Securing Electronic Mail in a Small Company

Nikolai Fetissov
Securing Electronic Mail in a Small Company

Nikolai N. Fetissov
As part of GIAC Certification
GSEC Practical Assignment Version 1.4b, Option 1
August 11, 2003

Abstract

Electronic mail has become a de facto standard for business communications. Companies, big and small, depend on email for their day to day operations, even though electronic mail is plagued with numerous security problems. Email might be read in transit. Messages may be altered or dropped by intermediate servers. Mail boxes are effectively unprotected with sniffable passwords. Yet people use email because it’s convenient. While numerous approaches and protocol enhancements have been proposed to reduce security risks in email communications, no comprehensive solution exists. This paper presents a typical email configuration of a small company, the associated vulnerabilities, and demonstrates how free open source tools help reduce the risks.

1 Introduction

Electronic mail, now at the same threat level as HTTP and peer-to-peer software, presents one of the major risks to corporate and home networks. Viruses and worms, such as Nimda and BugBear, enter the network via SMTP. Anonymous Internet hackers constantly scan for exposed email gateways to exploit. Email protocols are vulnerable to man-in-the-middle attacks. Unsolicited mail, also known as spam, fills user’s mailboxes. Spam blocking tools refuse to accept email from users with dynamic addresses\(^1\). Still the most important pitfalls of email are in its complete lack of confidentiality, integrity and availability.

Section 2 defines a small company and describes its network setup with a focus on email. Section 3 highlights the risks associated with such a setup. Section 4 lays out a simple plan to take control of the situation. Section 5 draws attention to the critical details of a real-life setup.

\(^1\)Once an address block has been put onto a public black list it’s likely that email from that block won’t reach its addressees. An increasing number of Internet service providers require correct reverse DNS resolution of the sender and/or a match of the source IP of SMTP connection to the official MX records for sender’s domain.
2 The Small Company

This paper focuses on an organization running a small private network. This environment might consist of a couple of computers and a printer protected by a consumer-grade firewall appliance; or it might be an office full of desktops sharing file and print server machines protected by multiple firewalls and routers. The connection to the Internet is through a local Internet Service Provider (the ISP.) The email, DNS and Web site services are externally hosted by a hosting provider. This configuration is important because no active SMTP and HTTP connections and no incoming DNS requests are made into the private network. The machines within the private network make active POP3 or IMAP connections to retrieve email and make active SMTP connections to send messages. Figure 1 shows this typical network setup. See Appendix A for the list of acronyms.

Figure 1: Typical Small Company.

The hosting provider, most often not the ISP, maintains the domain name service records for the company web site, the authoritative name servers (NS records) and the mail exchanger server (MX records). The hosted DNS almost never covers the public Internet IP space used by the company network (i.e. the IP address of the external network interface of the firewall box). That IP space belongs to the ISP and might or might not have proper reversed name resolution (in-addr.arpa. domain.) These external IP addresses could be assigned statically or provided dynamically via DHCP. Host name resolution within the private network is often served with NetBIOS/WINS for Windows machines or /etc/hosts files for Unix and Linux
To best understand the example configuration, it is necessary to look at the network traffic relevant to email in this setup. Of interest here are POP, IMAP, SMTP and DNS. IMAP is rarely an option with hosting providers, resulting in the much wider use of POP3. In the situation shown in Figure 1 every packet for these protocols travels across the firewall to and from the Internet. Tracing a single POP3 session from the desktop machine (desktop1, IP 192.168.0.100) to the hosting provider server (mail.hosting-provider-example.com, IP 1.1.1.1) would be helpful here. Figure 2 shows the tcpdump [1] output on a box within the private network. Figure 3 shows the same session on the outside of the firewall after network address translation (NAT). The external IP address of the firewall is 1.2.3.4. The connection establishment and termination packets are not shown. Timestamps, TCP window announcements and options are removed for readability. The command used in both cases is tcpdump -n port pop3.

Both traces are almost the same with the exception of NATed IP addresses and client-end ephemeral port numbers. Both pictures look like examples from W. Richard Stevens [2].

Capturing full-length packets gives more insight into what is actually being sent. Figure 4 shows the output of the command tcpdump -s 1460 -w pop3session.cap port pop3 opened in Ethereal [3] using the ‘Follow TCP Stream’ tool. Some ‘Received’ lines have been removed for clarity.

Not surprisingly, the plain TCP stream reveals not only the contents of the downloaded

---

2In contrast to the above, a setup in a bigger corporation would include an externally accessible mail exchanger (SMTP gateway) pointed to by an official MX record in public DNS. Such a host, along with web server(s) and DNS servers, would usually reside on a separate network segment (DMZ) with restricted and closely monitored access from the Internet and the private network. The IP space here is static and referenced via DNS. The firewall allows incoming SMTP and HTTP connections, as well as, DNS requests into the DMZ.
message, but also the POP3 user name and password. POP3 [4] is not a secure protocol by itself. Various extensions have been proposed to secure the authentication and message download traffic. One extension, Transport Layer Security (TLS), can be deployed to protect the POP3 connection.

The previous example looked at incoming email. People on a network also have a habit of sending email. Listening to SMTP traffic reveals similar flaws. Figure 5 shows the output of tcpdump -s 1460 -w smtpsession.cap port smtp opened in Ethereal. Again, the plain text of the message is revealed. Though the email is sent to a colleague in the same office, the message itself travels unprotected across the Internet.

3 What is at Risk

With passwords in clear text, internal email hopping over the Internet, and desktops connecting to unknown machines, this environment is waiting for trouble. Multiple security vulnerabilities exist.

- The first and most noticeable vulnerability is the lack of confidentiality caused by email messages traveling in clear text over an insecure network. This is similar to sending snail-mail messages on postcards. Almost all Internet email is transmitted in clear text due to the original design of SMTP [5] [6]. The most visible technologies that provide email confidentiality, S/MIME and PGP [7], do not scale well to a global level and are quite complex for the average user. While Internet mail might remain insecure for years to come, there is no excuse for disclosing private business communications to passive Internet
3 WHAT IS AT RISK

Figure 4: TCP stream of a POP3 session.

+OK Messaging Multiplexor (iPlanet Messaging Server ... 
USER peter 
+OK password required for user peter 
PASS secret 
+OK Maildrop ready 
STAT 
+OK 1 1534 
LIST 
+OK scan listing follows 
1 1534 
. 
UIDL 
+OK unique-id listing follows 
1 S=1067567905 
. 
XSENDENT 1 
+OK 
RETR 1 
+OK 1534 octets 
Return-path: <john-paul@customer-example.com> 
Received: from mta.customer-example.com (mta.customer-example.com [4.3.2.1]) 
    by mail.hosting-provider-example.com 
    (iPlanet Messaging Server ...) 
    with ESMTP id <...> for 
    peter+INBOX@ims-ms-daemon; Sun, 06 Jul 2003 12:51:09 -0400 (EDT) 
... 
Date: Sun, 06 Jul 2003 12:51:09 -0400 (EDT) 
From: "John-Paul ..." <john-paul@customer-example.com> 
Subject: Hope to see you soon 
To: peter@mail-company-example.com 
Message-id: <...> 
MIME-version: 1.0 
Content-type: ... 

Can't wait to meet you in person. 
-- 
John-Paul 
. 
DELE 1 
+OK message deleted 
QUIT 
+OK
WHAT IS AT RISK

The next vulnerability is the weak authentication provided by POP3 passwords in clear text. Multiple protocol extensions provide integration of strong authentication technologies, for example Kerberos, into POP3. None of the extensions are in wide use by the general Internet public. The main point here is that passwords could be easily stolen with passive network monitoring, giving an eavesdropper complete control over a POP3 account. The less obvious problem, however, is that the authentication scheme on the company network and the scheme of the hosting provider are independent. Thus every user has two different passwords, one for local access and one for the email account, even though, chances are, both are the same. Sniffing an email password off the wire grants access to the protected network, no matter how good the internal password policy is.
The third vulnerability is a lack of availability of email messages. Email messages are stored by POP users on their desktop machines. Individual desktops are rarely backed up. A single disk failure might cost the loss of years worth of messages exchanged with a business partner. Leaving the messages on the POP server is not a viable option either. Hosting providers enforce disk quotas for email storage. Even worse, storing mail history on the Internet would make more sensitive business information available to a cracker stealing a POP3 password.

Hypothetically, an active attacker could completely remove a message from the wire, which may also be considered a lack of availability (though this requires much more control over the network.)

Another vulnerability is the loss of confidentiality that can occur when both POP and SMTP clients rely on an external DNS for resolving server addresses. Given the completely unprotected nature of DNS, an active outside attacker can trick internal clients into sending messages or giving POP3 passwords to a rogue machine.

Not the least are the multiple vulnerabilities posed by spam and viruses. Users need to run filtering and anti-virus software on their desktops, which leads to inconsistent and insecure configurations.

It is also worth mentioning the lack of message integrity. Any message might be altered in transit or in POP3 storage since no integrity validation is done, unless PGP or S/MIME is in use.

The considerations above strongly suggest that the situation is well below an acceptable security level even for a non-profit organization 'with no secrets'. Although the lack of confidentiality might be acceptable for some businesses, most would want at least some data availability and integrity.

4 Steps in the Right Direction

What could be done to mitigate against these threats and vulnerabilities? First, bring the mail server into the protected network, self-host email, taking comfort behind the firewall. But does this also mean that DNS and web servers must be self-hosted as well? Does this require creating a DMZ, buying expensive machines, an Exchange license and hiring two full-time administrators? As it turns out, the answer to these questions is no. A middle ground between self hosting and external hosting does exist.

Before describing the solution, here are some realistic goals.

It is important to note that providing total confidentiality of email is not feasible. The primary goal is to provide integrity and confidentiality to internal company communications. Two levels of security are feasible. One allows that no entities from outside the private network can read internal mail. The other provides users inside the firewall with access to their messages and only their messages.

Prevent POP3 authentication in clear text.
• Improve message availability with a backup system. The backup also needs to be protected so not to compromise confidentiality.

• Centralize the transmission and receipt of email to provide a consistent and manageable treatment of spam and viruses.

With the above goals in mind the following observations can be made. Internal email should not leave the boundaries of the protected network. This leads to the conclusion that an internal mail server is needed to provide both mail delivery and retrieval. Mail delivery is provided by standard SMTP through an internal mail gateway. The retrieval is better provided with IMAP, allowing for centralized email storage and centralized backup. Does all this require full self-hosting? No. Needed here is a secure way of bringing email from the outside, filtering the messages and saving them into the IMAP store. This is where Fetchmail [9] comes to the rescue.

Fetchmail is a free mail retrieval and forwarding program by Eric S. Raymond. In fact, the development of Fetchmail was author’s ‘sociological experiment’ regarding the Linux development model and is fully described in Raymond’s famous paper, ‘The Cathedral and the Bazaar’ [10]. Fetchmail, not surprisingly, fetches mail from a server, dropping messages into local mailboxes via SMTP or LMTP. Fetchmail supports all open message access protocols and multiple authentication schemes. It includes extensive support for secure communications via SSL/TLS, provided the executable was compiled with the OpenSSL library. If IPv6 is available, Fetchmail can initiate IPSec. The particular feature to look at is the implementation of the STLS (Start TLS) POP3 command for secure mail box access. Fetchmail supports checking a server’s public key against a locally stored CA certificates or checking an MD5 fingerprint. See RFC 2595 [11] for the description of the Transport Layer Security in POP3 and IMAP.

The POP3 user names, passwords and connection settings reside in a configuration file. This needs to be carefully protected, since unauthorized access to this file would compromise all email accounts.

The next issue to discuss is mail delivery. Both local mail and inbound external mail downloaded with Fetchmail needs to be injected into the IMAP store. It is necessary to ensure that outbound messages leave the protected network via a known path, preferably through the hosting provider gateway. This would help prevent, but not guarantee, a good-intentioned email message from being classified as spam by its recipient.

Standard Sendmail [12] can deliver both local and remote mail. The specific method of local delivery is IMAP server specific. For example, Cyrus IMAP [13] provides its own mailer, while UW IMAP [14] works off the standard Unix mail spool. Sendmail offers multiple authentication mechanisms via the Simple Authentication and Security Layer [15]. For remote delivery, Sendmail supports the STARTTLS ESMTP command (see RFC 3207 [16]) and allows for controlling certificate verification via the tls_server ruleset.

The users access messages with IMAP. Depending on the server software, multiple authentication options are available. Most IMAP implementations either natively support SSL or allow for tunneling IMAP traffic with tools such as Stunnel [17]. Consider SSL as an option giving a higher degree of email confidentiality and fencing off internal man-in-the-middle attacks.

---

3This, of course, requires hosting providers to support POP3 with TLS. As it turns out many already do, just by using the latest releases of the server software, such as the Courier Mail Server [28] for example.
5 SOME CONFIGURATION DETAILS

Another thing to mention is the DNS setup. An internal DNS zone is strongly recommended. Standard ISC BIND [18] is easy to setup, see [19]. An internal DNS helps in several ways. First, it provides both self-contained direct and reversed name resolution. Second, it’s possible to setup your own private MX records. Third, machine names are centrally managed and backed up. DNS caching comes as a bonus to provide altogether better availability.

It’s now time to summarize the above considerations and point out the resulting security benefits.

- Internal email does not leave the protected network, the IMAP server is behind a firewall and optionally requires SSL and strong authentication.

- The centralized IMAP message store allows for regular backups, which in turn might be encrypted to administrative keys.

- External inbound email is retrieved in a secure fashion. The POP3 server at the hosting provider could be authenticated using SSL before Fetchmail discloses passwords to it. The administrator manages the POP3 passwords.

- The internal delivery gateway provides strong authentication and supports SSL. All outbound messages are forwarded through the hosting provider gateway, again with optional gateway authentication.

- Having both delivery and retrieval servers under local control allows for integrating them into a company-wide authentication scheme. This leads, if not to single sign-on, then to at least a single id and password per user, better password policy enforcement, and thus stronger passwords.

- Routing all messages through the same channel gives the perfect opportunity for spam filtering and virus scanning. Both Fetchmail and Sendmail provide anti-spam features. Virus detection is harder and is usually not free, though tools like Open Anti Virus [20] have begun development. The initial approach should be preventing dangerous attachments from getting through at the minimum.

An often overlooked option is to tighten the firewall. To prevent users from falling back into non-secure email habits, the firewall should block outbound POP3 and SMTP connections, except from the internal mail server. This is also relevant for the DNS. The internal name server provides recursive name resolution to the internal clients. The firewall blocks all DNS traffic except to and from the internal DNS server.

So far, better email availability and confidentiality are in sight. This setup shows all the benefits of self-hosting without its inherent problems. The picture 6 gives a possible server and network layout. The next section highlights several critical configuration details for the particular server software.

5 Some Configuration Details

All the server tools mentioned above work under a variety of ‘Unix-derivative’ and ‘Unix-like’ operating systems. The author’s choice is OpenBSD [21] led by the Canadian developer Theo de
Raadt. OpenBSD offers a ‘free, functional and secure’ OS with a multitude of security audited software packages. The latest updates to OpenBSD provide exclusive memory write/execute access, a non-executable stack and Propolice (‘Stackguard on steroids’), eliminating a whole class of buffer overflow attacks. OpenBSD offers support for cryptographic hardware, has excellent capabilities for building IPSec VPNs, and includes a powerful packet filter for firewalls, network address translation and bandwidth management. Even with all the bells and whistles, OpenBSD is very easy to install and configure. It also runs superbly on low-end i386 systems.

The software packages of choice are BIND 9 [18], Sendmail [12], Fetchmail [9], OpenSSL [22], SpamAssassin [23], and Cyrus IMAP [13]. Sendmail and OpenSSL are pre-installed with OpenBSD. BIND 9 and SpamAssassin come as packages, and Fetchmail is in the ports tree. Cyrus IMAP needs to be manually compiled and requires the CMU SASL [15] library, also in the ports tree. SASL provides an authentication layer for both Sendmail and Cyrus, which allows for the integration of various authentication schemes like LDAP and Kerberos4.

The full set of compilation and configuration options for all the above would easily make up a medium-sized book. As always, there’s no substitute for reading original manuals and working with the tools. The good news is that not all of the features need to be implemented immediately to significantly improve security. Information on setting up most of the components is available in print and on the Internet5. The rest of this section points out some important

---

4 OpenBSD includes the Heimdal Kerberos implementation [29] in the default install.
5 See, for example, [30] for a general overview of the Cyrus IMAP configuration and [31] for specifics on OpenBSD. See [19] for an excellent text on BIND.
configuration settings of the two critical components in the above setup: the mail retrieval daemon – Fetchmail, and the mail transfer agent – Sendmail.

5.1 Fetchmail Configuration

Fetchmail is configured to download POP3 mail every ten minutes, dispatching the retrieved messages locally via SMTP. Figure 7 shows the fetchmailrc file. Fetchmail is set to initiate a secure POP3 conversation to the mail server via the STLS command and to validate the public key of the server against the fixed MD5 hash. It cannot be stressed enough that the configuration file must be carefully protected. Unauthorized access to the file would compromise the entire email security scheme. One should consider putting the fetchmailrc on an encrypted file system. Figure 8 shows the beginning of the TCP stream capture of the POP3 session with the STLS command. After the client issues the STLS command, the parties negotiate a secure session and restart the dialog over an encrypted channel. As a result, passwords never travel in clear text over the network. The process of downloading the inbound mail is therefore, secure.

Figure 7: Fetchmail configuration file.

```
set postmaster "john"
set no bounceemail
set syslog
set daemon 600
poll mail.small-company-example.com with proto POP3
  user 'peter' there with password 'secret' is 'peter' here
  user 'luke' there with password 'terces' is 'luke' here
  user 'admin' there with password '0956437s23a' is 'john' here
sslproto tls1
smtphost localhost
```

Figure 8: Encrypted POP3 session.

```
+OK Hello there.
CAPA
+OK Here's what I can do:
STLS
TOP
USER
LOGIN-DELAY 10
PIPELINING
UIDL
IMPLEMENTATION Courier Mail Server
.
STLS
+OK Begin SSL/TLS negotiation now.
"V"C"A8C
...
```

See Kyle Amon’s Mini-HOWTO for the encrypted virtual file system under OpenBSD [32]

© SANS Institute 2003,
As part of the Information Security Reading Room.
Author retains full rights.
5 SOME CONFIGURATION DETAILS

5.2 Sendmail Configuration

Sendmail handles both local and remote delivery. While not the most secure piece of software ever written (see [24] for list of known exploits), Sendmail offers strong authentication via SASL, anti-spam and anti-relaying features. Sendmail version 8.11 and later supports the STARTTLS ESMTP command. Sendmail is used to deliver the internal mail to the local IMAP store\(^7\). All outbound messages are forwarded through the SMTP gateway of the hosting provider by including the SMART_HOST definition. Message relaying is restricted to authenticated internal users. Two options are available, either by requiring password authentication or by relaying based on the validity of the sender’s SSL certificate. The former is done with TRUST_AUTH_MECH macro. The latter requires maintaining a local certificate authority. This is possible using OpenSSL (see openssl(1) and starttls(8) OpenBSD manual pages). For now, client verification is disabled by setting confTLS_SRV_OPTIONS variable. Trusted authentication mechanisms are declared and the SMTP daemon listening on the non-loopback interface requires clients to login with one of these mechanisms. Fetchmail is made exempt from SMTP authentication by defining an additional local daemon with the DAEMON_OPTIONS macro\(^8\). All outgoing mail is masqueraded at the envelope level as coming from the company domain.

The last thing to take advantage of is SMTP over TLS for outbound connections. TLS provides point-to-point but not end-to-end security. This means a given email message is confidential and authentic while in transit between two SMTP gateways engaged in a TLS conversation. No security is guaranteed during further message forwarding. In our case, TLS provides for the authentication of a specific SMTP relay. Messages are routed through a known and, to some degree, trusted gateway. Encryption also makes life a bit more difficult to nosy outsiders watching network traffic. Sendmail provides options for configuring CA and TLS client certificates. The target gateway certificate must be signed by one of the locally configured CAs.

Figure 9 gives a snippet of a Sendmail m4 configuration file with the discussed settings. For an exhaustive list of Sendmail’s configuration options, see the third edition of Bryan Costales’ text [25]. Figure 10 shows the preamble of a TLS-enabled SMTP conversation. At this point the outbound message delivery is secure from the local SMTP server to the mail server of the hosting provider.

5.3 Further Improvements

There is always room for improvement. Using a defense-in-depth strategy, additional layers of protection would be introduced to improve the email security. In the example configuration, SSL and S/MIME should be used for the internal company communications. PGP should be used for external communication.

By using SSL, internal email traffic is encrypted to prevent packet sniffing and man-in-the-middle attacks. S/MIME provides message integrity based on the company-wide certificate trust. Both SSL and S/MIME require deployment of a private certificate authority. The OpenSSL and OpenCA [26] tools help here.

The integrity of external email communications can be enhanced by the use of PGP. The public PGP keys of externally visible company contacts should be published on a secure Internet

---

\(^7\)In case of Cyrus IMAP, the Sendmail configuration uses the MAILER macro. Other IMAP implementations provide different means of local delivery.

\(^8\)Fetchmail can also authenticate itself to ESMTP server with esmtpname and esmtppassword options.
Figure 9: Sendmail options.

```
define('CERT_DIR', '/etc/mail/certs')
define('confCACERT_PATH', 'CERT_DIR')
define('confCACERT', 'CERT_DIR/CAcert.pem')
define('confCLIENT_CERT', 'CERT_DIR/mycert.pem')
define('confCLIENT_KEY', 'CERT_DIR/mykey.pem')
TRUST_AUTH_MECH('DIGEST-MD5 CRAM-MD5 LOGIN')
define('confAUTH_MECHANISMS', 'DIGEST-MD5 CRAM-MD5 LOGIN')
FEATURE('always_add_domain', 'small-company-example.com')
FEATURE('masquerade_envelope')
FEATURE('masquerade_entire_domain')
MASQUERADE_AS('small-company-example.com')
define('SMART_HOST', 'mail.hosting-provider-example.com')
define('confTLS_SRV_OPTIONS', 'V')
FEATURE('no_default_msa', 'dnl')
DAEMON_OPTIONS('Family=inet, address=192.168.0.15, Name=FUBAR-MTA-38, M=a')
DAEMON_OPTIONS('Family=inet, address=127.0.0.1, Name=FUBAR-MTA-38-LOCAL')
CLIENT_OPTIONS('Family=inet')
define('confLOCAL_MAILER', 'cyrusv2')
MAILER(cyrusv2)
```

Figure 10: STARTTLS command in outbound SMTP connection.

```
220 mail.hosting-provider-example.com ESMTP
EHLO small-company-example.com
250-mail.hosting-provider-example.com
250-STARTTLS
250-AUTH LOGIN CRAM-MD5 PLAIN
250-AUTH=LOGIN CRAM-MD5 PLAIN
250-PIPELINING
2508BITMIME
STARTTLS
220 ready for tls
^@|A^-A^-e^-E@^A^@^A^
```

web site. These PGP keys allow the public to confirm the integrity of any email messages they receive from the company.

## 6 Conclusion

Email protocols are historically insecure as most were developed before the Internet community came to realize the dangers of an open network. Security solutions are hard to retrofit into widely deployed systems, often due to flaws in the original protocol design. Still, steps should be taken, even in restricted environments, to improve upon email security.

As often happens in practice, electronic mail is managed by a hosting provider. The cost and effort associated with self-hosting are often prohibitive to small or non-profit organizations. This paper presents a middle approach where internal email communications are kept within the boundaries of the protected network, while the existing hosting provider is used for the
remaining inbound and outbound mail.

The example configuration shows how to achieve a level of confidentiality with internal company communications, how to improve the availability of stored email messages and, optionally, how to provide for the integrity of email messages. The paper also demonstrates how to secure the interactions with the external systems of the hosting provider.

Using free open source tools, each supporting secure versions of message retrieval or message delivery protocols, enables the implementation of this low-budget, secure configuration.

7 Appendix A, List of Acronyms

<table>
<thead>
<tr>
<th>Acronym</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>BSD</td>
<td>Berkeley Software Distribution</td>
</tr>
<tr>
<td>CA</td>
<td>Certificate Authority</td>
</tr>
<tr>
<td>DHCP</td>
<td>Dynamic Host Configuration Protocol</td>
</tr>
<tr>
<td>DMZ</td>
<td>De-Militarized Zone</td>
</tr>
<tr>
<td>DNS</td>
<td>Domain Name Service</td>
</tr>
<tr>
<td>ESMTP</td>
<td>SMTP Service Extensions</td>
</tr>
<tr>
<td>HTTP</td>
<td>Hyper Text Transfer Protocol</td>
</tr>
<tr>
<td>IMAP</td>
<td>Internet Message Access Protocol</td>
</tr>
<tr>
<td>IMAP4</td>
<td>Internet Message Access Protocol, Version 4</td>
</tr>
<tr>
<td>IP</td>
<td>Internet Protocol</td>
</tr>
<tr>
<td>IPSec</td>
<td>IP Security</td>
</tr>
<tr>
<td>IPv6</td>
<td>Internet Protocol, Version 6</td>
</tr>
<tr>
<td>LDAP</td>
<td>Lightweight Directory Access Protocol</td>
</tr>
<tr>
<td>LMTP</td>
<td>Local Mail Transfer Protocol</td>
</tr>
<tr>
<td>MX</td>
<td>Mail Exchanger</td>
</tr>
<tr>
<td>LaTeX</td>
<td>Typesetting system used in this document</td>
</tr>
<tr>
<td>NAT</td>
<td>Network Address Translation</td>
</tr>
<tr>
<td>PGP</td>
<td>Pretty Good Privacy</td>
</tr>
<tr>
<td>POP</td>
<td>Post Office Protocol</td>
</tr>
<tr>
<td>POP3</td>
<td>Post Office Protocol, Version 3</td>
</tr>
<tr>
<td>SASL</td>
<td>Simple Authentication and Security Layer</td>
</tr>
<tr>
<td>S/MIME</td>
<td>Secure Multipurpose Internet Mail Extension</td>
</tr>
<tr>
<td>SMTP</td>
<td>Simple Mail Transfer Protocol</td>
</tr>
<tr>
<td>SSL</td>
<td>Secure Socket Layer</td>
</tr>
<tr>
<td>TCP</td>
<td>Transmission Control Protocol</td>
</tr>
<tr>
<td>TLS</td>
<td>Transport Layer Security</td>
</tr>
</tbody>
</table>

References

[1] TCPDUMP public repository. URL: http://www.tcpdump.org
REFERENCES


[18] Internet Software Consortium, ISC BIND. URL: http://isc.org/products/BIND/


[21] Open BSD Project. URL: http://www.openbsd.org

[22] OpenSSL Project. URL: http://www.openssl.org

[23] SpamAssassin Project. URL: http://www.spamassassin.org


[26] The Open CA Project. URL: http://www.openca.org


[29] Heimdal Project. URL: http://www.pdc.kth.se/heimdal/


<table>
<thead>
<tr>
<th>Course Name</th>
<th>City, Country</th>
<th>Dates</th>
<th>Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>SANS Las Vegas Summer 2020</td>
<td>Las Vegas, NVUS</td>
<td>Jun 08, 2020 - Jun 13, 2020</td>
<td>Live Event</td>
</tr>
<tr>
<td>SANS New Orleans 2020</td>
<td>OnlineLAUS</td>
<td>Jun 08, 2020 - Jun 13, 2020</td>
<td>Live Event</td>
</tr>
<tr>
<td>SANS OnDemand</td>
<td>Books &amp; MP3s OnlyUS</td>
<td>Anytime</td>
<td>Self Paced</td>
</tr>
</tbody>
</table>