hakin9

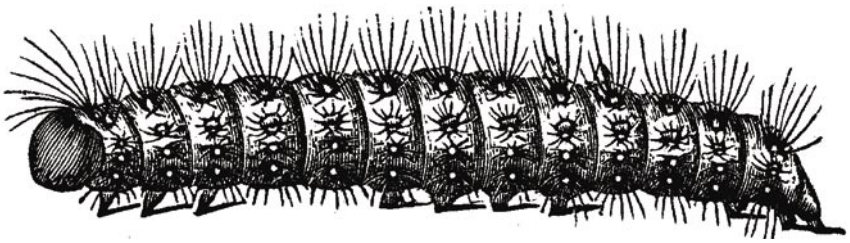
Wie Spam verschickt wird

Tomasz Nidecki

Der Artikel wurde in der Ausgabe 2/2004 des Magazins "Hakin9" publiziert.
Alle Rechte vorbehalten. Kostenlose Vervielfältigung und Verbreiten des Artikels ist in unveränderter Form gestattet.
Das Magazin “Hakin9”, Wydawnictwo Software, ul. Lewartowskiego 6, 00-190 Warszawa, piotr@software.com.pl

Wie Spam verschickt wird

Tomasz Nidecki



Spammer nutzen oft schlecht abgesicherte Systeme aus. Die beim Verschicken von sogar mehreren Zehn- oder Hunderttausend E-Mails entstehenden Folgen und Kosten werden auf Dritte übertragen. Sehen wir uns die von Spammern angewendeten Techniken an und überlegen wir uns, wie wir uns davor schützen können.

Große Massen von E-Mails zu verschicken ist sehr ressourcenaufwändig; eine leistungsfähige Leitung und ein dedizierter Rechner sind unumgänglich. Auch wenn der Spammer über diese Ressourcen verfügt, kann die ganze Operation Stunden dauern. Auch ist der ISP selten entzückt, dass seine Leitungen zum Spammen benutzt werden und der Spammer kann vom Netz getrennt werden, noch bevor er viele E-Mails verschicken konnte. Wenn er geschnappt wird, drohen ihm auch schwerwiegende rechtliche oder finanzielle Konsequenzen.

Um das Senden ihrer Nachrichten zu beschleunigen und effizienter zu organisieren, bedienen sich Spammer vor allem zwei Methoden.

Die erste ist, die Zeit für das Versenden einer E-Mail zu minimieren. Sie heißt *fire and forget – schieß und vergiss*. Der Rechner des Spammers wartet dann nicht auf die Antworten von den kontaktierten Vermittlungsservern.

Die zweite Methode besteht im Diebstahl von Ressourcen anderer Benutzer, die ihre Systeme undicht konfiguriert haben oder einem Virus zum Opfer gefallen sind. Sie tragen dann die meisten Kosten, und oft auch die Verantwortung für das Verschicken von Spam, und der eigentliche Spammer bleibt unbestraft.

Das Protokoll SMTP

Um die Methoden der Spammer verstehen zu können, müssen wir die Regeln kennen lernen, nach denen das häufigste Protokoll der E-Mail-übertragung funktioniert – und zwar das Protokoll SMTP. Es basiert, wie die meisten üblichen Internet-Protokolle, auf einfachen Textbefehlen.

Aus diesem Artikel erfahren Sie...

- wie Spammer ihren Spam (über die Rechner unschuldiger Benutzer) verschicken,
- wie Sie Ihren Server vor dem Missbrauch durch Spammer schützen,
- wie das SMTP-Protokoll funktioniert,
- was ein *Open Relay*, ein *Open Proxy* und ein Zombie ist.

Was Sie vorher wissen sollten...

- wie man die wichtigsten Tools unter Linux verwendet.

Für Einsteiger

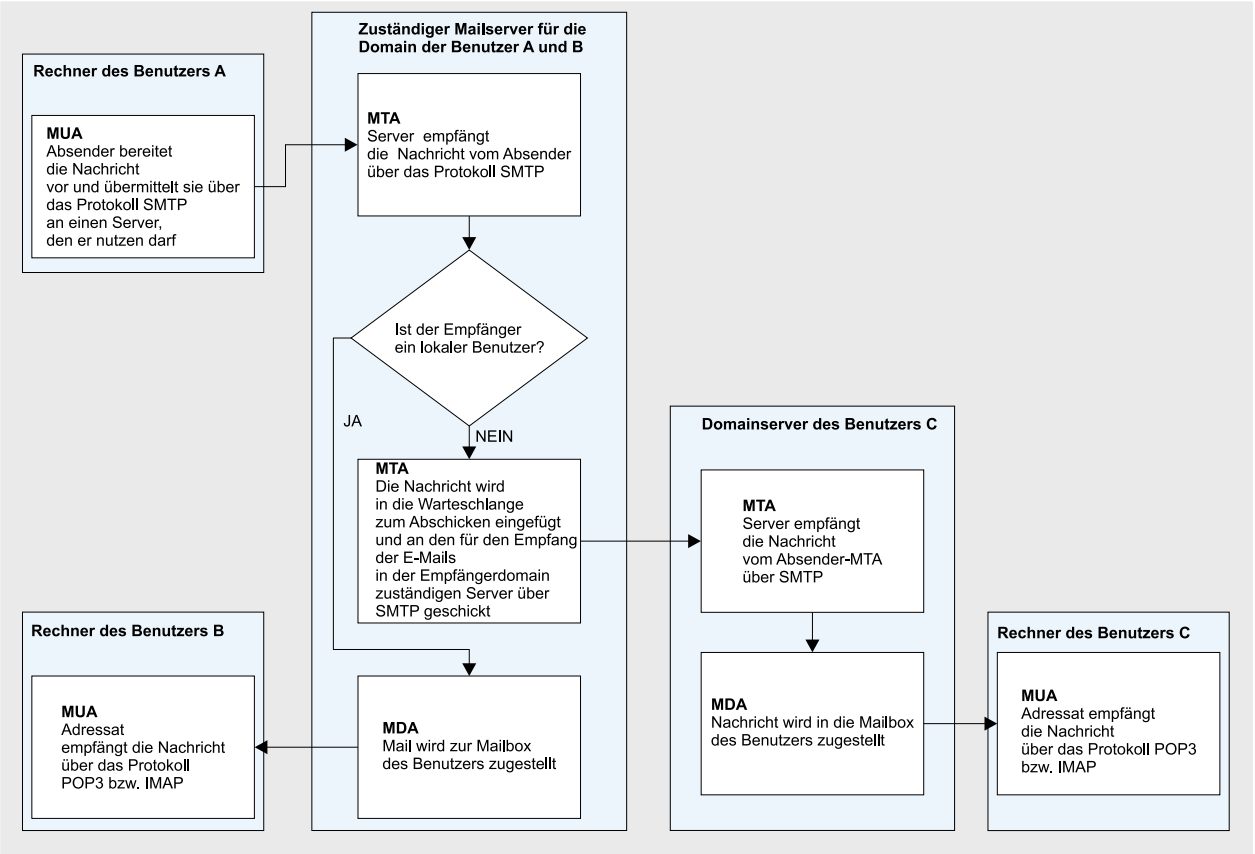


Abbildung 1. *Phasen des E-Mail-Versands*

Phasen des E-Mail-Versands

E-Mails werden in mehreren Etappen übertragen (siehe Abbildung 1). Um es besser zu verstehen, sehen wir uns ein Beispiel an, in dem wir eine Nachricht von *hakin9@hakin9.org* an *nobody@example.com* schicken wollen. Der Absender verwendet das Programm *Mozilla Thunderbird* im Lokalnetzwerk, und der Empfänger *Microsoft Outlook Express* über eine Dialup-Verbindung.

Im ersten Schritt kontaktiert *Mozilla Thunderbird* den in den Einstellungen des Benutzeraccounts *hakin9@hakin9.org* angegebenen SMTP-Server *mail.software.com.pl*. Die Nachricht wird an den Server mit dem Protokoll SMTP geschickt.

Anschließend werden die DNS-Einträge von *mail.software.com.pl* für die Domain *example.com* überprüft und wir erfahren, dass der Server *mail.example.com*. für die Verarbeitung von E-Mails für diese Domain zuständig ist.

Diese Information findet er in dem sog. MX-Eintrag (*Mail Exchanger*), wie von dem DNS-Server für die Domain *example.com* bekannt gegeben wird (wir können dieselben Angaben mit den Tools *host* und *dig* ermitteln: `host -t mx example.com` oder `dig example.com mx`).

Der dritte Schritt ist der Aufbau der Verbindung von *mail.software.com.pl*

Die Geschichte von SMTP

Der Wegbereiter für SMTP war das Programm *SNDMSG* (*Send Message*), das 1971 von Ray Tomlinson (in Verbindung mit seinem eigenen Projekt – *CYPNET*) in einer Anwendung zum Übertragen von E-Mails in dem Netzwerk *ARPANET* eingesetzt wurde. Ein Jahr später wurde die in *ARPANET* zum Transfer von Dateien verwendete Software, *FTP*, um die Befehle `MAIL` und `MLFL` erweitert. E-Mails wurden bis 1980 mit *FTP* verschickt – damals ist der erste Protokollstandard für elektronische Post erarbeitet worden – MTP (*Mail Transfer Protocol*) wurde in dem Dokument RFC 772 beschrieben. MTP wurde mehrmals modifiziert (RFC 780, 788), und im Jahre 1982 hat Jonathan B. Postel in der RFC 821 *Simple Mail Transfer Protocol* vorgeschlagen.

In seiner Grundform hat SMTP allerdings nicht alle Erwartungen erfüllt, es sind also zahlreiche Dokumente mit Protokollerweiterungen entstanden. Die wichtigsten sind:

- RFC 1123 – Anforderungen an Internetserver (auch SMTP-Server),
- RFC 1425 – der Standard für SMTP-Erweiterungen – ESMTP,
- RFC 2505 – eine Sammlung von Vorschlägen zum Spamschutz an Servern,
- RFC 2554 – Authentifizierung der Verbindungen – Einführung des Befehls `AUTH`,

Der aktuelle SMTP-Standard ist 2001 in der RFC 2821 beschrieben worden. Der komplette Satz von RFC's ist auf unserer Heft-CD zu finden.



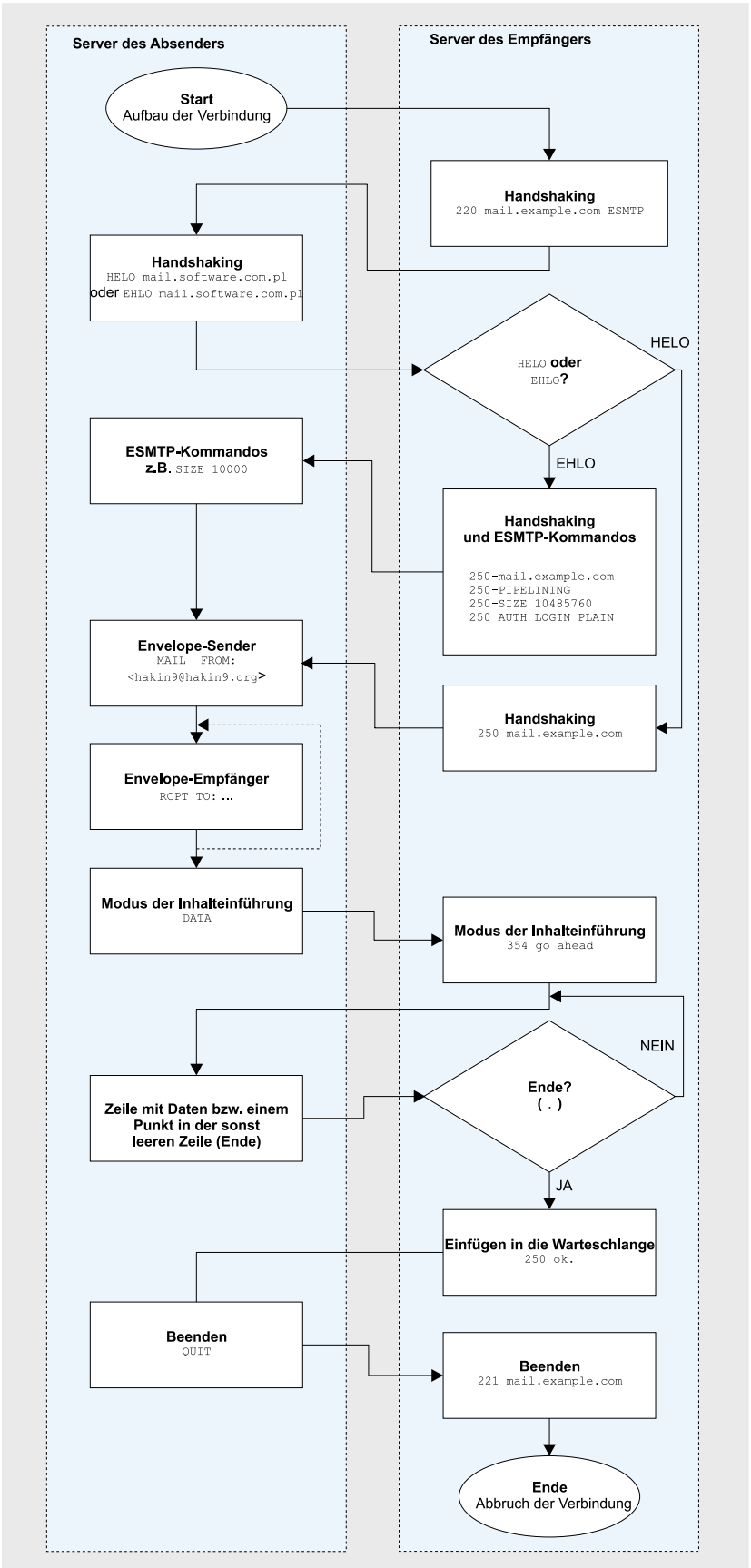


Abbildung 2. *Phasen der SMTP-Kommunikation*

zu *mail.example.com* und die Übertragung der Mail. In der vierten Phase stellt *mail.example.com* die empfangene Nachricht ins lokale Postfach des Benutzers *nobody* zu. Im fünften Schritt kontaktiert der Benutzer *nobody* über seine Dialup-Verbindung den Server *mail.example.com* über das Protokoll POP3 (bzw. IMAP) mit Hilfe des Programms *Outlook Express* und downloadet die E-Mail.

Es kommt auch vor, dass die Nachricht eine längere Strecke zurücklegt. Der Absender kann etwa separate Mailserver nutzen, zum Beispiel: *empfang.software.com.pl* und *senden.software.com.pl*. Dann wird die E-Mail vom Benutzer (*empfang.software.com.pl)* empfangen, weitergeleitet (*senden.software.com.pl*), und an *mail.example.com* geschickt. Ähnlich funktioniert es bei *mail.example.com* – für den Empfang und die Zustellung der Nachrichten an Benutzer sind separate Server zuständig.

Am Versand von E-Mail beteiligte Anwendungen

Zum Versand von E-Mail werden mehrere Programme eingesetzt:

- Das Programm des Endnutzers, mit dem E-Mails empfangen und geschickt sowie gelesen und erstellt werden, heißt MUA – *Mail User Agent*. Beispiele von MUA's sind: *Mozilla Thunderbird*, *Outlook Express*, *PINE*, *Mutt*.

Der Nachfolger von SMTP?

Prof. Dan Bernstein, der Autor von *qmail*, hat ein Protokoll mit dem Namen QMTP (*Quick Mail Transfer Protocol*) erarbeitet, das SMTP ersetzen sollte. Es behebt viele Mängel von SMTP, ist damit allerdings inkompatibel. Leider ist es auch praktisch nirgends außer in *qmail* implementiert worden. In der Praxis wird es u.a. in den Mailservern des polnischen Portals Wirtualna Polska eingesetzt.

Mehr über QMTP können Sie unter: *http://cr.yp.to/proto/qmtp.txt* nachlesen

Tabelle 1. *Die Übersicht der häufigsten Befehle des SMTP-Protokolls*

Befehl	Beschreibung
HELO <FQDN>	Der Client stellt sich dem Server vor
EHLO <FQDN>	Der Client stellt sich dem Server vor und bittet um die Liste der verfügbaren ESMTP-Befehle
MAIL FROM:<Adresse>	Angabe der Envelope-Absenderadresse – der Adresse, an die die E-Mail bei unmöglicher Zustellung zurück geschickt werden kann
RCPT TO:<Adresse >	Angabe der Empfängeradresse für die E-Mail
DATA	Wechsel zum Empfang des Mailinhalts
AUTH <Methode>	Authentifizierung der Verbindung (ESMTP) – die häufigsten Methoden sind: LOGIN, PLAIN und CRAM-MD5

Eine erweiterte Liste der SMTP- und ESMTP-Befehle ist unter http://fluffy.codeworks.gen.nz/esmtp.html zu finden.

- Der Serverteil, der sich mit der Kommunikation mit Benutzern (Empfangen von E-Mails) und mit dem Abschicken sowie Empfangen von Nachrichten von anderen Servern beschäftigt, ist der MTA – *Mail Transfer Agent*. Übliche MTA's sind: *Sendmail*, *qmail*, *Postfix*, *Exim*.
- Der Serverteil, der die E-Mails an lokale Benutzer zustellt, wird als MDA – *Mail Delivery Agent* bezeichnet. Beispiele selbständiger MDA's sind: *Maildrop*, *Procmail*. Die meisten MTA's verfügen über eigene Mechanismen der Mailzustellung an lokale Benutzer, also sind separate MDA's nicht notwendig.

Phasen der SMTP-Kommunikation

Das Verschicken einer E-Mail mit dem Protokoll SMTP erfolgt in mehreren Phasen. Nun besprechen wir eine SMTP-Beispielsitzung zwischen den Servern *mail.software.com.pl* und *mail.example.com*. Die von *mail.software.com.pl* geschickten Daten sind mit dem Zeichen >, und die von *mail.example.com* empfangenen mit < markiert.

Nachdem die Verbindung aufgebaut worden ist, stellt sich *mail.example.com* vor:

```
< 220 mail.example.com ESMTP Programm
```

indem er informiert, dass sein kompletter Hostname (FQDN) *mail.example.com* lautet. Wir erfahren auch, dass wir die Befehle des ESMTP (erweiterten SMTP – siehe Kasten) verwenden können und dass der verwendete MTA *Programm* ist. Der Programmname ist optional – einige MTA's, beispielsweise *qmail*, geben ihn nicht an. Wir stellen uns vor:

```
> HELO mail.software.com.pl
```

und erhalten als Antwort:

```
< 250 mail.example.com
```

also eine Mitteilung, dass *mail.example.com* bereit ist, E-Mails zu empfangen. Dann geben wir die sog. Envelope-Absenderadresse an – die E-Mailadresse, an die die E-Mail im Falle einer Abweisung zurück geschickt werden soll:

```
> MAIL FROM:<hakin9@hakin9.org>
< 250 ok
```

Tabelle 2. *Die wichtigsten Antwortcodes in SMTP*

Code	Beschreibung
220	Server bereit – der Server begrüßt uns und informiert, dass wir ihm Befehle zuschicken können
250	Befehl ausgeführt
354	Der Empfang des Mailinhalts startet
450	Das Postfach ist zeitweise nicht verfügbar (z.B. weil es ein anderer Prozess gesperrt hat)
451	Lokaler Fehler bei der Verarbeitung der E-Mail
452	Zeitweise ungenügend Festplattenspeicher verfügbar
500	Befehl nicht erkannt
501	Syntaxfehler im Befehl oder in seinen Parametern
502	Befehl nicht implementiert
550	Kein Zugriff auf das Postfach
552	Zugewiesene Festplattenquota überschritten

Eine komplette Liste der Codes und die Regeln, nach denen sie gebildet werden, ist in der RFC 2821 (auf der Heft-CD) zu finden.



Listing 1. *Der einfachste Open Relay-Server*

```
$ telnet lenox.designs.pl 25
< 220 ESMTP xenox
> helo hakin9.org
< 250 xenox
> mail from:<hakin9@hakin9.org>
< 250 Ok
> rcpt to:<nobody@example.com>
< 250 Ok
> data
< 354 End data with ↵
  <CR><LF>.<CR><LF>
> Subject: Test
>
> Das ist ein Test
> .
< 250 Ok: queued as 17C349B22
> quit
< 221 Bye
```

Als nächstes geben wir die Empfängeradressen für die E-Mail an:

```
> RCPT TO:<test1@example.com>
< 250 ok
> RCPT TO:<test2@example.com>
< 250 ok
> RCPT TO:<test3@example.com>
< 250 ok
```

Dann kommen nach dem Kommando `DATA` die Header und der Inhalt der Nachricht. Die Header separieren wir vom Inhalt durch eine Leerzeile, und die ganze E-Mail beenden wir mir einem Punkt in einer separaten Zeile:

```
> DATA
< 354 go ahead
> From: niemand@hakin9.org
> To: alle@example.com
> Subject: Nichts
>
> Das ist ein Test
> .
< 250 ok 1075929516 qp 5423
```

Nachdem wir die Nachricht abgeschickt haben, können wir die Verbindung schliessen:

```
> QUIT
< 221 Bye
```

Der Server ist nicht immer im Stande, unsere Befehle zu befolgen. Wenn wir als Antwort einen Code mit der Ziffer 4 am Anfang erhalten (eine Code der Serie 4xx) bedeutet es, dass der Server die Verarbeitung der Nachricht zeitweise verweigert. Wir sollten versuchen, die E-Mail etwas später noch einmal zu schicken. Wenn der Antwortcode mit einer 5 beginnt, weist der Server unsere Nachricht endgültig ab und weitere Versuche bringen nichts. Die Listen der wichtigsten SMTP-Befehle und -Antwortcodes stehen in den Tabellen 1 und 2.

Listing 2. *Ein Open Relay-Server, von dem nur existierende Benutzer E-Mails absenden können*

```
$ telnet kogut.o2.pl 25
< 220 o2.pl ESMTP Wita
> helo hakin9.org
< 250 kogut.o2.pl
> mail from:<ania@o2.pl>
< 250 Ok
> rcpt to:<hakin9@hakin9.org>
< 250 Ok
> data
< 354 End data with ↵
  <CR><LF>.<CR><LF>
> Subject: Test
>
> Das ist ein Test
> .
< 250 Ok: queued as 31B1F2EEA0C
> quit
< 221 Bye
```

Listing 3. *Ein Multistage Open Relay-Server, von dem nur existierende Benutzer E-Mails absenden können*

```
$ telnet smtp.poczta.onet.pl 25
< 220 smtp.poczta.onet.pl ESMTP
> helo hakin9.org
< 250 smtp.poczta.onet.pl
> mail from:<ania@buziaczek.pl>
< 250 2.1.0 Sender syntax Ok
> rcpt to:<hakin9@hakin9.org>
< 250 2.1.5 Recipient address ↵
  syntax Ok; ↵
  rcpt=<hakin9@hakin9.org>
> data
< 354 Start mail input; ↵
  end with <CRLF>.<CRLF>
> Subject: Test
>
> Das ist ein Test
> .
< 250 2.6.0 Message accepted.
> quit
< 221 2.0.0 ↵
  smtp.poczta.onet.pl Out
```

Open Relay-Server

Als das Protokoll SMTP entstand, war Spam noch kein Problem und User konnten Nachrichten über beliebige Server in alle Welt schicken. Jetzt, wo Spammer jede Gelegenheit suchen, Tausende von E-Mails über fremde Server zu versenden, bewährt sich dieser Ansatz nicht mehr. Server, die E-Mails ohne Autorisierung versenden lassen, werden als *Open Relay*-Server bezeichnet.

Jeder Server, der nicht autorisierte Benutzer E-Mails versenden lässt, wird früher oder später zum Ziel der Spammer. Das kann dann schwerwiegende Folgen haben. Erstens kann es die Leistung des Servers beeinträchtigen, da er – statt die Mails autorisierter Benutzer zu verarbeiten – mit dem Verschicken von Spam beschäftigt ist. Zweitens kann der ISP den Vertrag kündigen, da der Server für illegale und unmoralische Zwecke genutzt wird. Drittens kommt die IP-Adresse des Servers auf schwarze Listen und viele andere Server akzeptieren keine E-Mails mehr von ihm (die IP-Adresse aus schwarzen Listen zu entfernen ist schwierig, und manchmal sogar unmöglich).

Listing 4. *Received-Header in einer über einen Multistage Open Relay Server verschickten E-Mail*

```
Received: from smtp8.poczta.onet.pl (213.180.130.48)
  by mail.hakin9.org with SMTP; 23 Feb 2004 18:48:11 -0000
Received: from  mail.hakin9.org ([127.0.0.1]:10248 "helo hakin9.org")
  by ps8.test.onet.pl with SMTP id <S1348420AbUBWSrW>;
  Mon, 23 Feb 2004 19:47:22 +0100
```

Wie vermeiden wir es, ein *Open Relay* zu werden?

Das Protokoll SMTP lässt uns:

- E-Mails von einem Benutzer übernehmen (MUA) und an einen anderen Server schicken (MTA),
- E-Mails von einem anderen Server übernehmen (MTA) und an einen lokalen Benutzer zustellen,
- E-Mails von einem anderen Server übernehmen (MTA) und an noch einen anderen Server schicken (MTA).

Es gibt keinen Unterschied zwischen der Art und Weise, wie Nachrichten von MUA und MTA befördert werden. Am wichtigsten ist, ob die IP-Adresse vertrauenswürdig erscheint (z.B. sie befindet sich im lokalen Netzwerk) und ob sich der Adressat in einer lokalen oder in einer externen Domain befindet.

Das Versenden von Nachrichten nach außerhalb des Servers wird als *Relaying* bezeichnet. Damit Spammer unseren Rechner nicht für ihre Tätigkeit missbrauchen können, dürfen wir kein Relaying ohne Autorisierung erlauben. Daher müssen wir bei der Konfiguration des SMTP-Servers folgendes voraussetzen:

- Wenn die E-Mail für eine der von unserem Server bedienten Domains bestimmt ist, muss sie ohne Autorisierung akzeptiert werden
- Wenn die E-Mail von einem lokalen Benutzer (deren MUA sich auf dem Server befindet), einem Benutzer aus dem Lokalnetz bzw. einem Benutzer mit einer autorisierten, festen IP an einen externen Benutzer geschickt wird, kann die Mail ohne Autorisierung akzeptiert werden (obwohl sich die Forderung nach Autorisierung empfiehlt).
- Wenn die Nachricht von einem externen Benutzer (etwa einem mit einer dynamisch zugewiesenen IP-Adresse) an einen anderen externen Benutzer geschickt wird, ist Autorisierung erforderlich.

Ausnutzung der *Open Relays*

Mal sehen, wir einfach es ist, einen *Open Relay* zum Spammen auszunutzen. Als Beispiel verwenden wir einen fehlerhaft konfigurierten polnischen Server, mit dem Spammer ihr Gewerbe betreiben – *lenox.designs.pl*. Wie in Listing 1 demonstriert, müssen wir keine zusätzlichen Operationen unternehmen, um eine E-Mail abschicken zu können. Der Server betrachtet jeden Benutzer, der sich damit in Verbindung setzt, als berechtigt, Mails zu versenden. Dies ist der gefährlichste Typ von *Open Relays*, denn er lässt sich am leichtesten missbrauchen.

Es gibt auch andere, für Spammer etwas schwierigere Arten von *Open Relays*. Ein Beispiel wäre einer der fehlerhaft konfigurierten Mailserver des polnischen Portals *O2 – kogut.o2.pl*. Wie in Listing 2 präsentiert, muss man nur den Namen des Benutzers erraten, um sich für ihn auszugeben und eine E-Mail als dieser Benutzer abzusenden. Bei einigen Servern braucht man nur die lokale Domain zu nennen – der angegebene Benutzer muss gar nicht existieren.

Eine ähnliche Situation sehen wir in Listing 3 – auch hier handelt es sich um den Mailserver eines der größten polnischen Webportale – diesmal ist es *Onet* (das interessanterweise mit aktiver Spambekämpfung prahlt). Es sind die sog. *Multistage Open Relays*, bei denen Nachrichten auf einer IP empfangen und von einer anderen versendet werden.

Um das zu erfahren, müssen wir die `Received`-Header (siehe Kasten) der Mail analysieren. In Listing 4 z.B. wird die Nachricht von *ps8.test.onet.pl* (213.180.130.54) empfangen und an den Benutzer von *smtp8.poczta.onet.pl* (213.180.130.48) geschickt. Dies erschwert die Ermittlung, dass der Server als *Open Relay* konfiguriert ist, stört jedoch keineswegs beim Spamversand von diesem Server.

Ein anderer Typ von *Open Relay*-Servern sind Server mit mangelhaft konfigurierter Absenderauthentifizierung (SMTP-AUTH). Sie lassen uns E-Mails nach der Angabe eines beliebigen Benutzernamens und Passworts verschicken. Am häufigsten passiert so etwas den Newbies unter den Administratoren von *qmail*, die die Dokumentation für die SMTP-AUTH-Korrektur nicht gelesen haben und *qmail-smtpd* inkorrekt aufrufen.

Das gepatchte Programm *qmail-smtpd* bedarf drei zusätzlicher Argumente: Des FQDN, des Namens des Programms zur Passwortüberprüfung (das mit *checkpassword* kompatibel sein muss) und des zusätzlichen Parameters für das Passwortprüfprogramm. Beispiel: `qmail-smtpd hakin9.org /bin/checkpassword /bin/true`. Der häufigste

Received-Header

Die `Received`-Header sind ein Pflichtelement jeder Nachricht. Jeder Mailserver, von dem die E-Mail mitübertragen wird, fügt einen eigenen `Received`-Header über die schon vorhandenen hinzu. Wenn wir also die Header von unten nach oben lesen, lernen wir die Route der Nachricht vom Absender zum Empfänger kennen. Der Spammer kann natürlich über den vorhandenen Header eigene Felder einfügen, indem er seine Identität zu verschleiern und jemand anderem die Schuld in die Schuhe zu schieben sucht. Die Header, die mit Sicherheit echt sind, sind die vom Empfängerserver (ganz oben); alle übrigen können gefälscht sein.

Nur anhand der `Received`-Header können wir die IP des eigentlichen Absenders der Nachricht identifizieren und oft auch erkennen, ob sie mit einem *Open Relay* oder einem *Open Proxy* verschickt worden ist. Es ist allerdings nicht leicht, Header zu analysieren, denn es gibt keinen einheitlichen Standard zu ihrer Form und jeder Mailserver gibt die Header-Informationen in einer etwas anderen Reihenfolge an.



Listing 5. *Ein Open Relay Server mit fehlerhaft konfiguriertem SMTP-AUTH*

```
$ telnet mail.example.com 25
< 220 mail.example.com ESMTP
> ehlo hakin9.org
< 250-mail.example.com
< 250-PIPELINING
< 250-8BITMIME
< 250-SIZE 10485760
< 250 AUTH LOGIN PLAIN CRAM-MD5
> auth login
< 334 VXNlcm5hbWU6
> was_auch_immer
< 334 UGFzc3dvcmQ
> was_auch_immer
< 235 ok, go ahead (#2.0.0)
> mail from:<hakin9@hakin9.org>
< 250 ok
> rcpt to:<nobody@nowhere.com>
< 250 ok
> data
< 354 go ahead
> Subject: Test
>
> Das ist ein Test
> .
< 250 ok 1077563277 qp 13947
> quit
< 221 mail.example.com
```

Fehler ist, `/bin/true` als den zweiten Parameter anzugeben – dann ist die Passwortprüfung immer erfolgreich (egal, was für ein Benutzername und Passwort angegeben wurden).Der Spammer kann auch einen Wörterbuchangriff *(Dictionary attack)* versuchen – es ist also empfehlenswert, die SMTP-Passwörter nicht allzu banal werden zu lassen.

Open Proxy-Server

Ein anderer Typ von schlecht konfigurierten Servern, die von Spammern ausgenutzt werden können, sind *Open Proxy*, Proxyserver, mit denen sich nicht authentifizierte Benutzer verbinden können. *Open Proxy*- Server können mit verschiedener Software und verschiedenen Protokollen implementiert werden. Meistens ist es das Protokoll HTTP-CONNECT, es gibt aber auch *Open Proxies*, die Verbindungen über HTTP-POST, SOCKS4, SOCKS5 und andere ermöglichen.

Ein *Open Proxy* kann vom Spammer identisch ausgenutzt werden wie ein *Open Relay* – zum Verschicken unautorisierter E-Mails. Viele lassen zusätzlich die IP des Absenders verschleiern; so ein Proxy bildet also ein appetitliches Stück für das Spamgewerbe.

Ausnutzung der *Open Proxies*

In Listing 6 sehen wir ein Beispiel, wie ein *Open Proxy* HTTP-CONNECT -Verbindungen auf dem Port 80 bietet. Der größte Teil der Verbindung ist die Kommunikation mit einem *Open Relay* (dieselben Befehle, wie sie in Listing 2 dargestellt werden). Bevor wir jedoch die Verbindung zum SMTP-Server aufbauen, kontaktieren wir einen *Open Proxy* und verbinden uns von hier aus mit dem MTA. Beim Aufbau der Verbindung deklarieren wir die Verwendung von HTTP/1.0 als Kommunikationsmittel, müssen es aber gar nicht benutzen.

Die komfortabelste Situation für einen Spammer ist, einen *Open Proxy*-Server zu finden, der zugleich einen lokalen Mailserver installiert hat. In den meisten Fällen akzeptiert der MTA Verbindungen vom lokalen Proxy ohne eine Autorisierung zu erfordern, betrachtet sie also wie lokale Benutzer. Dann braucht der Spammer keinen einzigen *Open Relay*-Server zu kennen und kann komfortabel und anonym seine Umtriebe auf Kosten und Verantwortung Anderer ausüben, indem er die Ermittlung seiner Identität wesentlich erschwert (sei-ne IP-Adresse steht nur in den Logdateien des Proxyservers und der Empfänger unerwünschter

Listing 6. *Der Open Proxy-Server wird zum anonymen Mail-Versand über einen Open Relay-Server benutzt*

```
$ telnet 204.170.42.31 80
> CONNECT kogut.o2.pl:25 HTTP/1.0
>
< HTTP/1.0 200↵
  Connection established
< 220 o2.pl ESMTP Wita
> helo hakin9.org
< 250 kogut.o2.pl
> mail from:<ania@o2.pl>
< 250 Ok
> rcpt to:<hakin9@hakin9.org>
< 250 Ok
> data
< 354 End data with↵
  <CR><LF>.<CR><LF>
> Subject: Test
>
> Das ist ein Test
> .
< 250 Ok: queued as 5F4D41A3507
> quit
< 221 Bye
```

Wo finden Spammer die Adressen von *Open Relays* und *Open Proxies*?

Nachlässig abgesicherte Server auf eigene Faust zu suchen kann problematisch sein. Allerdings brauchen wir nur eine über einen *Open Relay*- bzw. *Open Proxy*-Server geschickte Spam-Mail, um das Gesuchte zu erfahren. Die Überprüfung, ob sich unter der gegebenen IP-Adresse ein *Open Relay* versteckt, kann etwa mit dem Skript *rlytest* (*http://www.unicom.com/sw/rlytest/*), und für *Open Proxies* mit dem Skript *pxytest* (*http://www.unicom.com/sw/pxytest/*) realisiert werden. Beide Tools sind auf unserer Heft-CD zu finden.

Spammer, denen an Effizienz gelegen ist, verwenden vor allem kostenpflichtige Datenbanken mit *Open Relay* und *Open Proxy* Adressen. Sie sind nicht schwer zu finden – in einer beliebigen Suchmaschine den Text *open proxy* bzw. *open relay* eingeben, dann auf einen der ersten Links auf der Seite klicken (etwa.: *http://www.openproxies.com/* – 20 USD pro Monat, *http://www.openrelaycheck.com/* – 199 USD für ein halbes Jahr).

Eine andere Methode ist, eine Zone mit Adressen von *Open Relays* und *Open Proxies* von einem DNSBL-Server zu beziehen (siehe Artikel *Serverseitige Spamabwehr*). Eine Liste solcher Server befindet sich u.a. unter *http://www.declude.com/junkmail/support/ip4r.htm*. Eine Zone kann mit dem Programm *host* gedownloadet werden: `host -l <Name_der_Zone> <Adresse_des_DNSBL>`. Leider bieten nicht alle DNSBL-Server die Möglichkeit, ganze Zonen zu beziehen.

E-Mails kann sie nur mit Hilfe des Administrators des Proxyservers herausfinden). Wenn dem Spammer sehr viel daran liegt, seine IP zu verschleiern, kann er mehrere kaskadierte *Open Proxies* verwenden, d.h. er baut eine Verbindung zu einem Proxy auf, von dort zur einem zweiten, dritten und erst zum Schluss mit dem Mailserver.

Zombies

Die neueste – und aggressivste – Methode, die Spammer zur Verlagerung der Kosten und der Verantwortung an Andere verwenden, sind die sog. Zombies. Diese Technik beruht auf einer Kombination eines Virus (Wurms) mit einem Trojaner. Das Ziel ist, einen *Open Proxy*-Server auf dem infizierten Rechner einzurichten. Auf diese Art und Weise entsteht ein enormes Netzwerk von anonymen *Open Proxies*, das von Spammern weltweit ausgenutzt werden kann.

Das bekannteste Beispiel für die Zombie-Methode sind die Viren der *Sobig*-Familie. Die Variante *Sobig.E* funktioniert etwa wie folgt:

- Das erste Segment infiziert den Rechner (nachdem der Benutzer den Mailanhang gestartet hat) und verschickt die Kopien von sich selbst an alle E-Mail-Adressen, die es in .txt- und .html-Dateien auf der Festplatte finden kann.
- Zwischen 19 und 23 Uhr UTC verbindet sich der Wurm mit einer der 22 in seinem Code aufgelisteten IP-Adressen auf dem UDP-Port 8998, um die Download-Adresse des zweiten Segments zu beziehen.
- Nachdem das zweite Segment (der Trojaner) gedownloadet worden ist, wird es installiert und gestartet; die IP-Adresse des infizierten Rechners wird an den Zombie-Autor geschickt und das dritte Segment wird gedownloadet.
- Das dritte Element ist eine modifizierte Version des Programms *Wingate*, das nach einer automatischen Installation einen *Open Proxy* auf dem angegriffenen Rechner einrichtet.

Weitere Informationen zum Funktionsprinzip der *Sobig*-Viren stehenunter *http://www.lurhq.com/sobig.html* bereit.

Die einzige wirksame Schutzmethode gegen Zombies ist, Antivirensoftware und IDS-Systeme (*Intrusion Detection System* – z.B. *Snort*) zu verwenden, die einen *Open Proxy* in unserem Netzwerk aufspüren können.

Vorsorge verhütet Nachsorge

Wie wir sehen, ist es gar nicht kompliziert, mangelhaft abgesicherte Server auszunutzen. Die Folgen können für den Administrator eines solchen Servers schwerwiegend sein, und der Spammer trägt höchstwahrscheinlich keine Konsequenzen. Daher können wir es uns nicht leisten, die Fragen der Absicherung unserer Server gegen Ausnutzung durch Spammer zu bagatellisieren.

Wenn wir also einen eigenen Proxyserver einrichten, sollten wir sofort sicherstellen, dass nur Benutzer aus dem lokalen Netzwerk ihn nutzen dürfen. Auch sollte unser Mailserver eine Authentifizierung verlangen, obwohl die größten polnischen Webportale hier kein gutes Beispiel liefern. Vielleicht ist es für unsere Benutzer etwas umständlicher – dass diese Lösung sinnvoll ist, wird wohl niemand bestreiten. ■

Zur Geschichte des Spams

Die Herkunft des Wortes *Spam* hängt mit der Bezeichnung des von der Firma Hormel Foods hergestellten Dosen-Hackfleiches namens "SPAM" zusammen. Der Markenname SPAM ist eine Abkürzung von "*Shoulder Pork and Ham*" oder "*SPiced hAM*".

Wie hat man eigentlich Hackfleisch in der Dose mit der unerwünschten Post in Verbindung gebracht? Eine Teilschuld tragen hier die Macher der Serie: *Monty Python's Flying Circus*. Ein Ihrer Sketsche präsentiert ein Restaurant, in dem die Eigentümerin SPAM empfiehlt, aus dem jedes Gericht besteht. Zu Gast ist gerade eine Wikingerhorde, die die Eigentümerin ab und zu mit dem Gesang "*spam, spam, spam, lovely spam, wonderful spam*" betäubt.

Es ist schwer zu sagen, wer und wann zum ersten Mal das Wort Spam als Synonym für den Netzmißbrauch benutzt hat. Es ist gut möglich, dass diese Pionierleistung den Benutzern des Textspiels *RPG* zusteht, die MUD (*Multi-User Dungeon*) genannt werden. Sie haben dieses Wort für das einmalige Versenden von vielen Daten benutzt (Zur Zeit wird dies häufiger *Flooding* genannt). Es könnte aber auch sein, dass dieser Terminus zuerst in dem Chat auftauchte, das der Vorläufer des IRC war – *Bitnet Relay.*

Als erster Fall vom Spam in Bezug auf die elektronische Post nimmt man oft die Nachricht an, die 1978 durch die Firma *Digital Equipment Corporation* versandt wurde. DEC hatte allen Benutzern von *Arpanet* auf der Westküste der USA eine Mitteilung über ihren neuen Computer DEC-20 zugestellt.

Das Wort Spam wurde erst in 1994 in dieser Form übernommen, als im Usenet die Anzeige der Lawrence Canter und Marthy Siegiel Anwaltskanzlei erschien, in der sie ihre Dienste in Bezug auf das Ausfüllen der Teilnahmeformulare für die amerikanische Green-Card-Lotterie anboten. Diese Anzeige wurde per Skript an alle damaligen Usenet-Diskusionsgruppen übermittelt.

Zur Zeit werden als Spam alle elektronischen Nachrichten bezeichnet, die zielgerichtet und in großen Mengen an Personen verschickt werden, ohne dass deren Einwiligung dafür vorliegt. Diese Nachrichten können (müssen nicht) ein Handelsangebot, politische Werbung etc darstellen. Seit kurzem wird elektronische Post, die kein Spam ist, als Ham (englisch: Schinken) bezeichnet.

Mehr Informationen über die Geschichte des Spams finden Sie unter der Adresse: *http://www.templetons.com/brad/spamterm.html*