQL-Injektion-Kampagnen wird immer wieder China als Ursprung der Angriffe genannt. Konkrete Ursachen dafür sind nicht bekannt. Allerdings wird vermutet, dass die fortgeschrittene Entwicklung komfortabler Scanningtools einer der Gründe sein könnte.[200]

b) RFI-Schwachstelle

SQL-Injektion-Schwachstellen sind zwar eine recht weitverbreitete Möglichkeit für die Kompromittierung von Servern, allerdings nicht die einzige. Eine weitere, sehr beliebte Methode der Serverkompromittierung ist in letzter Zeit die sogenannte Remote File Inclusion (kurz: RFI). Diese Schwachstellenart resultiert aus fehlerhaft geschriebenen Webserverskripten (meist auf Basis der Programmiersprache PHP[201]), die einen häufig per HTTP-GET-Parameter übergebenen Dateipfad

---

[199] http://www.findmysoft.com/news/Romanian-Hackers-Expose-Kaspersky-USA-Site-Open-to-SQL-Injection/.

[200] http://securitywatch.eweek.com/sql_injection/china_flooding_web_with_sql_injection_attacks.html.

[201] http://php.net.

interpretieren. Dieser Dateipfad muss nicht auf dem eigenen, sondern kann auch auf anderen Servern liegen. Für eine erfolgreiche Kompromittierung muss lediglich ein verletzliches Serverskript gefunden und der Dateipfad direkt in die URL mit eingebaut werden. Eine solche URL könnte wie folgt aussehen:

http://vuln-site/setup.php?ConfigPath=http://server/shell.txt.

Bei der eingebundenen Datei handelt es sich meist um ein Skript in txt-Form. Das Skript, in diesem Fall setup.php, lädt dann je nach Vorhaben ein bösartiges Skript nach, installiert eine PHP-Shell oder infiziert alle Indexdateien auf dem Server mit einer *IFrame*[202]-Referenz auf eine bösartige Seite. RFI-Schwachstellen sind als sehr kritisch einzuordnen, da sich ein Angreifer sehr leicht weitere Rechte auf dem System verschaffen kann. Solche Schwachstellen können unter anderem durch einen sicheren Quellcode und die detaillierte Überprüfung von Nutzereingaben verhindert werden.

RFI-Schwachstellen für populäre Portalskripte, Foren und Weblogs lassen sich leicht über präzise Suchanfragen in Suchmaschinen finden. Diese Methode ist als *Google Dorking*[203] bekannt und ermöglicht eine breite Suche nach verwundbaren Servern. Es gibt zudem bereits fertige Skripte und Programme, die eigenständig nach RFI-verwundbaren Skripten im Netz suchen.

Ein weiteres Problem, das unter anderem durch RFI-Schwachstellen auftreten kann, sind Massen-ARP-Poisoning-Angriffe[204] in Serverumgebungen. Durch eine RFI-Schwachstelle kann sich ein Angreifer Zugriff zu einem Server verschaffen. Dort startet er dann ARP-Poisoning-Angriffe auf weitere Server im Netzwerk und leitet sämtlichen Verkehr über das gekaperte System. Trickreich an diesem Angriff ist die Tatsache, dass der Angreifer nicht versucht, andere Server zu kompromittieren, sondern lediglich sämtlichen HTTP-Verkehr zum Client mit bösartigem JavaScript infiziert. Der Angreifer versucht somit, das System des Opfers zu infizieren. Die Untersuchung dieses Angriffs gestaltet sich sehr schwierig, da aufseiten des Servers die eigentlichen HTML-Seiten nicht infiziert werden, sondern nur der Netzwerkverkehr. Der Angriff kann aufseiten der Betreiber allerdings durch Abschottung der Server auf Netzebene, ständige Beobachtung des ARP-Verkehrs und einen statischen *ARP-Cache*[205] unterbunden werden.

c) Weitere Angriffsmöglichkeiten

Zwar stellen Sicherheitslücken in Webanwendungen eine der bedeutendsten Gefahren für Server dar, andererseits sind sie aber nur eine Möglichkeit des Einbruchs. Im Frühjahr 2008 wurde eine Schwachstelle in der *OpenSSL*[206]-Bibliothek des

[202] http://de.selfhtml.org/html/referenz/elemente.htm#iframe.

[203] http://www.wecon.net/files/37/AI3_2009-SH.pdf.

[204] http://de.wikipedia.org/wiki/ARP-Spoofing.

[205] http://de.wikipedia.org/wiki/Address_Resolution_Protocol.

[206] http://www.openssl.org.

Betriebssystems *Debian*[207] bekannt. Das Debian-Sicherheitsteam entdeckte, dass der Zufallszahlengenerator von OpenSSL in Debian und seinen Derivaten eine Schwäche enthält. Als ein Ergebnis dieser Schwäche waren bestimmte kryptografische Schlüssel sehr viel einfacher zu berechnen, als sie es sein sollten, sodass ein Angreifer den Schlüssel mittels einer Brute-Force-Attacke mit minimalem Wissen über das System erraten konnte. Dies betraf insbesondere die Verwendung von Verschlüsselungsschlüsseln in OpenSSH, OpenVPN und SSL-Zertifikaten. Aber auch die Schlüssel für DNSSEC und X.509-Zertifikate waren betroffen. Da Schlüssel auf Debian-Systemen erstellt und anschließend auf andere Systeme portiert werden, waren auch weitere Systeme und deren Anwendungen anfällig.

Bei Massenkompromittierungen von Servern spielen immer wieder FTP-Accounts eine entscheidende Rolle. So gab es beispielsweise Masseninfektionen von Dateien auf Webservern mit bösartigen IFrames, die durch gestohlene FTP-Zugangsdaten möglich wurden. Diese Accounts wurden durch Schadsoftware auf infizierten Clientsystemen gesammelt und an den Angreifer übertragen. Dieser konnte die Zugänge mit automatisierten Programmen testen und – im Falle der Validität – automatisch Dateien auf dem Webserver infizieren, um entweder ein Trojanisches Pferd zu verbreiten oder Seiten auf dem Webserver zu infizieren.

Im September 2009 wurde bekannt, dass eine Reihe von Linux-Webservern in ein Bot-Netz eingegliedert wurden, welches Malware an Clients ausgibt[208]. Neben dem Standardwebserver Apache wurde ein zweiter Webserver namens *nginx*[209] installiert, der die Malware ausliefert. Bei den kompromittierten Systemen soll es sich anscheinend um eines der ersten Server-Bot-Netze handeln. Zur weiteren Verschleierung wurde dieses Bot-Netz mit einem Bot-Netz verknüpft, das lediglich aus Clientsystemen besteht.

6. *Netzwerkbasierte Angriffe*

Prognose: Netzwerkbasierte Angriffe werden an Bedeutung gewinnen. Dies liegt zum einen daran, dass wichtige Netzwerkdienste wie das Domain Name System (DNS) an die Grenzen ihrer Sicherheit gelangt sind. Zum anderen werden immer mehr Daten und Anwendungen „ins Netz" verlagert (Stichworte: Web 2.0, Cloud Computing), sodass ein Angriff auf den lokalen Computer nur noch dazu dient, Zugriff auf die im Netz gespeicherten Daten zu erhalten.

Das Internet und insbesondere das World Wide Web sind zu einer kritischen Ressource geworden, ohne dass dies den Entscheidungsträgern bewusst geworden ist:

- Immer mehr klassische Kommunikationsdienste werden über das Internet abgewickelt. Die rasante Abwärtsentwicklung der Preise im Telekommunikations-

[207] http://www.de.debian.org.

[208] http://www.theregister.co.uk/2009/09/12/linux_zombies_push_malware/.

[209] http://nginx.net.

bereich ist nicht zuletzt auf den verstärkten Einsatz von Voice-over-IP (VoIP) zurückzuführen. Somit kann fast jedes Telefonat heute durch einen Angriff auf das Internet mitgehört werden, und Notrufnummern können durch internetbasierte Denial-of-Service-Angriffe lahmgelegt werden. Mit ENUM ist sogar ein Verfahren standardisiert worden, bei dem Telefonnummern bequem im Domain Name System (DNS) hinterlegt werden können. Ebenso bequem können aber dadurch auch Telefonate durch Angriffe auf das DNS [Kaminski08] abgehört werden.

- Durch das Auftreten eines neuen, besonders aggressiven Internetwurms (Conficker) wurden ganze Truppenteile der Bundeswehr und der französischen Luftwaffe lahmgelegt.
- In neuen Informatikparadigmen (Webservices, serviceorientierte Architekturen, Cloud Computing) spielen die Unique Ressource Identifier (URL) aus dem WWW eine Schlüsselrolle: Über sie (und nicht etwa über IP-Adressen oder ähnliche Adressierungsmechanismen) werden Softwarekomponenten für komplexe Anwendungen angesprochen oder geladen. URLs beinhalten als wichtigsten Bestandteil den Domainnamen, und zur Auflösung dieses Domainnamens hin zur eigentlichen IP-Adresse spielt das Domain Name System die Rolle einer kritischen Infrastruktur.
- Auf der Ebene unterhalb des DNS sichern Routing-Protokolle den Transport der Daten ab. Der YouTube-Pakistan-Zwischenfall[210] und ein aktueller Vorfall bei einem chinesischen Provider[211] haben allerdings gezeigt, wie unsicher Routing-Protokolle in der Realität sind. Dies wurde von einer aktuellen Studie untermauert [BFZ07].

Netzwerkbasierte Angriffe haben gegenüber Malwareangriffen den Vorteil, dass sie besser skalieren: Man kann viele Opfer gleichzeitig mit den gleichen Methoden attackieren.

Sicherheit des Domain Name System

In einigen Bereichen ist die Unsicherheit des DNS offenkundig: Z. B. wird im Bereich WLAN DNS-Spoofing ganz legal eingesetzt, wenn der Internetzugang über ein offenes WLAN durch eine Passworteingabe in einem HTML-Formular abgesichert wird. In diesem Fall wird jede Anfrage nach einem beliebigen Domainnamen auf den Log-in-Server des Anbieters umgeleitet (legales Pharming!).

Den Sargnagel zur Sicherheit des DNS hat Dan Kaminski im Jahr 2008 auf der Black-Hat-Konferenz eingeschlagen [Kaminski08]: Er kombinierte auf clevere Weise einen Geburtstagsparadoxon-Angriff auf die 16 Bit lange Zufallszahl, die zur Identifizierung legaler Antworten auf eine DNS-Anfrage dient (Anfrage und Antwort müssen die gleiche 16-Bit-Zufallszahl enthalten), mit sogenannten Additional Records. Dadurch wurde es möglich, einen DNS-Cache innerhalb von nur

[210] http://www.heise.de/meldung/Routing-Kleinkrieg-Ursache-fuer-YouTube-Ausfall-205345.html.

[211] http://www.heise.de/meldung/Chinesischer-Provider-entfuehrt-kurzzeitig-Teile-des-Internets-975137.html.

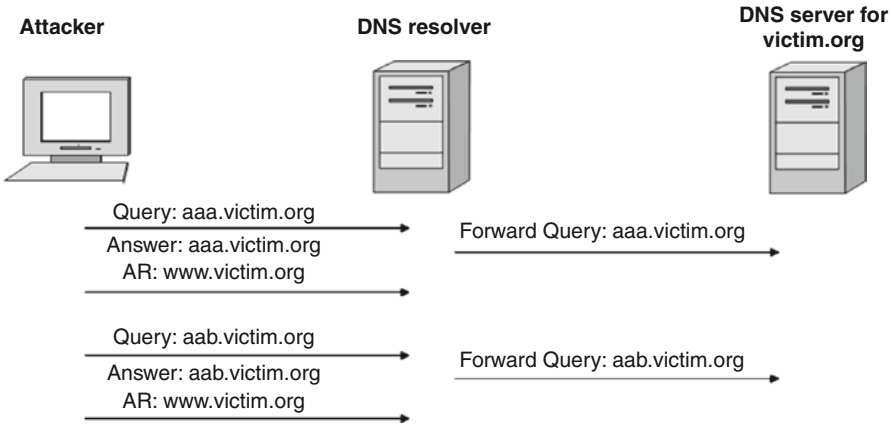


Abb. 22 Der Kaminski-Angriff auf das Domain Name System

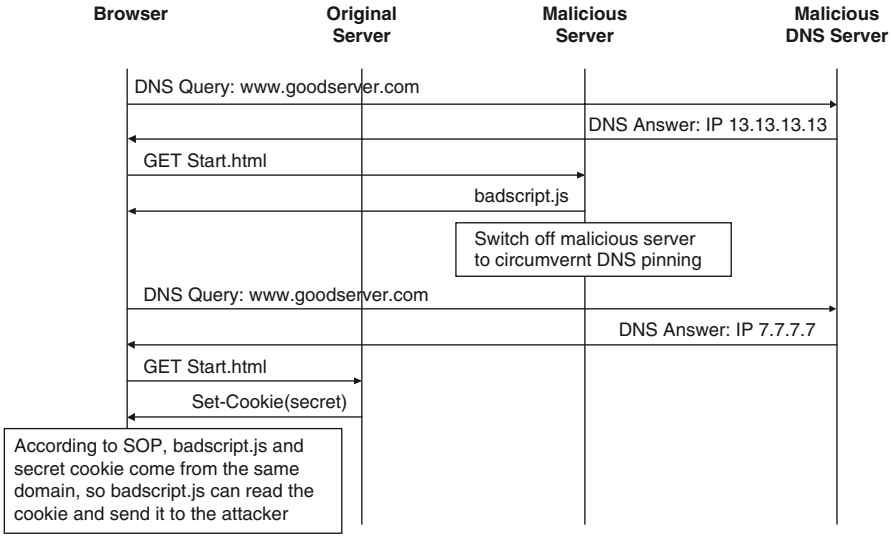


Abb. 23 Dynamic Pharming-Angriff zum Diebstahl von Identitätsdaten

10 s zu „vergiften" („DNS Cache Poisoning"). Dieser Angriff wurde lange Zeit vor seiner Veröffentlichung den Herstellern von DNS-Software mitgeteilt, woraufhin eine Ad-Hoc-Lösung integriert wurde. Diese Ad-Hoc-Lösung war mit DNS allein allerdings nicht zu erreichen, es musste eine Protokollgrenze überschritten und der UDP Source Port mit in diese Lösung einbezogen werden. Damit ist offensichtlich geworden, dass die Sicherheit des DNS an eine harte Grenze gestoßen ist, die nicht überschritten werden kann.

Weitere Angriffe auf das Domain Name System wurden in der wissenschaftlichen Literatur vorgestellt und gegen komplexe, kryptografisch abgesicherte Sys-

teme eingesetzt. So wurde z. B. mithilfe eines Dynamic Pharming-Angriffs das Cardspace-System von Microsoft gebrochen [GSSX09].

Für zukünftige Webanwendungen darf daher die DNS-Information allein nur als Hilfsinformation verwendet werden und nicht die Basis von Sicherheitsentscheidungen darstellen. Dies ist allerdings heute noch nicht der Fall.

DNS Security (DNSSEC[212] [RFC 4033], [RFC 4034], [RFC 4035]) könnte mit großem kryptografischen Aufwand einige dieser Probleme lösen. Allerdings ist unklar, ob DNSSEC überhaupt im großen Stil funktioniert, da Feldversuche fehlen, die den enormen organisatorischen Aufwand widerspiegeln. Bei diesen Feldversuchen muss es weniger um die schiere Masse der digitalen Signaturen, die erstellt werden müssen, als vielmehr um die Zertifizierung von Public Keys in der größten und strengsten, jemals geplanten Public-Key-Infrastruktur gehen.

Informationen zu aktuellen Feldversuchen sind im Internet verfügbar[213]. Es bleibt aber offen, ob DNSSEC alle Probleme lösen kann: Man kann zwar verhindern, dass fremde Domainnamen vom Angreifer an die eigene IP-Adresse gebunden werden, aber nicht, dass fremde IP-Adressen mit einer Domain des Angreifers verknüpft werden. Außerdem sind Angriffe auf die dem Internet zugrunde liegenden Routing-Protokolle weiterhin möglich.

Sicherheit von Routing-Algorithmen

Das Internet besteht aus sogenannten *Autonomen Systemen* (kurz: *AS*), die über das Border Gateway Protocol (kurz: BGP) untereinander den Transport von IP-Paketen aushandeln. Innerhalb der AS werden Routing-Protokolle eingesetzt, die das vorhandene Netz möglichst effizient nutzen sollen. Das BGP-Routing-Protokoll basiert dagegen überwiegend auf Regeln, die die Geschäftsbeziehungen der einzelnen AS zueinander widerspiegeln.

Über die (Un-)Sicherheit von Routing-Protokollen ist viel publiziert worden, eine praxistaugliche Standardisierungsinitiative wie bei DNSSEC ist aber nicht vorhanden.

Routing-Protokolle stehen nicht im Fokus aktueller Angriffe, einzelne Vorfälle sind aber dokumentiert. Im Februar 2008 wurde die Pakistanische Telekom (AS 36561) von ihrer Regierung angewiesen, eine Besonderheit des BGP-Protokolls auszunutzen und den Internetverkehr zu den IP-Adressen von YouTube umzuleiten [Brown08]. Pakistanischen Internetnutzern sollte es so unmöglich gemacht werden, die Seiten von YouTube aufzurufen. Die so absichtlich geänderten Routing-Tabellen wurden dann von den pakistanischen Routern über das BGP-Protokoll weitergeleitet, und viele andere AS leiteten ihre IP-Pakete, die für YouTube bestimmt waren, nach Pakistan um. Dieser Vorfall dauerte 2 h und 13 min und war unkritisch, da nur ein Unterhaltungsdienst betroffen war. Wäre der Adressbereich einer im Internet agierenden Firma, z. B. eines Börsenbrokers, betroffen gewesen, hätten die Folgen kommerziell verheerend ausfallen können, ganz zu schweigen von kritischen Infrastrukturen wie dem Stromnetz.

---

[212] http://www.dnssec.net/.

[213] http://www.dnssec.net/projects.

Ein weiterer Vorfall wurde auch China gemeldet.[214] Im Bereich der Sicherheit von Routing-Protokollen besteht erheblicher Forschungs- und Entwicklungsbedarf, insbesondere bei der Entwicklung von Schlüsselvereinbarungsmechanismen, die eine manuelle Konfiguration von Schlüsseln ersetzen.

Sicherheit von SSL

Die Sicherheit der meisten Webapplikationen beruht heute auf dem naiven Einsatz von SSL [TLS1.0]: Der Server sendet während des SSL-Handshakes ein Zertifikat an den Browser, das von einer der vielen Zertifizierungsstellen weltweit zu einem Preis zwischen 50 USD und einigen 10.000 USD (EV-Zertifikate[215]) ausgegeben wird. Der Browser prüft dieses Zertifikat zunächst gegen die in seinem Zertifikatsspeicher abgelegten Wurzelzertifikate. Ist diese Prüfung erfolglos oder stimmt der Domainname im Zertifikat nicht mit dem im Browser angezeigten Domainnamen überein, so wird der Nutzer um eine Entscheidung gebeten. Moderne Browser legen hier nahe, den Verbindungsaufbau abzubrechen. Trotzdem ist, wie eine Studie [DTH06] gezeigt hat, davon auszugehen, dass der durchschnittliche Internetnutzer nicht in der Lage ist, diese Entscheidung korrekt zu treffen. *Dies liegt vor allem in der Tatsache begründet, dass es visuell fast keinen Unterschied zwischen einer korrekt mit SSL geschützten Seite und einer vollständig ungeschützten Seite gibt: Gewarnt wird nur, wenn SSL auch eingesetzt wird.*

Erschwerend kommt hinzu, dass Browser immer noch Zertifikate akzeptieren, die unter Verwendung der Hash-Funktion MD5 erstellt wurden. In einem sorgfältig ausgearbeiteten Versuch haben holländische Forscher nachgewiesen, dass es möglich ist, durch Ausnutzung der ungenügenden Kollisionsresistenz von MD5 ein illegales CA-Zertifikat zu berechnen, mit dem es dann möglich ist, beliebige und von jedem Browser als gültig akzeptierte SSL-Zertifikate auszustellen [SSALMOdW08]. Zitat:

> In combination with known weaknesses in the Domain Name System (DNS) protocol such as Kaminsky's „DNS Flaw"…(see also…), the vulnerability we exposed opens the door to virtually undetectable phishing attacks. Without being aware of it, users can be redirected to malicious sites that appear exactly the same as the trusted banking or e-commerce websites they believe to be visiting. User passwords and other private data can fall into wrong hands.

Da alle genannten Angriffe die Sicherheit des SSL-Protokolls selbst nicht betreffen, wurden Vorschläge für eine verbesserte Nutzung von SSL gemacht. Dies betrifft die heute schon mögliche Nutzung von Clientzertifikaten [GJMS08] auf der einen Seite, die Entwicklung einer neuen Same Origin Policy für den Browser [KSTW07; MS07; GLS08] auf der anderen Seite. Auch hier besteht erheblicher Forschungsbedarf.

---

[214] http://www.heise.de/meldung/Chinesischer-Provider-entfuehrt-kurzzeitig-Teile-des-Internets-975137.html.

[215] http://en.wikipedia.org/wiki/Extended_Validation_Certificate.

7. Klassische Malware: neue Trends

Prognose: Malware wird in naher Zukunft das Problem Nummer 1 bleiben, da die Möglichkeiten der Angreifer hier noch lange nicht ausgeschöpft sind. Wegen der Anstrengungen von Microsoft zur Verbesserung der Sicherheit von Betriebssystemen wird sich der Fokus auf weitverbreitete Anwendungen wie Adobe Acrobat und Browser verlagern.

Drive-by-Downloads: Browser als Angriffswerkzeuge

Als markantes Beispiel dafür, wie schnell und effizient die Black-Hat-Community auf neue Sicherheitsmechanismen reagieren kann, müssen hier Drive-by-Downloads genannt werden.

Während vor einigen Jahren noch das Ausnutzen von Schwachstellen in Netzwerkdiensten auf PCs im Fokus der Angreifer stand, hat sich die Situation durch die Integration von Personal Firewalls in die Betriebssysteme und AV-Software grundlegend verändert. Angreifer konnten nun nicht mehr auf die Netzwerkdienste zugreifen, da diese Zugriffe durch die Firewall geblockt wurden.

Einige wenige Ports mussten aber geöffnet bleiben, um Standarddienste wie HTTP und E-Mail zu ermöglichen. Hier bot sich der Browser als Angriffsziel an, da neben einer zunehmend komplexer werdenden Browserengine auch noch zahlreiche, weitverbreitete Plug-ins als Angriffsziel dienen können. Mit JavaScript steht eine mächtige Skriptsprache zur Verfügung, um all diese Komponenten nach einem Download des entsprechenden Skripts anzusprechen. Und die Wege und Möglichkeiten, solche Skripte einzuschleusen, sind vielfältig.

In einer grundlegenden quantitativen Studie [Google08] haben Forscher der Firma Google die aktuelle Situation im Jahr 2008 untersucht. Sie konnten dabei auf die umfangreiche Datenbasis von Google zurückgreifen. Zitat:

> As the web continues to play an ever increasing role in information exchange, so too is it becoming the prevailing platform for infecting vulnerable hosts. In this paper, we provide a detailed study of the pervasiveness of so-called drive-by downloads on the Internet. Drive-by downloads are caused by URLs that attempt to exploit their visitors and cause malware to be installed and run automatically. Our analysis of billions of URLs over a 10 month period shows that a non-trivial amount, of over 3 Mio. malicious URLs, initiate drive-by downloads. An even more troubling finding is that approximately 1.3 % of the incoming search queries to Google's search engine returned at least one URL labeled as malicious in the results page. We also explore several aspects of the drive-by downloads problem. We study the relationship between the user browsing habits and exposure to malware, the different techniques used to lure the user into the malware distribution networks, and the different properties of these networks.

Bei einem Drive-by-Download wird die Malware über das HTTP-Protokoll an den Browser geschickt. Die Malware ist dabei modular aufgebaut: Meist wird nur ein kleiner Loader über eine im Browser oder in einem der Plug-ins vorhandene Schwachstelle eingeschleust. Dieser kontaktiert dann (wiederum über HTTP) einen Server des Angreifers und lädt Instruktionen und die benötigten Schadmodule nach.

Neben relativ neuen, noch kaum gepatchten Schwachstellen wird den Loadern teilweise auch ein ganzes Bündel alter Schwachstellen mit auf den Weg gegeben: Dies ist für den Angreifer billiger, da diese alten Schwachstellen bereits bekannt sind und der Exploitcode für sie frei verfügbar ist. Da es kein zentrales Update-Management für alle Softwarekomponenten eines PCs oder Browsers gibt, ist die Wahrscheinlichkeit sehr hoch, dass das Betriebssystem, der Browser oder ein Plug-in dieses Browsers noch einen veralteten Softwarestand mit diesen alten Schwachstellen besitzt. (Hier bietet sich als zentrale Dienstleistung des BSI für Bürger die Einrichtung einer Website an, die den Browser und alle geladenen Plug-ins auf ihre Aktualität hin prüft und Empfehlungen zu verfügbaren Updates gibt.)

Die Wege zum Einschleusen des Loader-Moduls in einen Browser sind vielfältig: Ein solches Modul kann auf einer nicht vertrauenswürdigen Website versteckt sein, wobei [Google08] aufgezeigt hat, dass es hier keine besonders gefährlichen Bereiche („Rotlichtbezirk“) im Internet gibt, sondern dass die Infektionsgefahr relativ gleichmäßig verteilt ist. Diese Tatsache ist auch dadurch zu erklären, dass immer wieder reguläre, vertrauenswürdige Websites mit Malware verseucht werden[216] und dass sogar die Internetwerbung zur Verbreitung von Schadsoftware genutzt wird [Google08].

Neben seiner Rolle als Einfallstor wird der Browser auch zur Umgehung von Sicherheitsmechanismen des Betriebssystems genutzt: Da die genaue Platzierung von Schadcode im Stack oder Heap des Betriebssystems in Windows Vista aufgrund der ASLR-Technologie immer schwieriger wurde, wurde von der White-Hat-Community die Technik des Heap Spraying[217] entwickelt: Mithilfe von JavaScript wird nicht nur eine, sondern werden gleich viele Versionen des Schadcodes an verschiedene Stellen des Heaps geschrieben. Die Wahrscheinlichkeit, diesen Schadcode dann auch ansprechen zu können, steigt entsprechend.

Malware schützt sich gegen AV-Signaturen

Da Malware nach dem Erfolg der Phishingangriffe zunehmend kommerziell motiviert erstellt wird, ist es nicht mehr wichtig, durch eine große Verbreitung Ruhm in der White-Hat-Community zu ernten, sondern möglichst unentdeckt zu agieren.

Um dieses Ziel zu erreichen, sind die Ersteller von Malware dazu übergegangen, ein in einer Hochsprache (C, C++) entwickeltes Schadprogramm immer wieder mit unterschiedlichen Optionen neu zu kompilieren. Diese Binärprogramme sind von ihrer Funktionalität her gleichwertig, werden aber als unterschiedliche Bytefolgen übertragen. Da die AV-Signaturen, die heute von allen AV-Firmen in unterschiedlicher Ausprägung verwendet werden, im Wesentlichen aus regulären Ausdrücken über diese Bytefolgen bestehen, benötigt man im Extremfall für jede kompilierte Variante eine neue AV-Signatur. Dies hat zur Folge, dass in jüngster Vergangenheit die Anzahl der benötigten Signaturen exponenziell gewachsen ist. Eine relativ

[216] http://securitylabs.websense.com/content/Alerts/3326.aspx.

[217] http://en.wikipedia.org/wiki/Heap_spraying.

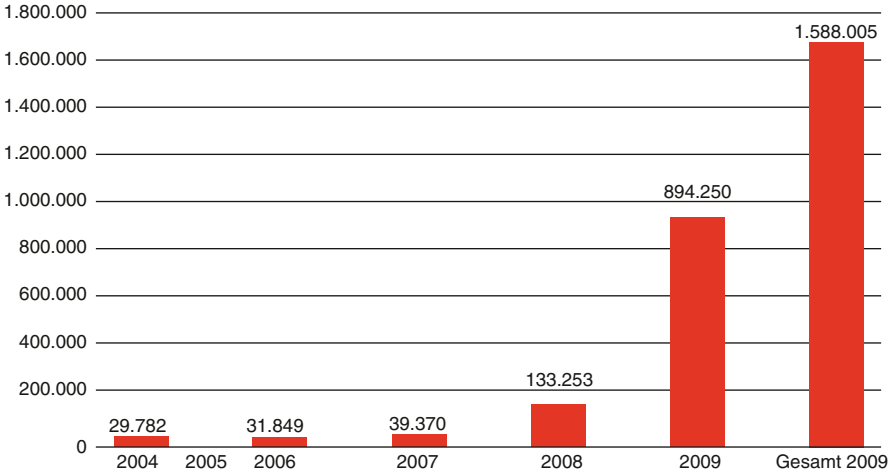


Abb. 24 Anzahl neuer Malwaresamples pro Jahr. (Quelle: GData)

anschauliche Begründung, warum diese Vorgehensweise zum Ziel führt, findet man in [GP09]. Zitat:

> Im ersten Halbjahr 2009 identifizierte GData 663.952 neue Schädlinge. Das sind mehr als doppelt so viele wie im gleichen Zeitraum ein Jahr zuvor. Gegenüber dem zweiten Halbjahr 2008 konnte lediglich eine leichte Steigerung um 15 % erzielt werden. Die Zahl der aktiven Malwarefamilien dagegen sank um 7 %.[218]

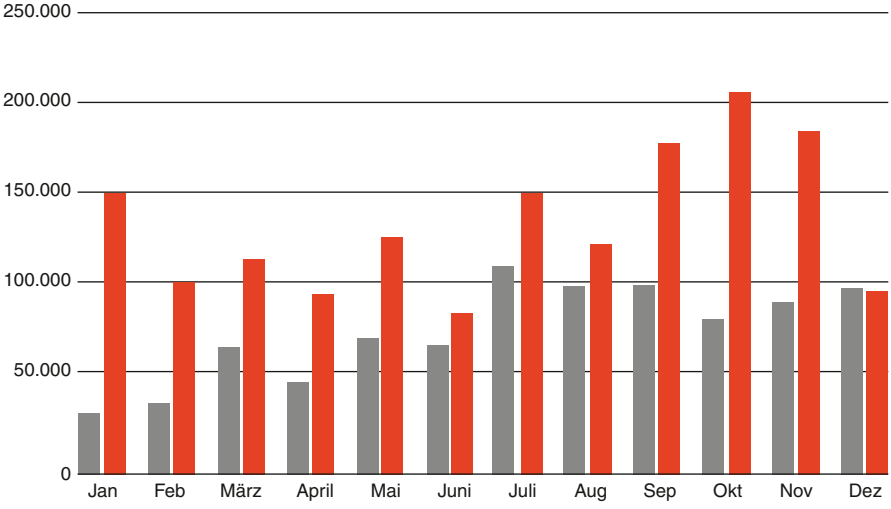


Abb. 25 Neue Malwarevarianten 2008 (*linker Balken*) und 2009 (*rechter Balken*). (Quelle: GData)

[218] http://www.gdata.de/virenforschung/info/whitepaper.html → MalwareReport 2009 1-6 DE 2 01.

Da absehbar ist, dass die benötigten AV-Signaturen nicht hinreichend schnell aktualisiert werden können, werden verschiedene Forschungsansätze verfolgt:

- Verhaltensbasierte Analyse: Hierbei wird das Verhalten einer unbekannten Software innerhalb einer „Sandbox" untersucht. Zeigen sich dabei Verhaltensmuster, die typisch für Schadsoftware sind (z. B. schreibender Zugriff auf die Windows Registry, kombiniert mit dem Nachladen von weiteren Softwarekomponenten aus dem Internet), so wird diese Software als schadhaft klassifiziert. Probleme bereiten bei diesem Ansatz die Vermeidung von Fehlalarmen („False Positives") und die Tatsache, dass die Malwareautoren als Gegenmaßnahme testen können, ob sie sich in einer „Sandbox" befinden, und daher unterschiedliche Verhaltensweisen für „in der Sandbox" und „außerhalb der Sandbox" in die Software mit einbetten. In Deutschland ist die Arbeitsgruppe um Felix Freiling an der Universität Mannheim hier mit führend.
- Strukturbasierte Analyse: Hierbei wird der gefundene Binärcode disassembliert und seine Struktur in einem Graphen auf zwei Ebenen dargestellt – auf der obersten Ebene als Callgraph, der das Hauptprogramm und seine verschiedenen Unterprogramme darstellt, und als Flussgraph, der den Kontrollfluss innerhalb jedes Unterprogramms darstellt. Durch Vergleich dieser Strukturen können unbekannte Programme sehr schnell bekannten Malwareklassen zugeordnet werden. False Positives können weitgehend ausgeschlossen werden. Das von der Firma Zynamics[219] entwickelte Verfahren ist sehr rechenaufwendig und eignet sich damit nicht zum Einsatz auf dem heimischen PC. Als „Nebenprodukt" ist hier aber eine Methodik entstanden, wie man AV-Signaturen für eine Vielzahl von äquivalenten Binärcodes entwickelt [Blichmann09].

Data Mining
Angriffe mit Malware sind so erfolgreich, dass die Angreifer mittlerweile vor dem Problem stehen, die Flut von erbeuteten Identitätsdaten manuell nicht mehr auswerten zu können. Daher wird entweder versucht, nur potenziell wertvolle Daten zu sammeln (z. B. durch Einsatz von Browser-Plug-ins, die gezielt nur Passwörter mitschneiden, als Ersatz für Keylogger), oder es werden Data Mining-Techniken eingesetzt, um die wertvollen Daten herauszufiltern. Hier bieten sich in Zukunft eine Fülle von Möglichkeiten für einen Datenabgleich an, z. B. soziale Netzwerke und die Datensammlungen von Suchmaschinen wie Google.

Arbeitsteilung
Die Arbeitsteilung in der Black-Hat-Community wird weiter zunehmen. Dieser Trend, der bereits bei der Vermarktung von Tools wie mPack oder IcePack zu beobachten war, ist sinnvoll, da die eingesetzten Technologien im Internet und die Endgeräte immer vielfältiger werden. Bei einer Einschätzung des Risikos nach Common Criteria ist daher die schwächste Angreiferkategorie anzusetzen.

JavaScript
Dass man allein mit JavaScript eindrucksvolle Angriffsszenarien realisieren kann, haben [Felten97] und [SRJ06] belegt. Weitere Angriffe sind zu erwarten.

[219] http://www.zynamics.com/.

Neue Forschungstrends in der White Hat-Community
Die White Hat-Community hat sowohl auf die Herausforderungen von Windows Vista und Trusted Computing als auch auf die zunehmende Differenzierung im Bereich der Endgeräte reagiert. Folgende Stichpunkte sollen hier als Beispiel für neue Trends benannt werden.

- Return Oriented Programming: Es gibt auf dem Markt Schutzmechanismen, die wirkungsvoll verhindern, dass ein ausführbarer Code in einen Prozess eingeschleust wird, der im Kernelmode des Betriebssystems läuft. Um diese Mechanismen zu umgehen, wurde die Technik des Return Oriented Programming entwickelt. Man benötigt zur Ausführung dieses Angriffs nur eine Schwachstelle, die es ermöglicht, beliebige Rücksprungadressen auf einen Stack zu schreiben. Diese Rücksprungadressen zeigen jeweils auf ein Codefragment einer Standardbibliothek kurz vor einer Returnanweisung. Diese Codefragmente werden durch die verschiedenen Rücksprungadressen auf dem Stack zu einem sinnvollen Schadprogramm zusammengebunden.[220]
- Heap Spraying[221]: Um neue Sicherheitstechniken wie Address Space Layout Randomization[222] (kurz: ASLR), die mit Microsofts Betriebssystem Windows Vista in den Massenmarkt eingeführt wurden, zu kompensieren, werden verstärkt Skriptsprachen wie JavaScript und ActionScript für Vorbereitungsangriffe eingesetzt.

Weitere Trends werden in den nachfolgenden Abschnitten beschrieben.

8. Social Engineering

Phishing ist heute sicherlich der bekannteste Angriff aus der Familie der Social Engineering-Angriffe, um Zugriff auf vertrauliche Dokumente zu erlangen. Allerdings liegen die ersten Identitätsdiebstähle mit Social Engineering-Methoden schon sehr viel länger zurück.

Grundsätzlich kann Social Engineering als das Erlangen vertraulicher Informationen durch Annäherung an Geheimnisträger mittels gesellschaftlicher oder gespielter Kontakte definiert werden. Das grundlegende Problem beim Social Engineering ist die Tatsache, dass Menschen manipulierbar und generell das schwächste Glied in einer Kette sind.[223] Egal ob in der IT oder irgendwo anders – der Mensch selbst ist der größte Risikofaktor, denn eine „Vertrauensbasis“ ist schnell aufgebaut bzw. vorgetäuscht, und so können Menschen relativ leicht manipuliert werden.

---

[220] http://en.wikipedia.org/wiki/Return-oriented_programming.

[221] http://en.wikipedia.org/wiki/Heap_spraying.

[222] http://en.wikipedia.org/wiki/Address_space_layout_randomization.

[223] http://www.britannica.com/bps/additionalcontent/18/28111529/The-Human-Element-The-Weakest-Link-in-Information-Security.

Identitätsdiebstahl mit Social Engineering-Methoden muss aber nicht zwangsweise im Internet stattfinden. Eine frühe Form des Social Engineering wurde in den 1980er-Jahren mit Phreaking[224] praktiziert. Hierbei riefen Phreaker unter anderem bei Telefongesellschaften an, gaben sich als Systemadministratoren aus und baten um neue Passwörter, mit denen sie schließlich kostenlose Modemverbindungen herstellten.

Mittlerweile gibt es im Internet Projekte, die versuchen, Social Engineering Awareness zu automatisieren und somit schnell und effizient Audits für z. B. die eigene Firma zu erstellen. Das Social Engineering Toolkit[225] kann im Rahmen von Metasploit ausgeführt werden und ermöglicht es, beispielsweise zu Awareness-Zwecken automatisiert Phishing-E-Mails zu verschicken.

Eine weitere, modernere Methode des Social Engineering ist das sogenannte Dumpster Diving.[226] Hierbei wird der Müll des Opfers durchwühlt und nach Hinweisen und Anhaltspunkten über das soziale Umfeld durchsucht. Diese können dann in einem darauf folgenden Anruf dazu verwendet werden, das Vertrauen des Opfers zu erschleichen. Manchmal ist dies jedoch gar nicht notwendig, wenn beispielsweise auf einem achtlos weggeworfenen Zettel bereits Log-in-Daten notiert sind.

Allerdings können auch im Bereich Social Engineering neue Methoden der Angreifer verzeichnet werden. Neugier, Angst und Gewinnsucht sind menschliche Eigenschaften, die Angreifer ausnutzen können. Diese psychologischen Tricks nehmen zu und sind auch eine Gefahr für erfahrene Surfer.

Scareware-, Rogueware- und Ransomware-Angriffe

Scareware[227] heißt eine neue Social-Engineering-Taktik, die darauf ausgelegt ist, den Benutzer zu verunsichern und zu verängstigen. Fällt das Opfer auf den Trick herein, so wird ihm häufig gegen Bezahlung eine Beseitigung der nicht vorhandenen Gefahr angeboten. In anderen Fällen soll das Opfer durch den Glauben an einen erfolgreichen Angriff zu Handlungen verleitet werden, welche den tatsächlichen Angriff erst ermöglichen. Scareware kann, muss aber nicht als eine Art Malware angesehen werden, da sich Scareware-Angriffe auch ohne eigentliche Schadprogramme durchführen lassen. Beispielsweise kann sich ein Nutzer lediglich durch den Besuch einer geschickt präparierten Website täuschen lassen.

Bei *Ransomware* handelt es sich definitiv um Malware, die sich auf dem System des Opfers einnistet. Dort versucht sie, den Nutzer zur Herausgabe von Geld zu zwingen, was etwa durch Verschlüsselung von Dateien, ganzen Festplatten oder durch Aussperren des Nutzers realisiert wird. Der Anwender erhält erst nach Zahlung eines bestimmten Betrags Zugriff auf seine Dateien.

---

[224] http://en.wikipedia.org/wiki/Phreaking.

[225] http://www.social-engineer.org/framework/Computer_Based_Social_Engineering_Tools:_Social_Engineer_Toolkit_%28SET%29.

[226] http://www.kuro5hin.org/story/2003/1/29/215523/088.

[227] http://www.stern.de/digital/computer/abzockmasche-scareware-der-betrug-mit-der-angst-649746.html.

a) Rogueware

Eine praktizierte Methode von Scareware-Angriffen ist die sogenannte *Rogueware*. Rogueware gibt vor, Schadsoftware wie beispielsweise einen Trojaner gefunden zu haben, und bietet gleichzeitig an, diesen kostenpflichtig zu löschen. Die tatsächliche Bedrohung durch die Schadsoftware hat aber niemals bestanden. Bekannte Beispiele für Rogueware sind Namen wie *AntiSpyCheck*, *Antivirus 2009*, *IE Defender* und *Pc-Antispyware*.

Das eigentliche Ziel von Rogueware ist es, den Nutzer beispielsweise mithilfe von Pop-Ups und visuellen Effekten, die einem tatsächlichen System sehr ähneln, so zu verunsichern, dass dieser einwilligt und bereit ist, Geld für die Entfernung der nichtexistenten Schadsoftware zu bezahlen.

Dem Angreifer stehen für die Realisierung verschiedene Möglichkeiten zur Verfügung, da gängige Betriebssysteme und Programme relativ viele Fehlermeldungen an den Nutzer zurückgeben. Dies machen sich manche Rogueware-Websites zunutze, indem sie beispielsweise Fehlermeldungen des Internetbrowsers Firefox nachahmen. Problematisch ist hier allerdings, dass diese Fehlermeldungen auch in Browsern wie dem Internet Explorer angezeigt werden, wie in der Abb. 26 zu sehen ist.

In der Abb. 27 ist eine Scareware-Website dargestellt, die eine fiktive Vireninfektion vortäuscht. Der Nutzer wird durch diese Warnung dazu verleitet, eine kostenpflichtige Software zur Entfernung herunterzuladen.

Oftmals werden Scareware- bzw. Rogueware-Kampagnen mit anderen Angriffsvektoren kombiniert, um die Effizienz des Angriffs zu erhöhen. So wurden Rogueware-Kampagnen in der Vergangenheit gerne mit *Blackhat-SEO-Methoden*

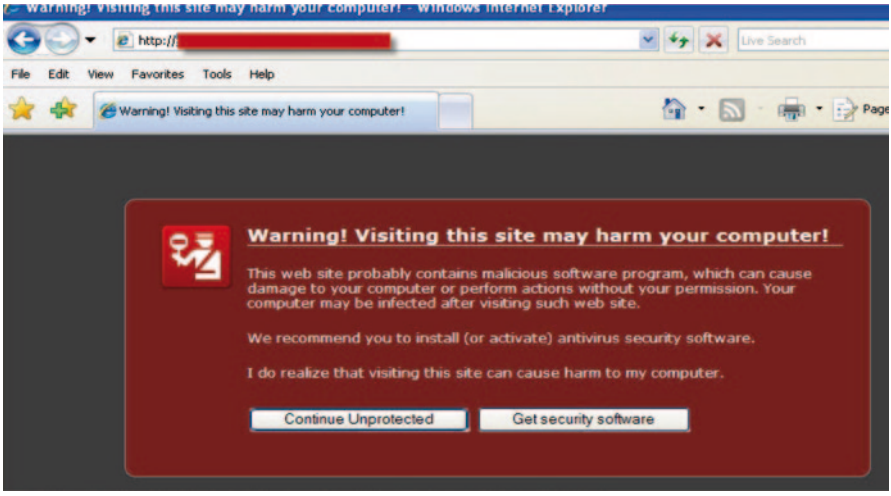


Abb. 26 Rogueware-Website nutzt Fehlermeldung des Mozilla Firefox. (Quelle: sunbeltsoftware.com)

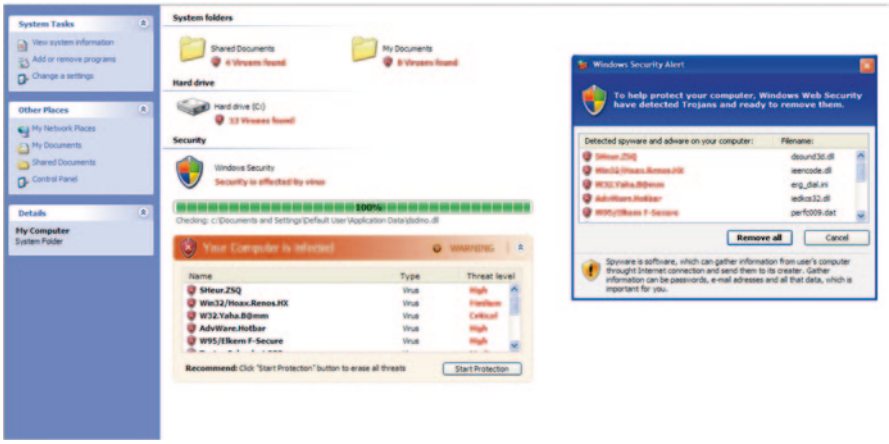


Abb. 27 Scareware-Website, die eine Vireninfektion vortäuscht. (Quelle: http://www.zdnet.com/blog/security/scareware-scammers-hijack-twitter-trending-topics/4389)

kombiniert.[228] Dies bedeutet, dass die Suchergebnisse nach beliebten Schlagwörtern von Angreifern bösartig modifiziert werden, um somit Websites von Rogueware-Kampagnen im Ranking der Suchmaschine weiter oben zu platzieren. Die Suchmaschinenergebnisse für beliebte Schlagwörter des aktuellen politischen Tagesgeschehens sind daher oftmals Opfer von Blackhat-SEO-Angriffen. Eine Liste von beispielhaften Schlagwörtern, die Ziel einer kürzlichen Blackhat-SEO-Kampagne auf Google waren, enthält unter anderem folgende Einträge:

> Obama Speech, GM group enterprises, Apple, Beatles, America, White House, Jon Gosselin, Live Interview, School Season, Swine Flu.

Laut dem Antivirenhersteller F-Secure[229] waren auch die Erdbeben in Samoa Ende September 2009 Thema einer Rogueware-SEO-Kampagne.

Rogueware verbreitet sich auch über Werbe-Adds auf großen Websites und Portalen. Dies ereignete sich im September 2009 auf den Websites der New York Times.[230] Besucher der Website NYTimes.com bekamen ein Pop-Up-Fenster angezeigt, das sie aufforderte, ein Antivirusprogramm zu installieren. Problem war in diesem Fall nicht der Inhalt der NYTimes.com-Website, sondern die Tatsache, dass Werbe-Adds von Dritten auf der Website eingebunden wurden. Diese Adds stammten von einem Werbenetzwerk, das anscheinend seine Anzeigen nicht gründlich genug überprüfte. So war es für Angreifer möglich, bösartigen Inhalt in den Publikationszyklus des Netzwerks einzuspeisen. Es wird in diesem Zusammenhang kontrovers diskutiert, wer die eigentliche Verantwortung für solch einen Vorfall übernimmt.

---

[228] http://blogs.zdnet.com/security/?p=3962.

[229] http://www.f-secure.com/weblog/archives/00001779.html.

[230] http://securitywatch.eweek.com/malware/nytimescom_users_hit_by_malicious_ad.html.

b) Ransomware

Rogueware versucht lediglich, den Nutzer zu täuschen. *Ransomware* hingegen zwingt den Nutzer, eine Tätigkeit durchzuführen.[231] Wie der Begriff schon sagt, handelt es sich bei dieser Malware um Schadcode, der den Nutzer dazu zwingt, etwa für Daten oder Informationen ein Lösegeld zu bezahlen. Ransomware tauchte in der jüngeren Vergangenheit auf unterschiedliche Art und Weise im Netz auf. Die eigentliche Motivation für diese Art von Malware war aber immer die finanzielle Bereicherung des Angreifers.

Im Gegensatz zu anderer Malware versucht Ransomware, dem Nutzer den Zugriff zu seinen Daten etwa durch Verschlüsselung oder Aussperren zu verweigern. Bekannte Namen von Ransomware sind *Trojan.Ransomlock*, *Trojan.Randsom* und *Trojan.Ransomcrypt*.

Je nach Art der Ransomware wird der Nutzer z. B. aus dem laufenden System ausgeperrt und muss, um sich erneut einloggen zu können, mit seinem Mobiltelefon eine teure Kurznachricht (SMS) an eine spezielle Nummer schicken oder eine kostenpflichtige Hotline anrufen. Meist wird dem Nutzer in diesem Kontext eine definitive Deadline gesetzt, deren Überschreitung dazu führt, dass das System nicht mehr genutzt werden kann.

Die Ransomware Trojan.Ransomcrypt hingegen verschlüsselt alle Dateien mit spezifischen Endungen (etwa .doc, .jpg, .rar, .zip, .txt usw.) auf der Festplatte und löscht die Originaldateien. Nach erfolgreicher Verschlüsselung und Löschung der Originaldateien wird das System automatisch neu gestartet. Anstelle des herkömmlichen Log-in-Screens stehen nun Anweisungen, denen der Nutzer Folge leisten muss, wenn er seine Daten zurückhaben möchte. Meist muss auch hier eine sehr teure Nummer angerufen werden. Das Starten des Windows-Systems im Safemode hilft bei dieser Art von Malware nicht. Weiterer Druck auf den Nutzer wird dadurch aufgebaut, dass alle 30 min eine zufällige Datei gelöscht wird.

Prinzipiell ist es bei bisheriger Ransomware möglich gewesen, die kompilierte Binärdatei mit Techniken des Reverse Engineering zu zerlegen und zu versuchen,

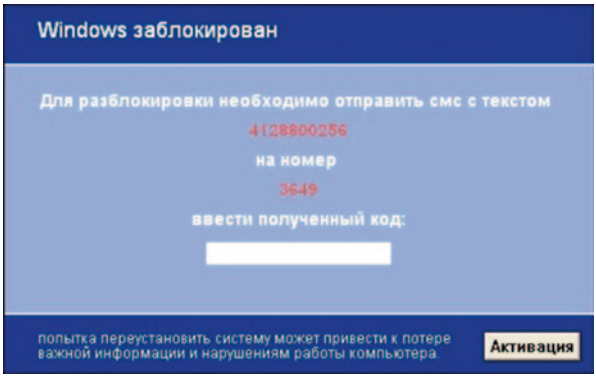


Abb. 28 Aufforderung einer russischen Ransomware-Malware, die den Nutzer zum Versand einer kostenpflichtigen SMS zwingen will. (Quelle: http://www.zdnet.com/blog/security/new-ransomware-locks-pcs-demands-premium-sms-for-removal/3197)

[231] http://www.heise.de/security/meldung/Scareware-wird-zu-Ransomware-209804.html.

den Schlüssel der Verschlüsselung zu erhalten. Dies hat sich grundlegend mit dem Einsatz des Public-Key-RSA-Verfahrens mit einer Schlüssellänge von 1024 Bit geändert. Die Daten des Opfers werden hierbei mit dem öffentlichen Schlüssel verschlüsselt, und nur der Betreiber der Ransomware besitzt den zugehörigen privaten Schlüssel. Ein Reverse Engineering der Binärdatei ist in diesem Fall also sinnlos.

Eine abgeschwächte Art der Ransomware wurde vom Antivirenhersteller Symantec entdeckt.[232] Die eigentliche Ransomware macht dabei nichts anderes, als einen persistenten Inline-Add in alle verfügbaren Browser auf dem System zu legen. Dieser Add weist das Opfer permanent darauf hin, dass eine kostenpflichtige Nummer kontaktiert werden muss, um entfernt zu werden. In diesem Fall werden die Nerven des Opfers konstant durch ein blinkendes Werbe-Add strapaziert. Allerdings konnte, im Gegensatz zu anderen Arten von Ransomware, das Problem durch eine Neuinstallation des Systems gelöst werden. Private Daten wurden nicht verschlüsselt oder geblockt.

9. Malware + JavaScript, Web 2.0-Angriffe

Das Web 2.0 ermöglicht durch seine Technologievielfalt diverse neue Funktionen für Betreiber und Nutzer. Allerdings haben sich durch die Technologien auch neue Angriffsvektoren gebildet, die von Angreifern dazu genutzt werden können, sowohl Nutzer als auch Betreiber von Webanwendungen zu gefährden. Die Statistik des Web Application Security Consortium (WASC)[233] für das Jahr 2008 zeigt, dass Sicherheitslücken in Webanwendungen weiterhin stark auf dem Vormarsch sind und höchstwahrscheinlich auch bleiben werden.

Interessant ist hierbei aber, dass verglichen zu 2007 die Anzahl der SQL-Injektion- und XSS-Schwachstellen zunächst um 13 % bzw. 20 % gefallen ist. Die Chance, ein System mit automatisierbaren Mitteln zu kompromittieren, stieg allerdings von 7 % auf 13 %. Im Folgenden werden einige dieser neuen Angriffsvektoren diskutiert und ihr Gefahrenpotenzial vorgestellt.

a) Cross-Site-Scripting (XSS)

Cross-Site-Scripting (XSS)[234] bezeichnet das Ausnutzen einer Sicherheitslücke in Webanwendungen, wobei Informationen aus einem nicht vertrauenswürdigen Kontext in einen anderen Kontext eingefügt werden, in dem sie als vertrauenswürdig gelten. Aus diesem vertrauenswürdigen Kontext kann dann ein Angriff gestartet werden. Ziel ist es meist, an sensible Daten des Opfers zu gelangen, um beispiels-

---

[232] http://www.symantec.com/connect/blogs/browsers-and-ransoms.

[233] http://projects.webappsec.org/Web-Application-Security-Statistics.

[234] http://www.cgisecurity.com/xss-faq.html.

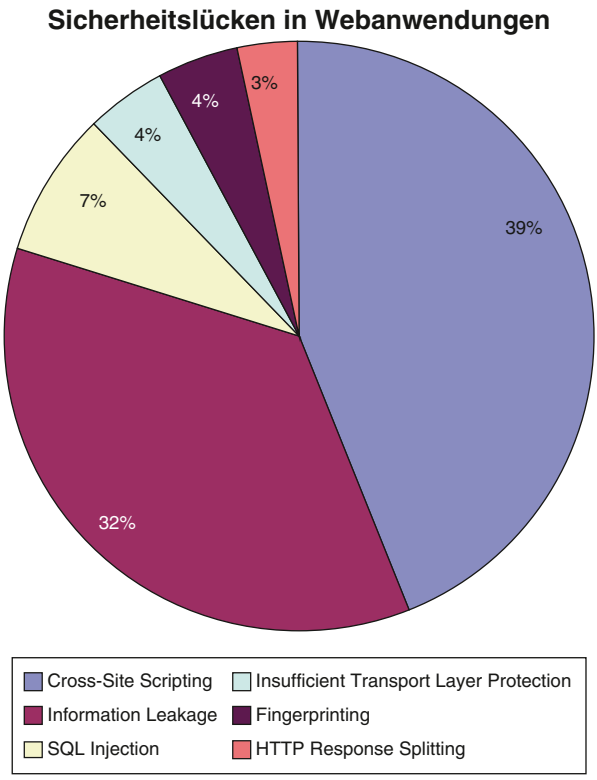


Abb. 29 Statistik der populärsten Sicherheitslücken in Webanwendungen; die Zahlenwerte wurden dem Bericht des WASC aus dem Jahre 2008 entnommen

weise Identitätsdiebstahl zu betreiben. Eine sehr verbreitete Methode hierfür ist, bösartiges JavaScript als Payload der XSS-Schwachstelle zu übergeben. Dieses JavaScript wird dann im vertrauenswürdigen Kontext im Browser des Opfers ausgeführt.

XSS-Schwachstellen werden meist über schlecht programmierte Serverskripte ausgelöst, wohingegen der eigentliche Payload auf dem Client ausgeführt wird. Bei XSS-Schwachstellen kann zwischen zwei verschiedenen Arten differenziert werden – zwischen *persistentem* und *nicht persistentem* (reflektivem) XSS.

Bei dem persistenten XSS wird der Payload aufseiten des Servers beispielsweise in einer Datenbank gespeichert und bei jedem Besuch durch den Browser des Clients ausgeführt. Bei dem nicht persistenten XSS muss das Opfer eine individuell kreierte URL aufrufen, was dazu führt, dass der XSS-Payload nur einmalig in diesem Kontext ausgeführt wird.

XSS-Schwachstellen sind gerade im Kontext von Onlinebanking- und Social Network-Anwendungen mit hoher Nutzerinteraktion kritisch, da dort zwei Ziele relativ einfach erreicht werden können: Identitätsdiebstahl und das Ausführen von bösartigem JavaScript auf vielen Nutzersystemen, um z. B. Nutzersysteme mit Malware zu infizieren.

Onlinebanking-Anwendungen mit XSS-Schwachstellen können relativ einfach für Identitätsdiebstahl missbraucht werden.[235] Handelt es sich beispielsweise um eine nicht persistente Schwachstelle, erstellt der Angreifer einen individuellen Link mit JavaScript-Payload und propagiert diesen im großen Stil, beispielsweise über Phishing-E-Mails. Der bösartige Link könnte wie folgt aussehen:

https://www.beispiel-bank.com/login.asp?content=<script>evil()</script>.

Der Vorteil eines Angriffs mit ausgenutzter XSS-Schwachstelle gegenüber einer normalen Phishingattacke liegt auf der Hand. Die Verbindung zum Server der Bankanwendung ist nach wie vor über SSL/TLS geschützt und wird beim Opfer kein Misstrauen erwecken. Die Funktion evil() wird nun im Kontext der vertrauenswürdigen Website ausgeführt und könnte beispielsweise das Cookie bzw. die SessionID des Nutzers an einen Server des Angreifers übertragen. Eine SessionID wird bei Anwendungen auf zustandslosen Protokollen als Identifikationsmerkmal verwendet. Dies bedeutet, dass es die SessionID ermöglicht, bei mehreren zusammengehörigen Anfragen eines Nutzers diese Anfragen einem Nutzer genau zuzuordnen. Meist besteht eine SessionID aus einer zufällig gewählten Zeichenkette und hat zusätzlich den Nachteil, dass sie prinzipiell nicht durch weitere Authentifizierungsmechanismen wie beispielsweise ein Passwort geschützt ist. Dies bedeutet, dass ein Angreifer, der in den Besitz einer SessionID gekommen ist, sich prinzipiell als der eigentliche Besitzer der zugehörigen Identität ausgeben kann. Viele große Websitebetreiber wie eBay und Amazon fragen mittlerweile das Passwort trotz Authentifizierung durch die SessionID vor einer Transaktion aus Sicherheitsgründen nochmals ab. Eine weitere Maßnahme gegen die feindliche Übernahme der SessionID wäre die Bindung an die IP-Adresse des Nutzers. Auf diese Maßnahme sollte man sich allerdings nicht ausschließlich verlassen, da sich beispielsweise Nutzer hinter Proxies nicht eindeutig zuordnen lassen.

Je nach Art des JavaScript-Codes wird der Nutzer darauf hingewiesen, dass Content von einer nicht vertrauenswürdigen Quelle nachgeladen wird. In der Abb. 30 wird die Sicherheitswarnung des Mozilla Firefox beim Versuch, unverschlüsselte Inhalte von fremden Quellen nachzuladen, veranschaulicht. Falls der Nutzer allerdings erstmalig kein Häkchen setzt, wird die Warnung beim nächsten Mal nicht wieder angezeigt.

Diese Warnmeldung wird viele Nutzer allerdings nicht davon abhalten, dem Inhalt von Websites zu vertrauen, wo Dateien von weiteren Parteien heruntergeladen

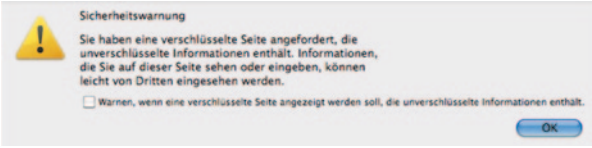


Abb. 30 Sicherheitswarnung des Mozilla Firefox beim Nachladen von unverschlüsselten Inhalten

[235] http://news.netcraft.com/archives/2008/01/08/italian_banks_xss_opportunity_seized_by_fraudsters.html.

werden. Problematisch ist allerdings, dass es sich bei den Inhalten Dritter um bösartige Skripte handeln könnte, die beispielsweise die Inhalte eines Webformulars an eine fremde Partei schicken. Viele Websites wie z. B. große Portale nutzen die Methode des Nachladens unverschlüsselter Dateien von Dritten für die eigene Finanzierung durch Werbe-Adds. In diesem Fall ist das Nachladen gewollt, birgt aber auch seine Risiken, da der nachgeladene Inhalt meist nicht im Vorfeld überprüft wurde.

Der Sessiondiebstahl im Kontext von sozialen Netzwerken oder anderen interaktiven Plattformen könnte anders verlaufen, vorausgesetzt, die Serverskripte enthalten eine XSS-Schwachstelle. Das Opfer erhält eine personalisierte Nachricht über die Plattform, die beim Öffnen automatisch bösartigen JavaScript-Code in die geladene Seite einbettet und dieser somit vom Browser des Opfers ausgeführt wird[236]. Je nach den technischen Möglichkeiten muss das Opfer auch zuerst durch Social Engineering-Methoden zum Aufruf eines Links verleitet werden. Der bösartige JavaScript-Code überträgt anschließend die SessionID zu einem Server des Angreifers. Der Angreifer kann anschließend die SessionID übernehmen und sich als Opfer ausgeben.

XSS-Schwachstellen sind nach wie vor sehr verbreitet und werden von vielen Webentwicklern weiterhin ignoriert. Dies dürfte wohl daran liegen, dass vielen Verantwortlichen das enorme Gefahrenpotenzial nicht bewusst ist. Oftmals werden XSS-Schwachstellen auch geduldet, da sie laut der Administratoren im spezifischen Kontext der Website keine Gefahr darstellen.

b) Cross-Site-Reference-Forgery (XSRF)

XSRF, auch bekannt als Cross-Site-Request-Forgery (CSRF)[237], ist eine Angriffsmethode, bei der ein Angreifer unberechtigt Daten in einer Webanwendung im Kontext eines Opfers verändert. Spezifische URLs wie die Folgende erlauben es dem Angreifer hierbei, den Browser eines Opfers dazu zu bringen, eine gewünschte Aktion durchzuführen.

```
http://www.example.com/user.php?action=new_user&name=Max&password=test123
```

Der obige Link erlaubt es beispielsweise dem Administrator der Website example.com, einen neuen Nutzer „Max“ mit dem Passwort „test123“ anzulegen. Um diese Aktion durchführen zu können, muss sich der Administrator gegenüber dem System authentifizieren. Dies geschieht dadurch, dass er sich mit Nutzername und Passwort dem System gegenüber ausweist. Ist ein Administrator im System eingeloggt und besucht z. B. in einem weiteren Tab des Browsers eine andere Website, die als Image-Tag den obigen Link verankert hat, so ruft der Browser diesen Link automatisch im Kontext des Administrators auf. Dies bedeutet, dass ein Angreifer den Browser des Opfers ohne Mitwissen des Nutzers dazu verleiten kann, nicht autorisierte Aktionen durchzuführen. Eine Schwierigkeit für den Angreifer könnte es sein,

---

[236] http://www.computer.org/portal/web/csdl/doi/10.1109/CSE.2009.424.

[237] http://www.owasp.org/index.php/Cross-Site_Request_Forgery_%28CSRF%29.

Hintergrundwissen über die Linkstruktur einer Website zu erlangen. Allerdings ist dies in vielen Szenarien sehr einfach, da der Angreifer z. B. selbst ein Nutzer mit eingeschränkten Rechten ist. In anderen Angriffsszenarien, wie beispielsweise dem Angriff auf die Webadministrationsoberfläche eines Routers, kann der Angreifer das benötigte Wissen durch den Kauf solch eines Routers erwerben.

Im Bereich des Onlinebanking könnte es sein, dass der Angreifer selbst Kunde bei der betroffenen Bank ist.[238] Nehmen wir beispielsweise an, eine Überweisung eines Onlinebanking-Systems wird mit folgendem Link durchgeführt:

http://www.bank.com/transfer.pl?konto=12345&betrag=5000.

Ein Angreifer könnte versuchen, diesen Link, versehen mit einer von ihm kontrollierten Kontonummer und einem hohen Geldbetrag, im großen Stil auf gut besuchten Websites einzubetten. Alle Besucher dieser Website, die gleichzeitig mit ihrem Bankaccount eingeloggt sind, führen automatisch diese Transaktion aus. Problematisch hierbei ist vor allem die Tatsache, dass der Nutzer von dieser Transaktion nichts mitbekommt.

Dass dieses Szenario durchaus realistisch ist, zeigten Ed Felten und Bill Zeller in ihrem Beitrag über Cross-Site-Request-Forgery-Schwachstellen. Denn da z. B. in Amerika beim Onlinebanking anscheinend keine TANs oder ähnliche Sicherheitsmaßnahmen verwendet werden, können auf den Websites diverser Banken Überweisungen über CSRF-Angriffe durchgeführt werden.

c) Web 2.0-Würmer

Der erste allgemein bekannt gewordene Web 2.0-Wurm wurde im Herbst 2005 auf die Plattform MySpace losgelassen. Ein Nutzer namens Samy war der Ansicht, er hätte zu wenig Freunde auf MySpace. Daher programmierte er einen Web 2.0-Wurm, der dieses Problem beseitigen sollte und seither als *Samy-Wurm*[239] in die Geschichte einging.

Der Autor fand eine persistente XSS-Schwachstelle auf der MySpace-Plattform. So war es ihm möglich, bösartigen JavaScript-Code auf seiner eigenen Profilseite einzubetten. Beim Besuch der Profilseite von Samy verbreitete sich der Wurm nun automatisch über eine persistente XSS-Schwachstelle auf die Profilseiten der Besucher. Denn durch die XSS-Schwachstelle interpretierte der Browser des Besuchers die in der Schadfunktion enthaltenen Steuerbefehle als legitime Befehle seitens der MySpace-Plattform. Die Schadfunktion wurde unbemerkt ausgeführt und enthielt zwei Steuerbefehle: „Kopiere die Schadfunktion in die persönliche Homepage des Besuchers“ und „Füge den Benutzernamen des Angreifers zu der Liste der eigenen Freunde hinzu.“ Auf diese Weise verbreitete sich der Wurm lawinenartig schnell.

---

[238] http://www.freedom-to-tinker.com/blog/wzeller/popular-websites-vulnerable-cross-site-request-forgery-attacks.

[239] http://namb.la/popular/tech.html.

Innerhalb von 24 h hatte der Nutzer Samy eine Million Freunde.[240] Allerdings war die Absicht des Wurms nicht, Passwörter zu stehlen oder eine andere Plattform zu attackieren. Trotzdem musste MySpace den Betrieb vorübergehend komplett einstellen, um den Wurm zu stoppen und alle befallenen Seiten zu reinigen.

Der Wurm Samy blieb allerdings nicht der einzige seiner Art. Im Laufe der letzten Jahre wurde fast jedes größere Portal oder soziale Netzwerk von einem Wurm befallen. Web 2.0-Würmer haben die Eigenschaft, dass sie sich ausschließlich auf Webapplikationsebene bewegen und fortpflanzen. Diese Art von Würmern kann daher mit „digitalen Krankheiten" von Massively Multiplayer Online Role-Playing Games (kurz: MMORPGs)[241] verglichen werden, da diese sich auch ausschließlich über das betroffene Spiel weiterverbreiten. Im Jahre 2005 war beispielsweise das MMORPG „World of Warcraft" durch einen Programmierfehler von einer digitalen Epidemie betroffen.[242]

Web 2.0-Würmer hatten in den meisten Fällen lediglich die Absicht, dem Autor Ruhm und eine Menge neuer Freunde zu bescheren. Sie befielen nur die Profile von Nutzern und nicht die eigentlichen Endsysteme. Allerdings könnten Würmer solcher Art auch für bösartige Zwecke erstellt werden, um beispielsweise die Nutzer auf Websites mit schädlichem Inhalt zu locken und anschließend zu versuchen, das Endsystem des Nutzers zu kompromittieren.

Die Ursachen für einen Wurm wie den Samy-Wurm sind bekannt und weitverbreitet. In der Regel benötigt der Programmierer eine XSS-Schwachstelle auf der Website, über welche er JavaScript-Schadcode im Kontext der Website ausführen kann. Noch einfacher für den Autor wird es, wenn es sich bei der XSS-Schwachstelle um eine persistente Cross-Site-Scripting-Schwachstelle handelt. Denn dann wird der Schadcode z. B. direkt im Profil des Autors abgelegt, und jeder eingeloggte Besucher führt diesen Code automatisch aus.

Anfang diesen Jahres wütete ein Wurm[243] auf der Plattform Twitter[244], der sich genau dieses Verfahren zunutze machte. Sobald ein eingeloggter Nutzer ein infiziertes Profil besuchte, wartete der Schadcode zunächst drei Sekunden, bevor er vom Browser den Namen des Nutzers und das Twitter-Cookie anforderte. Anschließend konnte der Schadcode über die Twitter-API im Namen des Nutzers sogenannte Tweets (Kurznachrichten) verschicken und sich im Profil des Nutzers verewigen, um auf diese Weise weitere Besucher zu infizieren.

Web 2.0-Würmer haben den Vorteil, dass sie meist aus nur relativ wenigen Zeilen Code bestehen. Verfügt die betroffene Website über eine persistente XSS-Schwachstelle, so kann der Schadcode bequem im Profil eingebettet werden und muss nicht über bösartige Links und Social Engineering-Methoden verbreitet werden.

---

[240] http://www.guardian.co.uk/media/2006/mar/09/newmedia.technology.

[241] http://de.wikipedia.org/wiki/Massively_Multiplayer_Online_Role-Playing_Game.

[242] http://www.securityfocus.com/news/11330.

[243] http://blog.twitter.com/2009/04/wily-weekend-worms.html.

[244] http://twitter.com.

Es ist daher davon auszugehen, dass Web 2.0-Würmer auch in Zukunft auf großen Web 2.0-Anwendungen wüten werden. Hierbei ist zu hoffen, dass die Endsysteme der Nutzer zukünftig weiterhin verschont bleiben.

10. Google-Hacking

Zusammenfassung: Internetsuchmaschinen bieten einem Angreifer auch im Rahmen des Identitätsdiebstahls bzw. -missbrauchs eine gute Hilfestellung. Google bietet dabei mit seiner Vielzahl von Möglichkeiten und Optionen eine besonders gute Ausgangsbasis. Zudem kann sich ein Angreifer Suchmaschinen auch durch Methoden der Suchmaschinenoptimierung zunutze machen und Anwender so auf malwareverseuchte Webserver locken.

Google[245] ist wohl die bekannteste und größte Suchmaschine (Marktanteil[246] [Stand Oktober 2009] etwa 80 %) und wird schätzungsweise täglich von mehr als 200 Mio. Nutzern zur Recherche im Internet genutzt. Neben der Suche über eine einfache Eingabe von Suchbegriffen besitzt Google aber auch die Möglichkeit, mit einer Google-eigenen Syntax gezielte Abfragen zu stellen. Neben Google existieren zudem noch weitere Suchmaschinen, bekannt sind beispielsweise noch MSN, Yahoo! und AltaVista. Darüber hinaus gibt es noch spezielle Dienste, die neben dem aktuellen Inhalt von Websites auch die zeitliche Entwicklung des Inhalts beobachten können. Dazu zählt beispielsweise die Way Back Machine.[247]

Google indexiert alle im Internet gefundenen Webinhalte, wenn dies nicht durch spezielle Maßnahmen – etwa durch Einträge in der Datei robots.txt[248] – verhindert wird. Einträge in robots.txt werden vor allem deshalb erstellt, damit Google sensitive Daten nicht indexiert und damit auch nicht in seinen Cache aufnimmt. Damit kann ein potenzieller Angreifer die Daten zwar nicht mittels Google (oder anderen Suchmaschinen) finden, bekommt aber über robotx.txt zugleich die Information, in welchem Verzeichnis des Webservers sich die vom Serveradministrator als sensitiv eingestuften Daten befinden.[249] Zudem existiert das Problem, dass sich nicht alle

---

[245] Zu den Hintergründen von Google siehe beispielsweise auch: http://de.wikipedia.org/wiki/Google.

[246] Zu den Nutzerstatistiken von Suchmaschinen siehe auch: http://www.webhits.de/deutsch/index.shtml?/deutsch/webstats.html.

[247] Siehe hierzu auch: http://www.archive.org/web/web.php.

[248] Zum Umgang mit der robots.txt siehe beispielsweise: http://de.wikipedia.org/wiki/Robots_Exclusion_Standard.

[249] Auch dazu kann eine spezielle Google-Abfrage ("robots.txt" "disallow:" filetype:txt) genutzt werden. Eine Beispielseite, die mittels dieser Anfrage gefunden wird, ist etwa http://www.focus.de/robots.txt. Diese Seite enthält alle Bereiche, die auf der Webpräsenz von www.focus.de nicht

Suchmaschinenanbieter an die Konventionen von robots.txt halten und trotzdem den gesamten Webauftritt einer Domain durchsuchen.

a) Funktionalität von Google

Google kann also in vielen Fällen Informationen liefern, die sich im Rahmen des Identitätsdiebstahls bzw. -missbrauchs verwenden lassen. Im Folgenden wird nun das entsprechende Potenzial aufgezeigt, das Google einem Angreifer bietet. Zunächst stellen wir die Suchmöglichkeiten mit Google im Detail dar.

Suchoptionen
Google bietet dem Nutzer eine Vielzahl von Suchoptionen und Möglichkeiten und lässt sich sogar als Taschenrechner *Google Calc*[250] und Übersetzungsdienst *Google Translate*[251] verwenden. Neben diesen speziellen Angeboten kann aber auch über die eigentliche Suchanfrage sehr viel an Funktionalität erzielt werden.

Suchen innerhalb einer speziellen Site[252] werden über das Keyword *site* ermöglicht. Die Abfrage *site:a-i3.org phishing* findet somit alle Websites auf a-i3.org, die das Suchwort Phishing enthalten. Auf ähnliche Art und Weise lässt sich die Suche auch auf spezielle Typen von Dokumenten (mittels filetype) oder Dateiendungen (mittels *ext*) einschränken. Die gezielte Suche innerhalb des Titels von Websites oder innerhalb des Unified Ressource Locators (kurz: URL) ist mittels intitle bzw. inurl ebenfalls möglich. Zudem ist es auch möglich, nach Links innerhalb von Dokumenten zu suchen. Dazu wird der Suchstring *link* benutzt: *link:www.google.de* findet somit alle Dokumente, die einen Link auf www.google.de enthalten.

Google erlaubt dabei auch Verknüpfungen verschiedener Suchanfragen. Im Standardfall werden die Suchbegriffe durch ein logisches *Und* verknüpft. Durch | bzw. *OR* kann ein logisches *Oder* eingebaut werden. *Haus OR Hütte* findet somit alle Dokumente, die entweder das Stichwort *Haus* oder das Stichwort *Hütte* enthalten. Auch Platzhalter sind möglich, so findet **stuhl* sowohl Dokumente mit dem Stichwort *Lehnstuhl* als auch mit dem Stichwort *Dachstuhl*. Zudem kann auch nach ganzen Phrasen gesucht werden. Dies geschieht über die Verwendung von. *"Dieser Suchstring sucht nach einer ganzen Phrase"* findet somit nur Dokumente, die die Phrase *Dieser Suchstring sucht nach einer ganzen Phrase* enthalten.

Suchen von Passwörtern
Die Suche nach Passwörtern unter Google lässt sich bspw. mit dem Suchstring *intext:"password|pass|passwd" (ext:sql|ext:dump|ext:dmp) intext:values* realisieren. Dadurch kann ein Angreifer allzu häufig Zugangsdaten zu Webservern erlangen und diese dann im Anschluss missbrauchen, um sich auf dem Webserver und/

---

durch Google indexiert werden sollen, liefert einem Angreifer aber gleichzeitig Informationen über die Struktur des Webauftritts.

[250] Google als Taschenrechner zu finden unter: http://www.google.com/help/features.html.

[251] Google als Übersetzungsdienst: http://translate.google.de/#.

[252] Mit „Site“ ist hier der Inhalt eines Webauftritts gemeint.

oder der Datenbank einzuloggen und die dortigen Inhalte auszulesen oder gar zu manipulieren.[253,254]

Suchen von Multimediageräten
Durch eine Suchanfrage wie bspw. *intitle:"live view/-axis"* lassen sich z. B. auch Webcams finden. Den Anwendern ist dabei oftmals gar nicht bewusst, dass ihre Webcam weltweit per Google gefunden werden kann. Unter Umständen kann sich ein Angreifer dadurch Informationen beschaffen, die ihm helfen, sein potenzielles Opfer besser zu charakterisieren. Darüber hinaus kann die Webcam auch das Einfallstor in das Netzwerk des Anwenders sein.[255,256,257]

Suchen von IT-Infrastruktur
Darüber hinaus lassen sich auch die Administrationsoberflächen von Druckern oder auch VPS-Zugangspunkten mittels Google-Anfragen finden. *intitle:"Cisco Systems, Inc. VPN 3000 Concentrator"* findet bspw. die Administrationsoberfläche des VPN-Concentrators. *inurl:hp/device/this.LCDispatcher* und *inurl:"printer/main.html" intext:settings* finden z. B. die Administrationsoberflächen von Netzwerkdruckern.

Suchen von vertrauenswürdigen Dokumenten
Auch die gezielte Suche nach als „vertraulich" gekennzeichneten Dokumenten ist mittels Google einfach möglich. Dazu reicht die Eingabe von *filetype:ppt "confidential"*, was dann ppt-Dateien zurückliefert, die das Stichwort „confidential" enthalten.

b) Suchmaschinenoptimierung

Eine weitere Form des Google-Hacking ist die sogenannte *Suchmaschinenoptimierung* (engl.: Search Engine Optimization, kurz: SEO). Diese wird auch im Rahmen der Funktionalität der Drive-by-Malware, etwa den Web Exploit Toolkits[258] (kurz: WET), genutzt, um Anwender auf die entsprechend manipulierten Webserver zu locken. Hier manipuliert der Angreifer die Reihenfolge, in der die Suchergebnisse von Google anzeigt werden. Er versucht, seine Website bzw. die des versuchten Servers möglichst weit oben in der Liste der Suchergebnisse zu platzieren, also hoch zu

---

253 Google liefert beispielsweise: http://euchina-cdm.org/cdm_db.sql.

254 Bei solchen Suchanfragen werden nicht nur einzelne Links, sondern auch Verweise auf ganze Archive von entsprechenden Daten gefunden: http://www.hotfile123.com/index.php?q=index+of+admin+inurl+passwd+txt&filetype=&page=2.

255 Ein Beispiel findet sich unter: http://24.231.158.230:8888/ViewerFrame?Mode=Motion&Language=0.

256 Ein Beispiel findet sich unter: http://cam102053.miemasu.net:60003/ViewerFrame?Mode=Motion.

257 Ein Beispiel findet sich unter: http://212.248.100.101:88/ViewerFrame?Mode=Motion.

258 Vgl. dazu Kap.3 I.2.b).

ranken. Bei einer bereits erfolgten Massenverbreitung entsprechender Malware geschieht dies – bedingt durch den häufigen (wenngleich ungewollten) Aufruf durch die Anwender – nahezu von selbst. Darüber hinaus sind aber auch diverse Methoden bekannt, diesen Effekt künstlich zu erzeugen. Google kann somit indirekt dazu beitragen, die Verbreitung von Malware zu intensivieren. Google bemüht sich zwar seit geraumer Zeit, Manipulationen mittels SEO zu minimieren. Dies ist allerdings nicht in allen Fällen möglich, da der Angreifer im Prinzip völlig legitime Methoden benutzt. Anwendern kann nur geraten werden, die Ergebnisse einer Google-Suche mit Vorsicht zu behandeln und zumindest die empfohlenen Standardsicherheitsmaßnahmen einzusetzen.[259]

c) Schutzmaßnahmen

Als Schutzmaßnahme gegen die in diesem Abschnitt beschriebenen Möglichkeiten kommt zunächst einmal eine sichere Servereinstellung in Frage. Der verwendete Webserver sollte dabei immer so konfiguriert werden, dass das sogenannte *Need-to-Know-Prinzip* eingehalten wird. Wann immer möglich, sollten die einzelnen Webpräsenzen auf dem Server mittels *Zugriffsregeln* (engl.: *Access Control Lists*, kurz: *ACL*) so abgesichert werden, dass der Zugriff nur noch durch berechtigte IP-Adressen möglich ist. Die Verwendung von Authentifizierungsmechanismen kann die Sicherheit weiter erhöhen. Dabei ist allerdings zu beachten, dass die Authentifizierungsdaten bei Verwendung der Standardauthentifizierung ohne weitere Maßnahmen oft im Klartext über das Netzwerk übertragen werden[260] und somit von einem potenziellen Angreifer mitgelesen werden können. Eine wesentlich sicherere Authentifizierung ermöglicht die Verwendung von Clientzertifikaten in Verbindung mit der Transport Layer Security[261] (kurz: TLS). Hierbei muss sich der Nutzer mittels eines Clientzertifikats authentisieren, das sich entweder auf einer Smartcard befinden kann und dort mittels der PIN vor unberechtigtem Zugriff geschützt ist oder als Softwarezertifikat[262] im Browser mittels eines Passworts geschützt vorliegt. Da hier sowohl der Besitz des Clientzertifikats als auch die Kenntnis des Geheimnisses (PIN oder Passwort) nachgewiesen werden müssen, handelt es sich um eine Zwei-Faktor Authentifizierung.

Darüber hinaus kann der zielgerichtete Einsatz einer robots.txt-Datei zumindest verhindern, dass Google sensitive Dokumente indexiert.[263] Wie bereits diskutiert, bietet dies aber nur einen rudimentären Schutz und kann einem Angreifer sogar

---

[259] Zu den Standardsicherheitsmaßnahmen vgl. S. 56 ff.

[260] Ein Beispiel hierfür ist die Verwendung der Basic Authentication beim Apache Webserver.

[261] Zur Transport Layer Security siehe beispielsweise: http://en.wikipedia.org/wiki/Transport_Layer_Security.

[262] Gegenüber der Verwendung einer Smartcard ist dies die wesentlich unsicherere Variante, da eine Malware Zugriff auf die Zertifikatsdatei und/oder das zugehörige Passwort erlangen könnte.

[263] Sensitive Dokumente sollten natürlich nie in einem öffentlich zugänglichen Bereich eines Webauftritts liegen.

wertvolle Informationen liefern. Wesentlich ist in jedem Fall der bewusste und sensitive Umgang mit Daten bei der Veröffentlichung im Internet. Die konsequente Umsetzung des Need-to-Know-Prinzips und des Ansatzes einer Absicherung auf mehreren Schichten (engl.: Layered Defense) kann dazu beitragen, das Schutzniveau entsprechend zu erhöhen. Da heute eine Vielzahl von Geräten vom Werk aus mit aktivierten Webschnittstellen ausgestattet ist, ist die Standardkonfiguration vor einer Inbetriebnahme auf die aktuellen Bedürfnisse anzupassen. Der Einsatz einer Firewall kann zudem dazu beitragen, den unkontrollierten Datenabfluss zu minimieren.

II. Prognosen: Zielplattformen

Prognose: Neben dem Windows-PC als bevorzugte Zielplattform für Angriffe werden weitere mobile (Windows Mobile) und stationäre (Apple OS) Plattformen im Fokus der Angreifer stehen. Daneben treten gleichberechtigt Angriffe auf plattformübergreifende Software wie Mozilla Firefox, Apple Safari, Google Chrome und Adobe PDF sowie Flash in den Vordergrund.

Im Bereich der Zielplattformen kann grundlegend zwischen vier Clientsystemen differenziert werden:

- Microsoft-Systeme wie Windows XP, Windows Vista und neuerdings Windows 7,
- Linux- und Unix-Systeme, zu denen die verschiedenen Linux-Derivate und Mac OS-Systeme gehören,
- mobile Plattformen, zu denen Android, Apples iPhone OS, Symbian, RIMs Blackberry und Microsoft Windows Mobile zu zählen sind, und
- Spielekonsolen wie XBOX-360, Wii und Playstation 3.

Diese Zielplattformen sind unterschiedlichen Angriffen und Szenarien ausgesetzt, sodass im Folgenden nun einzeln auf diese Plattformen und die damit verbundenen Gefahren eingegangen wird.

1. Zielplattformen

a) Microsoft Windows 7

Die Systemplattformen von Microsoft besitzen nach wie vor einen Marktanteil von rund 80 %, was sie zu attraktiven Zielplattformen für potenzielle Angreifer macht. Daher stehen Angriffe auf Microsoft Windows-Systeme immer noch an der Spitze der Angriffsstatistik.

Laut Microsoft wurde die Sicherheit des Systems in Windows 7 nochmals verbessert. Man hat von Anfang an einen *Security Development Lifecycle* (SDL)[264]

[264] http://blogs.msdn.com/sdl/.

mit in den Entwicklungsprozess integriert. Der SDL erweitert die Phasen der Softwareentwicklung um mehrere auf Sicherheit abzielende Aktivitäten. Zu diesen Aktivitäten zählen die Entwicklung von Bedrohungsmodellen beim Softwareentwurf, die Verwendung von Tools zur statischen Codeanalyse bei der Implementierung sowie das Durchführen von Codeüberprüfungen und Sicherheitstests bei einer gezielten Suche nach Sicherheitsmängeln. Beispielsweise wurden Verfahren wie Data Execution Prevention (DEP)[265], Address Space Layout Randomization (ASLR)[266] und Maßnahmen gegen das böswillige Patchen des Kernels eingeführt.

Des Weiteren wurden bei den Firewalleinstellungen Veränderungen vorgenommen. Wo es bei Windows Vista noch Probleme mit den verschiedenen Firewall-Policies gab, soll sich das bei Windows 7 nun geändert haben. Vista unterscheidet den Typ der Netzwerkverbindung: Home, Work, Public oder Domain. Dies kann zu Problemen führen, wenn Nutzer sich über das Internet einwählen, sich anschließend aber in das firmeninterne VPN einwählen wollen. Denn Vista wendet zuallererst die Firewall-Policy „Public" für das System an, die anschließend bei Einwahl in das Firmen-VPN nicht mehr auf „Domain" geändert werden kann, da bereits eine Policy besteht. Windows 7 soll dieses Problem dadurch lösen, dass nun mehrere Firewall-Policies auf verschiedene Netzwerk-Interfaces angewendet werden können.

In Windows 7 soll auch die sichere Verbindung zum Firmennetzwerk besser möglich sein. DirectAccess, basierend auf IPv6 und IPsec, soll bisherige VPN-Tunnel überflüssig und in Zusammenarbeit mit dem Windows-Server 2008 RC2 die Arbeit in unsicheren Netzen noch sicherer machen.

Die Verschlüsselung von externen Platten und USB-Sticks wird in Windows 7 per Bitlocker To Go realisiert.[267] Es handelt sich dabei um eine Erweiterung für das Bitlocker-Programm von Windows Vista.

Auch der Bereich Antimalware wurde in Windows 7 ausgebaut. Problematisch war bisher, dass viele Arten von Malware Administratorrechte auf dem eigentlichen Zielsystem benötigten. Sie laufen mit den Rechten eines Administrators und können so die meisten Funktionen ausführen. Das Entziehen der Administratorrechte für Nutzer ist daher ein hilfreicher Schritt in Richtung erhöhter Sicherheit für das System, löst allerdings das Malwareproblem nicht grundlegend. Denn viele Nutzer installieren Programme, die sie per USB-Stick mitbringen oder sich aus dem Netz laden. Mit der *AppLocker*-Anwendung von Windows 7 sollen Administratoren nun wieder mehr Kontrolle über die Programme auf dem System bekommen. AppLocker schaltet sich zwischen Kernelaufrufe, die versuchen, neue Prozesse zu erstellen oder Bibliotheken zu laden, und hinterfragt, ob der eigentliche Programmcode die notwendigen Rechte besitzt, um ausgeführt zu werden. AppLocker soll somit versichern, dass nur berechtigte Programme auf dem System ausgeführt werden.

---

[265] http://en.wikipedia.org/wiki/Data_Execution_Prevention.

[266] http://www.ngssoftware.com/papers/xpms.pdf.

[267] http://www.microsoft.com/windows/enterprise/products/windows-7/features.aspx.

Der Antivirenhersteller Sophos hat Windows 7 in Hinsicht auf derzeit gängige Malware getestet.[268] Auf einem frisch installierten Windows 7-System ohne zusätzliche Gegenmaßnahmen wurden allerdings acht von zehn Malwaresamples erfolgreich ausgeführt. Laut Sophos ist die UAC-Maßnahme in Windows 7 nicht dafür geeignet, das System vor Malware zu schützen.

Aber nicht nur Microsoft selbst hat eingesehen, dass zusätzliche Maßnahmen für die Sicherheit des Systems notwendig sind. Einige der großen Antivirenhersteller haben ihre Produkte bereits auf Windows 7 portiert.

Die oftmals bemängelten UAC-Sicherheitsanfragen von Windows Vista an den Nutzer in Form von Pop-Ups sollen um 30 % reduziert worden sein. Der Nutzer kann nun selbst einstellen, wie viele Sicherheitsanfragen er bekommen möchte. Ob dies zur erhöhten Sicherheit des Systems beiträgt, bleibt aber offen. Es wird sich in näherer Zukunft zeigen, ob Microsoft mit der Einführung von Windows 7 den Angreifern wieder einmal einen Schritt voraus sein konnte.

b) Linux und Mac OS

Zielgerichtete Angriffe auf Linux-Clientsysteme sind nach wie vor kaum zu verzeichnen. Dies liegt sicherlich am geringen Marktanteil von Linux-Derivaten im Desktopeinsatz. Die eventuellen Vorteile in Bezug auf die Sicherheit von Linux-Systemen gegenüber Windows-Systemen werden in der Szene weiterhin kontrovers diskutiert. Der hohe Marktanteil von Microsoft-Betriebssystemen wird häufig als Ursache für die überdurchschnittlich hohe Zahl an Angriffen gesehen.

Beispielsweise sind Drive-by-Angriffe auf Browser unter Linux bisher nicht bekannt. Denn gängige Web Exploit Toolkits (WETs) identifizieren das Opfersystem anhand der Informationen, die sich aus dem HTTP-Request-Header extrahieren lassen. Bei den bisher untersuchten WETs waren keine Linux-Signaturen auffindbar, die auf mögliche Drive-by-Angriffe auf Linux-Clients schließen lassen könnten.

Eine Drive-by-Infektion von Linux-Derivaten wäre aber eventuell möglich, wenn auf dem System Browser mit verwundbaren Plug-ins installiert sind. Über eine Schwachstelle im Plug-in könnte dann eventuell auch ein Linux-System mit Schadcode infiziert werden.

Das auf BSD-Unix basierende Mac OS X aus dem Hause Apple hat dagegen diverse Probleme mit Drive-by-Angriffen in der Vergangenheit aufzuweisen. Bei der Auslieferung des neuen Systems *Snow Leopard* wurde eine veraltete Version des Adobe Flash Players integriert.[269] Diese veraltete Version des Players enthielt Sicherheitslücken, die das Mac-System für Drive-by-Angriffe verletzlich machten. Es wurde daher durch die Presse empfohlen, den Adobe Flash Player manuell schnellstmöglich upzudaten.

Paradox erscheint hierbei die Tatsache, dass Apple sein neues System Snow Leopard mit einem rudimentären Malwareblocker ausgestattet hat, der jedoch

---

[268] http://www.sophos.com/blogs/chetw/g/2009/11/03/windows-7-vulnerable-8-10-viruses/.

[269] http://blogs.adobe.com/psirt/2009/09/flash_player_update_and_snow_l.html.

bislang nur zwei Malwaretypen erkennt. Der als *XProtect* bekannte Schutzmechanismus[270] greift zudem lediglich bei Downloads und bietet keinen Echtzeitschutz oder manuelles Scannen der Festplatte an. Es kann hier also keinesfalls von einem vollständigen Schutz vor Schadsoftware geredet werden.

Auch Apples Browser Safari war in der Vergangenheit immer wieder Opfer von Zero Day-Exploits. Dies zeigte der Sicherheitsexperte Charlie Miller beim Pwn2Own Contest in Vancouver, indem er ein vollständig gepatchtes Mac OS X durch eine Schwachstelle im Safari Browser innerhalb weniger Minuten kompromittierte.[271]

In der IT-Sicherheitsszene wird derzeit heiß diskutiert, wie das Gefährdungspotenzial von Mac-Systemen einzuschätzen ist. Denn Fakt ist, dass Apple-Nutzern derzeit weniger Schutzmaßnahmen vor Viren und anderer bösartiger Software zur Verfügung stehen, als dies bei Windows-Nutzern der Fall ist. Allerdings wird argumentiert, dass sich Mac-Nutzer immer noch sicherer im Netz bewegen als Windows-Nutzer, da die Anzahl der vorhandenen Schadsoftware für Mac-Systeme relativ gesehen sehr gering ist.[272] Der Virenspezialist Dmitry Samosseiko des Herstellers Sophos referierte auf der Konferenz *Virus Bulletin 2009* über die Verbreitung von Malware auf Mac OS X-Plattformen und kam zu dem Schluss, dass diese derzeit wohl noch als zu unattraktiv für Angreifer gelte.[273] In diesem Zusammenhang erklärte Samosseiko, dass für infizierte Mac-Systeme etwa 43 US-Cent auf dem Untergrundmarkt geboten wurden, wohingegen Windows-Systeme zwischen 50 und 55 Cent gehandelt werden. Nun bleibt die Frage offen, wie lange dieser Zustand noch anhält.

Da Apple zunehmend Marktanteile gewinnt, steigt auch die Attraktivität der Plattform für Angriffe jeglicher Art. So tauchten bereits die ersten speziell auf Mac OS X ausgerichteten Trojanischen Pferde im Netz auf, die über präparierte Websites verbreitet werden. Angreifer machen sich auch hier die Informationen aus den HTTP-Request-Headern zunutze, um zwischen Windows- und Mac OS X Nutzern zu differenzieren. Je nach verwendetem Browser und Betriebssystem wird entweder eine exe- oder dmg-Datei ausgegeben und mit Social Engineering-Methoden versucht, den Nutzer zum Ausführen der Datei zu verleiten.

Der erstmals Ende 2007 erkannte Mac-Trojaner *RSPlug*[274] ist seit seiner Erstentdeckung in verschiedenen Versionen im Netz verfügbar. Die Infektionsroutinen beruhen entweder auf Drive-by-Browser-Exploits, oder es wird versucht, das Schadprogramm als Video-Codec zu tarnen.

Die Anzahl von Mac-Schadsoftware ist zwar weiterhin relativ leicht überschaubar, das Risiko für eine Infektion ist heute jedoch sehr viel höher als noch vor zwei Jahren.[275]

---

[270] http://www.sophos.com/blogs/sophoslabs/v/post/6269.

[271] http://blogs.zdnet.com/security/?p=2917.

[272] http://www.wired.com/gadgetlab/2009/09/security-snow-leopard.

[273] http://www.sophos.com/sophos/docs/eng/marketing_material/samosseiko-vb2009-paper.pdf.

[274] http://www.intego.com/news/ism0705.asp.

[275] http://oreilly.com/catalog/9780596523039.

c) Mobile Plattformen

Mobile Plattformen bieten Nutzern die Möglichkeit, viele herkömmliche Desktopanwendungen auch auf dem mobilen Endgerät auszuführen. In den letzten Jahren konnte die Entwicklung im Bereich mobiler Plattformen einen enormen Fortschritt verzeichnen. Jedoch steigt mit der Portierung von Desktop-anwendungen auf mobile Plattformen auch das Gefahrenpotenzial. Nach Meinung vieler IT-Sicherheitsexperten ist es nur eine Frage der Zeit, bis entsprechende Schadsoftware auch für mobile Plattformen verfügbar sein wird.

Im Juni 2004 wurde die erste Proof-of-Concept-Malware für die Symbian-Plattform entdeckt. *EPOC.Cabir*[276] hatte ursprünglich keine bösartigen Absichten, sondern sollte lediglich zeigen, dass Malware auch für mobile Plattformen existieren kann. Der Cabir-Wurm verbreitete sich via Bluetooth-Schnittstelle und war so programmiert, dass jedes infizierte Gerät über Bluetooth weitere Geräte suchte, die infiziert werden konnten.

Prinzipiell kann mobile Malware verschiedene Ereignisse auf einem Smartphone auslösen.

- Ohne Wissen des Nutzers wird eine Massenaussendung von SMS und/oder MMS durchgeführt. Zusätzlich können teure Nummern, beispielsweise im Ausland, angerufen werden. Der Nutzer wird dieses Fehlverhalten erst bei der nächsten Monatsabrechnung bemerken.
- Daten des Nutzers wie beispielsweise das Telefonbuch oder private Dokumente können vernichtet werden. Es können zudem vertrauliche Informationen gestohlen und/oder einzelne Funktionen des Smartphones wie SMS, Spiele, Mikrofon oder Kamera blockiert werden.
- Der Akku des Smartphones könnte sich schneller als normal entladen.
- Unter dem Namen des Nutzers könnten per E-Mail, W-LAN oder Bluetooth andere Smartphones oder Systeme attackiert und infiziert werden. Die Einträge im Adressbuch des Opfers könnten hierbei als potenzielle Opfer gelten.
- Bei der Synchronisation von Smartphone und Desktop-PC oder Laptop könnte sich der Schadcode weiter ausbreiten. Hier wäre allerdings hybrider Schadcode notwendig, der sich sowohl auf Smartphone-Plattformen als auch auf PC-basierten Plattformen ausbreiten kann.

Nutzer und Entwickler von Smartphones sollten sich darüber im Klaren sein, dass jede neue Funktion, die dem Nutzer angeboten wird, auch für Angriffe missbraucht werden kann.

So auch im Falle der Bluetooth-Schnittstelle, die z. B. vom *CommWarrior-Wurm*[277] für Symbian-Plattformen zur Weiterverbreitung missbraucht wurde. Hierbei versucht der Wurm, den Nutzer zur Installation zu überreden, und verbreitet sich nach erfolgreicher Installation via Bluetooth-Schnittstelle, MMS oder Speicherkarte weiter. Bei der Weiterverbreitung per Bluetooth sucht das infizierte Mobiltelefon

[276] http://www.sophos.com/security/analyses/viruses-and-spyware/symbcabira.html.

[277] http://www.f-secure.com/v-descs/commwarrior.shtml.

nach offenen, potenziellen Empfangsgeräten und verschickt Dateien mit einem zufälligen Dateinamen und der Dateiendung *.SIS*. Der Bluetooth-basierte Infektionsmechanismus ist ohne spezielle Antennen auf ca. 10 m begrenzt.

Auch im Bereich der mobilen Malware gilt das einfache Prinzip: Je größer der Marktanteil, desto höher die Chance, dass für diese Plattform Schadcode entwickelt wird. So gilt Symbian mit einem Marktanteil von 60 % derzeit als die Nummer 1 unter den Smartphone-Plattformen.

Die entdeckte Schadsoftware für mobile Plattformen war bereits in der Vergangenheit recht vielfältig. Der Trojaner *Doomboot*[278], welcher im Juli 2007 entdeckt wurde, gibt vor, eine Version des Shooters Doom 2 zu sein, und hält Nutzer davon ab, das System auf dem Smartphone zu booten. Die im Februar 2006 entdeckte Schadsoftware *RedBrowser*[279] sendete Textnachrichten an eine teure Premiumnummer in Russland. Dahingegen hatte die Malware *FlexiSpy* andere Absichten. Sie versendete Logdateien von Telefonanrufen und Kopien von Text- und MMS-Nachrichten an einen Server.

Die Voraussetzungen für die Entwicklung von Malware für die Plattform Windows Mobile wurden von Boris Michael Leidner Anfang 2007 untersucht. In seiner Diplomarbeit untersuchte er die Voraussetzungen, die für die Entwicklung eines Computerwurms für Windows Mobile 5.0 benötigt wurden. Dazu erstellte er einen Baukasten, der als Machbarkeitsnachweis für einen Computerwurm die Bedrohung durch mobile Malware aufzeigt. Des Weiteren versuchte er, durch Fuzzing-Techniken Sicherheitslücken in Windows Mobile aufzudecken, was ihm aber nicht gelang. Anscheinend stellte die Implementierung von sogenannten *Security Cookies* ein großes Hindernis bei der Ausnutzung von stackbasierten Pufferüberläufen dar.[280] Im Phrack Magazin #63 wurde allerdings ein Proof-of-Concept für einen Pufferüberlauf in Windows CE, dem Vorgänger von Windows Mobile, veröffentlicht.[281]

Colline Mullier und Charlie Miller veröffentlichten auf der Usenix-WOOT-Konferenz eine neue Methode, um Sicherheitslücken in SMS-Implementierungen von Smartphones zu analysieren. Sie behaupten, dass ihr Ansatz unabhängig vom Provider sei, keine zusätzlichen Kosten produziere und reproduzierbare Resultate garantieren würde. Anscheinend gelang es ihnen, mit diesem Ansatz bisher nicht bekannte Sicherheitslücken zu identifizieren, die für Denial-of-Service-Angriffe gegen gängige Smartphones genutzt werden könnten.[282]

Die Autoren waren auch für Gerüchte um einen Bug in der SMS-Implementierung des iPhones verantwortlich, die Mitte des Jahres 2009 entstanden. Sie behaupteten, sie hätten einen Weg gefunden, einen speziell präparierten SMS-Code in das iPhone einzuschleusen, der dort dann mit Root-Rechten ausgeführt werden könnte. Mit so einer Lücke wäre es möglich gewesen, sich beispielsweise Zugriff auf die GPS-Koordinaten oder das Mikrofon des Handys zu verschaffen. Die Autoren

---

[278] http://www.netzwelt.de/news/71830-doomboot-a-trojaner-zerstoert-symbian-handys.html.

[279] http://www.f-secure.com/v-descs/redbrowser_a.shtml.

[280] http://pi1.informatik.uni-mannheim.de/filepool/theses/diplomarbeit-2007-leidner.pdf.

[281] http://www.phrack.org/issues.html?issue=63&id=6#article.

[282] http://www.usenix.org/event/woot09/tech/full_papers/mulliner.pdf.

schafften es anscheinend, das iPhone zum Absturz zu bringen, das Ausführen von beliebigem Schadcode gelang ihnen allerdings nicht. Apple publizierte einen Patch für diese Probleme, spezifizierte in diesem allerdings nicht, um was für eine Lücke es sich letztlich handelte.[283] Im entsprechenden CVE-Report wird hingegen behauptet, es wäre möglich, beliebigen Code auf dem kompromittierten System auszuführen.[284]

Für weitere Schlagzeilen sorgte die Smartphone-Version des Safari Browsers im September 2009. Das Problem war anscheinend eine Inkonsistenz bei der Entdeckung von bekannten Phishingwebsites. So ließen sich bereits erkannte und registrierte Phishingwebsites auf demselben iPhone mal ohne und mal mit Warnung aufrufen. Die Anbindung des iPhones an das Internet machte hierbei anscheinend keinen Unterschied. Die Safari-Variante für den Desktop nutzt, wie diverse Browser, die Google Safe-Browsing-API, um potenzielle Phishingwebsites zu identifizieren. Es ist hier ganz klar zu sagen, dass eine Inkonsistenz in der Erkennung von schädlichen Websites noch gefährlicher für den Endnutzer ist als gar keine Erkennung. Denn bei gar keiner Erkennung kann sich der Nutzer nicht in falscher Sicherheit wiegen, was automatisch der Fall ist, wenn die Erkennung eine Inkonsistenz wie in dem hier aufgezeigten Fall aufzeigt.

Anfang November 2009 wunderten sich etliche iPhone-Nutzer, die mit ihrem Telefon über das UMTS-Netz online gingen, über eine angezeigte Warnmeldung. Diese teilte dem Nutzer mit, dass sein iPhone gehackt worden sei. Für die Lösung des Problems forderte der Angreifer 4,95 USD. Ursache für diesen Sicherheitsvorfall war ein bereits bekanntes Problem mit gleichen Passwörtern für die Nutzeraccounts root und mobile, das aber nur auf iPhones mit Jailbreak vorhanden war. Wenn ein Nutzer nach dem Jailbreak einen SSH-Server installierte, konnte ein Angreifer diesen ausnutzen, um von außen auf das verwundbare iPhone zu gelangen. Der Netzwerkscanner nmap konnte das iPhone zudem über TCP-Port 62078 eindeutig identifizieren. Der Angreifer scannte die IP-Adressbereiche der UMTS-Netze und fand auf diese Weise verwundbare iPhones, auf die er das Hintergrundbild mit der Warnmeldung kopierte. Interessant ist dieser Vorfall vor allem deswegen, weil hier das erste Mal in größerem Umfang Sicherheitsprobleme von iPhones über ihre Mobilfunkverbindung automatisiert ausgenutzt wurden.

Die Sicherheit der Android-Plattform stand im Fokus der *Hack-in-the-Box-Konferenz* Anfang Oktober 2009, wo sich Sicherheitsexperten einig darüber waren, dass Android-Nutzer vermehrt darauf achten sollten, welche Anwendungen sie auf ihrem Smartphone installieren.[285] Zudem wurde vor bösartigen ROMs gewarnt, die das System von Grund auf kompromittieren könnten. Dieses Problem besteht vor allem bei sogenannter *gecrackter* Software, die illegal von Quellen im Netz oder über P2P-Netzwerke bezogen werden kann. Da die eigentliche Quelle dieser Softwarepakete nicht vertrauenswürdig ist, sollte man auf Software aus solchen

---

[283] http://support.apple.com/kb/HT3754.

[284] http://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2009-2204.

[285] http://sg.codeandroid.org/2009/10/13/hack-in-the-box-coverage-dangers-of-customized-android-roms-and-malware/.

verzichten. Ein Entwickler mit bösartigen Absichten könnte ein präpariertes ROM im Netz zum Download anbieten, welches einen modifizierten Kernel besitzt und somit in der Lage wäre, Tastatureingaben aufzuzeichnen, private Informationen abzugreifen und als Trojaner auf dem Smartphone zu agieren. Auf der anderen Seite haben modifizierte ROMs auch viele Vorteile wie die Möglichkeit, das System nach eigenen Vorstellungen zu modifizieren. Der Android Forensiker Andrew Hoog beschreibt[286] in seinem White Paper zu Android Forensik, dass der interne Android-Browser sensible Daten wie Nutzernamen, Passwörter und zugehörige URL im Klartext speichere.

Auch Blackberry Smartphones sind mittlerweile im Fokus von Angreifern. Im Oktober 2009 sorgte eine Software namens PhoneSnoop für Schlagzeilen, die für Awareness-Zwecke von einem Blogger mit dem Namen Sheran Gunasekera programmiert wurde.[287] Die Software installiert lediglich einen sogenannten PhoneListener-Service, der auf eingehende Anrufe von einer spezifischen Nummer wartet. Diese Nummer kann vorher vom Angreifer festgelegt werden. Im Falle, dass mit dieser Nummer der Blackberry des Opfers angerufen wird, erkennt der PhoneListener-Service die Nummer, hebt automatisch ab und stellt das Telefon in den SpeakerPhone-Modus. So kann der Angreifer, ohne vom Opfer direkt bemerkt zu werden, hören, was im Umfeld des Telefons gesprochen wird. RIM, der Hersteller des Blackberry, musste im September 2009 zusätzlich eine potenzielle Lücke für Phishingangriffe auf das Blackberry Smartphone patchen.[288] Mit einem bösartig modifizierten Zertifikat auf einer Website konnte ein Angreifer dem Nutzer des Blackberry vortäuschen, er würde sich auf einer vertrauenswürdigen, gesicherten Website befinden.

Abschließend ist zu sagen, dass das größte Sicherheitsproblem im Bereich mobiler Plattformen nach wie vor der Mensch ist. Eine Studie des Sicherheitsunternehmens TrendMicro zeigte im September 2009 auf, dass Nutzer von Smartphones sich nicht den Gefahren bei der Benutzung von mobilen Plattformen bewusst sind.[289] Anscheinend sind viele Nutzer nach wie vor der fälschlichen Meinung, dass sie bei der Benutzung von Smartphones den gängigen Sicherheitsrisiken anderer Plattformen nicht ausgesetzt sind. Fast 50 % der Studienteilnehmer gaben an, sich nicht wirklich Gedanken um ihre Sicherheit beim Surfen im Netz mit dem Smartphone zu machen. Eine größere Angst bestehe davor, durch Verlust oder Diebstahl des Telefons Datenverlust zu erleiden. Das Schockierende an den Ergebnissen der Studie war aber, dass etwa 45 % der Studienteilnehmer zugaben, bereits Opfer von Malwareangriffen auf ihr Smartphone gewesen zu sein.

Grund zur Panik besteht derzeit jedoch nicht, da die Hochkonjunktur mobiler Malware noch nicht angebrochen ist. Allerdings ist nachweislich erkennbar, dass die Anzahl von Würmern, Viren und Trojanern für mobile Endgeräte in den letzten

---

[286] http://viaforensics.com/android-forensics/android-browser-stores-passwords-sensitive-data-plain-text.html.

[287] http://chirashi.zensay.com/2009/10/remote-listening-for-the-blackberry/.

[288] http://blogs.zdnet.com/security/?p=4500.

[289] http://trendmicro.mediaroom.com/file.php/96/Trend±Smart±Smartphone±Report.ppt.

Jahren stetig gestiegen ist. Dies könnte daran liegen, dass nach wie vor keine stabile Weiterverbreitungsroutine gefunden wurde. Dies liegt sicherlich auch an der sich noch entwickelnden Verbreitung von Smartphones mit offenen Betriebssystemen. Von einem Monopol seitens Symbian kann trotz großer Marktanteile nicht geredet werden. Für eine effektive Malwareszene fehlt hier also ein Monopolist, so wie Microsoft im Desktopbereich.

Die Antivirenindustrie rüstet sich allerdings für den zukünftigen Kampf gegen Schadsoftware auf mobilen Plattformen. Dass Smartphones massenweise von Epidemien heimgesucht werden, ist nur eine Frage der Zeit. So haben Hersteller wie F-Secure, Kaspersky oder TrendMicro bereits Scanner für die gängigen Smartphone-Betriebssysteme entwickelt. Mit der zunehmenden Nutzung des Smartphones als Bezahlmittel werden diese sicherlich mehr und mehr in den Fokus von Angreifern rücken.

d) Spielekonsolen

Bei den Spielekonsolen lassen sich die Angriffsziele auf die drei größten Netzwerkplattformen beschränken: *PlaystationNetwork* (PSN)[290], *Xbox-Live*[291] und *WiiConnect24*[292]. Formen des Identitätsdiebstahls auf Spielekonsolen tauchten erstmalig auf, als die einzelnen Hersteller große Internetplattformen für ihre Konsolen entwickelten. Über diese Plattformen ist es für Konsolenspieler möglich geworden, gegen Spieler anderer Länder in verschiedenen Spielekategorien anzutreten, Erfahrungen auszutauschen oder sich digital zu vernetzen.

Die älteste Plattform ist die aus dem Hause Microsoft stammende Xbox-Live-Plattform, welche im Jahre 2002 gegründet wurde. Die wichtigste Funktion des Netzwerks stellt die Fähigkeit dar, online gegen Spieler auf der ganzen Welt spielen zu können, sofern das Spiel eine entsprechende Funktion anbietet. Des Weiteren kann ein Spieler seine Erfolge einem virtuellen Profil zuweisen und sich so mit Freunden vergleichen. Der Xbox-Live-Marktplatz bietet die Möglichkeit, sowohl kostenlos als auch kostenpflichtig Demos, Trailer und Addons herunterzuladen.

Der Nutzer kann mit einer Xbox 360 über zwei verschiedene Zugänge auf das Xbox-Live-Netzwerk zugreifen, die sich in Umfang und Funktion teilweise unterscheiden. Xbox-Live-Silber stellt den grundlegenden Zugang zu Xbox-Live dar und steht jedem Spieler kostenfrei zur Verfügung. Xbox-Live-Gold ist kostenpflichtig und kann entweder per Kreditkarte, Bankeinzug oder mit einer Guthabenkarte für drei oder zwölf Monate im Voraus bezahlt werden.

Durch die potenzielle Nutzung einer Kreditkarte sind vor allem Xbox-Live-Gold-Nutzer zum Ziel von Identitätsdieben geworden. Meist werden die Nutzer über persönliche Nachrichten oder bösartige Links in Gamer-Foren und -Portalen auf präparierte Websites gelockt, auf denen sie anschließend zur Eingabe ihrer Xbox-Live-

---

[290] http://de.playstation.com/psn/.

[291] http://www.xbox.com/de-DE/live/bestoflive/connectnow.htm.

[292] http://wiiportal.nintendo-europe.com/1351.html.

Zugangsdaten verleitet werden. Der Angreifer macht sich hierbei die Methoden des Social Engineering (siehe hierzu auch S. 96 ff.) zunutze und verspricht den Opfern etwa 2.000 Microsoft-Punkte. Es werden also nicht direkt Bankdaten angefordert, sondern lediglich der Name (Gamertag) des Nutzers, das Passwort und die dazugehörige E-Mail-Adresse. Mithilfe dieser Informationen kann sich der Angreifer anschließend im Namen des Nutzers am Xbox-Live-System anmelden und dort weitere persönliche Informationen, wie vollständiger Name, Adresse und Kreditkartendetails, auslesen. Microsoft-Punkte sind hierbei die Währung, die beispielsweise zum Kauf von Spielen und Inhalten auf dem Xbox-Live-Netzwerk genutzt werden kann. Sie ist aber unter anderem auch für die *Windows Live Gallery*[293] und *Zune Onlineshops*[294] gültig. Neben dem Abgreifen von Kreditkarteninformationen und persönlichen Informationen stellen diese Microsoft-Punkte sicherlich einen weiteren Anreiz für Angreifer dar. Denn mit ihnen kann man beispielsweise Geschenke für befreundete Spieler einkaufen. Ein Angreifer kann mithilfe eines übernommenen Xbox-Live-Accounts alle Microsoft-Punkte des Kontos in Geschenke für sich oder andere investieren und anschließend das Account kündigen.

Aber nicht nur über präparierte Phishingwebsites versuchen Angreifer Xbox-Live-Nutzern ihre Accounts zu stehlen. Auf der Onlineplattform YouTube werden schon seit längerem Videos propagiert, die einen sogenannten *Xbox Point-Generator* in Aktion zeigen. Es handelt sich hierbei um ein kleines Programm, das behauptet, kostenlos Microsoft-Punkte zu erstellen. Für die Generierung solcher kostenlosen Microsoft-Punkte ist jedoch das Nutzeraccount des Spielers notwendig. Sobald das Opfer die Accountdaten eingegeben hat, werden diese an den Angreifer übertragen.

Die PSN-Plattform ist eine Onlineplattform für Multiplayerspiele und digitalen Content, die im Jahre 2006 von Sony gegründet wurde. Das Angebot steht den Nutzern der Playstation 3-Konsole und der Playstation Portable zur Verfügung. Das Vertriebskonzept und das Nutzermanagement des PSN sind denen der Xbox-Live-Plattform sehr ähnlich – es wird auch hier ein Account benötigt, der mit Kreditkarteninformationen ausgestattet werden kann, um kommerzielle Inhalte konsumieren zu können.

Auch im Falle des PSN werden unbedarfte Nutzer mit kostenfreien Zusatzpunkten angelockt und dazu verleitet, ihre persönlichen Zugangsdaten freizugeben. Dabei läuft die Masche haarscharf genauso ab wie bei der Xbox-Live-Plattform. Den Opfern werden kleine Programme zum Download angeboten, die vorgeben, bei Eingabe des PSN-Accounts neue PSN-Prepaid-Karten zu generieren. Allerdings wird das PSN-Account bei Eingabe direkt an den Angreifer übertragen, und dem Opfer wird lediglich ein nutzloser Code ausgegeben, der keinerlei Bedeutung besitzt.

Eine weitere Methode von Angreifern besteht darin, unbedarfte Nutzer als Beta-Tester für Spiele anzuwerben, die noch nicht auf dem Markt sind.[295] In der Abb. 31 ist eine solche Nachricht zu sehen, die in einem Onlineforum für PSN-Nutzer pub-

[293] http://gallery.live.com.

[294] http://www.zune-online.com.

[295] Für einen beispielhaften Fall siehe etwa: http://www.anotha.com/f162/ive-been-selected-to-beta-test-3-ps3-t63900/.

Congrats!!! You have been invited by a Sony Mod to Sony's Beta Center. Here you will be required to do one thing and you can choose three betas from our list of choices below.

Darksiders
Guitar Hero 5
DiRT 2
G-Force
Final Fantasy XIII
MAG
Tekken 6
God of War III
Mafia II
Need for Speed SHIFT
The BIGS 2

Abb. 31 Auszug einer Phishingnachricht in einem Onlineforum für PSN-Spieler

liziert wurde. Die genauen Anweisungen des Phishers sind in der Abb. 32 zu sehen. Das Opfer wird hier unter anderem aufgefordert, eine präparierte Website aufzurufen und sich dort mit dem PSN-Account einzuloggen. Nach Betätigung des Submit-Buttons werden die Accountdaten an den Phisher übertragen, der, wie im Falle der Xbox-Live-Accounts, versuchen wird, persönliche Informationen und Kreditkartendaten aus den gephishten Accounts auszulesen. Laut einer Pressemeldung von Sony[296] kann ein Angreifer aber mit dem PSN-Account nicht die eingetragenen Kreditkarteninformationen auslesen.

Im Frühjahr 2009 wurden die ersten Scam-Angriffe für die Wii entdeckt. Die Währung der Wii nannte sich ursprünglich Wii-Punkte, wurde aber nach der Einführung des Nintendo DSi und dem Download-Angebot auf Nintendo-Punkte umbenannt. Die Nintendo-Punkte können als Währung im Wii- und Nintendo DSi-Shop genutzt und beispielsweise durch Belastung einer Kreditkarte erstanden werden.

Requirements

1st. Sign up to the Website below. Use the llink below.

2nd. You need to have one offer confirm.

3rd. You need more than 1 cent, not including the $1 dollar bonus in your July earnings.

4th. You must do the offer for us to send you the a beta code. Please read everything below!

Sign Up Here

Abb. 32 Auszug von Anweisungen eines PSN-Phishers in einem Onlineforum

[296] Quelle: http://uk.playstation.com/home/news/articles/detail/item98438/Notice-to-PLAYSTATION%C2%AENetwork-Users/.

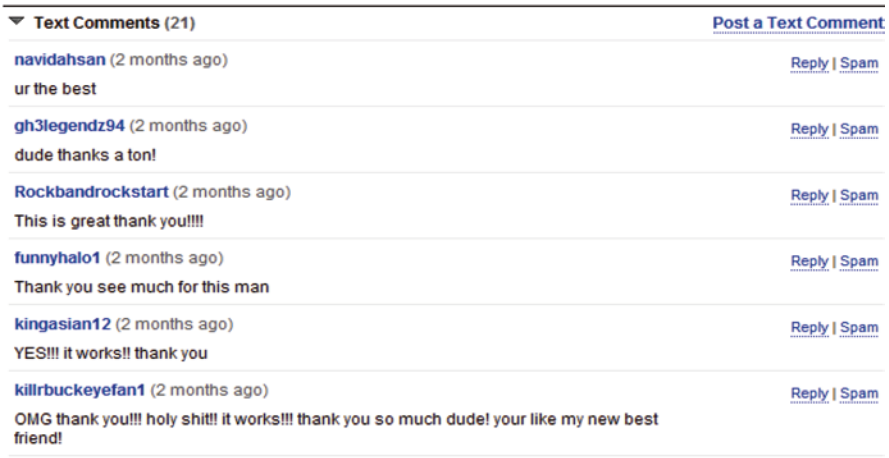


Abb. 33 Positive Kommentare zu Schadsoftware auf YouTube

Der Angriff auf die Nintendo-Punkte war, wie auch bereits bei Xbox-Live und PSN, ein sogenanntes *Generatorprogramm*, welches vorgibt, entsprechende Punkte kostenlos zu erstellen. Ein wesentlicher Unterschied zu den Generatorprogrammen der anderen Plattformen existiert allerdings. Das Programm *Wii Points Generator* infiziert das Endsystem des Nutzers mit einer Backdoor namens *Bifrose*[297], welche Angreifern die Kontrolle über das System übergibt. Die Verbreitung der Malware wurde auch im großen Stil über die Plattform YouTube betrieben, die bis heute keine effiziente Lösung für das Problem der Publikation von Schadprogrammen über ihre Plattform gefunden hat. Denn für die Meldung krimineller Inhalte gibt es auf YouTube nach wie vor keine geeignete Möglichkeit. Des Weiteren wird durch geschicktes Kommentieren der entsprechenden Einträge versucht, potenzielle Opfer von der Seriosität des Angebotes zu überzeugen. In der Abb. 33 sind einige dieser positiven Kommentare zu sehen.

Das grundlegende Problem ist dem anderer Plattformen und Systeme ähnlich: Der Druck der Spielergemeinschaft bringt die Konsolenhersteller in immer größere Zwänge, neue Software und Hardware in kürzeren Abständen auf den Markt zu bringen. Problematisch ist hierbei, dass eine saubere und sichere Implementierung unter diesen Umständen nicht mehr möglich ist. Es entsteht also ein ähnliches Phänomen, wie wir es aus der Softwareentwicklung für Desktopanwendungen bereits kennen. Durch unsichere Implementierung ist es unter Umständen möglich, die laufende Software der Konsole mit Schadcode zu kompromittieren. Im August 2007 berichtete der Antivirenhersteller Symantec über mögliche Sicherheitslücken in den Konsolen Wii und PS3[298]. Demnach können Sonys PS3 und die Nintendo Wii durch

[297] http://www.sophos.com/security/analyses/viruses-and-spyware/malbifrosex.html?_log_from=rss.

[298] http://www.techradar.com/news/gaming/consoles/wii-and-ps3-vulnerable-to-hacks-and-phishing-161313.

Schadsoftware und Phishingangriffe kompromittiert werden. Für die PSP von Sony gibt es bereits als Unlocker getarnte Schadsoftware, die bei Ausführung auf der PSP diese irreparabel zerstört.

Aufgrund der Anbindung an das Internet und der Möglichkeit, Geschäfte durchzuführen, sind Spielekonsolen heute ebenfalls ein lohnendes Angriffsziel. Die Tatsache, dass die meisten der heutigen Spielekonsolen ständig ans Internet angebunden sind, macht sie zu einem interessanten Ziel für Angreifer aller Art. Das Gefahrenpotenzial, welches von solchen Spielekonsolen ausgeht, ist nicht zu unterschätzen. Konsolenhersteller sollten diese Tatsache in ihre zukünftigen Produktentwicklungsprozesse mit aufnehmen.

Die Konsole Wii beispielsweise ist standardmäßig mit einem voll funktionsfähigen Browser ausgestattet, allerdings ohne zusätzliche Schutzmaßnahmen wie Schutz vor potenziellen Phishingwebsites. Fraglich ist hierbei auch, inwiefern Wert auf eine sichere Implementierung des Browsers gelegt wurde. Laut Symantec gilt die Xbox 360 aus dem Hause Microsoft als sicherste der drei großen Konsolen.

2. Neue Computing-Paradigmen: Browsertechnologien

Prognose: Der Browser spielt als zentrales Werkzeug in fast allen neuen Computing-Paradigmen eine herausragende Rolle. Identitätsdaten werden im Browser gespeichert oder über den Browser eingegeben. Der Schutz dieser Daten muss in Zukunft verbessert werden.

Im Browser gibt es drei zentrale Sicherheitsparadigmen: Sandboxing, Informationsflusskontrolle (mittels der Same Origin Policy) und SSL/TLS.

Sandboxing

Durch verschiedene Sandboxing-Mechanismen für Java-Applets und geladenen JavaScript-Code soll verhindert werden, dass über aus dem Internet geladene Codefragmente persönliche, auf dem PC des Internetnutzers gespeicherte Daten gelesen werden können.

Dieses Paradigma verliert zunehmend an Bedeutung, weil die wichtigen persönlichen Daten zunehmend im Browser gespeichert werden:

- Passwörter werden im Passwortmanager des Browsers gespeichert.
- Clientzertifikate und die dazugehörigen privaten Schlüssel werden im PKCS#11-Modul des Browsers gespeichert.
- Authentifizierungsdaten wie HTTP-Cookies oder SAML-Assertions werden im Document Object Model des Browsers dauerhaft oder vorübergehend gespeichert und können dort über die unter II.5. beschriebenen browserinternen Angriffe ausgelesen werden.

Außerdem wird das Sandboxing durch die Installation zahlreicher Plug-ins ausgehöhlt, die lokale Zugriffsrechte besitzen: Wenn ein Angreifer Zugriff auf lokale Ressourcen des PCs benötigt, wird er den Nutzer einfach bitten, ein Plug-in zu

installieren. Dies kann mithilfe von Social Engineering getarnt erfolgen, z. B. als angebliches Update für einen Flash Player.

Same Origin Policy

Die Same Origin Policy (SOP) ist von ihrem Ansatz her ein äußerst sinnvoller Versuch, eine Informationsflusskontrolle im Browser zu erzwingen. Die Idee ist hierbei, nur solche aktiven (JavaScript) und passiven Inhalte miteinander interagieren zu lassen, die von dem gleichen Webserver geladen wurden. Die SOP legt dabei folgende Parameter aus der URL zugrunde: das Protokoll (z. B. http oder ftp), den Domainnamen und die Portnummer (die entweder explizit angegeben sein kann oder sich aus dem Protokoll ergibt).

Die Abbildung 34 stellt an einem fiktiven Beispiel dar, wie die SOP funktioniert: Im Browser sind gleichzeitig eine Onlinebanking-Seite und die Seite des Angreifers geladen. Der Angreifer hat in seine Seite ein Skript eingebettet, das versucht, die Kontonummer für die aktuelle Transaktion zu verändern. Dies wird von der SOP unterbunden, weil zwar Protokoll (https) und Portnummer (443) gleich sind, die Domainnamen sich aber unterscheiden.

Dieses Konzept wurde aber sukzessive aufgeweicht [Zalewski09] und wird im Moment in der Forschung neu definiert [Gazelle]:

- Um eine Lastverteilung zwischen mehreren Servern zu erlauben, wurde die SOP auf dem Konzept einer Domain aus dem DNS aufgebaut. Inhalte von Servern, die zur gleichen Domain gehören (ftp.example.org, images.example.org und www.example.org gehören demnach alle zur Domain example.org). Die Definition der Domain kann unter gewissen Randbedingungen von einem der Server vorgenommen werden [Zalewski09].
- Für verschiedene im DOM des Browsers gespeicherte Objekte wurden unterschiedliche SOPs entwickelt. So wurde z. B. die Cross-Domain-Übertragung von HTTP-Cookies, die ursprünglich erlaubt war, verboten. Die Cross-Domain-

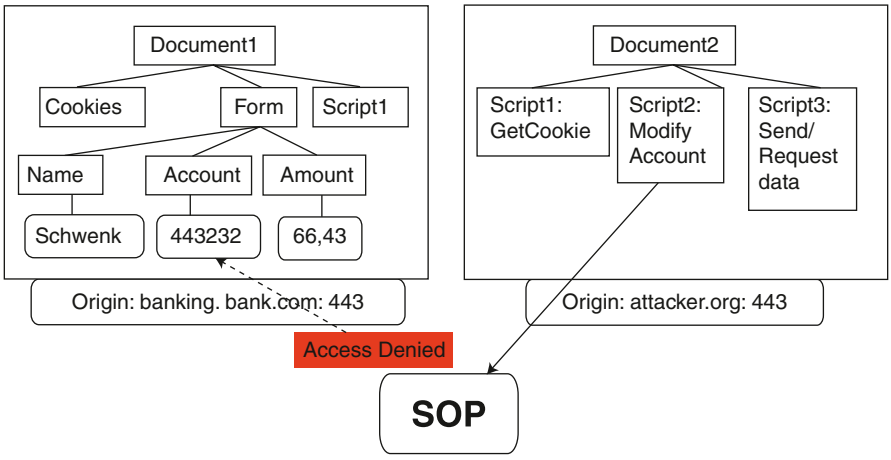


Abb. 34 Funktionsweise der Same Origin Policy

Übertragung von Daten, die in einem HTML-Formular gespeichert sind, ist aber weiterhin erlaubt.

- Durch die Einführung von mehreren, gleichzeitig geöffneten Browserfenstern und später von Tabs wurden die Inhalte verschiedener Webanwendungen im gleichen Prozess auf dem PC, und damit im gleichen Adressraum, ausgeführt. Die Informationsflusskontrolle ging somit auf der Ebene des Prozesses verloren. Seit der Veröffentlichung von Google Chrome ist die strikte Trennung von Tabs in verschiedene, durch eine Sandbox gesicherte Prozesse ein Forschungsthema.
- Neue Entwicklungen im Bereich Web 2.0 machen eine strikte Anwendung der SOP unmöglich. So werden z. B. Mashups zunehmend beliebter. Hierbei handelt es sich um Websites, die Inhalte von verschiedenen Websites kombinieren (z. B. die Einbettung von Google Maps in die Webanwendung eines Immobilienmaklers).

SSL/TLS

Das Transport Layer Security-Protokoll (der Name „Secure Socket Layer" stammt von der Firma Netscape und wurde mit der Standardisierung durch die IETF mit Version 3.1 aufgegeben) kann dazu eingesetzt werden, die über eine TCP-Verbindung übertragenen Bytes zu verschlüsseln und ihre Integrität zu sichern.

Der große Erfolg von TLS beruht vor allem darauf, dass in fast allen bisherigen Einsatzszenarien der normale Internetnutzer nichts tun musste, damit TLS funktionierte. Lediglich der Systemadministrator des Webdienstes musste hier tätig werden.

Die Intelligenz des TLS-Protokolls steckt im TLS-Handshake, der in der Abb. 35 dargestellt ist. Ein einfacher Drei-Wege-Schlüsselaustausch, meist auf Basis des RSA-Algorithmus (in der Abbildung dargestellt durch die Briefkästen), wird umrahmt von Protokollnachrichten, die ein automatisches Aushandeln der kryptogra-

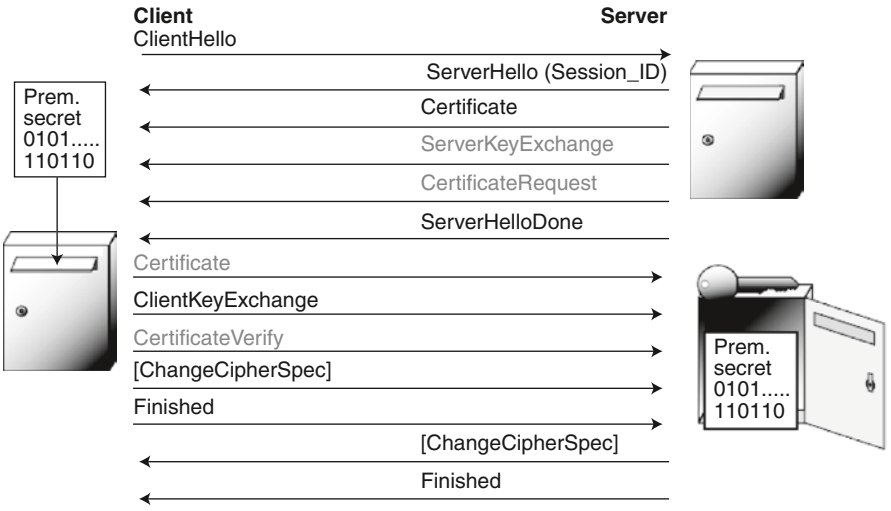


Abb. 35 Das SSL/TLS Handshake-Protokoll

fischen Algorithmen ermöglichen und das Handshake-Protokoll gegen alle bekannten Angriffe absichern.

In seiner am häufigsten eingesetzten Form bleibt der Browser für den Server vollständig anonym. In diesem Fall werden die grau dargestellten Nachrichten nicht benötigt. Die Sicherheit dieser Einsatzform beruht auf der Struktur und Leistungsfähigkeit der im Browser verankerten PKIs und der Fähigkeit des Nutzers, eine kryptografische Fehlermeldung korrekt zu interpretieren. Beide Voraussetzungen sind, wie auf S. 87 ff. dargestellt, nach den Erfahrungen aus den seit 2004 stattfindenden Phishingangriffen nicht zu halten.

3. Neue Computing-Paradigmen: Servertechnologien

Prognose: Mächtige Clientanwendungen wie Textverarbeitung oder Tabellenkalkulation werden in naher Zukunft abgelöst von entsprechenden Serveranwendungen. Diese Serveranwendungen sind jeweils Einzellösungen und werden voraussichtlich viele individuelle Schwachstellen aufweisen. Die Verantwortung für die Absicherung der Anwendung wird sich von großen, mächtigen Softwarekonzernen (z. B. Microsoft) hin zu kleinen Firmen verlagern. Da diese die Sicherheit ihrer Applikationen kommerziell nicht mehr gewährleisten können, müssen standardisierte Lösungen gefunden werden.

Multi-Tier-Architekturen

Moderne Serveranwendungen laufen nicht innerhalb nur eines Rechners ab, sondern sind physikalisch und logisch untergliedert. Diese Unterteilung umfasst typischerweise folgende Komponenten:

- Load Balancer: Ein Rechner nimmt HTTP-Anfragen zentral entgegen und verteilt diese nach einem möglichst optimalen Algorithmus auf die einzelnen Server der Serverfarm. Ein SSL-Tunnel muss in der Regel hier terminieren, damit Verteilungsstrategien angewandt werden können, die mehr als nur die IP-Adresse des Anfragenden berücksichtigen.
- Web Frontend: Dieser Rechner hat die Aufgabe, HTTP-Anfragen möglichst effizient (d. h. mit Rückgriff auf Inhalte im Cache) zu beantworten und unter Berücksichtigung der Antwort der Applikationslogik die Antwort-HTML-Seite und den HTTP-Header der Antwort zusammenzustellen.
- Applikationslogik: Hier werden Berechtigungen überprüft, Datenbankanfragen generiert und Berechnungen durchgeführt. Dieser Teil der Serveranwendung wird meist individuell entwickelt, hier können gravierende Sicherheitslücken bestehen.
- Datenbankanwendung: Alle persistenten Daten werden in einer Datenbank gehalten, die zentral aktualisiert werden kann. Die Applikationslogik muss hier geeignete SQL-Anfragen erzeugen und ggf. die Autorisierung dieser Anfragen nachweisen.

Eine feinere Aufteilung der Aufgaben ist möglich. Aber bereits diese Standardarchitektur offenbart Probleme beim Einsatz moderner Technologien des Identitätsschutzes:

- Passwörter sind sehr gut geeignet, um eine Authentifizierung in einer Multi-Tier-Architektur abzubilden: Die gültigen Nutzername-/Passwortpaare werden einfach in einer Datenbank oder in einem LDAP-Directory zusammen mit den jeweiligen Berechtigungen abgelegt, und bei jeder Anfrage können Web Frontend, Applikationslogik und Datenbankanwendung leicht das der jeweiligen Anfrage mitgegebene Nutzername-/Passwortpaar an dieses LDAP-Directory senden und erhalten die passenden Berechtigungen zurück.
- Die Probleme der Integration anderer Authentifizierungsmethoden sind dagegen in der Praxis noch nicht gelöst. So endet z. B. die starke Authentifizierung über ein SSL-Clientzertifikat in der oben skizzierten Architektur bereits am Load Balancer. Es reicht nun nicht mehr, die Anfrage einfach nur weiterzuleiten, sondern der Load Balancer müsste aus dem Clientzertifikat einen Wert generieren, der als Schlüssel für die in der LDAP-Directory gespeicherten Berechtigungen dient, und diesen Wert zusammen mit der HTTP-Anfrage weiterreichen. Ähnliche Probleme ergeben sich bei der Integration chipkartenbasierter Authentifizierungsmethoden.

REST-basierte Dienste

Die Abkürzung REST steht für Representational State Transfer[299]. Sie beschreibt eine Alternative zu SOAP und XML-RPC als Basis für Webservices.

REST-basierte Webanwendungen sind schon seit einigen Jahren erfolgreich im Einsatz. Sie basieren auf HTTP als Kommunikationsprotokoll, mit den beiden einfachen Methoden GET und POST. Auf Clientseite werden HTML und JavaScript eingesetzt, auf Serverseite Skriptsprachen (PHP), Java oder andere objektorientierte Entwicklungsumgebungen (.NET).

REST-Anwendungen sind im Massenmarkt sehr erfolgreich, da ihre Entwicklungsparadigmen gut zu verstehen und die erforderlichen Softwarekomponenten oft frei verfügbar sind und gut skalieren.

Das größte Problem bei diesem Paradigma besteht in der unstrukturierten Art der Datenübertragung vom Client zum Server: Sowohl GET als auch POST erlauben nur eine einfache, schwach strukturierte Übertragung von Daten in der Form „Name=Wert“, wobei „Name“ und „Wert“ jeweils nur Strings sein dürfen. Prinzipiell sind damit alle Arten von Daten übertragbar, aber durch die erforderliche Codierung wird die Suche nach bösartigen Stringbestandteilen, die Angriffe wie XSS oder SQL-Injection auslösen können, erschwert.

Die Beschränkung auf HTTP-GET und -POST gilt nur für die Verbindung Client–Web-Frontend. Danach müssen die Daten in der Regel umcodiert werden, bevor sie weiterverarbeitet werden können. (Einfaches Beispiel: In PHP wird jedes

[299] http://en.wikipedia.org/wiki/Representational_State_Transfer.

Paar „Name=Wert“ in eine Variable mit dem Namen „$Name“ und dem Inhalt „Wert“ umgewandelt.)

SOAP-basierte Dienste
Durch die Einführung von XMLHttpRequest in den modernen Browsers und von SOAP zur Kommunikation zwischen den Servern wurde die Möglichkeit geschaffen, Daten durchgehend stark strukturiert und typisiert zu übertragen. Die Verwendung von XML ermöglicht hier eine komplexe Strukturierung, eine starke Validierung mittels XML Schema und komplexe Kommunikationsmuster zwischen den Servern dank SOAP.

Eine räumliche Trennung der einzelnen Tiers auf der Serverseite und eine noch stärkere Modularisierung der Serveranwendung kann erstmals praktisch im großen Maßstab umgesetzt werden. Frühere Initiativen in diese Richtung waren entweder plattformgebunden (.NET), an eine Programmiersprache gekoppelt (JAVA RMI) oder schlecht unterstützt (CORBA).

Die sich hieraus ergebenden Sicherheitsprobleme werden im Abschnitt zu serviceorientierten Architekturen noch näher erläutert.

4. Neue Computing-Paradigmen: Kommunikationstechnologien

Prognose: Der Großteil der neuen Kommunikationstechnologien wird XML-basiert sein und auf einer HTTP/TCP/IP-Infrastruktur laufen. Die XML-Standards sind von so beispielloser Komplexität, dass diese selbst von Experten nicht mehr überblickt werden können. Daher gibt es hier zahlreiche Ansatzpunkte für Identitätsmissbrauch.

HTTP
In der Netzwerktechnik ist ein deutlicher Trend zu beobachten, immer neue Protokollschichten auf den bereits existierenden Protokollen aufzusetzen. Gab es vor einigen Jahren noch Netzwerkdienste, die nicht auf TCP/IP basierten, verschob sich der Fokus zunächst auf die Entwicklung immer neuer Protokolle, die TCP/IP oder UDP/IP als Grundlage mit unterschiedlichen Ports benutzten. Mit dem Siegeszug des WWW und der zunehmenden Abschottung vieler Ports durch (Personal) Firewalls (als Ergebnis zunehmender Angriffe auf diese Netzwerkdienste) wurden ähnliche Dienste auf Basis von HTTP entwickelt.

Neben der universellen Verfügbarkeit (Port 80 für HTTP ist in allen Firewalls offen) war die Einfachheit und Natürlichkeit des Kommunikationsparadigmas von HTTP der Grund für diesen Erfolg: Das Muster Frage – Antwort entspricht sehr dem menschlichen Kommunikationsverhalten und kann fast alle Standardsituationen der Datenübertragung abbilden.

HTTP ist zustandslos und daher performant und kann über neue HTTP-Headerzeilen sehr einfach erweitert werden.

Die große Gefahr bei HTTP besteht darin, dass dieses Kommunikationsmodell allzu unkritisch als das einzig mögliche angesehen wird. Als Beispiel seien hier

Cross-Site-Request-Forgery-Angriffe (CSRF) genannt, die in [Felten97] auch für den Identitätsmissbrauch genutzt wurden:

- Da HTTP zustandslos ist, speichern viele Webanwendungen eine erfolgreiche Authentifizierung persistent im Browser, z. B. in Form eines HTTP-Cookies. Bei jeder HTTP-Anfrage wird dieses Cookie mitgeschickt, um eine erneute manuelle Authentifizierung zu vermeiden.
- Dies erscheint zunächst sinnvoll, solange man annimmt, dass jede HTTP-Anfrage auf eine Aktion des authentifizierten Nutzers zurückgeht. Dies ist aber nicht der Fall. HTTP-Anfragen können auch automatisch generiert werden, und dies entzieht sich häufig der Kontrolle des Nutzers:
  - Beim Laden eines innerhalb eines <img>-Tags spezifizierten Bildes wird eine HTTP-GET-Anfrage an den Server generiert, der im „src“-Attribut spezifiziert ist. Diese HTTP-Anfrage darf beliebige „Name=Wert“-Paare im Query-String beinhalten. Der Webserver kann diese Anfrage nicht von einer über ein HTML-Formular generierten Anfrage unterscheiden.
  - HTML-Formulare dürfen beim Laden vom Server des Angreifers bereits ausgefüllt sein. Sie können als „hidden“ deklariert werden, sodass sie nicht angezeigt werden. Eine GET- oder POST-Anfrage an den im Formular spezifizierten Server kann durch ein beliebiges JavaScript-Event ausgelöst werden, z. B. durch „onload“ direkt nach Laden der Seite. Dieser seltsam anmutende Mechanismus wurde eingeführt, da für viele HTTP-basierte Webanwendungen eine Cross-Domain-Kommunikation erforderlich ist, diese aber aus „Sicherheitsgründen“ für besser zu kontrollierende Mechanismen wie HTTP-Cookies verboten wurde.
- Da in beiden Fällen ein ggf. vorhandenes Authentifizierungscookie automatisch mit gesendet wird, kann ein Angreifer so unter der Identität des Opfers, das nichts anderes getan hat, als eine fremde Website zu betrachten, agieren. In [Felten97] gelang es sogar, mit dieser speziellen Form des Identitätsmissbrauchs ein Bankkonto einer amerikanischen Bank leer zu räumen.

XML

Als erste Einführung in XML hat das World Wide Web Consortium[300] eine Liste mit zehn Punkten erstellt:[301]

- XML steht für strukturierte Daten.
- XML sieht ein wenig wie HTML aus.
- XML ist Text, aber nicht zum Lesen.
- XML ist vom Design her ausführlich.
- XML ist eine Familie von Techniken.
- XML ist neu, aber nicht so neu.
- XML überführt HTML in XHTML.
- XML ist modular.

[300] http://www.w3.org.

[301] http://www.w3c.de/Misc/XML-in-10-points.html.

Tab. 1 Ein einfaches XML-Dokument

```
<?xml version="1.0"?>
<conversation linefeed="yes">
Hier ist eine typische englische Konversation wiedergegeben:
<greeting style="italic">Hello, world!</greeting>
  <response style="bold">
      Stop the planet, I want to get off!
  </response>
</conversation>
```

- XML ist die Basis für RDF und das Semantic Web.
- XML ist lizenzfrei, plattformunabhängig und gut unterstützt.

Das kleine Beispiel aus Tab. 1 soll die Struktur von XML-Dokumenten veranschaulichen. XML wurde einerseits als Markup-Sprache für Textdokumente (wobei dieses Einsatzgebiet allerdings heute nur noch am Rande eine Rolle spielt) und als textbasierte, plattformunabhängige Spezifikationssprache für Datenformate (ASN.1 ist plattformunabhängig, aber nicht textbasiert) entwickelt. Besonderer Wert wurde auf eine konsequente Internationalisierung gelegt. D. h., „textbasiert" bedeutet nicht „ASCII-basiert", sondern als Defaultwert wurde der Zeichensatz UTF-8 verwendet, der alle international gebräuchlichen Schriftzeichen enthält.

Die erste Zeile aus Tab. 1 gehört nicht zum XML-Dokument selbst und taucht daher in der Abb. 36 nicht auf. Es handelt sich hierbei um eine Anweisung an den XML-Prozessor, und dies wird durch das „?" hinter der sich öffnenden spitzen Klammer angezeigt. Das Dokument selbst ist valide, d. h., öffnende und schließende Tags sind korrekt verschachtelt, und daher kann das Dokument in der Abb. 36 in Form eines Baumes wiedergegeben werden.

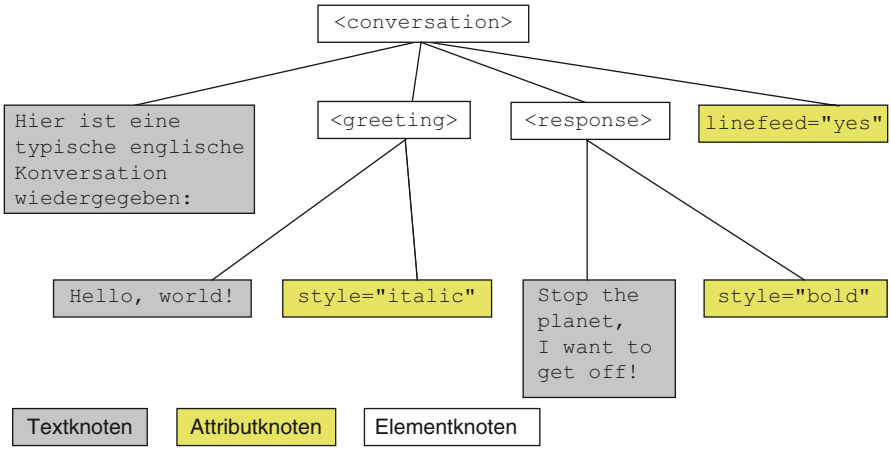


Abb. 36 Darstellung des XML-Dokuments aus Tab. 1 als DOM-Baum

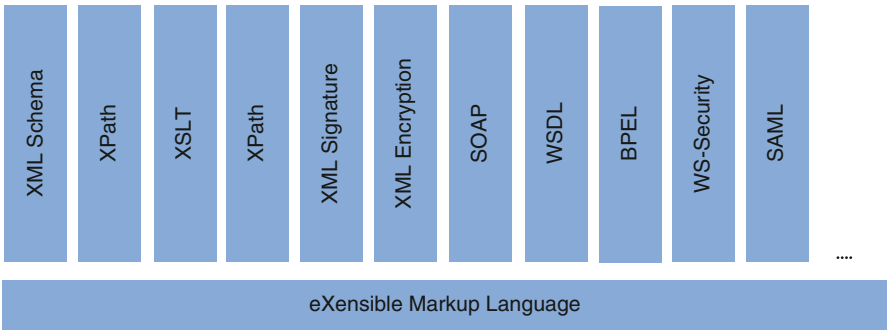


Abb. 37 Kleiner Ausschnitt der XML-Welt

Der Baum aus der Abb. 36 enthält mehrere Typen von Knoten. Nur die Elementknoten dürfen Nachfolger haben, sie sind somit das wichtigste Werkzeug zur Definition komplexer Datenstrukturen. Textknoten enthalten Werte, die in der Regel elementaren Datentypen wie Strings oder Zahlen entsprechen. Attributknoten bestehen wie bei HTML-Attributen aus einem „Name=Wert"-Paar.

Auf der Basis von XML wurde eine ganze Reihe von weiteren Standards definiert, die in XML eine wichtige Rolle spielen. Hier sollen nur kurz diejenigen Standards genannt werden, die schon zu Sicherheitslücken geführt haben:

- [XSLT] ist eine Turing-vollständige Sprache zur Transformation von XML-Dokumenten in eine andere Form (XML, HTML, …). In [Hill07] wurde gezeigt, wie man hier XSLT-Malware sogar in eine digitale Signatur einschleusen kann.
- Mit [XML Schema] soll eigentlich die Struktur von XML-Dokumenten überprüft werden. Da aber geschachtelte Typdefinitionen erlaubt sind, kann man hier sehr leicht DoS-Angriffe mit relativ kleinen Schemadateien ausführen.
- Mit der Web Service Description Language [WSDL 2.0] können die Schnittstellen, die Webservices nach außen hin anbieten, sehr genau beschrieben werden. Sie sind eine wichtige Basis für die Anwendbarkeit des SOAP-Paradigmas.
- Nach dem XML Signature-Standard [XML Signature] darf eine Datei beliebigen Transformationen unterworfen werden, bevor der Hash-Wert des Ergebnisses dieser Transformationen in die digitale Signatur einfließt. Es sind so relativ einfach Signaturen konstruierbar, die für jedes Dokument gültig sind, weil jedes Dokument vor der Hash-Wert-Bildung in einen konstanten Wert transformiert wird. Allerdings wurden schon 2005 von Michael McIntosh und Paula Austel (IBM TJ Watson NY) Möglichkeiten beschrieben, wie ein Angreifer signierte Daten im Netzwerk abfangen, manipulieren und den Empfänger schließlich dazu bringen kann, trotz Signaturprüfung gefälschte Daten zu verarbeiten [MA05]. *Diese gravierende Schwachstelle ist bis heute nicht behoben!*
- XML Signature erlaubt es, nur Teile eines XML-Dokuments zu signieren. Nahezu alle Implementierungen dieses Standards verwenden zur Referenzierung des signierten Teils ID-Attribute. Dies ermöglicht die in [MA05] beschriebenen Wrapping-Angriffe.

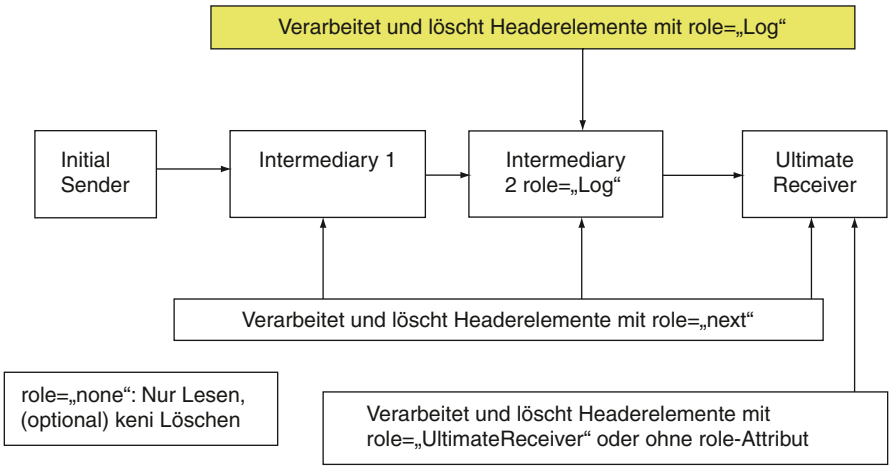


Abb. 38 Das „role“-Attribut zur Steuerung der SOAP-Kommunikation über mehrere beteiligte Instanzen

- Als weiterer wichtiger Standard ist XML Encryption [XML Encryption] zu erwähnen, der im Gegensatz zur Signatur eine einfachere Struktur besitzt und für den bislang keine gravierenden Sicherheitsprobleme bekannt geworden sind.

SOAP

SOAP [SOAP 1.2] ist das wichtigste Kommunikationsprotokoll in XML. Bewährte Paradigmen aus HTTP, E-Mail und anderen Protokollen werden übernommen, kombiniert und um weitere Features ergänzt.

- Die Struktur einer SOAP-Nachricht entspricht der bewährten Aufteilung in Header und Body (vgl. HTTP, E-Mail). Der Header enthält Metadaten, der Body die eigentlichen Nutzdaten.
- Im Gegensatz zu HTTP – und in Analogie zur E-Mail – ist SOAP prinzipiell in der Lage, eine Kommunikation zwischen mehr als zwei Parteien zu modellieren. Zwischen Sender und Empfänger können mehrere „Intermediaries“ agieren, die die SOAP-Nachricht in einem bestimmten Rahmen ändern dürfen (vgl. die Abb. 38).
- SOAP ist, analog zu HTTP, leicht erweiterbar: Der Header kann neue Funktionalitäten aufnehmen. Eine dieser Erweiterungen ist WS-Security; hierbei werden Felder für XML Signature, die Metainformationen zu XML Encryption und Security-Tokens in einem <Security>-Element im SOAP-Header aufgenommen.
- Da SOAP meist über HTTP gesendet wird und bei HTTP die Rollen von Client und Server fest und unveränderlich sind, wurde ein PAOS-Binding [PAOS] (PAOS = Reverse SOAP) eingeführt. Hierbei wird eine SOAP-Anfrage mit einer HTTP-Antwort und eine SOAP-Antwort mit einer HTTP-Anfrage geschickt.

WS-Security-Roadmap

Es ist IBM und Microsoft hoch anzurechnen, dass diese beiden Firmen schon sehr früh den riesigen Sicherheitsbedarf der neuen Ideen erkannt und eine Roadmap [Roadmap] zur Absicherung von Webservices vorgelegt haben. In dieser Roadmap werden alle wichtigen Sicherheitsaspekte angesprochen.

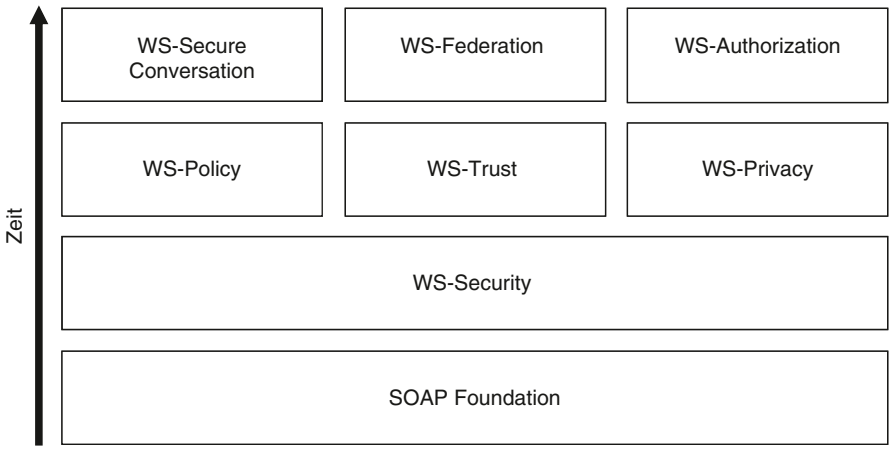


Abb. 39 Die IBM/Microsoft-Roadmap zur Absicherung von Webservices

Die grundsätzliche Struktur der Absicherung von SOAP-Nachrichten wird in [WS-Security] beschrieben. Bei diesem Standard geht es im Wesentlichen darum, die vielfältigen Möglichkeiten, Sicherheitselemente in einer SOAP-Nachricht unterzubringen, auf ein vernünftiges Maß zu reduzieren. Dazu wurde ein neues SOAP-Headerelement <Security> definiert, in dem alle Metainformationen zur Sicherheit der SOAP-Nachricht untergebracht sind. Existierende XML-Standards werden verwendet und nur leicht erweitert. Tabelle 2 gibt einen Überblick über die Struktur des <Security>-Headers und seine Position in der SOAP-Nachricht. Eine wesentliche

Tab. 2 Schematische Darstellung des <Security>-Headers aus WS-Security

```
<?xml version="1.0" encoding="utf-8"?>
<env:Envelope xmlns:env="...">
 <env:Header>
   <wsse:Security xmlns:wsse="..." xmlns:xenc="..." >
     <!-- Security Token -->
       <xxx:CustomToken wsu:Id="MyID" xmlns:xxx="...">
          FHUIORv...
       </xxx:CustomToken>
     <!-- XML Signature -->
       <ds:Signature>
          ...
       </ds:Signature>
     <!-- XML Encryption -->
       <xenc:EncryptedKey>
          ...
       </xenc:EncryptedKey>
   </wsse:Security>
 </env:Header>
 <env:Body wsu:Id="MsgBody"> ...</env:Body>
</env:Envelope>
```

Kritik an diesem Standard ist, dass er neben vielen sinnvollen Festlegungen auch eine sicherheitstechnisch verheerende Empfehlung enthält: Es wird empfohlen, zur Referenzierung der signierten Nachrichtenteile ID-Attribute einzusetzen. In XML Signature ist diese nur erlaubt, nicht empfohlen. Die war auch der Auslöser für die Publikation von [MA05].

[WS-Policy] löst das Problem der automatischen Aushandlung von (Security-)Policies relativ elegant: Policies werden als einfache boolesche Formeln beschrieben, in denen atomare Security-Bausteine nur durch AND, OR und Klammerung gegliedert sein dürfen. Dadurch ist es möglich, zwei Policies einfach zu vergleichen, indem sie in die disjunktive Normalform transformiert werden und dann die geklammerten Ausdrücke in beiden Normalformen auf identische Ausdrücke hin durchsucht werden. Wird ein solcher identischer Ausdruck gefunden, so gelten die in ihm enthaltenen Regeln als ausgehandelt. Eine große Sammlung von Bausteinen für Security Policies ist im Standard [WS-SecurityPolicy] enthalten.

[WS-Trust] beschreibt den Mechanismus des Trust Brokers, mit dessen Hilfe es z. B. Mitarbeitern der Firma A, in welcher Nutzername/Passwort als Authentifizierungsmechanismus zum Einsatz kommt, ermöglicht wird, sich mithilfe eines vom Trust Broker ausgestellten, temporären X.509-Zertifikats in Firma B für die Dauer eines Projekts zu authentifizieren. Dieses Konzept wird in [WS-Federation] auf lose föderierte Verbünde von Firmen erweitert.

[WS-SecureConversation] erweitert das WS-Security-Framework um sitzungsbasierte Sicherheit: Ein vorab ausgehandelter symmetrischer Schlüssel kann mittels dieses Standards in SOAP-Nachrichten referenziert und modifiziert werden.

Die beiden restlichen Standards der Roadmap fristen ein Schattendasein, dafür gewinnen folgende XML-basierte Standards zunehmend an Bedeutung:

- [XrML] wurde durch die Firma ContentGuard vor allen Dingen in die MPEG-21-Standardisierung eingebracht, um Rechte an digitalem Content zu beschreiben. Es findet sich im Standard ISO/IEC 21000-5 wieder.
- [XACML] hat im Bereich der Zugriffskontrolle an Bedeutung gewonnen. Man kann in dieser Sprache alle gängigen Access Control Policies abbilden, auch die gängigen Discretionary access control (DAC) und Role-based access control (RBAC).
- SAML: Siehe unten.

Als wichtigster Kritikpunkt ist an dieser Stelle die Tatsache zu nennen, dass die Implementierungen nicht mit der Standardisierung Schritt halten können. Dies hat zur Folge, dass insbesondere die Sicherheitsstandards nicht voll eingesetzt werden können.

Security Assertion Markup Language

Die Security Assertion Markup Language[302] [SAML] hat sich als *der* Standard herauskristallisiert, um Aussagen über Identitäten zu machen. Ihr soll daher hier ein

[302] http://www.oasis-open.org/committees/tc_home.php?wg_abbrev=security.

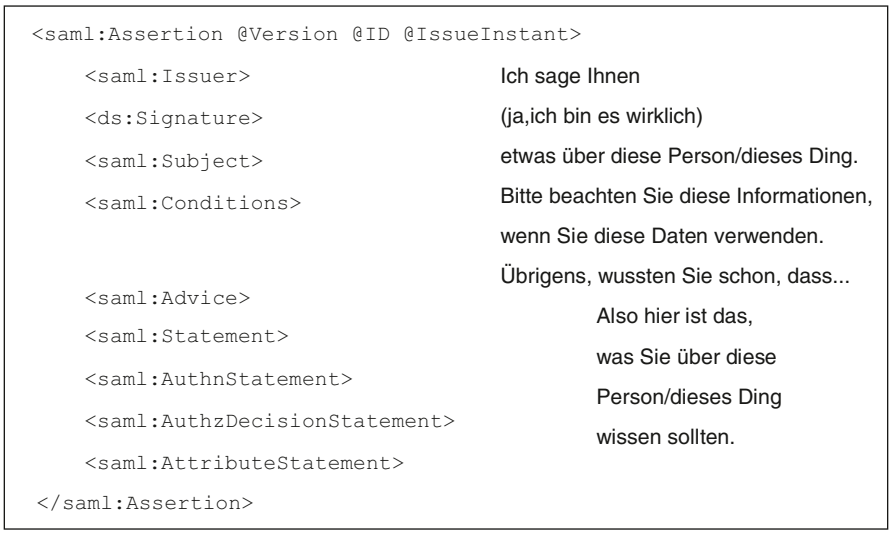


Abb. 40 Beispielhafter Aufbau einer SAML-Assertion („@“ bezeichnet ein Attribut)

eigener Abschnitt gewidmet werden. Eva Maler hat in [Maler06] versucht, SAML in 15 Wörtern zusammenzufassen:

> XML-based framework for marshaling security and identity information and exchanging it across domain boundaries.

Man kann SAML grob als äußerst flexible Erweiterung von X.509-Zertifikaten ansehen. Bekannte Felder aus X.509 wie „Issuer“, „Subject“, „NotBefore“, „NotAfter“, „Signature“ und auch Analoga zu Zertifikatserweiterungen findet man wieder (vgl. die Abb. 40). Darüber hinaus kann eine SAML Assertion aber auch weitere Funktionen übernehmen, z. B. die Rolle von Attributzertifikaten bei der Festlegung der Berechtigungen eines Subjekts oder eine maschinenlesbare Beschreibung der Sicherheits-Policy in <AuthnStatement>, nach der diese Assertion ausgestellt wurde.

Der Erfolg von SAML ist aber nicht allein aus der sorgfältigen Modellierung von Aussagen über Identitäten zu erklären. Darüber hinaus hat es sich die SAML-Arbeitsgruppe zum Ziel gemacht, praktisch anwendbare Use Cases für die wichtigsten Einsatzgebiete von SAML zu spezifizieren. Dies erleichtert die Akzeptanz in der Industrie enorm. Wir werden daher auf SAML noch einmal in Abschn. II.7. zu sprechen kommen.

5. Neue Computing-Paradigmen: Web 2.0 und SaaS

Prognose: Durch neue Paradigmen in der Informatik werden persönliche und geschäftliche Daten zunehmend zentral gespeichert. Der Browser wird als universelle Clientsoftware zur Schnittstelle zwischen Nutzer und Daten, und seine Sicherheits-

features werden eine immer wichtigere Rolle spielen. Identitätsdaten werden im Browser gespeichert und sind nur durch dessen Sicherheitsmechanismen geschützt.

Webmail, Moodle & Co.
Der Erfolg von Webmail-Anbietern im Privatkundenbereich hat schon früh gezeigt, dass ein serverzentriertes Computing-Paradigma durch seine einfache Bedien- und Wartbarkeit besticht. Mittlerweile ist bei vielen Nutzern in Vergessenheit geraten, dass E-Mail eigentlich ein Push-Dienst ist, der auch offline genutzt werden kann: E-Mail wird heute weitgehend online genutzt, und auch viele professionelle E-Mail-Nutzer sind ohne eine schnelle UMTS-Verbindung auf Reisen nicht in der Lage, mit E-Mails zu arbeiten.

Ein weiteres, noch offensichtlicheres Beispiel sind Kalenderdienste wie Moodle oder Google Calendar. Jedem Nutzer, der Termine auf Server, Laptop und iPhone synchronisieren möchte, weiß, wie schwierig sich dies in der Praxis gestalten kann. Packt man den Kalenderdienst aber auf einen Server, auf dem alle Teilnehmer ihre Einträge machen können, verschwinden alle Probleme.

Der Erfolg von Webmail hat, von der Öffentlichkeit unbemerkt, schon Auswirkungen auf die Sicherheit gehabt: Da verschlüsselte E-Mails (OpenPGP oder S/MIME) über Webmail-Interfaces nicht abgerufen werden können, haben diese Standards in der breiten Öffentlichkeit nicht die benötigte Akzeptanz finden können. Bei Kalenderdiensten ist nicht einmal ansatzweise klar, wie diese kryptologisch gesichert werden könnten.

Web 2.0
Unter dem Schlagwort Web 2.0 werden Dienste zusammengefasst, die diesen Trend fortsetzen. In typischen Web 2.0-Anwendungen wie Facebook, Flickr, YouTube, Xing, SchülerVZ, Blogger etc. werden dem Anbieter auch sehr persönliche Daten überlassen. Web 2.0-Dienste können in der Regel mit wenig Investitionen ins Leben gerufen werden, weil nur noch eine Serveranwendung geschrieben werden muss und der Browser als universeller Client kostenlos verfügbar ist.

Konzepte zur Absicherung dieser Daten, unter denen sich bei Web 2.0-Anwendungen viele Identitätsdaten befinden, existieren nicht. Lediglich SSL wird vereinzelt zur Absicherung des Kommunikationskanals während der Übertragung herangezogen. So kam es wiederholt zu Datendiebstahl[303] oder zur Manipulation dieser Daten.[304] Moderne Computerwürmer nutzen die gespeicherten Identitätsdaten, um sich zu verbreiten.[305]

Die komplexe Interaktion zwischen dem Browser und den einzelnen Serverkomponenten kann durch eine Reihe spezialisierter Angriffe ausgenutzt werden:

- iFrame Injection[306]: Bei diesem Angriff wird ein fremder Inhalt in eine Website eingeschleust. Dies kann nicht persistent (z. B. durch Übergabe des HTML-

[303] http://www.tagesschau.de/inland/datenmissbrauch120.html.

[304] http://en.wikipedia.org/wiki/Samy_(XSS).

[305] http://www.kaspersky.com/news?id=207575670.

[306] http://eisabainyo.net/weblog/2009/04/06/iframe-injection-attack/.

Quelltextes als Suchparameter, der dann zusammen mit der Fehlermeldung vom Server an den Browser zurückgesandt und dort nicht als String, sondern als interpretiertes HTML dargestellt wird) oder persistent (indem der Angreifer sich Zugang zum Webserver verschafft und ein unsichtbares iFrame dort speichert) erfolgen. In einem auf solche Art injizierten iFrame sind oft Hyperlinks eingebunden, die Schadcode auf den Browser von der Seite des Angreifers nachladen.

- Cross-Site-Scripting (XSS)[307]: Analog zu iFrame Injection wird ein JavaScript-Programm unberechtigt in eine fremde Website, die im Browser ausgeführt wird, eingeschleust. Dadurch erlangt das Programm die Berechtigung, auf alle im Browser vom fremden Webserver geladenen Inhalte lesend und schreibend zuzugreifen. Dies kann unter anderem dazu benutzt werden, Identitätsdaten zu stehlen.
- SQL Injection[308] ist ein Angriff auf Serverseite: Hier wird ausgenutzt, dass fast jede Webanwendung eine SQL-Datenbank einbindet. Durch geeignete Formatierung einer Eingabe in ein HTML-Formularfeld kann der Angreifer bei dieser Form des Angriffs sicherstellen, dass seine Eingabe von den Frontend-Servern als SQL-Befehl an die Datenbank weitergeleitet wird und dort den gewünschten Effekt erzielt.
- Cross-Site-Request-Forgery[309]: Bei dieser Angriffsart wird ausgenutzt, dass die Authentifizierung eines Nutzers gegenüber einer Webanwendung oft sitzungsbasiert ist. Außerdem kann die Webanwendung oft nicht unterscheiden, ob eine HTTP-Anfrage unter Mithilfe des Nutzers erzeugt wurde oder ob sie automatisch generiert oder vorberechnet ist. Im einfachsten Fall muss der Angreifer auf seiner eigenen Website nur einen Imagelink einbetten, der als src-Parameter einen Query-String für eine Webanwendung enthält, für die sich das Opfer bereits authentifiziert hat. Der Server der Webanwendung sieht nur die HTTP-Anfrage und beantwortet diese, da die sitzungsbasierten Identitätsdaten vom Browser mitgeschickt wurden. Durch den Einsatz von JavaScript kann man diese Art von Angriffen ausweiten, bis hin zu vollautomatischen Angriffen auf Onlinebanking.[310]

Software-as-a-Service (SaaS)

Die unter dem Kürzel SaaS angebotenen Dienste beziehen klassische Desktopanwendungen in das serverzentrierte Computing-Paradigma mit ein. Dies wurde insbesondere durch die in neuen Webbrowsern eingeführte AJAX-Technologie möglich, die eine asynchrone Kommunikation zwischen Browser und Webserver ermöglicht. AJAX steht für „Asynchromous JavaScript And XML“ und basiert auf dem verstärkten Einsatz von umfangreichen JavaScript-Bibliotheken und der Kapselung von automatischen HTTP-Anfragen im XMLHttpRequest-Objekt [XMLHttpRequest].

---

[307] http://www.owasp.org/index.php/Cross-site_Scripting_(XSS).

[308] http://en.wikipedia.org/wiki/SQL_injection.

[309] http://www.owasp.org/index.php/Cross-Site_Request_Forgery_(CSRF).

[310] http://www.freedom-to-tinker.com/tags/csrf-attack.

Durch diese neuen Techniken wurde es möglich, die für eine Clientserver-Anwendung im Web 1.0 typischen Wartezeiten zwischen Absenden der Anfrage und Antwort des Servers für den Nutzer unsichtbar zu machen: AJAX fragt im Hintergrund bereits Daten ab, die der Nutzer noch nicht zu sehen bekommt. Dies kann am Beispiel von Google Maps illustriert werden: Ist hier eine hinreichend große Bandbreite vorhanden, kann die JavaScript-Bibliothek, die beim Aufruf der Website von Google Maps geladen wurde, die noch nicht sichtbaren Randbereiche sowie die nächsthöhere Auflösung der sichtbaren Bereiche der Karte im Voraus laden. Auf Anforderung des Nutzers hin werden diese lokal zwischengespeicherten Daten dann sofort angezeigt.

Die Sicherheitsdiskussion zu SaaS steckt ganz in den Anfängen: In einem offenen Brief an den CEO von Google haben 37 amerikanische Sicherheitsforscher darum gebeten, die Daten für die Google Apps doch wenigsten mit SSL während des Transports zu schützen [Acquisti et al. 09]. Eine Verschlüsselung der auf dem Google-Server gespeicherten Daten wird nicht einmal erwähnt. SSL soll in der „naiven" Form genutzt werden, d. h., der Google-Nutzer muss die Echtheit des Serverzertifikats überprüfen. Die daraus resultierende Sicherheitsproblematik wurde bereits auf S. 87 ff. angesprochen, mögliche Lösungsansätze werden auf S. 144 ff. dargestellt.

6. *Neue Computing-Paradigmen: Webservices, SOAP und Cloud Computing*

Prognose: Durch die gesteigerte Komplexität und höhere Dynamik in den Serveranwendungen entstehen neue Schwachstellen, die für DoS-Angriffe auf Firmeninfrastrukturen und für aktive Angriffe ausgenutzt werden können. Relevante Angriffe sind bereits seit mehreren Jahren publiziert, wurden aber in der Industrie noch nicht berücksichtigt. Identitätsmissbrauch kann durch Angriffe auf neue Technologien wie XML Signature durchgeführt werden.

Webservices und serviceorientierte Architekturen

Der Erfolg von Webanwendungen hat in der Industrie Bestrebungen geweckt, die dort genutzten Paradigmen zu verallgemeinern, um Kosten für immer komplexer werdende Softwaresysteme einzusparen. Hier gibt es drei Erfolgsmodelle, die weiterentwickelt wurden:

- HTML wird zu XML. HTML war zu Beginn eine einfache Markup-Sprache, die Formatierungs- und Strukturelemente vereinte. Im Laufe der Entwicklung wurde sie um mächtige Konstrukte erweitert, z. B. um Formulare, Skriptsprachen (JavaScript) und ein Objektmodell. Gleichzeitig entstand eine große Entwicklercommunity. Diese Tatsachen wurden von den XML-Entwicklern genutzt. Sie entwickelten, ausgehend von älteren Initiativen (z. B. ASN.1), eine universelle Sprache zur Beschreibung von Datenstrukturen und fügten gleich ein klares Objektmodell und eine Fülle von Skriptsprachen zur Manipulation dieses Objekt-

modells hinzu. Dabei lehnten sie sich grob an die Syntax von HTML an, um die Community für die neue eXtensible Markup Language (XML) zu gewinnen.
- HTTP wird das grundlegende Kommunikationsparadigma. Während sich ältere Netzwerkprotokolle noch an komplexen Kommunikationsmustern versuchten (z. B. FTP mit einem Daten- und einem Steuerkanal), hat sich mittlerweile das einfache und klare Kommunikationsmuster von HTTP durchgesetzt: Eine Frage – eine Antwort. Die meisten neuen Kommunikationsprotokolle (z. B. SOAP) bauen darum auch auf HTTP auf.
- URLs bezeichnen jeden auf der Welt vorhandenen Datensatz eindeutig. Damit sind alte Standardisierungspläne (X.400 und X.500) mit einem rein pragmatischen Ansatz doch noch realisiert worden. Wichtig ist hier das Domain Name System, da es die grundlegende, eindeutige Namensvergabe an die Server regelt. URLs wurden damit zur natürlichen Basis, um auf fremde Webservices zuzugreifen.

Diese Entwicklung wurde rein durch die neuen Funktionalitäten getrieben. Die Sicherheit der neuen Paradigmen selbst wurde nicht reflektiert.

Es wurde zwar eine ganze Reihe neuer Sicherheitsstandards entwickelt, aber diese waren überwiegend darauf ausgerichtet, neue Sicherheitsfunktionalitäten zu ermöglichen, und nur teilweise, um die neuen Paradigmen abzusichern:

- XML Signature [XML Signature] ist der „älteste“ Sicherheitsstandard, und er wurde gemeinsam vom World Wide Web Consortium (W3 C, www.w3.org) und der IETF (www.ietf.org) entwickelt. Er erweitert herkömmliche Standards zu digitalen Signaturen um die Idee, auch den *Ort* der geschützten Daten mit zu signieren. Darüber hinaus enthält der Standard aber viel zu viele Optionen, die letztlich die Sicherheit von digitalen Signaturen selbst untergraben können. So wurden z. B. schon 2005 von Micheal McIntosh und Paula Austel (IBM TJ Watson NY) Möglichkeiten beschrieben, wie ein Angreifer signierte Daten im Netzwerk abfangen, manipulieren und den Empfänger schließlich dazu bringen kann, trotz Signaturprüfung gefälschte Daten zu verarbeiten [MA05]. Aufgrund der großen Komplexität der XML-Standards kann man diesen Angriff sogar noch verfeinern [JLS09]. Die Tatsache, dass die Unterstützung der Turing-vollständigen Skriptsprache XSLT zwingend vorgeschrieben ist, ermöglicht darüber hinaus die Ausführung von Schadcode innerhalb des Signaturprüfprozesses [Hill07]. Diese Probleme sind in den gängigen Implementierungen noch nicht gelöst und können als Basis für Angriffe verwendet werden.
- Da es Ziel der Entwicklung bei den neuen Sicherheitsstandards für Webservices war, neue Features zu integrieren, wurde das Datenformat sehr komplex – dies zieht eine komplexe Verarbeitung nach sich: Sicherheitskonstrukte müssen heute immer komplett in den Hauptspeicher des verarbeitenden Rechners geladen werden, um dort nach dem DOM-Prinzip verarbeitet zu werden. Die restlichen (Nutz-)Daten werden aber aus Performanzgründen eventbasiert nach dem SAX-Prinzip verarbeitet. Diese Verarbeitungsstrategie kann zu Sicherheitsproblemen führen [GL09].

Cloud Computing
Während SOAP noch ein abstraktes Konzept ist, ist Cloud Computing schon Realität: Die Firma Amazon hat ein Vermarktungskonzept gefunden, mit dem sie ihre freien Rechenzentrumsressourcen am Markt anbieten kann. Damit werden erstmals auch geschäftliche Daten im großen Maßstab in einem fremden Rechenzentrum verarbeitet, basierend auf sehr losen vertraglichen Vereinbarungen. Die Firma Gartner hat hierbei sieben Sicherheitsprobleme in den Fokus gestellt. Zitat aus [Gartner08]:

- **Privileged user access** – inquire about who has specialized access to data and about the hiring and management of such administrators
- **Regulatory compliance** – make sure a vendor is willing to undergo external audits and/or security certifications
- **Data location** – ask if a provider allows for any control over the location of data
- **Data segregation** – make sure that encryption is available at all stages and that these „encryption schemes were designed and tested by experienced professionals“
- **Recovery** – find out what will happen to data in the case of a disaster; do they offer complete restoration and, if so, how long that would take
- **Investigative Support** – inquire as to whether a vendor has the ability to investigate any inappropriate or illegal activity
- **Long-term viability** – ask what will happen to data if the company goes out of business; how will data be returned and in what format.

Identitätsdaten sind durch diese Entwicklung in zweierlei Hinsicht betroffen: Zum einen können sich unter den verarbeiteten Datensätzen Identitätsdaten befinden, zum anderen wird dem Identitätsdiebstahl oder Identitätsmissbrauch ein neues Ziel geboten: Diebstahl/Missbrauch der Identitätsdaten zur Authentifizierung in einem Cloud Computing-Portal.

Die Problemliste von Gartner ist daher noch wie folgt zu ergänzen/präzisieren:

- **User access** – Wie wird der Nutzer im Internet identifiziert? (→ Identitätsdiebstahl/Identitätsmissbrauch)
- **Legal compliance** – Wen wird die Rechtsprechung bei Verlust von Daten/illegalen Daten haftbar machen?
- **Data encryption and key management** – Wenn der Nutzer die Verschlüsselung der Daten selbst kontrolliert, sollte data location und data seggregation kein Problem sein. Wie kann der Nutzer seine Schlüssel managen?
- **Recovery of encrypted data** – Wie sind Backup und Recovery von verschlüsselten Daten organisiert?

Konkrete Angriffsszenarien ergeben sich durch die Kombination der einzelnen Angriffsszenarien für die verwendeten Komponenten. Cloud Computing verwendet hier Webservices (dieser Abschnitt) und den Browser als zentrale Sicherheitskomponenten auf Clientseite (S. 87 ff.). Ansätze zur Absicherung dieser Komponenten werden S. 144 ff. diskutiert. Ein Überblick zu Sicherheitsfragen des Cloud Computing wurde in [JSGL09] publiziert.

7. Neue Computing-Paradigmen: Single-Sign-On

Prognose: Da jeder Internetnutzer in Zukunft eine Fülle von Identitätsdaten zu verwalten hat, werden Single-Sign-On-Technologien stark an Bedeutung gewinnen. Stehen heute noch die einfache Bedienbarkeit und der Schutz gegen einfache Phishingangriffe im Vordergrund der Entwicklungen, so müssen diese Systeme, da sie ein ausgesprochen lohnendes Angriffsziel darstellen, auch gegen komplexere Angriffe geschützt werden.

Das Kerberos-Protokoll

Das Kerberos-Protokoll wurde vom MIT entwickelt [KNT91] und von Microsoft in der Version 5 ab Windows 2000 als Standardauthentifizierungsdienst eingesetzt. Das Kerberos-Protokoll basiert auf der Tatsache, dass es ein Vertrauensverhältnis in Form eines gemeinsamen symmetrischen Schlüssels zwischen den einzelnen Hosts und dem Kerberos-Server gibt. Es stand nicht eine Vereinfachung des Log-in-Vorgangs im Mittelpunkt, sondern eine Vereinfachung des (symmetrischen) Schlüsselmanagements in geschlossenen Netzwerken.

Die Sicherheit von Kerberos ist gut untersucht, es sind keine größeren Lücken bekannt geworden. Es lag daher nahe zu untersuchen, ob ein ähnlicher Mechanismus nicht auch für die Authentifizierung im Internet verwendet werden kann.

Microsoft Passport und Microsoft Cardspace

Da Microsoft bereits über umfangreiche Erfahrungen mit Kerberos verfügte, war es nicht verwunderlich, dass die erste technisch ausgereifte Lösung unter dem Namen „Microsoft Passport“ vermarktet wurde.

Leider unterscheiden sich Kerberos und seine Adaption für das WWW deutlich, wie die Abb. 41 zeigt: Die paarweise Authentifizierung zwischen dem Kerberos-Server auf der einen und Client und Server auf der anderen Seite, die über einen paarweise vereinbarten, symmetrischen Schlüssel zumindest implizit sichergestellt war, geht bei MS Passport verloren: Der Browser ist in diesem Szenario völlig anonym.

Dies hat eine Reihe von Angriffen nach sich gezogen [KR00]. In [Slemko01] ist es sogar gelungen, über einen komplexen XSS-Angriff das Passport-Account eines Nutzers vollständig zu übernehmen. Dieser musste dazu nur eine vom Angreifer an ihn geschickte E-Mail über sein MS-Hotmail-Account abrufen.

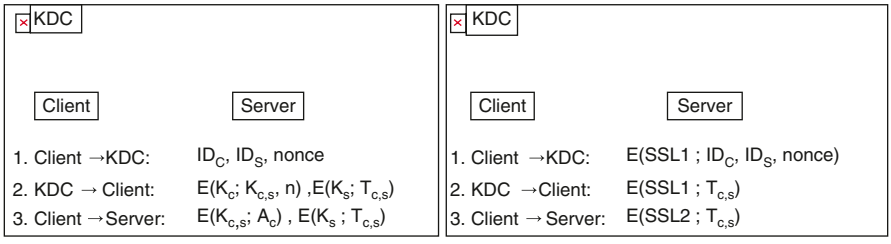


Abb. 41 Gegenüberstellung von Kerberos und einem webbasierten Single-Sign-On-Protokoll

Letztlich war für das Scheitern von MS Passport nicht dieser Angriff entscheidend, sondern die Tatsache, dass Microsoft vorgeworfen wurde, das WWW zu monopolisieren.

Microsoft ist der einzige Player im Single-Sign-On-Markt, der zugleich die Kontrolle über einen Großteil der Browserpopulationen besitzt. Dadurch ist es Microsoft möglich, neuartige Konzepte sehr schnell einzuführen. Über eine Erweiterung des Internet Explorers, die Verwendung der offenen WS-Security-Standards und eine wettbewerbsneutrale Methode, den Identity Provider auszuwählen, hat Microsoft es mit Einführung von Microsoft Cardspace geschafft, den Monopolvorwurf auszuräumen.

Leider wurde diese einzigartige Marktposition nicht dazu genutzt, auch die Sicherheit von Single-Sign-On zu verbessern: Microsoft Cardspace besitzt immer noch die generischen Sicherheitsprobleme aller browserbasierten SSO-Protokolle (vgl. die Abb. 41), was mit einem auf dynamischem Pharming basierenden Proof-of-Concept-Angriff bereits nachgewiesen werden konnte [GSSX09].

Diese in der Abb. 42 dargestellten generischen Sicherheitsprobleme haben ihren Ursprung in der unsicheren Same Origin Policy des Browsers: Alle Sicherheitstoken, die der Identity Provider über den Browser an die Relying Party übermittelt, werden zumindest temporär im Document Object Model des Browsers gespeichert. Kann man die Same Origin Policy ausschalten, so ist es entweder möglich, das Sicherheitstoken aus dem DOM auszulesen ([GSSX09], [Slemko01]) oder das Token mittels eines Pharming-Angriffs zu übermitteln.

Standardisierung: SAML 2.0

Standardisiert wurden SSO-Protokolle im Zusammenhang mit SAML 2.0. In dieser Arbeitsgruppe wurde ganz klar erkannt, dass SSO ein wichtiges Werkzeug für die Verwaltung von Identitäten im Internet sein wird. Es wurden daher verschiedene SSO-Szenarien beschrieben und eine Vereinheitlichung der Terminologie herbeigeführt.

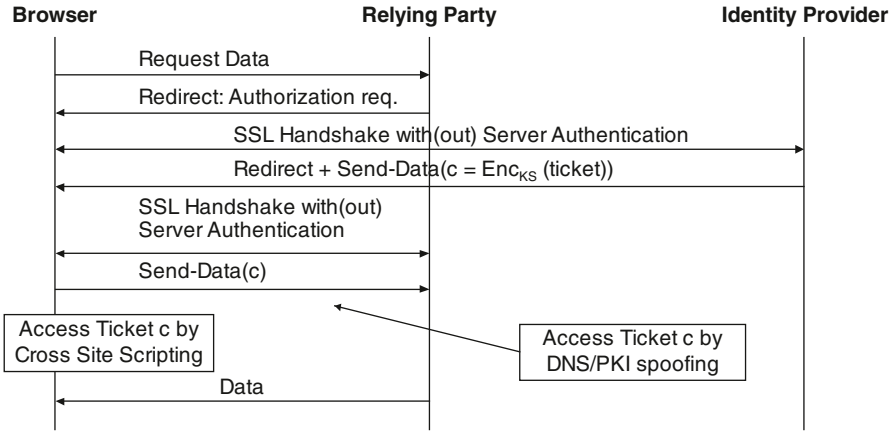


Abb. 42 Sicherheitsprobleme eines generischen Single-Sign-On-Protokolls

Terminologie: In einem SSO-Protokoll agieren drei Parteien: Ein **Client** versucht, Zugriff auf eine geschützte Ressource der **Relying Party** zu erhalten. Diese führt die Authentifizierung (und evtl. auch die Autorisierung) nicht mehr selbst durch, sondern delegiert diese an eine spezialisierte Partei, den **Identity Provider**. Der Identity Provider führt die Authentifizierung des Clients durch, beschreibt das Ergebnis dieser Authentifizierung in einer SAML Assertion und sendet diese über den Browser an die Relying Party.

SSO-Szenarien: Verschiedene Möglichkeiten, wie SAML Requests und SAML Responses über einen Browser transportiert werden können, sind in [SAMLProfiles] beschrieben. Hierunter fällt z. B. der Transport als Query-String in einer URL (Redirect Binding) oder der Transport als Daten in einem vorab ausgefüllten HTML-Formular (POST-Binding). Will man die durch das HTTP-Protokoll vorgegebenen Rollen zwischen Client und Server vertauschen, so benötigt man einen Client, der Daten gemäß [PAOS 1.0] senden und empfangen kann (Enhanced Client Profile). Sehr positiv hervorzuheben ist, dass in der SAML-Arbeitsgruppe ein Vorschlag zur Absicherung von SSO gemacht wurde, der kryptografisch sicher ist [SAMLHoK].

SAML wurde mittlerweile von den beiden nachfolgend genannten Standardisierungsgremien als Datenformat festgelegt:

- Shibboleth[311]: „The Shibboleth System is a standards based, open source software package for web single sign-on across or within organizational boundaries. It allows sites to make informed authorization decisions for individual access of protected online resources in a privacy-preserving manner."
- Liberty Alliance[312]: „Thus the Liberty Alliance emerged: a first-of-its-kind standards organization with a global membership that provides a holistic approach to identity."

SSO für den Massenmarkt
Als Entwicklung für den Massenmarkt ohne große Sicherheitsansprüche ist die OpenID-Initiative[313] zu sehen. Es gibt hier keine klaren Sicherheitsrichtlinien im Standard, und OpenID-Phishing ist ein in Internetforen[314,315] diskutierter Punkt.

Verwandte Lösungen
Als verwandte Lösungen sind alle Lösungen anzusehen, die den Nutzer bei der Verwaltung oder der Auswahl starker Passwörter unterstützen. Zu nennen wären hier zum einen Passwortmanager in Browsern, zum anderen die Idee, domainspezifische Passwörter einzusetzen, oder auch eine Hardwarelösung wie der Passwortsitter.[316] Allerdings sind all diese Lösungen wiederum anfällig gegen Phishingattacken.

---

311 http://shibboleth.internet2.edu/.

312 http://www.projectliberty.org/.

313 http://openid.net/.

314 http://wiki.openid.net/OpenID_Phishing_Brainstorm.

315 http://www.identityblog.com/?p=659.

316 http://www.passwordsitter.de/.

8. Neue Computing-Paradigmen: neue Schutzmaßnahmen

Da Standardschutzmaßnahmen immer weniger wirksam sind, sollen hier kurz einige neue Schutzmaßnahmen aufgezählt werden, die in Zukunft an Bedeutung gewinnen könnten.

Schutzmaßnahmen auf Serverseite

- Web Application Firewalls[317] filtern Eingaben für Webapplikationen, um so XSS- und SQL-Injektions zu verhindern. Dies entlastet die Programmierer von Webanwendungen, die oft gar nicht in der Lage sind, einen Schutz gegen solche Angriffe mit zu berücksichtigen. Auf diese Weise können zumindest Standardangriffe verhindert werden.
- Maßnahmen gegen CSRF: In dem Artikel zur Pressemeldung über den bislang weitgehendsten CSRF-Angriff[318] (zurzeit nicht online) sind einfache Gegenmaßnahmen beschrieben. Um zwischen automatisch mittels JavaScript generierten Anfragen und Anfragen als Resultat einer vom Server gelieferten Website unterscheiden zu können, müssen (Pseudo-)Zufallswerte in diese Anfragen integriert werden, die leicht vom Server verifiziert werden können.
- Bessere Nutzung von SSL: Clientzertifikate werden bislang kaum genutzt, obwohl sie ein kryptografisch sehr sicheres Wiedererkennen des Browsers ermöglichen. Erste Ideen hierzu wurden in [SAMLHoK] aufgegriffen.

Schutzmaßnahmen auf Clientseite

Auf Clientseite spielen der Browser und seine Sicherheitsfeatures die zentrale Rolle.

- Strong Locked Same Origin Policy (SLSOP): Mithilfe der SLSOP [SLSOP07] und mit modernen Browsern, die für jeden geöffneten Tab einen eigenen Prozess starten, lässt sich eine bessere Isolation von Webinhalten im Browser durchführen. Die SLSOP entscheidet über Zulassung oder Unterbindung eines Zugriffs auf Webinhalte nicht mehr auf Basis des Domainnamens, sondern auf Basis des Public Keys des SSL-Serverzertifikats.
- Verbessertes DOM: Das Document Object Model der Browser[319] muss harmonisiert und an neue Anwendungsformen wie Mashups angepasst werden. Als Ausgangspunkt kann hier die Arbeit von Google dienen [Zalewski09].
- Whitelisting für Browser: Ähnlich zu Whitelisting-Ansätzen für Betriebssysteme, bei denen bei Ausführung einer Applikation mit hohem Sicherheitsbedarf alle nicht auf der Whitelist geführten Prozesse vorübergehend suspendiert werden, könnte man auch für die Dauer des Aufrufs einer Bankingwebsite alle nicht benötigten Browserfunktionalitäten (z. B. JavaScript) oder Plug-ins (Adobe Flash, Malware) deaktivieren.
- Vollständige Virtualisierung jeder Browserinstanz: Jegliche Browserprozesse werden in einer vollständig virtualisierten Umgebung ausgeführt. Auf diese Wei-

[317] http://www.owasp.org/index.php/Web_Application_Firewall.

[318] http://www.freedom-to-tinker.com/blog/wzeller/popular-websites-vulnerable-cross-site-request-forgery-attacks.

[319] http://en.wikipedia.org/wiki/Document_Object_Model.

se könnte selbst eine kompromittierte Browserinstanz das Betriebssystem und andere (virtualisierte) Browserinstanzen nicht in Mitleidenschaft ziehen.
- Eine Integration von XML Signature und XML Encryption in den Browser wäre sinnvoll, da er dann als vollwertiger Endpunkt von sicherer Webservicekommunikation agieren kann.

Schutzmaßnahmen aufseiten der Daten
- XML Signature und XML Encryption bilden eine Basis, auf der man Daten vollständig schützen kann, ohne Einbußen bei der Funktionalität hinnehmen zu müssen. Auf diese Art und Weise können Daten in Zukunft ihren Schutz selbst in sich tragen.

9. Kombination mehrerer Angriffstechniken

Im Bereich der Angriffe über mehrere beteiligte Instanzen hinweg müssen verschiedene Angriffsvektoren betrachtet und kombiniert werden. Denn sobald mehrere Instanzen an einem Prozess beteiligt sind, wird die Absicherung dieses Prozesses komplizierter. Der Angreifer besitzt bei solchen Szenarien einen entscheidenden Vorteil: Er muss lediglich eine Lücke in einer beteiligten Instanz finden und ausnutzen, wohingegen der Verteidiger alle beteiligten Instanzen vollständig absichern muss. Dies gestaltet sich oftmals sehr schwierig, da der Verantwortliche beispielsweise keinen Zugriff auf alle Instanzen hat oder diese nicht in seinem Zuständigkeitsbereich liegen.

Ein Vorteil könnte sein, zunächst das schwächste Glied der Kette abzusichern. Um den eigentlichen Prozess zu kompromittieren, muss der Angreifer in vielen Fällen lediglich eine Instanz erfolgreich angreifen. Im Folgenden werden Angriffe auf verschiedene Prozessszenarien mit mehreren beteiligten Instanzen durchgespielt und analysiert.

a) 1. Szenario: HTTP-Clientkompromittierung

In diesem Abschnitt wird ein Angriff gegen eine Organisation aufgezeigt, der damit endet, dass sensible Informationen in die Hand von Angreifern gelangen können. Das Szenario zeigt auf, wie sich Angreifer durch veraltete bzw. nicht gepatchte Software auf dem Client Zugang zu sensiblen Daten und Informationen innerhalb des Intranets einer Organisation verschaffen können.

aa) Schritt 1: Vorbereitung

Die Angreifer platzieren Schadcode auf Websites, die bei den Nutzern der Organisation als vertrauenswürdig eingestuft sind. Bedingung ist hierbei allerdings, dass öffentliche Nutzer Inhalte auf diesen Websites publizieren können. Dies könnten beispielsweise Websites sozialer Netzwerke (Facebook, XING), Webblogs (Blog-

ger), Fotowebsites (Flickr) oder Videoplattformen (YouTube) sein. In vielen Organisationen sind die genannten Websites für die jeweiligen Mitarbeiter aufrufbar und werden nicht blockiert. Der Schadcode ist dafür programmiert worden, nicht gepatchte Software des Clients (Browser, PDF-Reader etc.) zu kompromittieren.

bb) Schritt 2: Kompromittierung des Clients

In diesem Schritt wird die Software auf dem Clientsystem des Opfers durch den platzierten Schadcode auf der vertrauenswürdigen Website kompromittiert. Dies könnte per Drive-by-Angriff auf den Browser (Mozilla Firefox, Internet Explorer etc.), durch Öffnen eines bösartigen PDF-Dokuments (Acrobat Reader, Foxit-Reader etc.), durch Abspielen eines präparierten Videos (Real Player, Windows Media Player, iTunes etc.) oder durch Öffnen eines modifzierten Office-Dokuments (Microsoft Word, Excel, Powerpoint) geschehen.

Durch die erfolgreiche Ausnutzung der Sicherheitslücke in der Software des Clientsystems ist es dem Angreifer nun möglich, beliebige Programme auf das System nachzuladen und dort auszuführen. Die Programme werden mit den Rechten des Nutzers der kompromittierten Anwendung ausgeführt. Im Regelfall werden dies keine Administratorrechte sein.

cc) Schritt 3: Öffnen einer Reverse-Shell-Hintertür über HTTPS

In diesem Schritt installiert der Angreifer eine Hintertür auf dem System des Opfers. Diese gibt dem Angreifer die Möglichkeit, Befehle direkt über eine Shell auf dem System des Opfers auszuführen. Der erzeugte Netzwerkverkehr wird verschlüsselt über einen HTTPS-Tunnel zwischen dem System des Angreifers und dem des Opfers ausgetauscht. Er wird daher von der Firewall der Organisation als regulärer HTTPS-Verkehr angesehen und fällt nicht weiter auf.

dd) Schritt 4: Erlangung von erweiterten Rechten auf dem Clientsystem

Durch die Reverse-Shell-Hintertür wird der Angreifer eine lokale Schwachstelle im System ausnutzen, um sich erweiterte Rechte (meist Administratorrechte) auf dem System zu verschaffen. In vielen Organisationen werden Patches für entsprechende lokale Schwachstellen nicht regelmäßig ausgeliefert, da man primär Patches für Schwachstellen, die von extern ausgenutzt werden können, bevorzugt.

ee) Schritt 5: Kompromittierung weiterer Systeme im Intranet der Organisation

Mit Administratorrechten besitzt der Angreifer die vollständige Kontrolle über das System des Opfers. Von hier aus kann er nun Angriffe auf weitere Systeme im

Intranet der Organisation starten. Ein Keylogger auf dem Clientsystem protokolliert alle Tastatureingaben der Nutzer an diesem System und gibt dem Angreifer gegebenenfalls weitere Informationen über mögliche Angriffsziele. Viele Nutzer benutzen für verschiedene Dienste die gleichen Passwörter. Dies könnte durch das Mitlesen eines Accounts auf dem Clientsystem dazu führen, dass der Angreifer weitere Systeme im Intranet kompromittieren kann. Dadurch, dass der Angreifer Administratorrechte auf dem System besitzt, kann er seine weiteren Schritte mit einem Rootkit tarnen, sodass eine Entdeckung durch die Nutzer noch schwieriger wird.

ff) Schritt 6: Auslieferung von sensiblem Datenmaterial durch den Angreifer

Der Angreifer verschafft sich Zugang zu einem Datenbankserver, indem er das Account eines Nutzers aus der Logfile des Keyloggers benutzt. Er exportiert die Inhalte der Datenbank, bei denen es sich um Kundendaten mit zugehörigen Kontoverbindungen einer Customer-Relationship-Anwendung handelt. Über den gesicherten HTTPS-Tunnel transferiert der Angreifer rund 500 MB sensible Daten an einen Server in Korea – diese Datenübertragung bleibt unentdeckt.

gg) Schlussfolgerung

Der Angreifer attackiert das schwächste Glied in der Kette: das nicht gepatchte System des Endanwenders. Dieses System besitzt im Intranet diverse Rechte und Möglichkeiten. Durch die vollständige Kompromittierung des Systems werden diese Rechte und Möglichkeiten indirekt an den Angreifer übergeben. Dieser macht sich diese Eigenschaften zunutze, indem er beispielsweise weitere Systeme im Intranet angreift. Er benutzt diesen Rechner als Ausgangsbasis für weitere Angriffe. Die Gefahr dieses Angriffsszenarios liegt darin, dass fälschlicherweise angenommen wird, der Client des normalen Nutzers befinde sich geschützt hinter der Firewall im Intranet der Organisation. Aber Angriffe auf Clientsysteme wie die in diesem Szenario vorgestellten sind immer häufiger Ursache für Clientkompromittierungen. Meist ist die eigentliche Absicht eine Eingliederung des Clients in ein Bot-Netz; dies wäre in diesem Fall aber rein optional. Um zielgerichtete Angriffe auf Organisationen durchzuführen, wird sich der Angreifer so unauffällig wie möglich verhalten.

Dieses Szenario zeigt, dass durch die Kompromittierung einer schwächeren Instanz eventuell auch stärker geschützte Instanzen angegriffen werden können.

b) 2. Szenario: Prozesskompromittierung durch fortgeschrittenes ARP-Spoofing

ARP-Spoofing ist nicht nur ausschließlich in Clientumgebungen relevant, sondern auch in reinen Serverumgebungen (siehe I.5.c)). In diesem Szenario wird ein trickreicher Man-in-the-Middle-Angriff auf einen Onlinebanking-Prozess gestartet,

ohne die beteiligten Server- und Clientinstanzen direkt zu attackieren. Es ist daher relativ schwer, diesen Angriff zu entdecken.

aa) Schritt 1: Kompromittierung der Man-in-the-Middle-Instanz

Serverfarmen in Rechenzentren, die sensible Daten besitzen, sind meist nach außen gut abgesichert. Durch strikte Firewallregeln kann ein Außenstehender nur auf spezielle Rechner zugreifen, ohne dass er die gesamte Serverfarm zu sehen bekommt. Die Absicherung der Server untereinander ist hingegen meist nicht so strikt geregelt. Dies bedeutet, dass die Server eventuell untereinander kommunizieren können, ohne dass die Firewall, die als Regelinstanz fungiert, davon etwas mitbekommt. Genau diese Tatsache könnte sich ein Angreifer zunutze machen. Ist es dem Angreifer möglich, über einen schwächer gesicherten Server in dieser Serverfarm die Kontrolle zu bekommen, kann dieser als Basis für weitere Angriffe genutzt werden. Dieser kompromittierte Server braucht in erster Instanz nicht unbedingt mit der Onlinebanking-Applikation oder der Bank an sich in Verbindung stehen.

bb) Schritt 2: ARP-Poisoning-Angriffe auf umliegende Serverinstanzen

Sobald die kompromittierte Instanz unter der Kontrolle der Angreifer ist, wird von dort ein ARP-Poisoning-Angriff auf die restlichen Serverinstanzen im Subnetz gestartet. Mithilfe dieser Angriffsmethode ist es möglich, den Netzwerkverkehr aller Server nun primär über die kompromittierte Instanz laufen zu lassen, die diesen dann an den Router weiterleitet. Der kompromittierte Server agiert daher als Man-in-the-Middle.

cc) Schritt 3: Infektion des HTTP(S)-Netzwerkverkehrs

Die Man-in-the-Middle-Instanz loggt nun primär HTTP-Netzwerkverkehr mit und könnte auch aktiv Inhalte mit in den Verkehr einbinden. So könnten beispielsweise Fehlinformationen verbreitet, Formulardaten umgeleitet oder Browser mit bösartigem JavaScript attackiert werden. Da in diesem Szenario einige der Server Onlinebanking-Anwendungen betreiben, fokussiert der Angreifer primär den HTTP-Verkehr von und zu diesen Servern. Sollten diese Verbindungen per SSL/TLS gesichert sein, wovon auszugehen ist, könnte eine Angriffsmethode mit SSLStrip weiterhelfen. Dies bedeutet, die Man-in-the-Middle-Instanz täuscht dem Client vor, dass lediglich eine ungesicherte HTTP-Verbindung zum Server möglich ist. Nebenbei kommuniziert der Server aber über eine gesicherte SSL/TLS-Verbindung mit dem Onlinebanking-Webserver. Der Angreifer könnte so direkt auf den Onlinebanking-Prozess von Clients Einfluss nehmen, ohne dass der Webserver etwas davon merken würde. Aufseiten des Clients könnte man versuchen, durch verschiedene Verschleierungsmethoden dem Nutzer vorzutäuschen, es handele sich doch um eine gesicherte TLS/SSL-Verbindung zum Webserver.

dd) Schlussfolgerung

Dieses Szenario stellt ein sehr hohes Gefahrenpotenzial für die Nutzer von Onlinebanking-Anwendungen dar. Denn der Angreifer attackiert weder Server noch Client direkt, sodass eine Erkennung durch diese Instanzen sich primär als sehr schwierig herausstellen wird. Eine dritte, eigentlich nicht an diesem Prozess beteiligte Instanz agiert als Man-in-the-Middle und greift die eigentliche Verbindung zwischen Onlinebanking-Server und Client an.

Bei einer eventuell folgenden forensischen Untersuchung wird sich herausstellen, dass weder Server noch Client kompromittiert wurden. Man wird daher davon ausgehen, dass einer der Router auf dem Weg als Man-in-the-Middle agiert.