



CODENOMICON

Intelligent Bluetooth Fuzzing – Why bother?

T.Mäkilä & J.Taimisto



Overview

CODENOMICON

- Test results
- Anomaly gallery
 - What works
 - What will work
- Why anomalies work
- Security measures
- Closing



CODENOMICON

Disclaimer

- We get paid to make fuzzers, not exploits
- We concentrate on how to make implementations crash or malfunction
- Once we crash an implementation, our work is done



CODENOMICON

Test results

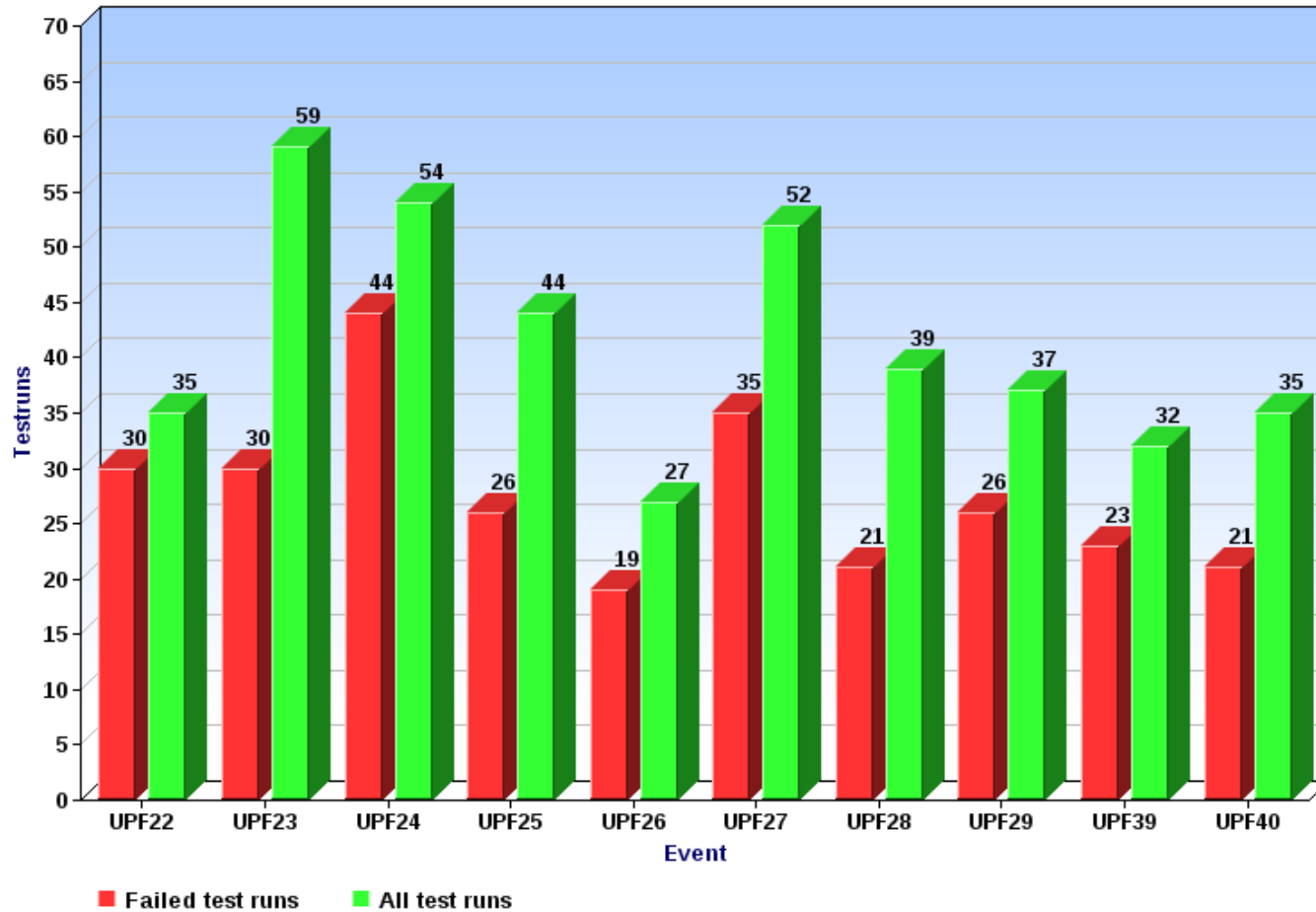
- We've been doing Bluetooth testing for 5 years
- Attending UPF's since 2006 (UPF #22)
- Test result have not shown significant improvement



Test results

CODENOMICON

Bluetooth UnPlugFest Results





CODENOMICON

Test results

- Carkit testing bonanza 2011
 - 15 different carkits tested
 - Found problems with 13 of them
 - Some in-car carkits required dealer service after testing
 - 10 crashed with L2CAP - No pairing required!
 - Tested profiles HFP, A2DP



Fuzzing

CODENOMICON

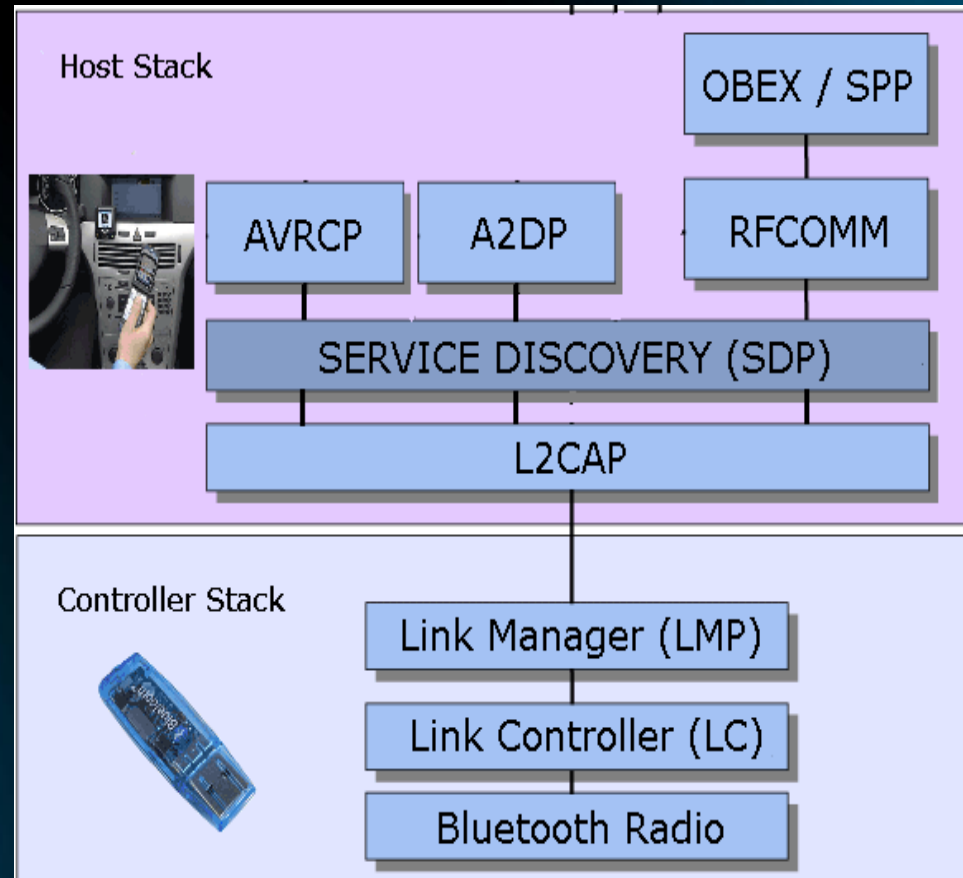
- Automated and efficient black-box testing method for finding software flaws
- Large amounts of anomalous (unexpected, invalid) inputs are sent to SW
- If the program fails, it indicates a bug in the software
- Fuzzing is used for security testing and as a quality assurance tool



Bluetooth Fuzzing

CODENOMICON

- No baseband, HCI or LMP fuzzing (Controller stack)
- Every protocol/profile from L2CAP above (Host stack)





CODENOMICON

Bluetooth Fuzzing

- Model-based fuzzing
- About 25 different protocols/profiles
- On average 10 000 test cases / profile
- No special hardware needed



CODENOMICON

EFFECTIVE ANOMALIES



Anomalies that work: Length field

- Under- and overflows of length fields are very efficient
- This is a good-old technique used for ages
- Especially good when applied to TLV structures



CODENOMICON

Anomalies that work: Length field

- Normal vs. Anomalous input

L2CAP Connect Request (VALID)

HEADER

ACL-Length : 0x0800
ACL-Channel : 0x0100

ACL_FRAME

Code : 0x02 [Connection Request]
Identifier : 0x01
Length : 0x0400
PSM : 0x0100 [Connect to SDP]
Source CID : 0x4000

L2CAP Connect Request (INVALID)

HEADER

ACL-Length : 0xFFFF
ACL-Channel : 0x0100

ACL_FRAME

Code : 0x02 [Connection Request]
Identifier : 0x01
Length : 0x0400
PSM : 0x0100 [Connect to SDP]
Source CID : 0x4000



CODENOMICON

Anomalies that work: TLV Length field

- Normal vs. Anomalous input

A2DP Set Configuration (VALID)

HEADER

Transaction-Label : 0b0001
Packet-Type : 0b00 [Single-Packet]
Message-Type : 0b00 [Command]
Signaling-ID : 0x03 [Set Configuration]
SEID-ACP : 0x00
SEID-Internal : 0x04

SERVICE CAPABILITIES

Type : 0x01 [Media Transport]
Length : 0x00
Value : ()

Type : 0x07 [Media Codec]
Length : 0x06
Value : 0b0000 [Audio]
0b0000 [RFA]
0x00 [SBC Codec Type]
0b0010 [44.1KHz]
0b0001 [Joint Stereo]
0b0001 [16 Blocks]
0b01 [8 Subbands]
0b01 [Loudness]
0x02 [Minimum Bitpool]
0x32 [Maximum Bitpool]

A2DP Set Configuration (INVALID)

HEADER

Transaction-Label : 0b0001
Packet-Type : 0b00 [Single-Packet]
Message-Type : 0b00 [Command]
Signaling-ID : 0x03 [Set Configuration]
SEID-ACP : 0x00
SEID-Internal : 0x04

SERVICE CAPABILITIES

Type : 0x01 [Media Transport]
Length : 0x00
Value : ()

Type : 0x07 [Media Codec]
Length : 0x00
Value : 0b0000 [Audio]
0b0000 [RFA]
0x00 [SBC Codec Type]
0b0010 [44KHz]
0b0001 [Joint Stereo]
0b0001 [16 Blocks]
0b01 [8 Subbands]
0b01 [Loudness]
0x02 [Minimum Bitpool]
0x32 [Maximum Bitpool]



CODENOMICON

Anomalies that work: TLV

- TLV structures are hard to parse.
- Anomalies in different parts of TLV structures yield good results
- Multi-field anomalies!



Anomalies that work: TLV

CODENOMICON

- Normal vs. Anomalous input

OBEX OPP Connect (VALID)

HEADER

Code : 0x80 [Connect]
Length : 0x0022

FRAME_DATA

OBEX Version : 0x01
SupportIrLMP : 0x00
Max Packet Length : 0x0400

HEADERS

Tag : 0x01 [NAME Header]
Length : 0x001B
Value : 0x0043006f0064 [Null terminated UTF16BE]
0x0065006e0066
0x006d00690063
0x006e0000

OBEX OPP Connect (INVALID)

HEADER

Code : 0x80 [Connect]
Length : 0x0022

FRAME_DATA

OBEX Version : 0x01
SupportIrLMP : 0x00
Max Packet Length : 0x0400

HEADERS

Tag : 0x00
Length : 0x001B
Value : 0x000000000000
0x000000000000
0x000000000000
0x00000000



Anomalies that work: Data structure fuzzing

- Oldest trick in the book
- Send too much or too little data
- Repeats of single character (on ascii-based profiles) are surprisingly effective



Anomalies that work: Data structure repeats

- Repeat complete valid TLV blocks, or valid messages
- Not just about the length
- Causes resource exhaustion etc.



CODENOMICON

Anomalies that work: Data structure repeats

- Normal vs. Anomalous input

A2DP Set Configuration (VALID)

HEADER

Transaction-Label : 0b0001
Packet-Type : 0b00 [Single-Packet]
Message-Type : 0b00 [Command]
Signaling-ID : 0x03 [Set Configuration]
SEID-ACP : 0x00
SEID-Internal : 0x04

SERVICE CAPABILITIES

Type : 0x01 [Media Transport]
Length : 0x00
Value : ()

Type : 0x07 [Media Codec]
Length : 0x06
Value : 0b0000 [Audio]
0b0000 [RFA]
0x00 [SBC Codec Type]
0b0010 [44.1KHZ]
0b0001 [Joint Stereo]
0b0001 [16 Blocks]
0b01 [8 Subbands]
0b01 [Loudness]
0x02 [Minimum Bitpool]
0x32 [Maximum Bitpool]

A2DP Set Configuration (INVALID)

HEADER

Transaction-Label : 0b0001
Packet-Type : 0b00 [Single-Packet]
Message-Type : 0b00 [Command]
Signaling-ID : 0x03 [Set Configuration]
SEID-ACP : 0x00
SEID-Internal : 0x04

SERVICE CAPABILITIES

Type : 0x01 [Media Transport]
Length : 0x00
Value : ()

Type : 0x01
Length : 0x00
Value : ()
Type : 0x01
Length : 0x00
Value: : ()
Type : 0x01
Length : 0x00
Value : ()
...
... [Repeat 1000 times]



Anomalies that work: Flooding data

- Some chipsets can't handle flooded data
- Sending for example 32k of data may cause the chipset to halt
- Requires hardware reset to recover



CODENOMICON

Common components

- Many profiles use either AT commands (SPP) or OBEX
- Different profiles may not share AT or OBEX parsers
- Same anomalies, same device, different profile -> different results
- Example : AT command anomaly does not work against HSP, but crashes HFP



CODENOMICON

FUTURE IMPROVEMENTS



CODENOMICON

Intelligent Bluetooth fuzzing

- We could develop more intelligent anomalies
- Multi-field anomalies
- Instead of parsers, attack state machines
- Multi-profile anomaly injection
- Bluetooth Low Energy?



Multi-field anomalies

CODENOMICON

- Anomalize multiple fields at once
- Downside: Root cause analysis more difficult

OBEX OPP Connect (VALID)

```
HEADER
Code       : 0x80      [Connect]
Length     : 0x0022

FRAME_DATA
OBEX Version : 0x01
SupportIrLMP : 0x00
Max Packet Length : 0x0400

HEADERS
Tag        : 0x01      [NAME Header]
Length     : 0x001B
Value      : 0x0043006f0064 [Null terminated UTF16BE]
            0x0065006e006f
            0x006d00690063
            0x006e0000
```

OBEX OPP Connect (INVALID)

```
HEADER
Code       : 0x80      [Connect]
Length     : 0x0022

FRAME_DATA
OBEX Version : 0x01
SupportIrLMP : 0x00
Max Packet Length : 0x0400

HEADERS
Tag        : 0x00      [Unknown Header Tag]
Length     : 0x001B
Value      : 0x000000000000
            0x000000000000
            0x000000000000
            0x00000000
```



Sequence anomalies

CODENOMICON

- Rearrange or repeat messages within the sequence
- The messages themselves are completely valid - only in wrong place
- More efficient against profiles with complex sequences
- Combine with anomalized messages for additional fun



Multiprofile

- Use combination of profiles to open attack vector
- Useful against client roles!
 - SDP client
 - AVRCP to trigger A2DP
 - HFP to trigger PBAP



CODENOMICON

WHY FUZZING WORKS



CODENOMICON

Why stuff breaks?

- “You are not supposed to do that”
 - Surprisingly common response when anomaly is explained.
 - Lack of negative testing and education.



CODENOMICON

Why stuff breaks?

- “So what?”
 - Why should anyone care if the 5€ headset breaks?
 - What if attitude is the same when implementing for example medical devices?



CODENOMICON

Why stuff breaks?

- It is not easy!
- Writing code which is robust is hard, even if you know what you are doing
- Combine this with ambiguous specifications and tight schedules, it is surprising that things work even now and then



CODENOMICON

SECURITY MEASURES



Bluetooth security measures

Pairing

- Most common way to protect
- Legacy pairing traditionally uses 4-digit code, Simple Secure Pairing has more advanced authentication mechanisms
- L2CAP connection to PSM 1 (SDP) rarely requires pairing



Evading pairing

CODENOMICON

- Social engineering.
- SSP downgraded to legacy pairing
- In SSP JustWorks pairing does not require any authentication.



Evading pairing

CODENOMICON

- Known PIN codes: 0000,1234;
- After serving some anomalies to L2CAP, we've seen devices to stop requesting pairing for any services.



CODENOMICON

Other security measures

- Non-discoverable devices
 - May still accept connections
- Dedicated 'pairing' mode
- Use SDP query to enable services
 - Requires SDP server on "attacker"



Other security measures: downsides

- Complex sequence needed for service activation.
 - Increases the number of errors in legitimate use and make it more complex.
- You need to make sure all profiles use the same measures



Hiding the problem

CODENOMICON

- Focus on preventing unauthorized access.
- The underlying implementations are still crap
- Instead of developing more complex security measures, focus more on the robustness of the implementation.



CODENOMICON

Fault propagation

- Anomalies can propagate to interconnected systems.
- Unprotected backend systems
 - Authentication servers etc.
- You could apply the same to Bluetooth-enabled carkits, payment terminals etc.



CODENOMICON

Not just about security

- There doesn't have to be evil hax0rs breaking your phone.
 - 5€ headset sending invalid data does the job
- Bluetooth applications are becoming more critical!
 - Stacks used in medical devices for example must be more robust
- We have seen headsets that die after one anomalous package and never recover



CODENOMICON

THANK YOU – QUESTIONS?