Hybrid III: designed to gather data from frontal impacts

CODENOMICON

Fuzzing Embedded Devices

Finding unknown vulnerabilities in home electronics

Rikke Kuipers and Ari Takanen

DEFEND. THEN DEPLOY.

Industry is Slowly Waking Up to the Unknown Threats

"All software has undetected exploitable vulnerabilities"
- Security Vendor 2009

"All our zero-day vulnerabilities were found with Fuzzing."
– Software Vendor 2010

"You would be a fool not to Fuzz."
– Analyst 2011

CODENOMICON

What is Fuzzing?

- A testing technique where purposefully unexpected and/or invalid input data is fed to tested system in hope to find robustness and security problems

CODENOMICON

Fuzzing Techniques

- Mutation/Template-Based Fuzzing
  - Quality of tests is based on the used template (seed) and mutation technique
  - Slow to execute, least bugs found

- Generational/Specification-Based Fuzzing
  - Full test coverage, as the model is built from specification
  - Fast to execute, most bugs found

CODENOMICON

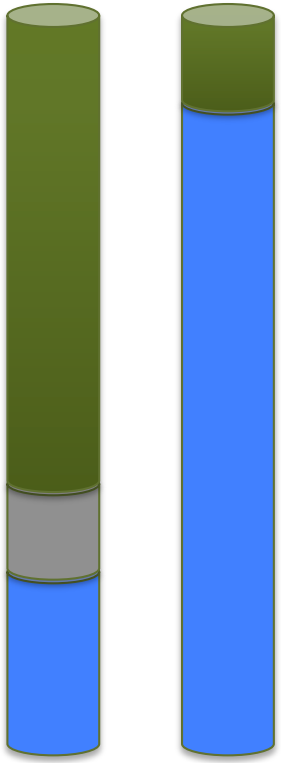
Fuzzer Efficiency Case Study

- Most important efficiency metric for fuzzers:
  - How many bugs does it find
  - How much time does it take to find them

"Smart" model-based generational fuzzer found 10 unique bugs

Both found 2 same bugs

Mutation fuzzer found 4 unique bugs



Generation fuzzer executed for 17 hours

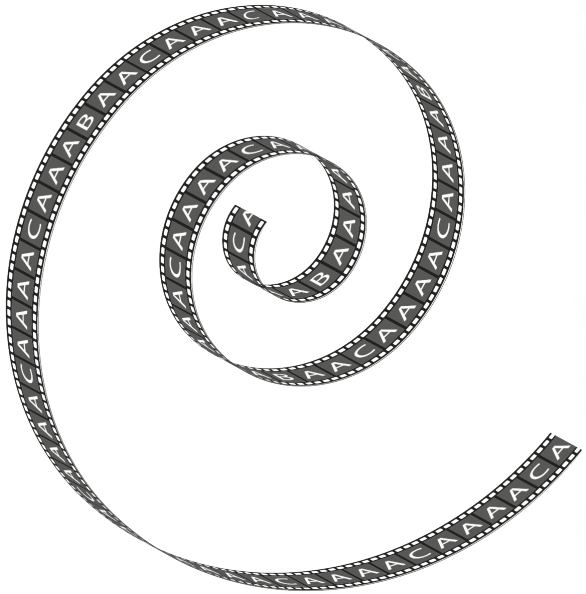
Mutation fuzzer took 118 hours (5 days) to execute, **after which no more new bugs were found**

Codenomicon Labs

- We have tested and released test reports on:
  - WiFi access points
  - Bluetooth devices (including cars and medical devices)
  - NAS devices
  - Printers
  - Browsers
  - **Smart TVs**
- **Any idea what we should test next?**

CODENOMICON

TV Attack Vectors



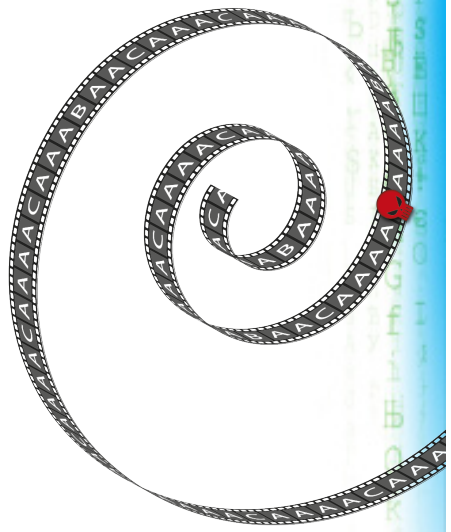
- Dumb DVB-enabled TVs
  - DVB-C/T
  - IR
- Media center TVs
  - basic network connectivity: IPv4, UPnP, DLNA, DHCP, HTTP, FTP
  - Digital media: images, videos, audio
  - USB and memory cards
  - Bluetooth and WiFi (client)
  - limited network services
- Internet-enabled TVs
  - “Web 2.0 client”
  - Applets, applications, widgets
  - Full browser
  - **Capability very similar to smart phones**

CODENOMICON

Demo/Video about JPG fuzzing

New attack vector: DVB

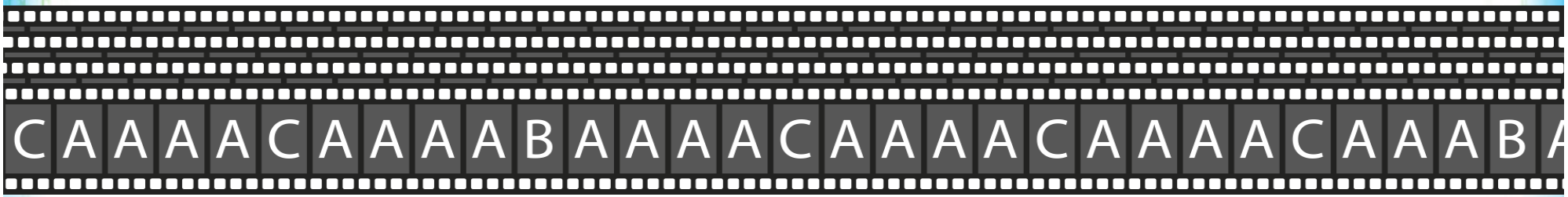
- Not just to transmit video/audio streams
  - vehicle to vehicle networks
  - navigation systems
  - handheld communications (DVB-H)
  - internet transport (IP-over-DVB/MPEG)
  - military (DVB-S2)



- DVB stream can contain several “channels” multiplexed into one stream, de-multiplexed at the receiver

Structure of MPEG2-TS / DVB

- Audio/video streams, in channel bundles
- Informational “tables” about the payload content, such as Program Association Table (PAT)



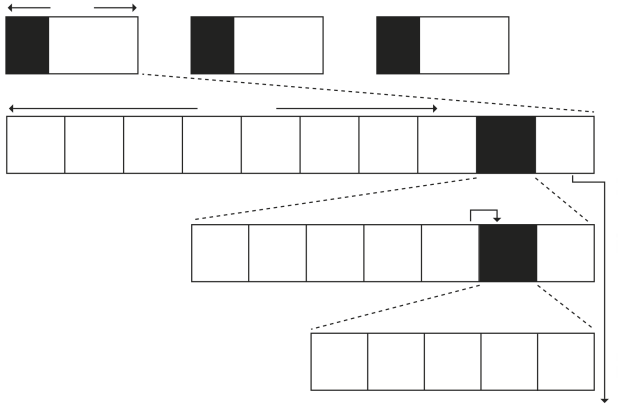
- Each type of frame or table needs to be fuzzed

CODENOMICON

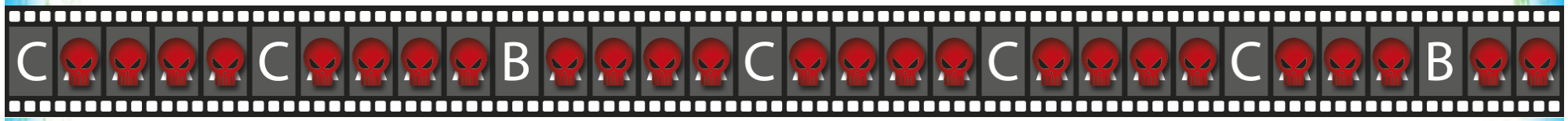
DVB Fuzzing

- Selection of tool:
  - Mutation fuzzing is the easiest, and most interoperable, but can be country-specific
  - Model-based, generation fuzzing, is more optimized to find bugs faster
  - Our solution was easy: Defensics MPEG2-TS fuzzer (available since 2010)
- Injection:
  - Our solution: after “file fuzzing”, multiplexing the stream back to right format, and then injecting using an off-the-shelf modulator

Fuzzing MPEG2-TS



- MPEG-TS acts like a protocol
- When fuzzing, each feature needs to be fuzzed separately



- “Common frames” require pauses between tests so that stream stays in sync

- “Rare frames” can require long streams so that all functionality is tested

Anomalization

input ANOMALIES ⟷ expose VULNERABILITIES

FIELD LEVEL
overflows,
integer anomalies

STRUCTURAL
underflows,
repetition of elements,
unexpected elements

SEQUENCE LEVEL
out of sequence
omitted / unexpected
repetition / spamming

WHAT FUZZING FINDS

crashes

denial of service (DoS)

security exposures

performance degradation

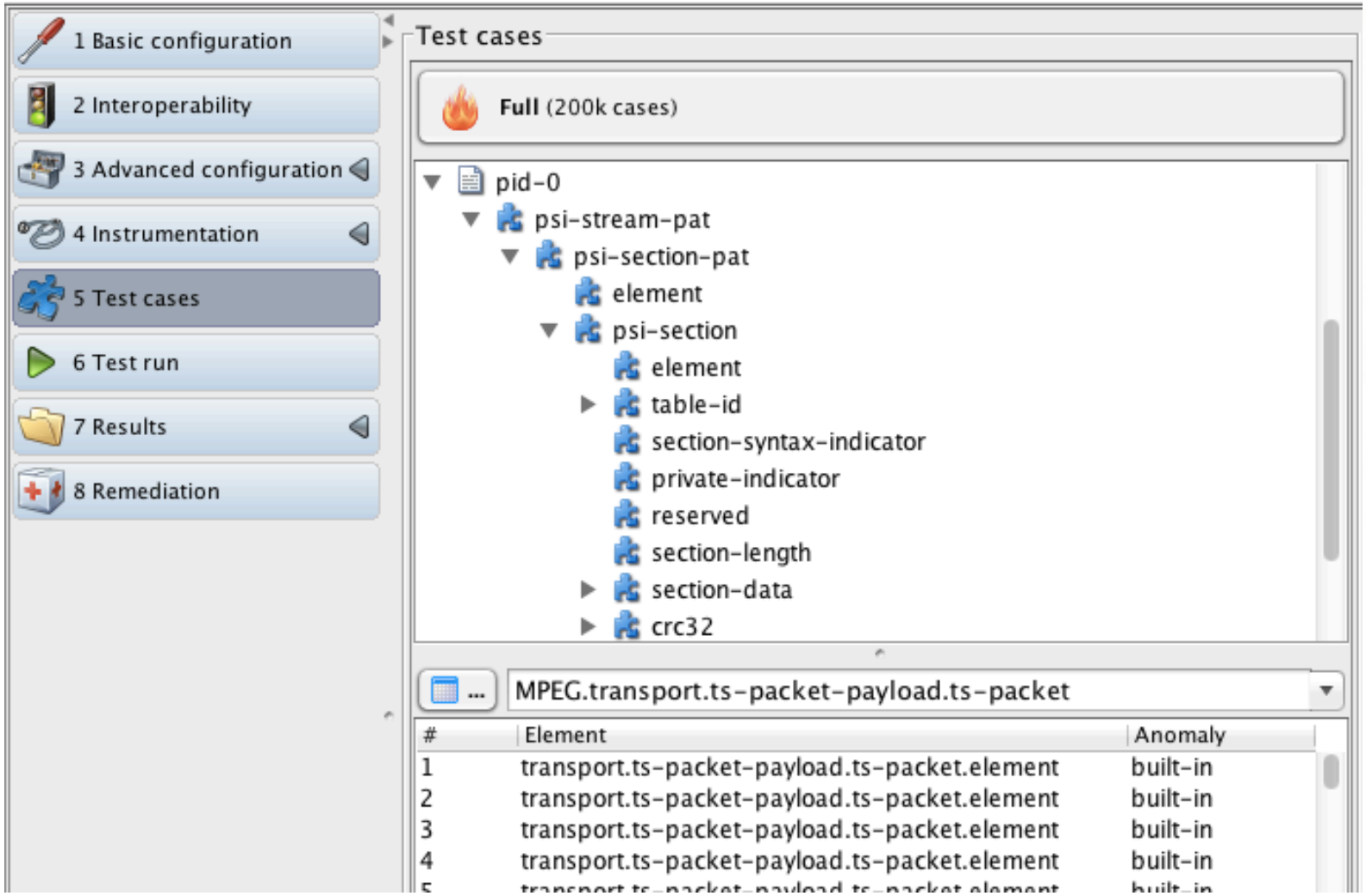
slow responses

thrashing

anomalous behavior

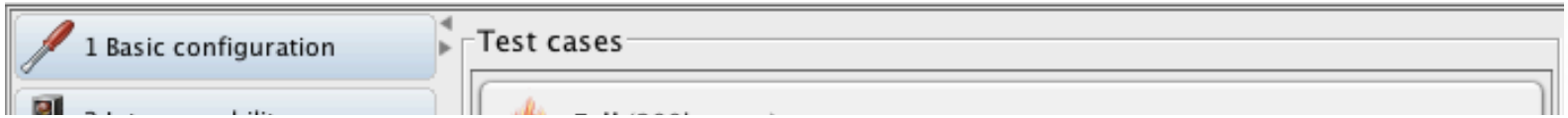
CODENOMICON

Example Anomalies



CODENOMICON

Example Anomalies



Message

000000	stream		
000000	ts-packet-payload		
000000	ts-packet		
000000	ts-prefix		
000000	sync-byte	G	47
000001	trans-error	1bit	0
	pay-start	1bit	0
	trans-prio	1bit	0
	pid	13bit	00000 00110001
000003	tsc		
000003	not-scrambled	2bit	00
	afc		
	adaptation-payload	2bit	11
	ctr	4bit	1111
000004	adaptation-payload		
000004	ts-adaptation-field		
000004	length	*	2a
000005	ts-payload		
000005	ts-payload-data		( )
000005	stuffing		( )


Message
000000 stream
000000 pes-stream-ac3
000000 pes-packet-ac3
000000 ...
000003 stream-id
000003 private-1
000004 length
000006 pes-packet-header
000006
scrambling-control
not-scrambled
priority
data-aligment-indicator
copyright
original-or-copy
000007 flags
000007
pts-dts
escr
es-rate
dsm
aci
crc
ext
000008 length
000009 variable-length
000009
presentation-timestamp
base-32-30
mbit
00000a base-29-15
mbit
00000c base-14-0
mbit
00000e stuffing
00000e ac3-bitstream
00000e ac3-syncframe
00000e ac3-syncinfo
00 00 01
. bd
.. 00 01
2bit 10
2bit 00
1bit 0
1bit 1
1bit 1
1bit 0
1bit 0
1bit 0
1bit 0
. 05
4bit 0010
3bit 101
1bit 1
15bit 00100101 0011100
1bit 1
15bit 11001100 1010110
1bit 1
( )
( )
( )

CODENOMICON

Modulation

"Thanks Sofia Digital!"

p0c@code:~$ DtPlay stream.ts -n 1 -t 215 -ml -27.5 -i 1 -mt OFDM -mC QAM64 -mf 530 -mG 1/8 -mc 2/3 -mT 8k -r 22100000

-n Device number to use: **1**

-t Device type to use: **215** (DTU-215)

-i Port number of the output channel to use: **1**

-r Transport-Stream Rate in bps or sample rate in case of IQ-modulation mode: **22100000**

-ml Output level in dBm: **-27.5**

-mG DVB-H/DVB-T guard interval: **1/8**

-mc Convolutional rate: **2/3**

-mf Modulation carrier frequency in MHz: **530**

-mt Modulation type: **OFMD**

-mC ATSC/DVB-H/DVB-T/DTMB constellation: **QAM64**

-mT DVB-H/DVB-T transmission mode: **8k**

Interoperability and Instrumentation

- Before fuzzing, the features in the target device need to be scanned
- Valid sequences are the easiest method, by building valid traffic directly from protocol specification
- Target device can be “picky” on what data it will accept
- Our solution: Capture of the local national TV stream is fed to the fuzzer

- In DVB, you cannot use valid sequences for instrumentation, as tests are unidirectional
- ICMP heart-beat “ping” is a simple instrumentation

CODENOMICON

Results

Protocol/ TV	TV 1	TV 2	TV 3	TV 4	TV 5	TV 6
IPv4	pass	**FAIL**	**FAIL**	pass	pass	**FAIL**
DVB	**FAIL**	**FAIL**	**FAIL**	**FAIL**	**FAIL**	**FAIL**
UPnP	n/a	**FAIL**	pass	n/a	n/a	**FAIL**
Images	pass	**FAIL**	**FAIL**	n/a	n/a	**FAIL**
Audio	pass	pass	n/a	n/a	n/a	pass
Video	**FAIL**	**FAIL**	n/a	**FAIL**	**FAIL**	**FAIL**

"FAIL" means multiple repeatable crashes were found
"pass" means the system did not crash (more fuzzing needed?)
"n/a" means the interface did not exist, or was not tested

We did not analyze the failures for exploitability.

CODENOMICON

Analysis

- As far as we know, there are no other fuzz tests against TVs

- No IPv6 yet in any TV
- Bad quality IPv4 still around
- DVB was easiest attack vector, probably because no DVB fuzzing available before this
- Video testing was with one container/codec only, but still lots of failures

- **TV fuzzing might not be high priority, but most likely same DVB/MPEG codecs are also used in other industry domains**

CODENOMICON

THANK YOU!
QUESTIONS to:
ari.takanen@codenomicon.com

“Thrill to the excitement of the chase! Stalk bugs with care, methodology, and reason. Build traps for them.
....
Testers!
Break that software (as you must) and drive it to the ultimate
- but don’t enjoy the programmer’s pain.”

[from Boris Beizer]