DATE 16

DESIGN, AUTOMATION & TEST IN EUROPE

14 – 18 March, 2016 · ICC · Dresden · Germany

The European Event for Electronic
System Design & Test

Trust but verify:
Why and how to establish trust in embedded devices

Aurélien Francillon

EURECOM
Sophia Antipolis

This talk

- Introduction
- Economic aspects
- Why make secure products?
- Trust in embedded devices
- Verifying trust
- Conclusion

Introduction

- **This talk: a “consequence” of about 10 years of working on the security of embedded systems software**
- **Practical approach**
  - **Attacking systems**
  - **Analysing systems (real products)**
  - **Developing new security mechanisms to make software more secure**
- **Unfortunately**
  - **A lot of this “systems security” knowledge is not public**
  - **Why is it so often so bad ?**

Problems found in a large scale analysis

- **Analysed ~30000 Firmware images**
- **Hard-coded passwords, SSL keys…**
  - **SSL private keys which are used by 40,000 IP on the internet...**
- **Same vulnerabilities across different products**
  - **Code sharing, Vulnerability sharing**
- **Several hundreds of vulnerable firmware images… tens of CVEs**
- **Web analysis: Many basic problems**

Automated Dynamic Firmware Analysis at Scale: A Case Study on Embedded Web Interfaces
A. Costin, , A. Zarras, A. Francillon AsiaCCS 2016
A Large Scale Analysis of the Security of Embedded Firmwares
A. Costin, J. Zaddach, A. Francillon, D. Balzarotti, Usenix Security 2014

Security for the 99%

- There are some very secure devices
  - Smartcards, HSMs, …
  - Not flawless but with a reasonable level of security
  - This is “1%” of the devices

Security for the 99%

- There are some very secure devices
  - Smartcards, HSMs, ...
  - Not flawless but with a reasonable level of security
  - This is “1%” of the devices
- The remaining 99% is not
  - Soho equipment
  - Computers peripherals
  - (Some) Industrial systems, etc.
- Security for the 99% ?

An economic problem

- **Intuitively security requires an extra effort**
  - **Costs money**
  - **Customers may not want to pay for it**

A bit more complicated…

- **Anderson / Schneier “economics of security”**
- **Security an externality:**
  - **Manufacturer often not responsible for operating the device**
  - **No direct loss in case of breach**
  - **So “why bother with security”**

Market for Lemons or silver bullets?

- **Markets with asymmetric information**
- **Market for Lemons: Used car market (Akerlof)**
  - **When selling a product seller knows more, buyer less**
  - **This drives down the average price of an used car**
- **Security products: Both seller and buyer lack information (Grigg)**
  - **Spafford: how to test a unicorn detection device?**
  - **Market for silver bullets**
- **Security products v.s. Product security**
  - **Product security is a lemons' market**

Motivations for trust/security on Manufacturers' side

Security considered when:

- **There are active attacks on asset to protect**
  - **Conditional access for Pay TV**
  - **Actual goal is to resist to the attacks**
- **Must not fail**
  - **E.g., critical military system**
  - **No need to be profitable**
- **Regulations, standards, certifications to pass**
  - **ID documents, payment processing**
  - **Actual goal is to get the certification**
- **For the 99% ?**

Economically speaking: Security or not?

- **In the short term, probably no...**
  - **Time to market, Cost**
  - **Users wants features**

    Schneier:

    “Any smart software vendor will talk big about security, but do as little as possible, because that's what makes the most economic sense.”

- **In the long term**
  - **A big problem**
  - **Maintenance, legacy, users defiance**
  - **Costs can be higher than the initial development**
  - **Life Cycle (How long will the manufacturer support it?)**

Transparency v.s. security

- Kerckhoffs 2nd design principle:
  - “... It should not require secrecy, and it should not be a problem if it falls into enemy hands”
- Often interpreted as:
  - “if the system is not open and does not receive public scrutiny then it is not secure”
  - Or ”Security by obscurity is bad”
  - A wrong interpretation
- Hiding the system details is actually making attacks much harder
  - Many more factors
- However, this has other bad effects...

Small digression...

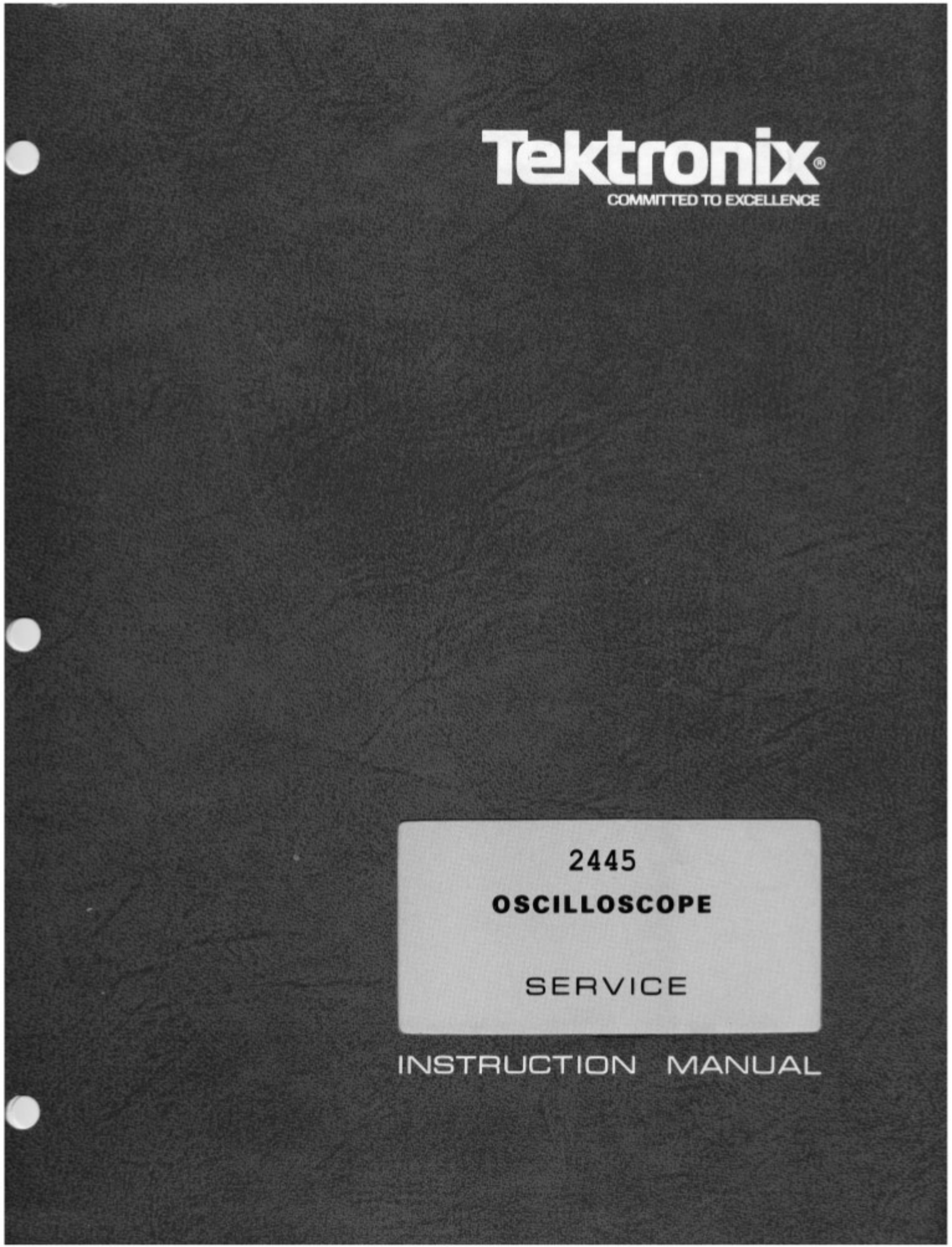
- One day I was given old scope for free to play at home...
- It worked 5 minutes and then the Magic Smoke escaped...

Small digression...

Tektronix 2445

Service manual

330 pages



Tektronix®
COMMITTED TO EXCELLENCE

2445
OSCILLOSCOPE

SERVICE

INSTRUCTION MANUAL

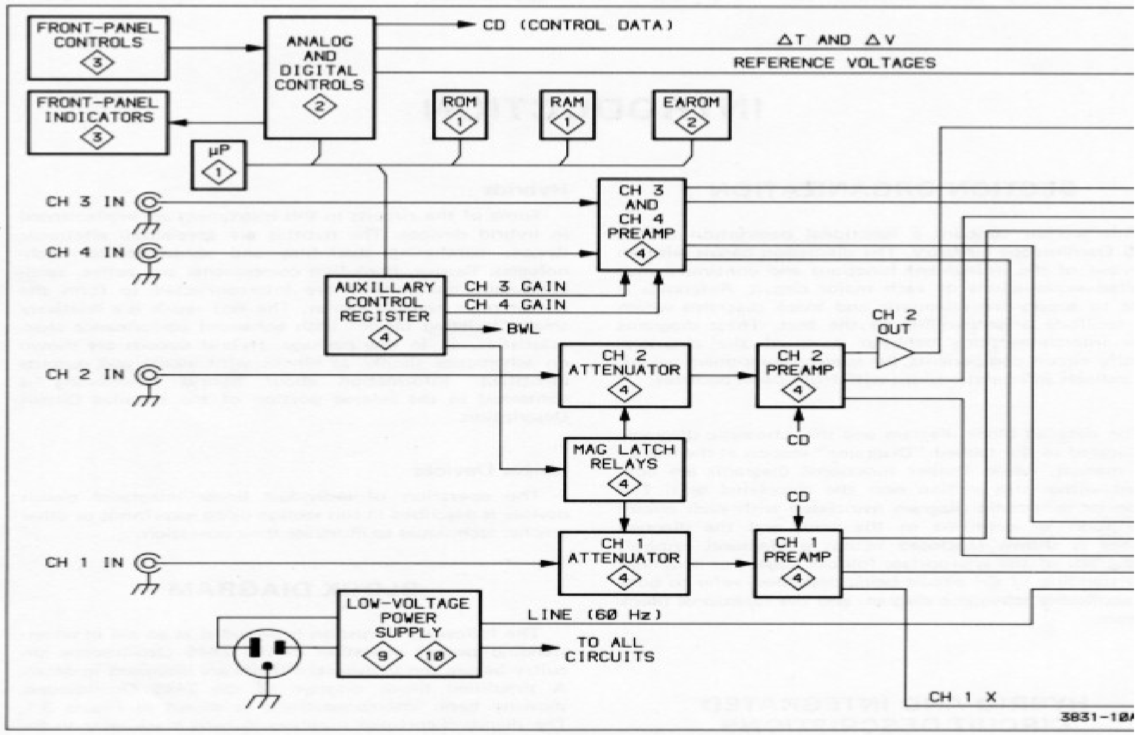


Figure 3-1. Block diagram.

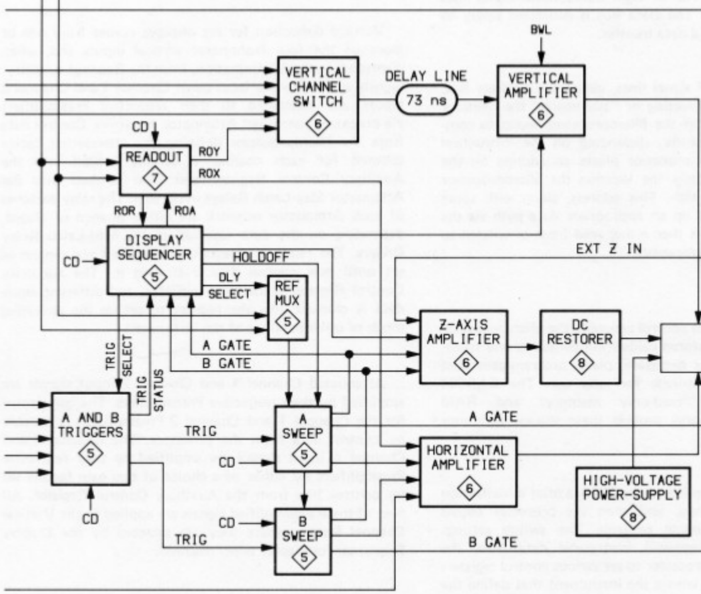
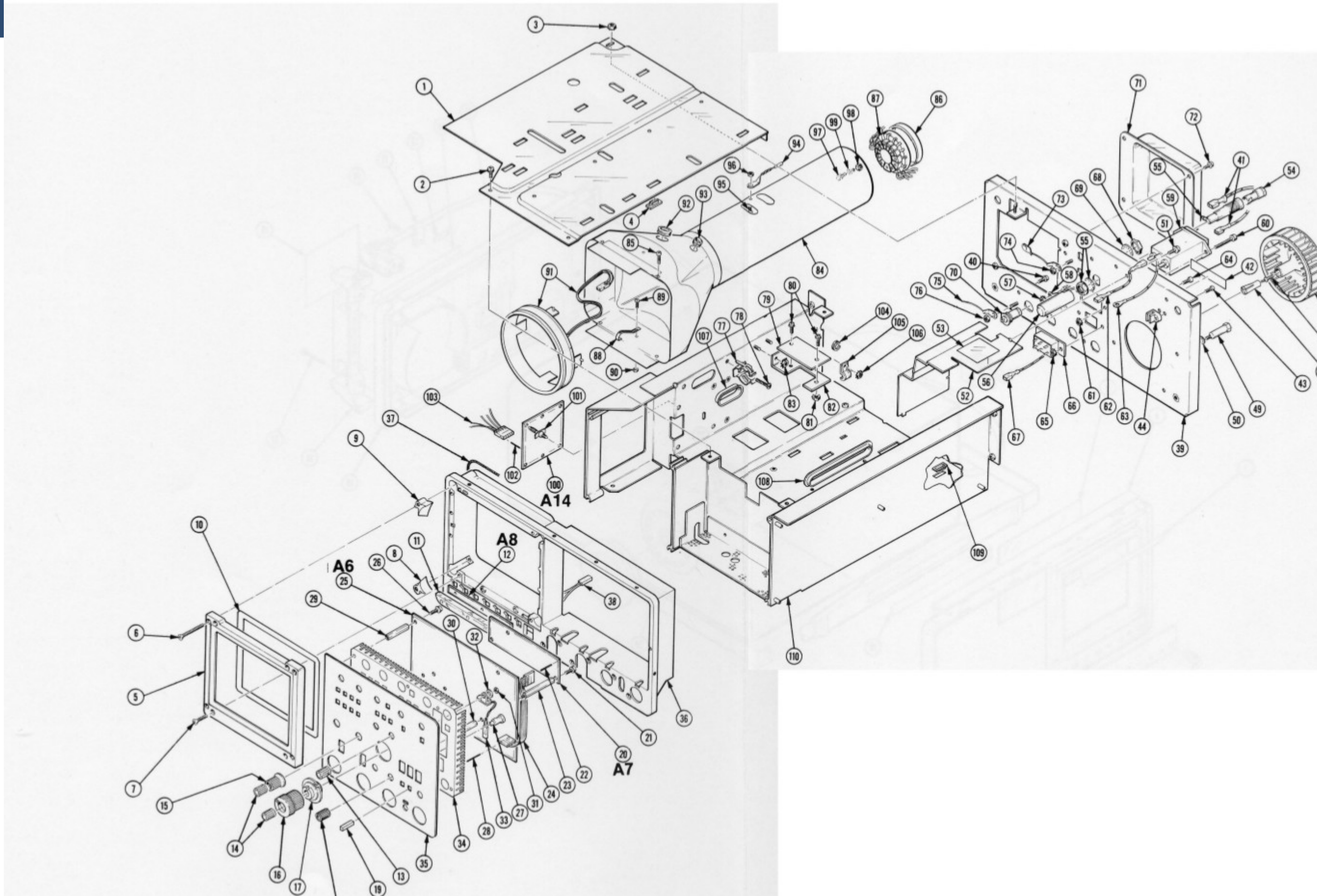
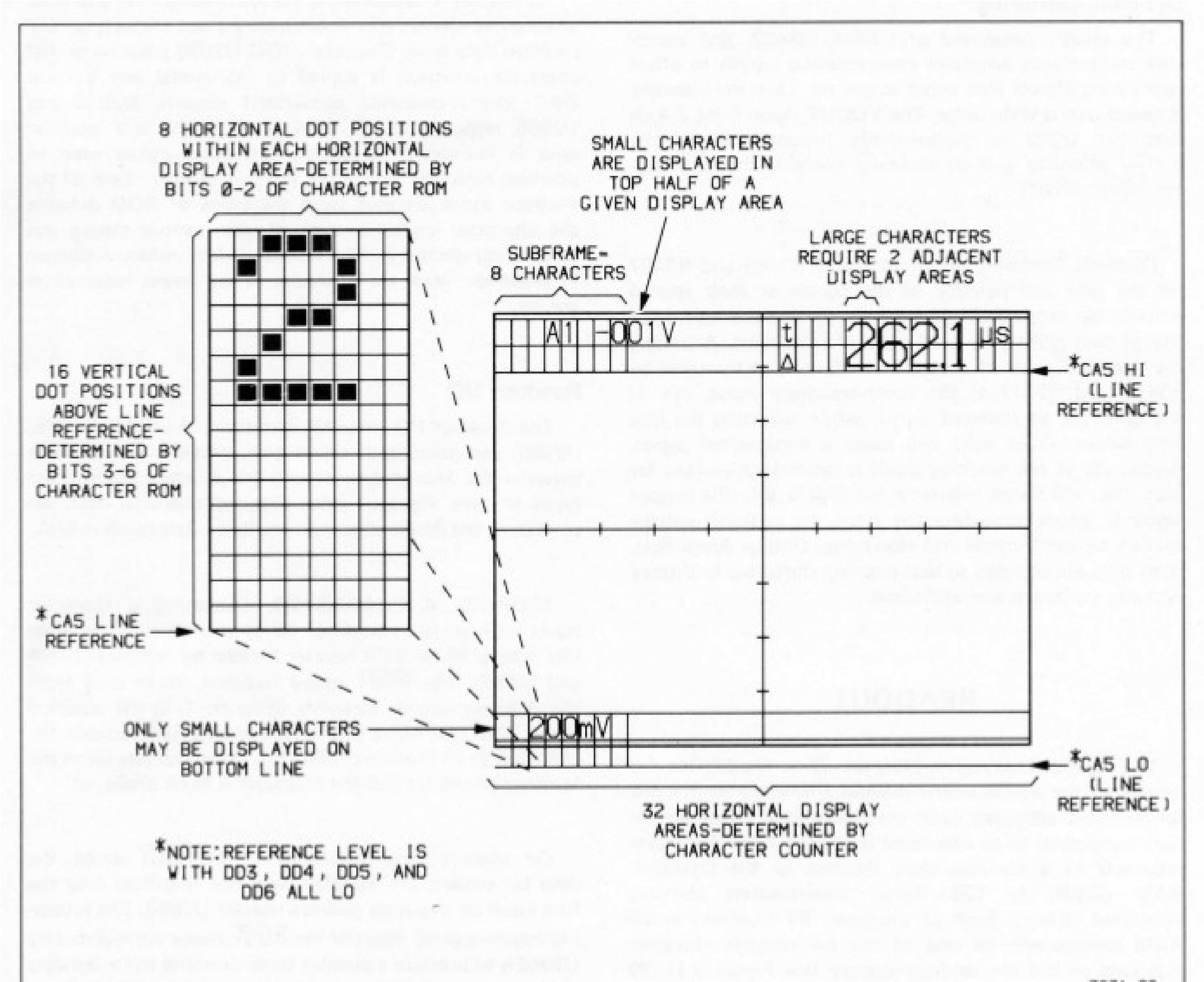
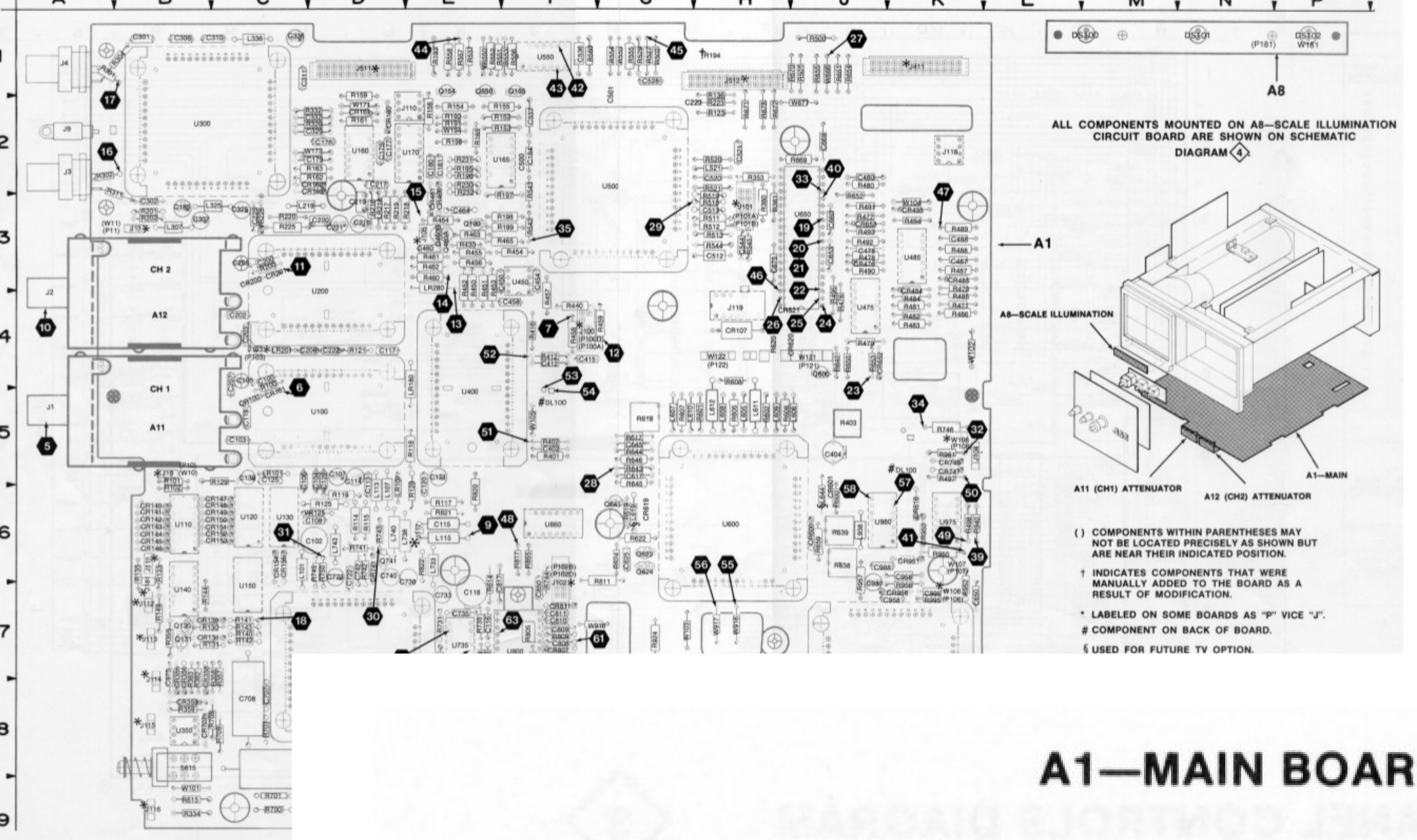


Figure 3-1. Block diagram (cont).


3
1
2
4
92
93
85
91
89
88
90
96
95
94
97
99
98
87
86
84
71
72
41
54
55
59
60
73
69
68
51
74
40
55
58
57
64
42
75
70
76
80
79
78
77
107
104
105
106
53
56
52
83
82
81
67
65
66
61
62
63
44
39
43
50
49
103
101
37
102
100
A14
9
108
109
110
10
A6
25
26
8
11
A8
12
38
29
6
30
32
5
36
21
20
A7
22
23
24
31
27
33
28
34
35
7
15
14
16
17
13
19


8 HORIZONTAL DOT POSITIONS WITHIN EACH HORIZONTAL DISPLAY AREA-DETERMINED BY BITS Ø-2 OF CHARACTER ROM
SMALL CHARACTERS ARE DISPLAYED IN TOP HALF OF A GIVEN DISPLAY AREA
SUBFRAME= 8 CHARACTERS
LARGE CHARACTERS REQUIRE 2 ADJACENT DISPLAY AREAS
A1 -001V
t
Δ
262.1 µs
*CA5 HI (LINE REFERENCE)
16 VERTICAL DOT POSITIONS ABOVE LINE REFERENCE-DETERMINED BY BITS 3-6 OF CHARACTER ROM
*CA5 LINE REFERENCE
ONLY SMALL CHARACTERS MAY BE DISPLAYED ON BOTTOM LINE
200mV
*CA5 LO (LINE REFERENCE)
32 HORIZONTAL DISPLAY AREAS-DETERMINED BY CHARACTER COUNTER
*NOTE: REFERENCE LEVEL IS WITH DD3, DD4, DD5, AND DD6 ALL LO



ALL COMPONENTS MOUNTED ON A8—SCALE ILLUMINATION CIRCUIT BOARD ARE SHOWN ON SCHEMATIC DIAGRAM 4

( ) COMPONENTS WITHIN PARENTHESES MAY NOT BE LOCATED PRECISELY AS SHOWN BUT ARE NEAR THEIR INDICATED POSITION.

† INDICATES COMPONENTS THAT WERE MANUALLY ADDED TO THE BOARD AS A RESULT OF MODIFICATION.

* LABELED ON SOME BOARDS AS "P" VICE "J".

COMPONENT ON BACK OF BOARD.

§ USED FOR FUTURE TV OPTION.

A1—MAIN BOARD

CIRCUIT NUMBER	SCHEM NUMBER	CIRCUIT NUMBER	SCHEM NUMBER	CIRCUIT NUMBER	SCHEM NUMBER	CIRCUIT NUMBER	SCHEM NUMBER	CIRC NUM
C100	4	C723	11	CR747	5	P100	4	R232
C102	11	C731	11	CR807	11	P100	4	R301
C103	4	C733	11	CR811	11	P101	5	R302
C104	4	C735	6	CR950	5	P101	5	R304
C105	4	C738	11	CR951	5	P102	5	R311
C106	11	C740	11	CR956	6	P102	5	R329
C107	11	C742	5	CR966	6	P103	4	R332
C108	11	C803	6	CR972	6	P106	6	R334
C113	11	C805	6	CR987	11	P107	5	R353
C114	11	C806	6	DL100	6	P108	5	R355
C115	4	C808	6	J9	5	P121	11	R357
C116	4	C809	6	J10	4	P121	11	R358

In the “good old days”...

- **Before there was documentation for:**
  - **Mini computers,**
  - **Apple II**
- **Today datasheets often not available, even for:**
  - **Raspberry PI**
  - **Intel Edison**
  - **Any secure device**

Trust and transparency

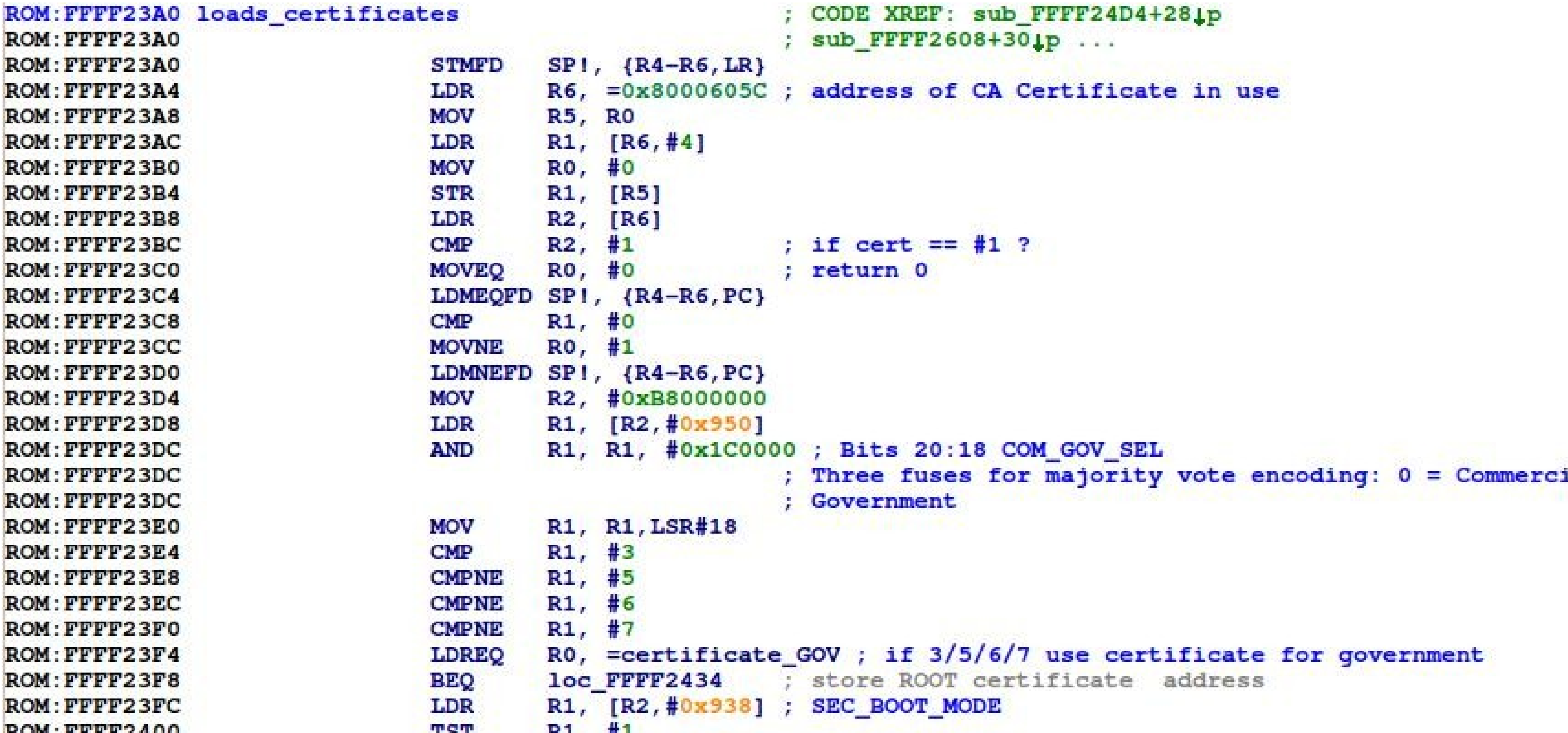
- To trust something we need to
  - Blindly trust?
  - Verify it, inspect it?
- Asymmetric market
  - Manufacturer knows
  - Customer cannot evaluate security
- Lack of transparency damages market of secure devices, users cannot:
  - Educate themselves
  - Learn about security
  - Evaluate security
  - Compare devices
- How could they want to pay more for security?

Lack of transparency

- **Basic security measures often make them less transparent**
- **Makes third party audit very hard**
  - **But does not mean the device is secure...**
- **Secrecy leads to suspicion**
  - **What is the device doing with my data?**
  - **Trying to hide a poor level of security?**
  - **Something nasty to hide?**

From an actual smartphone chip

- **Dumped a bootloader in Mask ROM**
  - **No FBI, it's not an iPhone!**



```
ROM:FFFF23A0 loads_certificates                      ; CODE XREF: sub_FFFF24D4+28↓p
ROM:FFFF23A0                                         ; sub_FFFF2608+30↓p ...
ROM:FFFF23A0                 STMFD   SP!, {R4-R6,LR}
ROM:FFFF23A4                 LDR     R6, =0x8000605C ; address of CA Certificate in use
ROM:FFFF23A8                 MOV     R5, R0
ROM:FFFF23AC                 LDR     R1, [R6,#4]
ROM:FFFF23B0                 MOV     R0, #0
ROM:FFFF23B4                 STR     R1, [R5]
ROM:FFFF23B8                 LDR     R2, [R6]
ROM:FFFF23BC                 CMP     R2, #1          ; if cert == #1 ?
ROM:FFFF23C0                 MOVEQ   R0, #0          ; return 0
ROM:FFFF23C4                 LDMEQFD SP!, {R4-R6,PC}
ROM:FFFF23C8                 CMP     R1, #0
ROM:FFFF23CC                 MOVNE   R0, #1
ROM:FFFF23D0                 LDMNEFD SP!, {R4-R6,PC}
ROM:FFFF23D4                 MOV     R2, #0xB8000000
ROM:FFFF23D8                 LDR     R1, [R2,#0x950]
ROM:FFFF23DC                 AND     R1, R1, #0x1C0000 ; Bits 20:18 COM_GOV_SEL
ROM:FFFF23DC                                         ; Three fuses for majority vote encoding: 0 = Commerci
ROM:FFFF23DC                                         ; Government
ROM:FFFF23E0                 MOV     R1, R1,LSR#18
ROM:FFFF23E4                 CMP     R1, #3
ROM:FFFF23E8                 CMPNE   R1, #5
ROM:FFFF23EC                 CMPNE   R1, #6
ROM:FFFF23F0                 CMPNE   R1, #7
ROM:FFFF23F4                 LDREQ   R0, =certificate_GOV ; if 3/5/6/7 use certificate for government
ROM:FFFF23F8                 BEQ     loc_FFFF2434    ; store ROOT certificate  address
ROM:FFFF23FC                 LDR     R1, [R2,#0x938] ; SEC_BOOT_MODE
ROM:FFFF2400                 TST     R1, #1
```

Security or lock out

- Who is in control of the device
  - Your Manufacturer?
  - Your government, another one?
  - Trusted Computing as “Treacherous Computing” (R. Stallman)
- Users should eventually be in control

Design problem

We need systems to be designed for:

- User Trust
  - Letting the choice to the user, owner of the device to which software is running on the device
  - Let the user know which software it is running
- Security Analysis
  - We need to be able to independently inspect those systems

Design for User Trust

- To trust the systems, users needs to:
  - Know what is running
  - Chose what can be running
  - Be in control
  - Be able to verify
- Currently there are devices which
  - We can control, but have zero security (e.g., unlocking android)
  - Are secure but under the control of someone else (iPhone)

Design for User Trust: examples

- My new laptop
  - Has an UEFI Firmware
  - Loaded with my own keys
  - Secure boot, only code I signed

- Joanna Rutkowska proposal of a state-less laptop
  - Without R/W memories
  - All firmware loaded from an external, trusted, device

Design for Security Testing

- **When do we really need to be able to analyse embedded devices?**

• **Each firmware version**	**Detect vulnerabilities**
• **Each independent device**	**Shipped with bad FW**
• **Regularly**	**Check for compromise**
• **Exceptionally**	**Forensics**

- **Need for independent analysis**
- **Requires some access to the device (DFUT)**
  - **But not reducing the security of the device… Authenticate users?**

Design for Security Testing

- Currently first security measures in an embedded system makes it harder to test:
  - Locking JTAG
  - Encrypt/Sign code
- Testing embedded systems is difficult We developed a tool for security testing
  - Avatar http://s3.eurecom.fr/tools/avatar/

In Summary

- We need more transparency
  - Datasheets!
  - Access to debug ports!
- Not because it makes devices more secure but it makes:
    - Auditable
    - Trustworthy
    - Forensics possible
- We need mechanisms that
  - Put users in control
  - Do not introduce new vulnerabilities
  - Are easy to integrate in products

Questions?

Backup slides

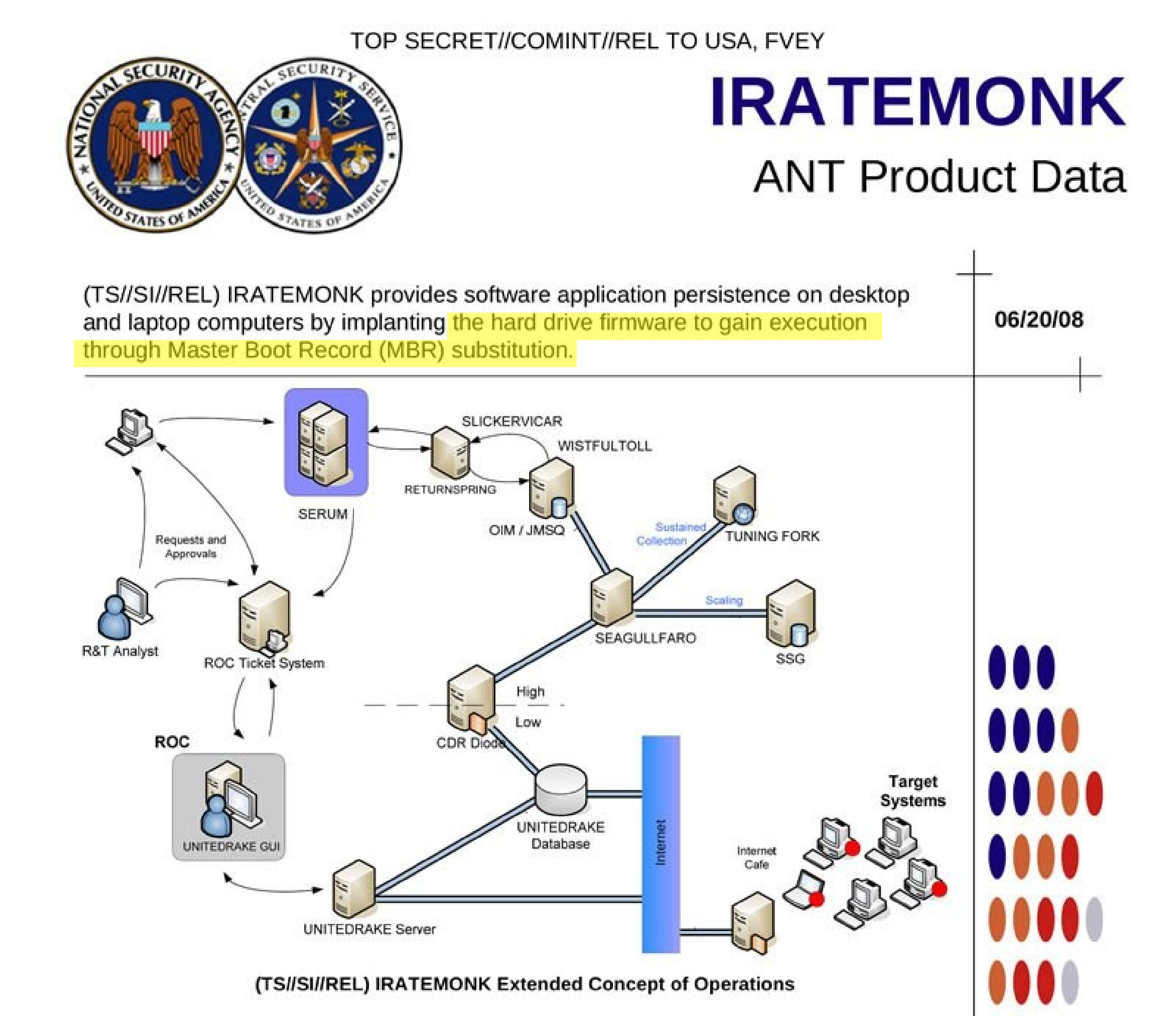
Liability

- Schneier argues for liability
  - Did not happen… will it one day?
- Probably in some regulated / life threatening markets?
  - Toyota sudden unintended acceleration
    - 9 Million cars recalled
    - 37 deaths alleged
- Will this occur for the 99%?
  - I guess not

Hard disk drive security

- A disk Drive runs a firmware
  - with its own OS
  - Can be updated
- Could be compromised
  - what would be the consequences ?
  - The required effort
- To discover it we did it
  - Took a disk and reverse engineered it
  - designed a backdoor
- So yes, feasible but difficult, but a few days later...

Implementation and Implications of a Stealth Hard-Drive Backdoor
J. Zaddach, A. Kurmus, D. Balzarotti, E. Blass, A. Francillon, T. Goodspeed, M. Gupta, I. Koltsidas, best student paper award, ACSAC 2013,



TOP SECRET//COMINT//REL TO USA, FVEY

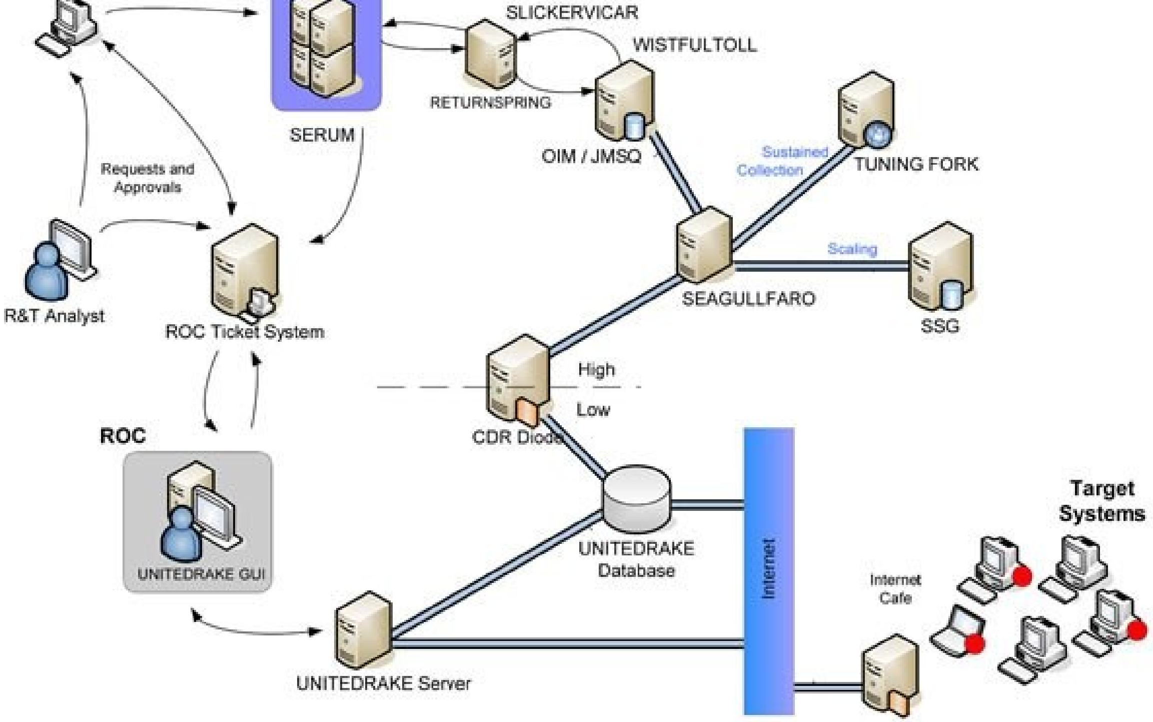
IRATEMONK

ANT Product Data

(TS//SI//REL) IRATEMONK provides software application persistence on desktop and laptop computers by implanting the hard drive firmware to gain execution through Master Boot Record (MBR) substitution.

06/20/08

(TS//SI//REL) IRATEMONK Extended Concept of Operations



(TS//SI//REL) IRATEMONK Extended Concept of Operations

(TS//SI//REL) This technique supports systems without RAID hardware that boot from a variety of Western Digital, Seagate, Maxtor, and Samsung hard drives. The supported file systems are: FAT, NTFS, EXT3 and UFS.

(TS//SI//REL) Through remote access or interdiction, UNITEDRAKE, or STRAITBAZZARE are used in conjunction with SLICKERVICAR to upload the hard drive firmware onto the target machine to implant IRATEMONK and its payload (the implant installer). Once implanted, IRATEMONK's frequency of execution (dropping the payload) is configurable and will occur when the target machine powers on.

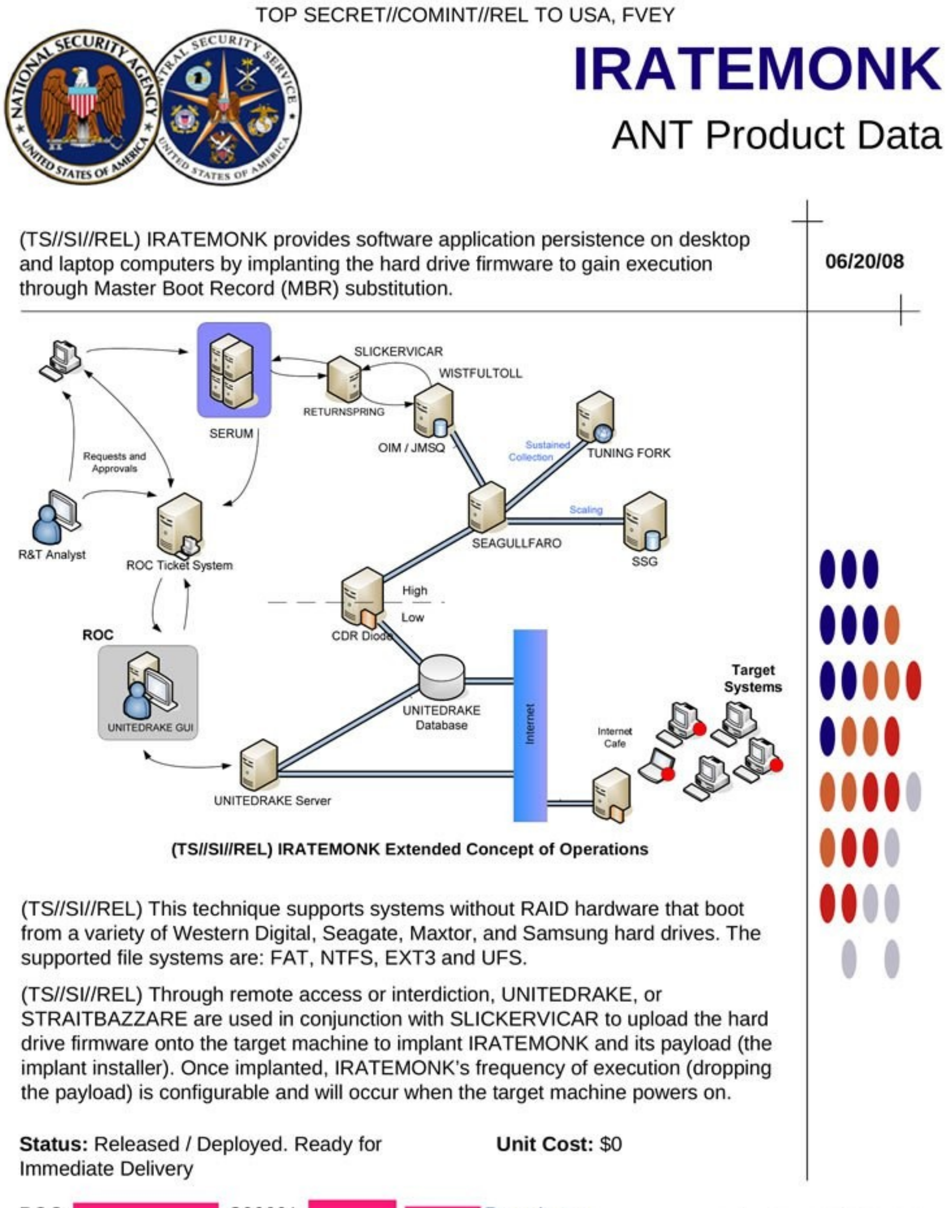
Status: Released / Deployed. Ready for Immediate Delivery

Unit Cost: $0

POC: [redacted], S32221, [redacted], [redacted]@nsa.ic.gov

Derived From: NSA/CSSM 1-52
Dated: 20070108
Declassify On: 20320108

IRATEMONK (12/2013)



TOP SECRET//COMINT//REL TO USA, FVEY

IRATEMONK

ANT Product Data

06/20/08

(TS//SI//REL) IRATEMONK provides software application persistence on desktop and laptop computers by implanting the hard drive firmware to gain execution through Master Boot Record (MBR) substitution.

SLICKERVICAR
WISTFULTOLL
RETURNSPRING
SERUM
OIM / JMSQ
Sustained Collection
TUNING FORK
Requests and Approvals
Scaling
R&T Analyst
ROC Ticket System
SEAGULLFARO
SSG
High
Low
ROC
CDR Diode
UNITEDRAKE GUI
UNITEDRAKE Database
Internet
Target Systems
Internet Cafe
UNITEDRAKE Server

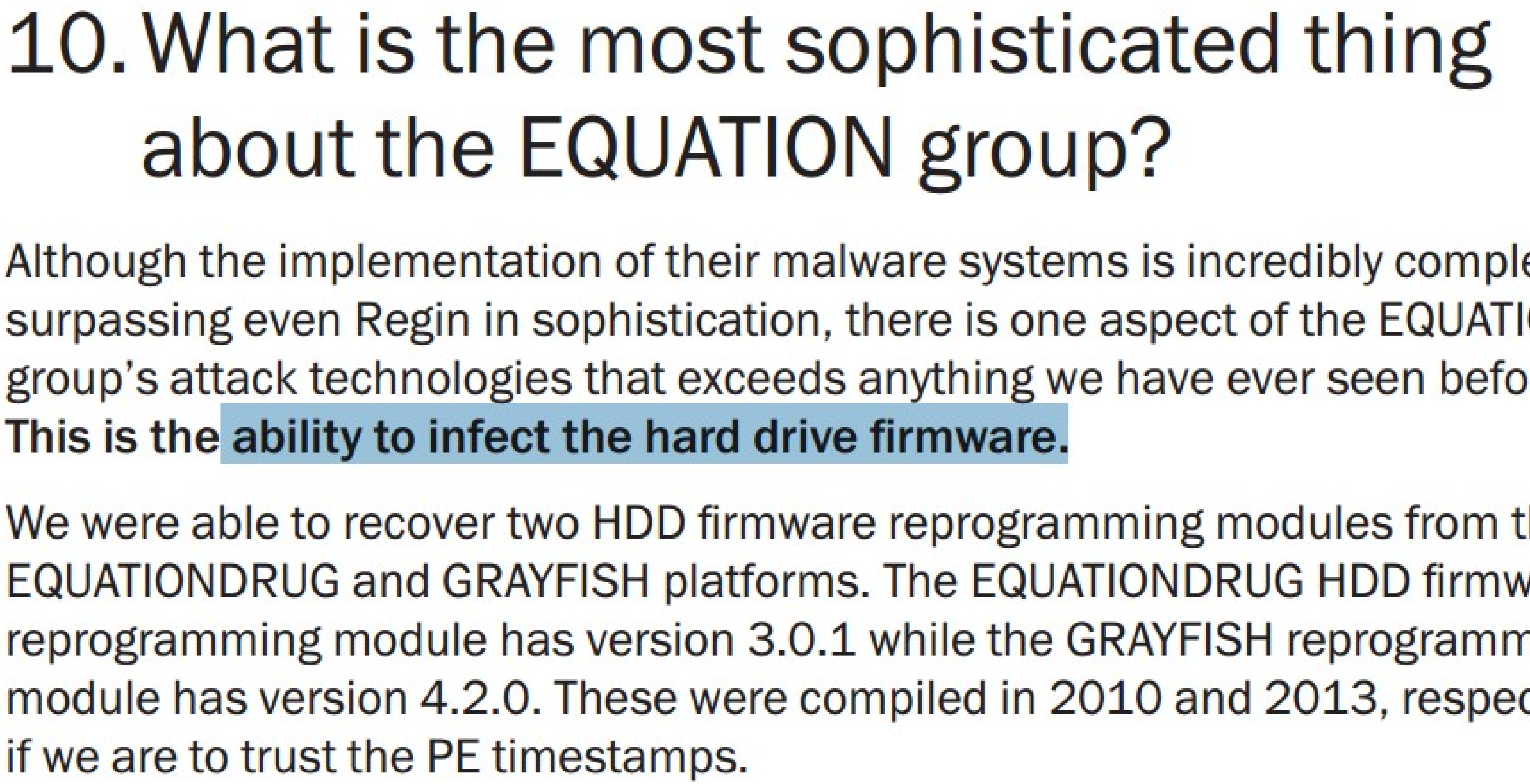
(TS//SI//REL) IRATEMONK Extended Concept of Operations

(TS//SI//REL) This technique supports systems without RAID hardware that boot from a variety of Western Digital, Seagate, Maxtor, and Samsung hard drives. The supported file systems are: FAT, NTFS, EXT3 and UFS.

(TS//SI//REL) Through remote access or interdiction, UNITEDRAKE, or STRAITBAZZARE are used in conjunction with SLICKERVICAR to upload the hard drive firmware onto the target machine to implant IRATEMONK and its payload (the implant installer). Once implanted, IRATEMONK's frequency of execution (dropping the payload) is configurable and will occur when the target machine powers on.

Status: Released / Deployed. Ready for Immediate Delivery

Unit Cost: $0

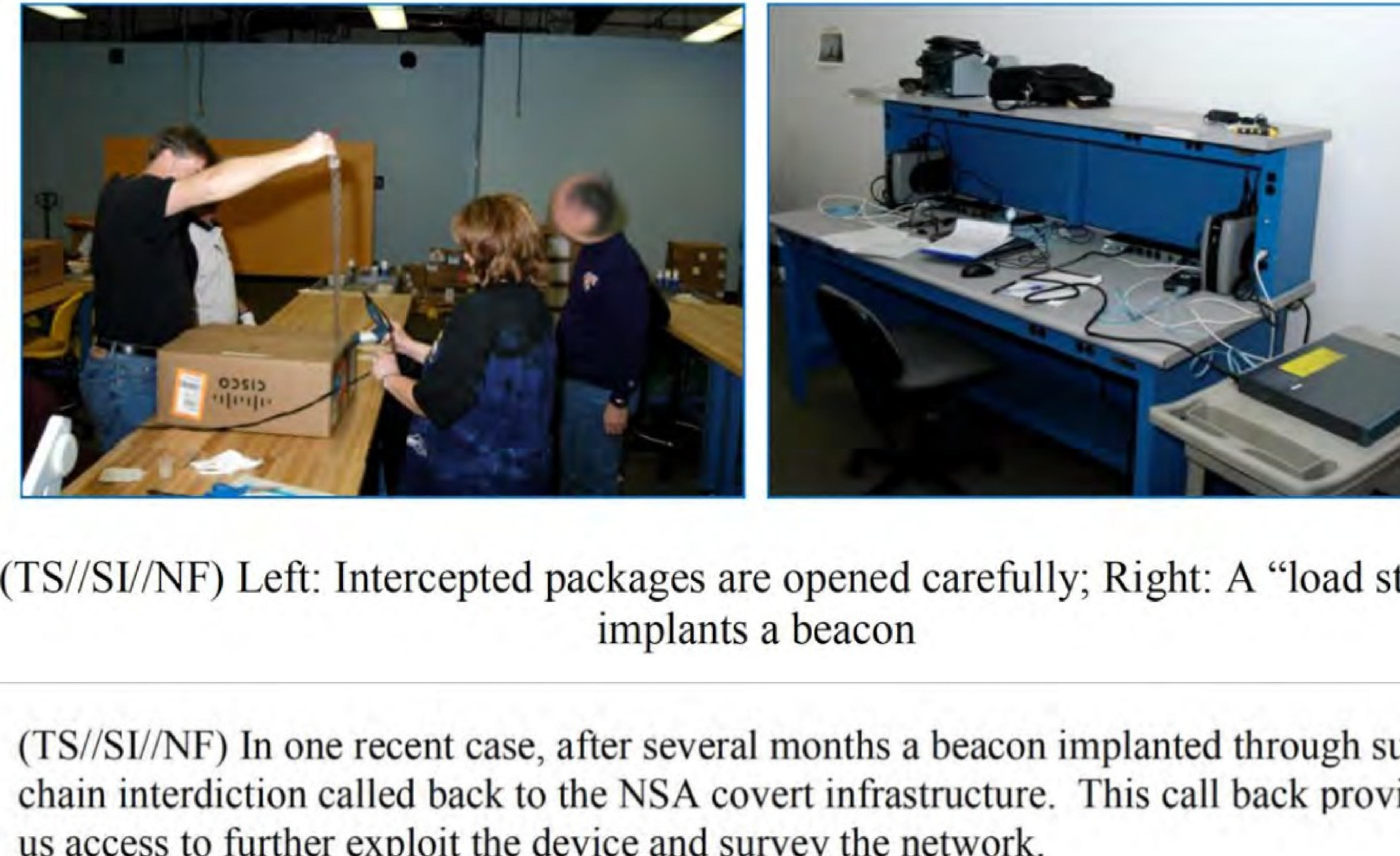


10. What is the most sophisticated thing about the EQUATION group?

Although the implementation of their malware systems is incredibly comple
surpassing even Regin in sophistication, there is one aspect of the EQUATI
group's attack technologies that exceeds anything we have ever seen befo
This is the ability to infect the hard drive firmware.

We were able to recover two HDD firmware reprogramming modules from t
EQUATIONDRUG and GRAYFISH platforms. The EQUATIONDRUG HDD firmw
reprogramming module has version 3.0.1 while the GRAYFISH reprogramm
module has version 4.2.0. These were compiled in 2010 and 2013, respec
if we are to trust the PE timestamps.

Snowden documents on “interdiction”



(TS//SI//NF) Left: Intercepted packages are opened carefully; Right: A “load st
implants a beacon

(TS//SI//NF) In one recent case, after several months a beacon implanted through su
chain interdiction called back to the NSA covert infrastructure. This call back provi
us access to further exploit the device and survey the network.