2 Point-To-Point Sicherheit

Übersicht

Die Sicherungsschicht (OSI-Schicht 2) dient zur verlässlichen Übertragung von Datenrahmen („Frames“) zwischen zwei Computern (oder aktiven Netzwerkkomponenten) über ein einheitliches physikalisches Medium (z.B. eine direkte Kupferdraht-Verbindung) [Tan03].

2.1 PPP-Sicherheit

Das *Point-to-Point-Protocol* (PPP) [Sim94] hat sich in den letzten Jahren als Standard für Einwahlverbindungen in Computernetze durchgesetzt: Internet Service Provider

7	Anwendungsschicht	Anwendungsschicht	Telnet, FTP, SMTP, HTTP, DNS, IMAP
6	Darstellungsschicht		
5	Sitzungsschicht		
4	Transportschicht	Transportschicht	TCP, UDP
3	Vermittlungsschicht	IP - Schicht	IP
2	Sicherungsschicht	Netzzugangsschicht	Ethernet, Token Ring, PPP, FDDI, IEEE 802.3/802.11
1	Bitübertragungsschicht		

Abb. 2.1 TCP/IP-Schichtenmodell: Das Point-to-Point-Protocol (PPP)

J. Schwenk, *Sicherheit und Kryptographie im Internet,*
DOI 10.1007/978-3-658-06544-7_2, © Springer Fachmedien Wiesbaden 2014

(ISP) binden ihre Kunden mit PPP an das Internet an, und Firmen bieten ihren Außendienstmitarbeitern PPP-Einwahlmöglichkeiten ins Firmennetz.

Der Erfolg von PPP ist wohl auch in den skalierbaren und in der Praxis erprobten Authentisierungsmöglichkeiten begründet. Mit Hilfe der Client-Server-Architektur von RADIUS (des bekanntesten Beispiels für ein Authentifizierungs-, Autorisierungs- und Abrechnungsprotokoll, AAA) und des Password Authentication Protocol (PAP) können ISPs Millionen von Kunden verwalten und Rechnungen erstellen.

Wir wollen daher kurz PPP und seine Authentisierungsprotokolle vorstellen, dann auf die Infrastruktur dahinter eingehen und schließlich Vorschläge zur „Verlängerung“ von PPP durch das Internet diskutieren. Dass diese Vorgehensweise nicht ungefährlich ist, zeigten Mudge und Schneier [SM98a] anhand von Sicherheitslücken in Microsofts PPTP-Protokoll auf. Da diese Lücken auf typische Implementierungsfehler zurückzuführen sind, wollen wir näher auf sie eingehen.

Auch die zweite Version von PPTP kann mittlerweile nicht mehr als sicher angesehen werden: Moxie Marlinspike and David Hulton haben gezeigt, dass PPTPv2 mit einer Komplexität von nur 2^{56} gebrochen werden kann [MH12].

2.2 PPP

Das *Point-to-Point-Protocol* (PPP) wurde 1994 als [Sim94] publiziert. Es benötigt eine Vollduplex-Verbindung (gleichzeitiger Datentransport in beide Richtungen ist möglich) zwischen zwei Hosts, wie sie z.B. ISDN, DSL oder eine Modemverbindung bieten.

In PPP können beliebige Protokolle verpackt werden; meist ist dies heute das Internet Protokoll (IP). PPP verwendet folgende Hilfsprotokolle:

- *Link Control Protocol (LCP)*: Hier werden die PPP-Optionen ausgehandelt, z.B. Datenrate und Rahmengröße; auch die Authentisierung der Teilnehmer findet hier statt.
- *Network Control Protocol (NCP)*: Unter diesem Begriff werden zusätzliche Protokolle zusammengefasst, die jeweils für ein bestimmtes Nutzlast-Protokoll bestimmt sind. So gibt es z.B. für IP ein Protokoll, über das die Zuweisung von IP-Adressen an einen Client erfolgen kann.

In einem PPTP-Frame gibt das Feld „Protocol“ mit einem Zahlenwert an, welches Protokoll im Datenfeld transportiert wird (z.B. 0x0021 für IP, oder 0xc023 für PAP).

Ein PPP-Verbindungsaufbau läuft wie folgt ab:

1. LCP-Protokoll: PPP-Pakete mit protocol=0xc021 handeln die PPP-Parameter aus.
2. Authentisierung: PPP-Pakete mit protocol=0xc023 PAP/0xc223 CHAP (s.u.) überprüfen die Authentizität des sich einwählenden Clients.
3. NCP-Protokoll: PPP-Pakete mit protocol=0x80** handeln weitere Parameter für das Nutzprotokoll aus (z.B. 0x8021 für IP).

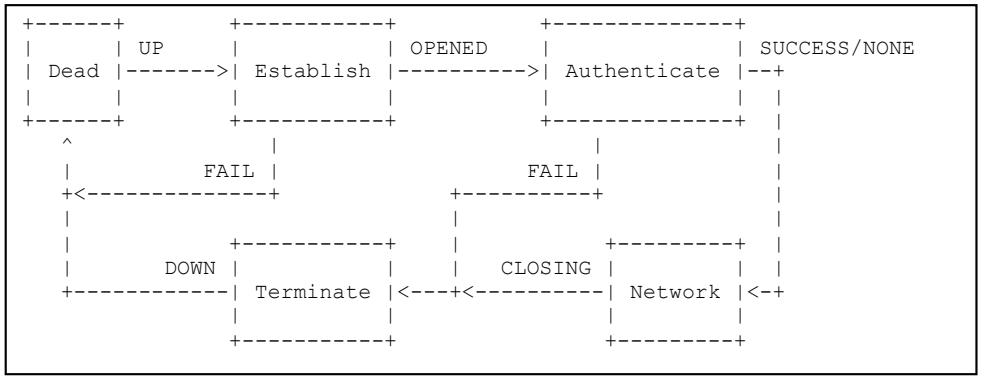

Abb. 2.2 Phasendiagramm eines PPP-Verbindungsaufbaus aus [Sim94]. Wichtig ist hier die Authenticate-Phase, die zum Kern des Verbindungsaufbaus gehört.

4. Nutzprotokoll: Jetzt können PPP-Rahmen mit dem Nutzprotokoll ausgetauscht werden.

2.3 PPP-Authentisierung

Wir wollen nun etwas näher auf die Authenticate-Phase eingehen. Für diese Phase können die beiden standardisierten Protokolle *Password Authentication Protocol* (PAP) [LS92] oder *Challenge Handshake Authentication Protocol* (CHAP) [Sim96], oder eine Vielzahl von mehr oder weniger proprietären Verfahren zum Einsatz kommen.

Beim Password Authentication Protocol (PAP) sendet der Client das Paar (ID, Passwort) im Klartext. Ein Network Access Server (NAS) überprüft das Passwort gegen den zur ID gespeicherten Wert, oder besser gegen den gespeicherten Hashwert des Passworts. Dieses Verfahren ist natürlich nur dann halbwegs sicher, wenn die Verbindung, auf der das Passwort übertragen wird, physikalisch gegen Abhören geschützt ist. Dies kann für Telefonverbindungen angenommen werden, da hier nur das Personal der Firmen Zutritt zu den Vermittlungsstellen hat. Für LAN-, WAN- oder Internet-Verbindungen ist das in der Regel nicht der Fall, hier muss man anders vorgehen.

Das Challenge Handshake Authentication Protocol (CHAP) ist ein typisches Challenge-and-Response-Protokoll, wie es auch im Mobilfunk eingesetzt wird. Bei korrekter Implementierung ist es kryptographisch sicher. Voraussetzung ist, dass Client und Network Access Server (NAS) ein gemeinsames Geheimnis *secret* besitzen.

Der NAS muss hier zunächst eine *challenge*-Nachricht an den Client senden. Dieser berechnet $response = hash(secret, challenge)$ und sendet *response* an den NAS. Dieser überprüft, ob $response = hash(secret, challenge)$ ist. Als Hashalgorithmus ist bisher MD5 reserviert.

Viele weitere Vorschläge zu PPP findet man unter [ppp]. Dazu zählen Vorschläge, ein SSL-ähnliches Verfahren in der Authenticate-Phase einzusetzen, konkrete Implementierungen zu CHAP (MS-CHAP) und Vorschläge, wie man ein Schlüsselmanagement zur Verschlüsselung der PPP-Nutzlast realisieren könnte.

- The PPP Encryption Control Protocol (ECP) [Mey96]
- PPP Extensible Authentication Protocol (EAP) [BV98]
- The PPP DES Encryption Protocol, Version 2 (DESE-bis) [SM98b]
- The PPP Triple-DES Encryption Protocol (3DESE) [Kum98]
- Microsoft PPP CHAP Extensions [ZC98]
- PPP EAP TLS Authentication Protocol [AS99]
- Microsoft PPP CHAP Extensions, Version 2 [Zor00]
- Microsoft Point-To-Point Encryption (MPPE) Protocol [PZ01]

Auf einige dieser Vorschläge gehen wir später noch ein. Zunächst aber noch ein Abschnitt darüber, wie die Authentisierung für ISPs skalierbar gemacht wird.

2.4 PPP-Verlängerungen

Wenn Außendienstmitarbeiter auf das Intranet der Firma zugreifen möchten, können sie im Prinzip jedes Sicherheitsprotokoll einsetzen: SSL, SSH oder IPSec. Diese Verfahren haben aber alle einen Nachteil: Zur Authentisierung von Nutzern in heterogenen Umgebungen liegen nur wenig Erfahrungswerte vor.

Dies ist aber gerade die Stärke von PPP: Die Authentisierungsfunktionen dieses Protokolls sind seit Jahren in großem Maßstab im Einsatz, und mit RADIUS oder TACACS+ liegen skalierbare Sicherheitsarchitekturen vor. Hinzu kommt, dass sich Außendienstmitarbeiter auch bislang schon über Modemverbindungen und PPP direkt ins Firmennetz einwählen konnten. Die Infrastruktur und die Erfahrung zur PPP-Authentisierung sind also vorhanden.

Die direkte Einwahl per Telefon und PPP hat nur einen, dafür aber gravierenden, Nachteil: die Kosten. Ein Mitarbeiter, der in Übersee seine E-Mails mit großen Anhängen von einem Server in München abruft, wird seine Spesenrechnung deutlich erhöhen.

Die Lösung des Problems ist naheliegend: Anstelle einer Telefonverbindung von Kapstadt nach München könnte er doch eine Internet-Verbindung nutzen, indem er sich lokal per PPP bei einem Internet Service Provider einwählt. Damit trotzdem PPP- und nicht IP-Pakete beim NAS der Firma ankommen, muss die PPP-Verbindung durch das Internet verlängert werden. Dazu gibt es prinzipiell zwei Möglichkeiten: Der Client des Mitarbeiters kann die PPP-Verbindung verlängern, oder der NAS des ISP vor Ort.

Client-initiierter Tunnel. Beim Client-initiierten Tunnel (Bild 2.3) besorgt sich der Client durch PPP-Einwahl beim lokalen ISP zunächst einmal eine IP-Adresse IP1.

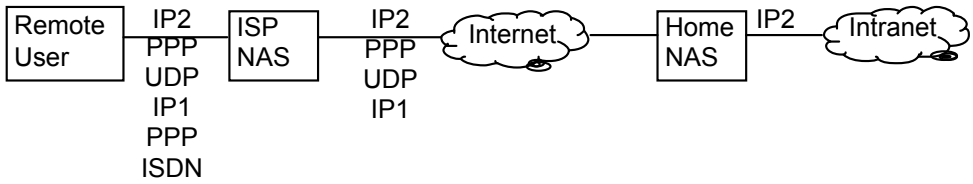

Abb. 2.3 Client-Initiierter Tunnel. Die Beschriftung der Verbindungen gibt einen typischen Protokollstack an. (Z.B. sind im Internet die IP2-Pakete in PPP-Rahmen verpackt, diese stecken in UDP-Transportsegmenten, die wiederum die Nutzlast eines IP1-Pakets bilden.)

Mit dieser Adresse kann er eine IP-Verbindung durch das Internet zum NAS („Home-NAS“) seiner Firma herstellen.

Über diese IP-Verbindung sendet er dann PPP-Pakete. Die IP-Verbindung vom Client zum Home-NAS wird also als eine Art „virtuelle Telefonverbindung“ behandelt.

Nach erfolgreicher Authentisierung durch den Home-NAS erhält er eine zweite IP-Adresse, IP2. Dies ist dann eine IP-Adresse aus dem Adressbereich der Firma. Mit dieser Adresse kann er sich im Intranet frei bewegen. Im Internet müssen die IP-Pakete mit Adresse IP2 in PPP-Rahmen und letztlich auch in IP-Paketen der Adresse IP1 eingepackt sein.

Bei dieser Art der Tunnelung bestehen gleichzeitig zwei PPP-Verbindungen: Eine zwischen dem Client und dem NAS des ISP, und eine zwischen dem Client und dem NAS der Firma. Da die zweite Verbindung in die erste verpackt ist, und zu dieser Verpackung auch noch ein IP-Header und ein UDP-Header (oder ein anderes Transportprotokoll) gehören, ist der Protokolloverhead beträchtlich.

NAS-initiierter Tunnel. Bei einem NAS-initiierten Tunnel (Bild 2.4) gibt es nur eine PPP-Verbindung: Die PPP-Verbindung, die der Client mit dem NAS des ISP aufbaut, wird von diesem durch das Internet an den Home-NAS weitergeleitet. Das spart Protokoll-Overhead, wie man beim Vergleich der Bilder 2.3 und 2.4 am kleineren Protokoll-Stack sehen kann. Ein NAS-initiierter Tunnel kann allerdings nur dann aufgebaut werden, wenn die Network Access-Server des ISP dies unterstützen.

Mit diesen beiden Lösungsmöglichkeiten hat man die Stärken von PPP voll ausgenutzt, seine Schwächen aber noch nicht beseitigt:

- Werden die PPP-Rahmen über das Internet transportiert, statt über eine Telefonleitung, so sind sie für eine große Zahl potenzieller Angreifer plötzlich sichtbar.

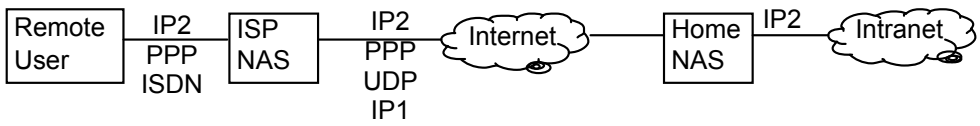

Abb. 2.4 NAS-initiierter Tunnel

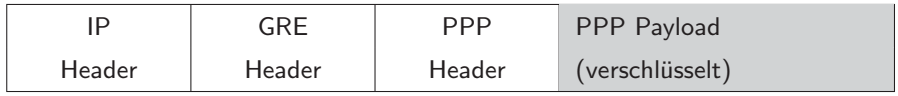

Abb. 2.5 Ein Paket des Point-to-Point Tunneling Protocol (PPTP). Die grau unterlegten Felder sind verschlüsselt.

Insbesondere sollten PAP-Pakete nicht unverschlüsselt über das Internet übertragen werden.

- Das Internet ist ein sehr viel unzuverlässigerer „Draht" als eine Telefonverbindung. Bei einer Modemverbindung hat PPP eine garantierte Datenrate, die es nutzen kann, im Internet nicht. Es wird noch eine Steuerleitung benötigt, um diese Effekte abzufangen.

Zwei große Firmen hatten ungefähr gleichzeitig diese Probleme erkannt und präsentierten ihre Lösungen: Microsoft mit dem *Point-to-Point Tunneling Protocol* (PPTP), und Cisco mit dem Layer 2 Forwarding Protocol (L2F). Dass das nicht gut gehen konnte, lag auf der Hand, denn Microsoft konnte PPTP auf dem NAS-Markt nicht durchsetzen, und Cisco hatte keinen Einfluss auf die Windows-Betriebssysteme. Man einigte sich daher relativ problemlos auf einen gemeinsamen Standard, das Layer 2 Tunneling Protocol (L2TP).

PPTP. Hier kurz die Eckdaten von PPTP, das wir uns später im Rahmen des von Mudge und Schneier [SM98a] entdeckten kryptologischen Angriffs noch einmal näher anschauen werden:

PPTP verlängert PPP mit Hilfe von Generic Routing Encapsulation (GRE, [HLFT94]), einem Protokoll, das zwischen Routern zum Aufbau von IP-in-IP-Tunneln verwendet wird. Kontrollnachrichten werden bei PPTP mit Hilfe von TCP übertragen.

Verschlüsselung und Authentikation finden auf PPP-Ebene statt („link encryption"), d.h. der PPP-Payload selbst wird verschlüsselt. Der kryptographische Schlüssel zum Verschlüsseln der Nutzlast wird bei MS-CHAP aus dem Client und NAS bekannten Passwort abgeleitet. Es gibt aber auch Ansätze (und PPP bietet dazu die Möglichkeit), ein SSL-ähnliches Schlüsselmanagement einzusetzen. Dies wird unter dem Namen EAP-TLS in [SAH08] beschrieben.

L2TP. Das Layer 2 Tunneling Protokoll wurde als „Best of" PPTP und L2F beschrieben. Wichtig ist, dass es einen einheitlichen Standard gibt.

- In L2TP werden PPP-Pakete in UDP gekapselt. Auch die Kontrollnachrichten verwenden UDP, was gegenüber der Verwendung von GRE und TCP bei PPTP eine Vereinheitlichung bedeutet.

Abb. 2.6 Ein Paket des L2TP-Protokolls. Die grau unterlegten Felder sind mit IPSec ESP verschlüsselt.

- Die Authentifikation, die ja die Stärke von PPP darstellt, findet weiter auf PPP-Ebene statt (PAP, CHAP, ...).
- Die Verschlüsselung der Daten überlässt man hier aber IPSec ESP. Dies mag mit den Sicherheitsproblemen von PPTP zusammenhängen, aber auch mit der Tatsache, dass die Microsoft-Kryptomechanismen nur schwer in andere Umgebungen zu übertragen sind.

Weitere Informationen zu L2TP findet man bei der IETF unter [l2t].

2.5 Der PPTP-Angriff von Mudge und Schneier

Die Sicherheitsmechanismen von PPTP wurden von Microsoft aus den bereits in den Windows-Betriebssystemen vorhandenen Mechanismen heraus entwickelt. Wenn eine solche Vorgabe, für das neue Protokoll möglichst wenig an den alten Mechanismen zu ändern, noch mit der Forderung nach Rückwärtskompatibilität zusammenfällt, so ergeben sich daraus Gefahren für die Sicherheit des Ganzen.

1998 konnten Bruce Schneier und Mudge [SM98a] den Nachweis erbringen, dass PPTP (damals in Version 1) tatsächlich gravierende Sicherheitsmängel besitzt, die auf die oben genannten Vorgaben zurückzuführen sind. Diese Sicherheitsmängel können dazu führen, dass ein passiver Angreifer, der nur die Kommunikation zwischen PPTP-Client und NAS belauscht, das gemeinsame Geheimnis der beiden berechnen und das System so vollständig kompromittieren kann.

2.5.1 Wörterbuchangriffe

Grundlage des Angriffs ist ein sogenannter Wörterbuchangriff. Diese Attacken wurden entwickelt, um Passwortdateien zu knacken, in denen nicht das Passwort selbst, sondern nur sein Hashwert abgespeichert ist.

Die Sicherheit von Passwörtern ist ein schwieriges Problem der Internetsicherheit: Einerseits sind Username/Password-Verfahren als Authentifizierungsverfahren im Internet weit verbreitet, und erst langsam werden Alternativen dazu akzeptiert (vgl. Abschnitt 11.4). Andererseits ist ein Passwort selbst relativ leicht zu stehlen: Mit einer Keylogger-Schadsoftware können Tastaturanschläge mitgelesen werden; mit einem Cross-Site-Scripting-Angriff kann der Eingabewert eines Passwort-Feldes ausgelesen werden; und über unverschlüsselte Verbindungen (oder mit Hilfe eines Man-in-the-middle-Angriffs) können Passwörter mitgelesen werden. In all diesen Fällen hilft es nichts, „starke“ Passwörter zu verwenden, da eine Wahl von starken Passwörtern nur gegen das „Raten“ von Passwörtern, und gegen Wörterbuchangriffe schützt.

Bei einem der oben genannten Angriffe auf den Client kann man nur relativ wenige Passworte erbeuten. Ein solcher Angriff lohnt sich daher eigentlich nur gegen hochwertige Webanwendungen wie z.B. Onlinebanking (wo dies in der jüngsten Vergangenheit

ja relativ häufig der Fall war). Wesentlich attraktiver sind Angriff auf Webserver, auf denen Abertausende von Passwörtern gespeichert sind. Dass dies relativ häufig vorkommt, kann man z.B. der Webseite [Daz] entnehmen.[1] Als grundlegende Sicherheitsmaßnahme speichert man daher auf Servern fast nie die Passwörter selbst ab, sondern nur Hashwerte dieser Passwörter.

Um einen Nutzer anhand von Username/Password zu authentifizieren, muss ein Webserver wie folgt vorgehen: Erhält er z.B. die Kombination $(Bob, seCretpaSsword)$, so muss er zunächst in seiner Datenbank den Username Bob suchen. Dort ist der Hashwert 0xd2febd6589d223edc9ac33ba26396a19b0b888b0 abgespeichert. Der Webserver vergleicht nun $hash(seCretpaSsword)$ mit diesem Wert und gibt bei Gleichheit den Zugriff auf das Konto von Bob frei.

Passwort	**SHA-1(Passwort)**
allissecret	d89d588db39f1ebfaf841c4b0962a6e60a974367
seCretpaSsword	d2febd6589d223edc9ac33ba26396a19b0b888b0
carolspassword	5f23413de0e51a1a9c0ec9ab50d26453d0e80782
verylongandsecurepassword	4a58bce3ba0b01c19214536d38c83308b4098cf2

Tab. 2.1 Passwörter und ihre SHA-1-Hashwerte: Das Passwort-Hashwert-Wörterbuch.

Bei einer Wörterbuch-Attacke versucht man, ein „Hashwert-Passwort"-Wörterbuch zu erstellen, ähnlich wie bei einer Fremdsprache: Wenn man ein englisches Wort nicht kennt, so schlägt man es im „Englisch-Deutsch"-Wörterbuch nach. Wenn man das Passwort zu einem Hashwert haben möchte, so schlägt man im „Hashwert-Passwort"-Wörterbuch nach.

SHA-1(Passwort)	**Passwort**
4a58bce3ba0b01c19214536d38c83308b4098cf2	verylongandsecurepassword
5f23413de0e51a1a9c0ec9ab50d26453d0e80782	carolspassword
d2febd6589d223edc9ac33ba26396a19b0b888b0	seCretpaSsword
d89d588db39f1ebfaf841c4b0962a6e60a974367	allissecret

Tab. 2.2 Passwörter und ihre SHA-1-Hashwerte: Das hexadezimal aufsteigend sortierte Hashwert-Passwort-Wörterbuch. Der dritte Eintrag entspricht dem Hashwert von Bob's Passwort, und so kann der Angrifer die Login-Daten (Bob,seCretpaSsword) ermitteln.

Ein solches Wörterbuch kann man unter bestimmten Voraussetzungen effizient erstellen:

- **Es darf nicht zu viele verschiedene Passwörter geben.** Der Begriff „nicht zu viele" muss hier in kryptographischen Größenordnungen gemessen werden:

[1] Dort findet man auch eine Liste der 2.151.220 häufigste Passwörter, die man für einen Wörterbuchangriff verwenden kann.

Das oben erwähnte Wörterbuch mit 2.151.220 verschiedenen Passwörtern scheint zunächst groß zu sein, aber ein sortiertes Hashwert-Passwort-Wörterbuch zu erstellen würde nur etwa 2^{26} Operationen benötigen, was bei einem Angriff auf Verschlüsselung einer vollständigen Schlüsselsuche für einen Schlüssel der Länge 26 Bit entsprechen würde.

- **Es wird jeweils nur das Passwort gehasht (ohne „Salt“).** Eine wirkungsvolle Schutzmaßnahme gegen Wörterbuchangriffe ist, nicht nur das Passwort zu hashen, sondern auch noch ein für jedes Passwort zufällig gewähltes *salt*. Auf dem Server würde dann $(username, salt, hash(password||salt))$ abgespeichert. Dies verhindert einen Wörterbuchangriff zwar nicht vollständig, zwingt den Angreifer aber, für jeden solchen Eintrag ein eigenes vollständiges Wörterbuch zu verwenden, bei dem *salt* an jedes Passwort im Wörterbuch angehängt wird.

Sind beide Voraussetzungen gegeben, so wird der Angriff wie folgt durchgeführt:

1. Der Angreifer „hackt“ sich beim Webserver ein und kopiert die Datei/Datenbank mit den Einträgen (Nutzername, Hashwert des Passworts).
2. Der Angreifer bildet den Hashwert aller Worte aus einem geeigneten Wörterbuch WB. Ist $n = |WB|$, so sind dazu n Hashwertbildungen erforderlich. Dies ist Tabelle 2.1.
3. Die Paare (Passwort, Hashwert) werden nach Hashwert sortiert. Um eine Liste mit n Einträgen zu sortieren, benötigen effiziente Sortieralgorithmen (z.B. Quicksort oder Heapsort) ungefähr $n \cdot log(n)$ Vertauschungsoperationen. Dies ist Tabelle 2.2.
4. Der Angreifer nimmt nun die Hashwerte aus der gehackten Datenbank und sucht diese in Tabelle 2.2. Unter Verwendung des „divide-and-conquer“-Algorithmus sind hierzu jeweils nur $log(n)$ Vergleiche erforderlich. Wird ein passender Hashwert gefunden, so kann der Angreifer das dazugehörige Passwort als zweiten Eintrag aus Tabelle 2.2 entnehmen, und hat zusammen mit dem Nutzernamen aus der Datenbank alle Werte, um das Konto des Opfers auf dem Webserver zu übernehmen.

Was mussten Mudge und Schneier nun tun, um diesen Angriff auf PPTP anzuwenden? Zur Generierung der kryptographischen Schlüssel wird meist auf das einzige Geheimnis zurückgegriffen, das PPTP-Client und Server noch aus PAP-Zeiten teilen: das Passwort. Aus dem Passwort werden (ohne „salt“) bei PPTP zwei verschiedene Hashwerte gebildet. Die Grundvoraussetzungen für einen Wörterbuch-Angriff sind also gegeben.

2.5.2 Übersicht PPTP-Authentifizierungsoptionen

Zur Authentisierung und ggf. Verschlüsselung gibt es bei PPTP drei verschiedene Optionen:

1. Passwort im Klartext senden/keine Verschlüsselung möglich: Diese unsichere Variante sollte auf keinen Fall eingesetzt werden.

2. Hashwert des Passworts senden/ keine Verschlüsselung möglich: Hier kann der Hashwert des Passworts abgehört und im Wörterbuch nachgeschlagen werden.
3. MS-CHAP: Die Hashwerte des Passworts werden zum Verschlüsseln der Challenge benutzt; dieser verschlüsselte Wert stellt die Response dar (vgl. Abbildung 2.8). Verschlüsselung ist möglich. Aber auch bei dieser anscheinend sicheren Variante kann man aus der Response und der Challenge einen Hashwert und anschließend auch das Passwort berechnen [SM98a].

2.5.3 Angriff auf Option 2

Das Erstellen eines Wörterbuchs ist relativ einfach. In PPTP werden zwei verschiedene Hashfunktionen benutzt: Die alte LAN-Manager-Hashfunktion (aus Gründen der Rückwärts-Kompatibilität auf NAS-Seite), und die neuere WindowsNT-Hashfunktion. Das Wörterbuch wird für die LM-Hashfunktion erstellt, weil es hier besonders einfach ausfällt (vgl. Bild 2.7):

- Passwörter können nie länger als 14 Byte sein, weil alle Bytes ab dem 15. Zeichen nicht verwendet werden. (Dies gilt auch für den WinNT Hash.)
- In der Regel sind Passwörter kürzer als 14 Zeichen. Durch das Auffüllen mit Nullen gibt es eine Standard-Methode zur Verlängerung, die das Wörterbuch nicht vergrößert.
- Die Umwandlung der Klein- in Großbuchstaben verringert den Umfang des Wörterbuchs erheblich.
- Durch das Aufspalten in zwei 7-Byte-Hälften kann man statt eines Wörterbuchs zwei wesentlich kleinere Wörterbücher einsetzen, indem man die ersten 7 Byte des modifizierten Passworts mit den ersten 8 Byte des Hashwerts vergleicht, und analog auch für die zweite Hälfte vorgeht.

Als Ergebnis einer erfolgreichen Wörterbuchattacke erhält man das modifizierte Passwort, das aus den ersten 14 Zeichen des echten Passworts in Großschreibung besteht. Die korrekte Groß- und Kleinschreibung kann man dann mit dem zweiten Hashwert ermitteln, indem man alle möglichen Varianten (höchstens 2^{14}) ausprobiert.

2.5.4 Angriff auf Option 3

Mit der Erstellung der beiden Wörterbücher für den LAN-Manager-Hash ist die Authentisierungsvariante 2 geknackt. Es bleibt Variante 3, bei der der Hashwert nie übertragen wird und somit auch nicht abgehört werden kann. Durch ungeschickte Wahl der Verschlüsselungsfunktion für die Challenge kann man aber hier die Hashwerte aus der Challenge und der Response mit vertretbarem Aufwand berechnen.

Die Erzeugung der Response ist in Bild 2.8 dargestellt. Es werden zwei verschiedene Responses berechnet: Eine mit dem *LAN-Manager-Hash* (für alte Versionen des NAS),

http://www.springer.com/978-3-658-06543-0

Sicherheit und Kryptographie im Internet
Theorie und Praxis
Schwenk, J.
2014, XV, 351 S. 211 Abb., Softcover
ISBN: 978-3-658-06543-0