

Operating Systems Concepts: Chapter 9: Some Security Issues

Olav Beckmann

Huxley 449

<http://www.doc.ic.ac.uk/~ob3>

Acknowledgements: There are lots. See end of Chapter 1.

- Home Page for the course:

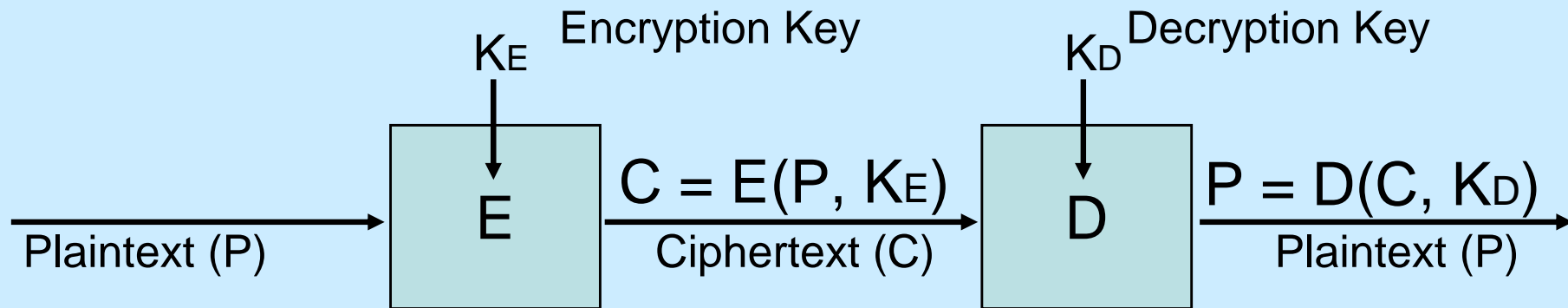
<http://www.doc.ic.ac.uk/~ob3/Teaching/OperatingSystemsConcepts/>

- This is only up-to-date after I have issued printed versions of the notes, tutorials, solutions etc.

Chapter 9: Some Security Issues

- Protecting data against unauthorised use is a major concern for operating systems designers and administrators.
 - Data confidentiality – prevent unauthorised access
 - Data integrity – prevent unauthorised tampering
 - System availability – prevent denial of service
- Issues to consider (at design stage)
 - Casual users browsing information that's not for them
 - Snooping by skilled insiders
 - Organised and determined attempts to make money
 - Espionage etc.
- System “bloat” is definitely bad for security
- Accidental data loss probably more common than determined security evasion

Very Basic Cryptography



- Secret (symmetric) key cryptography

- Plain: ABCDEFGHI J Example: Hello dear
- Cipher: KLMNOPQRST Rollo nokr
- Q: What is (are) the problem(s) here?

- Public key cryptography

- What is $314159265358979 * 314159265358979$?
- What is $\text{sqrt}(39125571506419387090594828508241)$?
- Exploits the fact that there are functions that, although invertible, are much more complex in one direction than the other.

- One –way functions

Digital Signatures

- How can you prove that an email claiming to come from ob3@doc is from me?
 - Public-key cryptography is quite computationally expensive
 - Apply a hard-to-invert hash function to the document to generate a small (e.g. MD5 – Message Digest: 16 bytes) result
 - Sender applies private key: $D(\text{hash})$
 - Receiver
 - Recalculates hash from document
 - Applies sender's public key to signature: $E(D(\text{hash}))$
 - If $\text{hash} == E(D(\text{hash}))$ success

User Authentication (Passwords)

- Some simple problems
 - Should you be able to see the number of letters in a password?
 - When will authentication fail (id/password)?
 - Easy to guess passwords (e.g. “baby names books”)
 - But people won’t try **my** computer...
- War dialers / IP scanning
 - Start off by collecting machines that accept logins
 - Dialing, systematic scanning of ip addresses in ic.ac.* etc
 - Then systematically try logins and passwords
 - Boring? That’s what computers are good at...

UNIX Passwords

- One-way-function f is applied to password at login
- “Encrypted”, i.e. $f(p)$, passwords stored on disk, in older systems usually in publically readable file
- No-one can tell a user what their password is
- Problem(s):
 - What if 2 people choose the same password?
 - What if someone chooses “hello” or “password”?
- To guard against pre-computed password dictionaries:
 - Add salt...
 - Password entry goes from $f(\text{“doggie”}) \rightarrow f(\text{“doggie1234”})$
 - Need to store the random number unencrypted
 - $f(\text{“doggie1234”})$ and $f(\text{“doggie5678”})$ no obvious relation
- Still possible to copy actual password file and search
 - Make password file unreadable

One-Time Passwords

- Sequence of passwords, each used only once
 - What is the idea behind this?
 - Do not lose the book where the sequence is written down
☺
- Better plan (Lamport 1981):
 - One-way function f
 - Suppose we need 4 passwords: start with $f(f(f(f(p))))$, then $f(f(f(p)))$ etc
 - The point is that although the previous password is very easy to calculate from the current one, the **next** one is impossible / very hard to compute

Authentication using Physical Objects

- An example we all know: ATM card plus PIN to show that it is us who are using the card
 - Magnetic strip: stores about 140 bytes, cost \$0.10 - \$0.50
 - Real problems if this stores the PIN
- Smart cards
 - Small CPU (maybe 4MHz, 8-bit, some RAM and ROM), small EEPROM (memory that doesn't need power to keep its value)
 - Cost more like \$5...
 - Challenge-response authentication
 - Server sends random string
 - Smart card adds user's password, encrypts, sends back part
 - Server does the same and compares
 - Even better to run a small Java VM on the card – allows substituting the encryption algorithm on the fly

Biometric Authentication

- Authentication via palm- / finger-print reader
 - Some concerns regarding use in criminal cases
- Alternative: Iris recognition tools



- Need to make sure there is a live person there!

Attacks from Inside a System

- “Trojan Horse”
 - Path searched for executable programs (see echo \$PATH)
 - If the current directory is in the path, what can happen?
 - As a user, **if** you must have the current directory in the path, where should it be?
 - As root?
- Login Spoofing
 - Write a program that prints
 - Login:
 - Password:
 - Start this, walk away...
 - This can be guarded against by having a key sequence in the login process **that user programs cannot catch**, e.g. Ctrl-Alt-Del.

Covert Channels



- Beautiful, right?
- Demo...
- Other ways of sending covert information
 - Files
 - Response time
 - Busy/idle
 - Power