

Generally Accepted Computer Security Goals

- in terms of data: CIA
 - **confidentiality** (privacy)
 - **integrity** (non-alteration)
 - **availability** (reliable timely access)
- in terms of roles
 - **authentication** (proof of identity)
 - **authorization** (what are you allowed to do?)
- linking these goals
 - **authenticity** (proof of source of that data)
 - **accountability** (who is responsible for that modification?)
 - **assurance** (why should I trust this system?)

If Data is Altered, Hash is Altered

If Data is Non-Alterable and Available but not Confidential

Message (Data)

„The summer time in Europe ends on October 26th, 2014.”

Cryptographic hash:

f2a04587b9c43d28326fcab065b182f748a94455ad1a81d30d325c5a71906077

The summer time in Europe ends on October 26th, 2014.



f2a04587b9c43d28326
fcab065b182f748a9445
5ad1a81d30d325c5a71
906077

„The summer time in Europe ends
on November 26th, 2014.”

a39c22c3b09c557f24223efb1b1331d9
4c575a121eb700bd6bad98440f8e1eb4

Computing Hashes

You may compute cryptographic hashes by

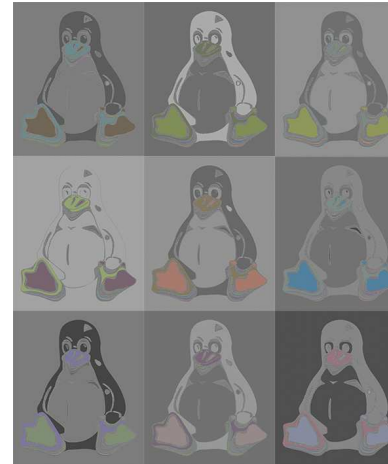
```
echo -n "(insert some text here)" | sha256
```

Cutting edge: SHA-3 (Keccak), not yet implemented

Other SHA-3 finalists

- BLAKE (very fast, authors Aumasson, Meier et al.)
- Grøstl (Knudsen, Rechberger et al.)
- JH (author Wu)
- Skein (authors Schneier, Callas et al.)

ECB-Mode Encrypted Penguin



encryption is block by block

~> every block gets another color

If Data is Confidential and Available but Altered

```
$ hexdump -C test.enc.orig
53 61 6c 74 65 64 5f 5f f2 aa 19 c9 ef 85 af 5e
48 ec 88 a0 28 99 11 41 bd 43 b3 c5 38 84 07 ab
3c de e4 4b 4f c3 b6 11 08 59 1e 0f b5 99 ab dc

$ hexdump -C test.enc
53 61 6c 74 65 64 5f 5f f2 aa 19 c9 ef 85 af 5e
48 ec 88 a0 28 99 11 41 aa 43 b3 c5 38 84 07 ab
3c de e4 4b 4f c3 b6 11 08 59 1e 0f b5 99 ab dc
```

decryption of altered data usually gives garbage

exception: electronic-codebook-mode (ECB)

(uses independent blocks)

Encryption

- do not use ECB-Mode
- use CBC- or CTR-mode (recommendation Schneier/Ferguson)
- use AES or one of the finalists
 - Twofish (Schneier, Ferguson, Kelsey, Whiting, Wagner, Hall)
 - Serpent (Anderson, Biham, Knudsen)
 - MARS (Coppersmith et al., IBM)
 - RC6 (Rivest, patented by RSA)

If Data is Non-Alterable and Confidential but not Available

„Your message with authenticator

08931281763e1de003e5f930c449bf791c9f0db6

has been received, but unfortunately the server is down.

Your mail-service will never be accessible.”

Example: lavabit.com, Snowden’s e-Mail-Provider

Problem: Person/Process/Role \leftrightarrow String (1)

Person identified by picture



this is Bob

String identified by equality relation.

Authorization: Who is Allowed to Do All This?



Authorized entities only.

Only **Bob** is allowed to enter here.

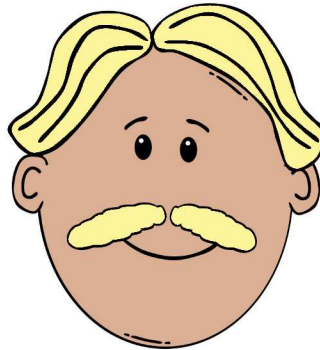
We have to identify persons, processes and their roles.

Problem: Person/Process/Role \leftrightarrow String (2)

How to link a person to a string?

- Person knows something (password, secret cryptographic key).
- Person has something (token, USB-key, chipcard).
- Person is something (biometrics, fingerprint etc.).

Proof of Identity is Called Authentication



this ~~is~~ Bob
looks like

Proof of True Source is Called Authenticity

Proof of Identity: Links Person to a String



Third party guarantees real identity. Has something: ID-card.

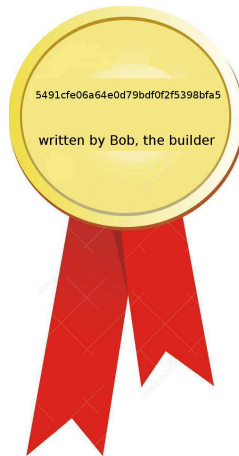


Authentic Messages

depends on **author (key) and message**

Either symmetric: *Message authentication code*

Or by public key: *Digital signature*



What this course is about...

1. UNIX Philosophy
2. Files / Inodes
3. Shell
4. Processes
5. User Identities
6. File System
7. Network
8. Kernel & Booting
9. Security Issues
10. Local Security
11. Network Security

Management of a Computer System

the user of a computer system needs

- reliability of his processes
- security of his data
- access to the attached devices
- access to the network

→ job of system administrator

Literature

- FreeBSD Handbook
http://www.freebsd.org/doc/en_US.ISO8859-1/books/handbook/
- the manual pages of the FreeBSD system, for example
`man security`
- the Internet
sans.org, schneier.com, www.heise.de/security/

Parts of an OS

An operating system consists of software to enable

- process management
- memory management
- file system
- input/output

of a computer system.

1973 first UNIX system by AT&T

1979 UNIX V6 (1BSD) with C, UUCP, Bourne Shell

1980 Microsoft: Xenix

1984 U. of California (Berkeley): 4.2BSD with TCP/IP

1986 U. of California (Berkeley): 4.3BSD with DNS and NFS

1987 MIT: X11 (→ XFree86 → X.org)

1991 U. of California (Berkeley): stop financial support of BSD

1991 386BSD splits into FreeBSD and NetBSD

1991 Linus Torvalds: Linux

1993 Novell buys AT&T UNIX Lab

1993 FreeBSD 1.0

1994 Linux 1.0

UNIX History

"...the number of UNIX installations has grown to 10,
with more expected..."

- Dennis Ritchie and Ken Thompson, June 1972

1995 OpenBSD splits from NetBSD (security)

...

2005 X.org splits from XFree86 (license debate)

...

2008 OpenSolaris

...

2010 Linux 2.6.33.2, FreeBSD 8.1, OpenBSD 4.7, NetBSD 5.0.2

...

2012 Linux 3.3.2, FreeBSD 9.0, OpenBSD 5.1, NetBSD 5.1.2

...

2015 Linux 3.19.5, FreeBSD 10.1, OpenBSD 5.7, NetBSD 6.1.5

UNIX Features

- multi-tasking
- multi-user
- virtual memory
- portable (written in C)
- symmetric multiprocessing (SMP)

Virtual Memory

If it's there, and you can see it, it's real.

If it's not there, and you can see it, it's virtual.

If it's there, and you can't see it, it's transparent.

If it's not there, and you can't see it, you erased it.

- IBM, 1978, on virtual memory

UNIX Standards

Consortium OpenGroup www.opengroup.org

- Capgemini (Netherlands)
- Hewlett-Packard (USA)
- IBM (USA)
- SAP (Germany)
- Shenzhen Kingdee Middleware (China)
- Sun Microsystems (USA)

POSIX (since 1988) en.wikipedia.org/wiki/POSIX

~(since 2003) OpenGroup+POSIX = Single UNIX Specification UNIX03

Philosophy

"It is not UNIX's job to stop you from shooting your foot.

If you so choose to do so,

then it is UNIX's job

to deliver Mr. Bullet to Mr Foot

in the most efficient way it knows."

-- Terry Lambert

UNIX Basic Commands

cp	copy files
mv	move/rename files
rm	remove files
cat	show file contents
more	show file pagewise
less	show file pagewise
gzip/gunzip	compress files
bzip2/bunzip2	compress files
tar	create/extract file archive
head	show first lines of files
tail	show last lines of files

Manual Pages

section meaning

- 1 Shell commands (tools, utilities) [ls]
- 2 System calls (kernel functions) [open]
- 3 Library calls (C library functions) [sqrt]
- 4 Device Drivers [lp]
- 5 File formats [/etc/passwd]
- 6 Games
- 7 Macro packages and conventions [man, groff]
- 8 System administration commands [mount]
- 9 (BSD) Kernel internals [suser]

Manual pages are searched in this order – first match is displayed.

Manual pages are located in /usr/share/man/

```
man -t grep > grep.ps # Save the PostScript version to a file
```

General Structure

command options files

Options are preceded by a *hyphen*: -a -b -c ...

Files are denoted by name or pattern

Patterns contain a

- “?” for an arbitrary character
- “*” for an arbitrary string

Manual pages contain possible options

Manual Page: Example

```
TOUCH(1)      FreeBSD General Commands Manual      TOUCH(1)
```

NAME

```
touch -- change file access and modification times
```

SYNOPSIS

```
touch [-acfhm] [-r file] [-t [[CC]YY]MMDDhhmm[.SS]] file ...
```

DESCRIPTION

The touch utility sets the modification and access times of files to the current time of day. If the file doesn't exist, it is created with default permissions.

Manual Page: Example

OPEN(2) FreeBSD System Calls Manual OPEN(2)

NAME

open -- open or create a file for reading or writing

LIBRARY

Standard C Library (libc, -lc)

SYNOPSIS

```
#include <fcntl.h>
```

```
int
```

```
open(const char *path, int flags, ...);
```

ASCII(7) FreeBSD Miscellaneous Information Manual ASCII(7)

NAME

ascii -- octal, hex and decimal ASCII character sets

...

00 NUL	01 SOH	02 STX	03 ETX	04 EOT	05 ENQ	06 ACK	07 BEL
08 BS	09 HT	0A NL	0B VT	0C NP	0D CR	0E SO	0F SI
10 DLE	11 DC1	12 DC2	13 DC3	14 DC4	15 NAK	16 SYN	17 ETB
18 CAN	19 EM	1A SUB	1B ESC	1C FS	1D GS	1E RS	1F US
20 SP	21 !	22 "	23 #	24 \$	25 %	26 &	27 '
28 (29)	2a *	2b +	2c ,	2d -	2e .	2f /
30 0	31 1	32 2	33 3	34 4	35 5	36 6	37 7
38 8	39 9	3a :	3b ;	3c <	3d =	3e >	3f ?
40 @	41 A	42 B	43 C	44 D	45 E	46 F	47 G
48 H	49 I	4a J	4b K	4c L	4d M	4e N	4f O
50 P	51 Q	52 R	53 S	54 T	55 U	56 V	57 W
58 X	59 Y	5a Z	5b [5c \	5d]	5e ^	5f _
60 `	61 a	62 b	63 c	64 d	65 e	66 f	67 g
68 h	69 i	6a j	6b k	6c l	6d m	6e n	6f o
70 p	71 q	72 r	73 s	74 t	75 u	76 v	77 w
78 x	79 y	7a z	7b {	7c	7d }	7e ~	7f DEL

...

Manual Page: Example**Manual Page: Structure**

NAME

SYNOPSIS

DESCRIPTION

OPTIONS (Linux)

FILES

EXAMPLES (BSD)

SEE ALSO

BUGS

COMPATIBILITY (BSD)

STANDARDS (BSD)

HISTORY (BSD)

AUTHOR (Linux)

UNIX Design (Files)

- everything *is a file*, which means, *has a file interface*
 - plain files
 - directories
 - devices
 - main memory
 - ...
- users are contained in groups
- files are owned by a user and a group
- there is *one* super-user with special privileges

Process: Layer-Model

User-Interface (Shell/GUI)

Application Program

Library (OS/Apl.)

System Calls (OS)

- delegate responsibility
- simplify bug fixes
- discourage reinventing the wheel

UNIX Design (Processes)

- a program is an executable file, this can be
 - a compiled program (from non-executable source C, C++)
 - an interpreted program (JAVA, Shell, Perl,...)
 - but Shell, JAVA-, Perl-Interpreter are compiled programs
- a process is a program currently executed by a processor
- a process uses code of the programmer, C-library, operating system
- a process returns the value ...
 - 0 in case of success \rightsquigarrow *true*
 - > 0 and ≤ 255 in case of failure \rightsquigarrow *false*
- processes are owned by the user starting it, group owner of the process is the user's login group
- processes manipulate files
- the process table may be viewed using the command `ps` (process status)

Process: Starting Other Processes

User-Interface (Shell/GUI)

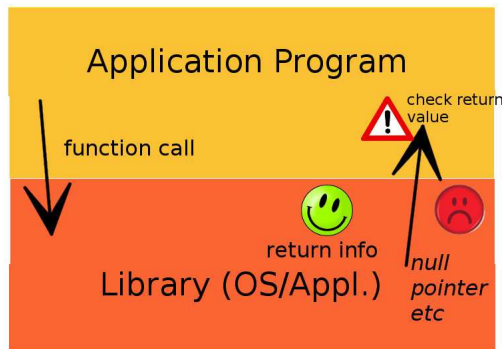
new process

Application Program



- processes are created by OS
- termination is accompanied by **exit value** \rightsquigarrow must be checked

Process: Programs Calling Libraries



- functions are called
- termination is accompanied by **return value** ~ must be checked

Errors on System Call

The system call returns -1.

CHECK ALL RETURN VALUES OF ALL FUNCTION CALLS!

The system call sets a global variable `int errno`.

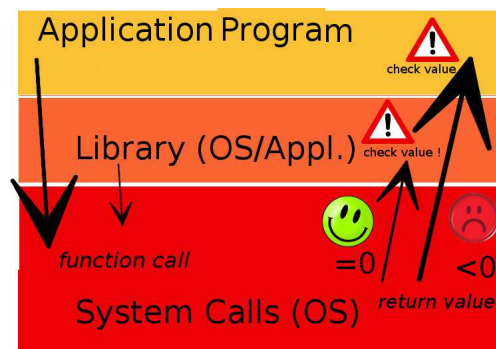
The global variable `errno` is defined in `errno.h`.

All possible values of `errno` are given by constants starting with E: `ENOMEM`, `EACCES`, `EINVAL`, ...

An error text for `errno` may be printed with `perror()`.

A string containing the error text may be created with `strerror()`.

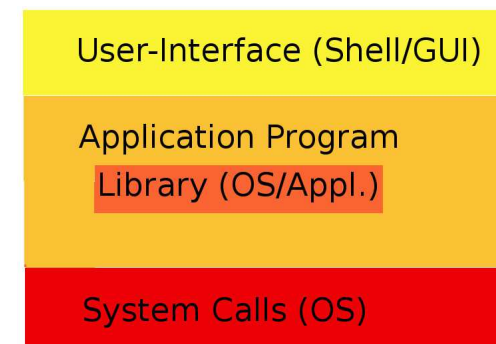
Process: Programs/Libraries Calling System Interface



- system calls (functions) are called
- termination is accompanied by **return value**
- return value **must** be checked for `< 0`

Library: Static Linking (1)

library functions incorporated into program (at compile time)



Library: Static Linking (2)

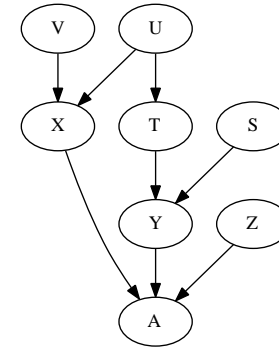
- + applications are more robust (run *standalone*)
- + cases saving performance: many calls to library routines
- + errors in new library version limited to corresp. programs
 - program size bigger
 - changes in library require recompiling program

static linking is enabled by

```
gcc -static ...
```

The Dependency Hell (with Dynamic Linking)

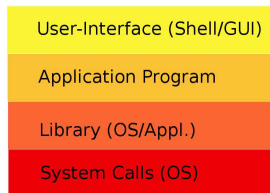
also called DLL-hell in Windows, JAR-hell in Java,...



~>package manager

Dynamic Linking

library functions load on demand into program (default, at run-time)



- + saves main memory, smaller size of executable
- + cases saving performance: already in mem, less likely paged out
- + simplifies security patches
 - errors are devastating
 - compatibility with library has to be ensured ~>*dependency hell*

Case Study: A C-Program