Wrocław University of Technology

Internet Engineering

Tomasz Surmacz

Secure Systems and Networks

Wrocław 2011

Contents

List of Figures

Chapter 2

Basic security in computer systems

2.1 The main problem are people

Although it is possible to automate many access rules and procedures and deploy very sophisticated security policies, research shows that the weakest link in the computer system security are the people who use it. This is not even because people want to break the security intentionally, but due to several factors. These include:

- Lack of understanding for the rules.

  People tend to create simple, easy to remember passwords or store them in easily accessible places, such as a yellow sticky note under the table. Or are easily tricked to give away information that should be kept confidential. If they do not understand that their actions can sabotage the security of the whole network, they will not pay enough attention to real problems.

Characters used	no. of passwords	breaking time required
4-digit password (PIN code)	10000	1.2 seconds
typical language vocabulary	100000	12 s
4-letter password	456976	1 minute
8-digits password	$1 \cdot 10^{8}$	3 hours 25 mins
6-letter password	$3 \cdot 10^{8}$	11 hours
8-letter password, lowercase	$2 \cdot 10^{10}$	302 days
8-character password, lowercase and digits	$2.8 \cdot 10^{12}$	11 years
8-letter password, mixed case	$5 \cdot 10^{13}$	212 years
8 characters, all ASCII 32-127	$7 \cdot 10^{15}$	28594 years
8 characters, all keyboard 1-127	$6.7 \cdot 10^{16}$	268246 years
All possibilities for MD5 hash function	$3.4 \cdot 10^{38}$	$1.3 \cdot 10^{27}$ years
Estimated age of the Universe		10^{10} years

Table 3.1: Strength of different passwords

or “recovering” passwords used to protect MS-Windows systems (e.g. Cain and Abel or SolarWinds). A useful list of available crackers may be found in [Lyo06].

3.2 One Time Passwords

The obvious solution to the problem of password sniffing are the passwords, that are changed frequently. So frequently, that a new password is required every time that the user logs in, so even when the password is sniffed, it will be useless. There are several systems allowing such way of operation, some of them requiring specialized hardware, others using just some clever ideas and a proper setup. The next few pages will show the most popular systems in use.

calculate its MD5 hash, and if the password is correct it will match password no. 99 stored in the */etc/skeykeys* file. In such case, the stored password (i.e. encrypted password 98) will be replaced with the just entered password which is both the encrypted version of password 98 and encrypted version of password 97. So, next time the user wants to log in, he will be asked for password no. 97.

Security and usefulness of this system is based on simplicity of generating all subsequent passwords from any password we already have when compared to difficulties of doing the same in the opposite direction. If password no. 40 is known, generating passwords no. 41, 42, 43 ... is trivial, but finding passwords numbered between 0 and 39 is impossible or at least improbable (it would require processing power taking billions of years to finish).

3.2.2 Tokens

Tokens are small pieces of hardware that can be used as authentication devices, much like the keys may be used to open locks. For token authentication to succeed, the user has to prove that he has the token with him which may be checked in many ways. Tokens connected to RS232, parallel or USB port of a computer system may be used as a kind of a hardware license key, but nowadays it is much easier to depend on some short cryptographic data to be shown by a token itself and entered by a user.

Tokens may either work as standalone devices independent of the outside world (i.e. only producing some output needed for authentication) or work bidirectionally –

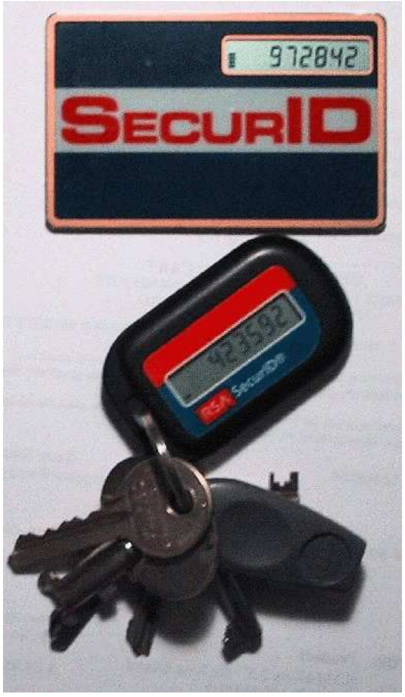
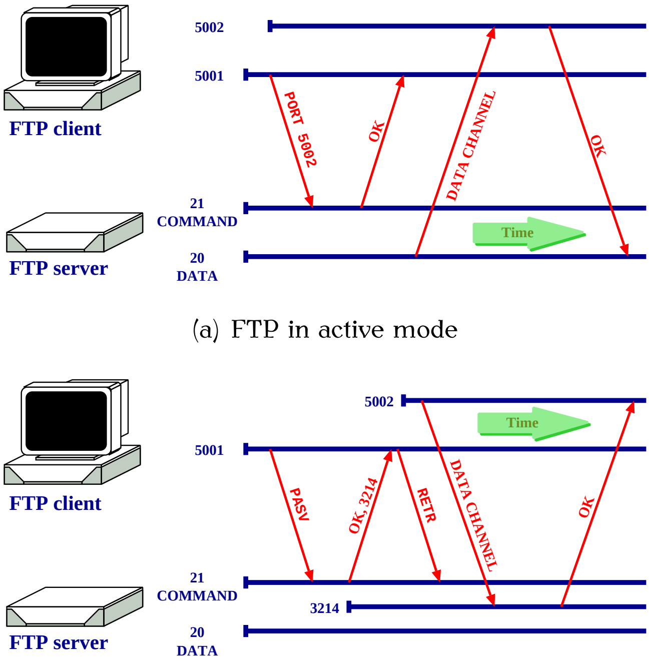


Figure 3.3: RSA OTP tokens

opens the connection to the client from its own port 20 and the data is transferred using this new connection, while the old one may still be used to issue other commands (e.g. to abort the transfer). When transfer is complete, this connection is closed automatically.

Data transfer in passive mode (fig. 4.8(b)) uses additional port on the server. The client first uses the `PASV` command to indicate its willingness to initiate the passive mode data transfer to which the server opens some random additional port and informs the client of its number with the `OK` response. The client then opens connection to the given port and transmits the data.



(a) FTP in active mode

(b) FTP in passive mode

Figure 4.7: FTP modes of operation: (a) active and (b) passive

4.5.3 Anonymous FTP

Anonymous FTP poses even more threats to the computer system, as by the definition, it allows access for any users of the Internet and if any problem with the service is discovered, it may be exploited by anybody. So the proper setup of anonymous FTP service is crucial for the security of the server running it.

5.3 Worms

Computer viruses propagate in passive way – they cannot survive on their own, but need a program or a file they can attach to and spread to other programs when their host is executed. They may also infect other computers, but mainly when transferred by e-mail, or when the infected program is copied from one system to another and then executed.

Computer *worms*, on the other hand, actively seek ways to propagate to other computers by exploiting know vulnerabilities in some programs or typical configuration problems.

Worms need some security holes in order to propagate, so the protection against them is basically the same as protection against break-ins. This includes monitoring the system for unusual activities, such as rapidly increasing umber of emails sent, skyrocketing CPU usage, rapidly growing size of the log files and number of logged errors – especially the ones that are generated by `identd` which show unusual number of outgoing connections.

5.4 Phishing attacks

The term *phishing* comes from joining together two words – *password* and *fishing* and describes the fraudulent actions of setting up fake servers collecting sensitive information from unsuspecting users – such as passwords (hence the name), usernames, credit card numbers, authorization tokens for bank operations, etc. Phishing attacks are usually targeted at popular social web sites, auction sites, online banks, but also the IT departments of various big institutions.

In the sample e-mail shown in fig. 5.1 the link is shown twice – once in the mail body where it looks like the Amazon reference (but in fact is a *link name*), and second time – in the “References” section, where all links are expanded to show their actual targets. This is how textual e-mail clients (such

```
From: "Amazon.com" <account-update@amazon.com>
Subject: Revision to Your Amazon.com Account

  Amazon Information
  We are contacting you to inform you that our Account Review
  Team identified some unusual activity in your account. In
  accordance with Amazon’s User Agreement and to ensure that
  your account has not been compromised, access to your account
  was limited. Your account access will remain limited until this
  issue has been resolved.  This process is mandatory, and if not
  completed within the nearest time your account or credit card
  may be subject for temporary suspension. To securely confirm
  your Amazon information please click on the link bellow:
  [1]https://www.amazon.com/cgi-bin/webscr?cmd=login-run

  We encourage you to log in and perform the steps necessary to
  restore your account access as soon as possible. Allowing your
  account access to remain limited for an extended period of time
  may result in further limitations on the use of your account
  and possible account closure.

References

  1. http://96.10.247.13/evo/www.amazon.com/index.html
```

Figure 5.1: Phishing e-mail targeting Amazon.com users

as mutt or pine) show such e-mails. Webmail-type browsing in Firefox or Internet Explorer will show just the name, and the actual link target will be shown in the browser status line only when the mouse cursor is over the link name. And even not for certain, as some bugs in Internet Explorer may truncate the link being shown, or some special characters in the link may cause only the trailing part of the link being visible, and that part may look like the normal link.

Mode	Description
ECB Electronic Code Book	Each plaintext block is encoded independently of other blocks
CBC Cipher Block Chaining	Instead of just the plaintext, the algorithm is fed with the XOR of the plaintext and the previous ciphertext.
CFB Cipher Feedback	Input is processed n bits at a time (les then or equal to the block size). Previous ciphertext is used as input to produce the pseudorandom data which is then xored with current plaintext.
OFB Output Feedback	Similar to CFB, except the input is not the final ciphertext, but the output of the DES encryptor (i.e. before XOR-ing it with the plaintext)
CTR Counter	Every plaintext block in XORed with an encrypted counter, which is incremented with each block
CTS Ciphertext Stealing	Used in addition to ECB or CBC for padding the last plaintext block with the high order bits of the second to last cipherblock (stealing it from the output). The new full last block is then encrypted and exchanged with the stolen cipherblock, truncated by removing the stolen bits, so that the overall message length does not change.

Table 7.1: Block cipher modes of operation

or altered during transmission, it will be not possible to decrypt the rest of the transmission. A little modification of this mode called *Output Feedback Mode* (OFB, see fig. 7.2) helps overcoming this problem by moving where this chaining information is attached – if a single block is malformed, it will be lost and impossible to decrypt, but the synchronisation is regained with the next block.

Bibliography

[Ale01] Aleph One. Smashing the stack for fun and profit. *Phrack*, 49, April 2001. http://www.phrack.org/archives/49/.

[Bau05] Michael D. Bauer. *Building secure servers with Linux*. O'Reilly & Associates, Inc., second edition, 2005.

[BH10] M. Brown and R. Housley. Transport Layer Security (TLS) Authorization Extensions. RFC 5878 (Experimental), May 2010.

[BSB05] Daniel J. Barrett, Richard E. Silverman, and Robert G. Byrnes. *SSH: The Secure Shell: The Definitive Guide*. O'Reilly & Associates, Inc., second edition, 2005.

[Che92] W.R. Cheswick. An evening with Berferd, in which a cracker is lured, endured, and studied. In *Proceedings of the Winter USENIX Conference (San Francisco)*, January 1992.

[Cor10] The MITRE Corporation. Top 25 most dangerous software errors. http://cwe.mitre.org/top25/index.html, December 2010.

[CZ95] D. Brent Chapman and Elizabeth D. Zwicky. *Building Internet Firewalls*. O'Reilly & Associates, Inc., September 1995.

[DA99] T. Dierks and C. Allen. The TLS Protocol Version 1.0. RFC 2246 (Proposed Standard), January 1999.

Index