PX Enterprise Pentesting Appliance (PX-EPA) User Manual

Note: The online version of this manual is maintained here:
http://www.pwnieexpress.com/support.html

Table of Contents:

Introduction

Legal stuff

1. All Pwnie Express / Rapid Focus Security products are for legally authorized uses only.

1. By using this product you agree to the terms of the Rapid Focus Security EULA: (http://pwnieexpress.com/pdfs/RFSEULA.pdf)

1. As with any software application, any downloads/transfers of this software are subject to export controls under the U.S. Commerce Department's Export Administration Regulations (EAR). By using this software you certify your complete understanding of and compliance with these regulations.

1. This product contains both open source and proprietary software. Proprietary software is distributed under the terms of the Rapid Focus Security EULA: (http://pwnieexpress.com/pdfs/RFSEULA.pdf).

1. The following software applications are governed by the bsd-3-clause license (http://www.opensource.org/licenses/bsd-3-clause)

   autossh
   darkstat
   dsniff
   wepbuster
   wpa_supplicant
   metasploit

1. The openssl/ssleay software applications are governed by the openssl toolkit dual license (http://www.openssl.org/source/license.html)
2. All other open source software is governed by the GNU General Public License: (http://www.gnu.org/licenses/gpl.html)

Appliance Features

1. Enterprise-class, wall-mountable, small form-factor enclosure
2. Supports Nessus server, Metasploit Express/Pro, and Cobalt Strike
3. Supports Backtrack, Qualys, Acunetix, nCircle, etc. as virtual guest machines
4. Hardened per NSA, NIST, DoD, and DISA guidelines, including encrypted volumes for pentest results
5. Simple web-based administration with "Pwnix UI" (http://pwnieexpress.com/pages/our-tech)
6. Includes most Pwn Plug features (http://pwnieexpress.com/pages/pwn-plug-software-release-1-1)
7. One-click Evil AP, stealth mode, passive recon, and history wipe
8. Fully-automated NAC/802.1x/RADIUS bypass (http://pwnieexpress.com/pages/our-tech)
9. Out-of-band SSH access over 4G/GSM cell networks

10. Maintains persistent, covert, encrypted SSH access to your target network (http://pwnieexpress.com/pages/our-tech)
11. Tunnels through application-aware firewalls & IPS
12. Supports HTTP proxies, SSH-VPN, & OpenVPN
13. Sends email/SMS alerts when SSH tunnels are activated
14. Preloaded with Ubuntu Server 12.04 LTS, Metasploit, SET, Fast-Track, w3af, Kismet, Aircrack, SSLstrip, nmap, Hydra, dsniff, Scapy, Ettercap, Bluetooth/VoIP/IPv6 tools, & many more (http://pwnieexpress.com/pages/our-tech)
15. Unpingable and no listening ports in stealth mode

Base hardware specs

1. Intel dual-core i5 processor with HT (4 cores total) at 2.66 GHz
2. 8GB DDR3 memory
3. 60GB internal solid-state storage
4. Onboard high-gain 802.11 a/b/g/n wireless supporting packet injection & monitor mode
5. Onboard high-gain Bluetooth (up to 1000' range) supporting packet injection & monitor mode
6. Onboard 6-band (worldwide) 4G GSM cellular data
7. Optional support for Zigbee/Zwave, RFID, and Software-Defined Radios (SDR)
8. Optional physical tamper detection using an internal 6-axis accelerometer

Getting started

1. Connect the provided wireless antenna to the SMA jack on the rear of the appliance.
2. Connect the appliance to a power source
3. Connect the onboard Ethernet jack to a local network or switch
4. Power up the appliance (oval button on front-panel)
5. The default appliance IP address is 192.168.9.10 (netmask 255.255.255.0).
6. To access the appliance for the first time, configure your Linux/Mac/Windows system with the following IP settings:

   IP address: 192.168.9.11
   Netmask: 255.255.255.0

   Tip: On Linux hosts you can configure a virtual interface as shown:

   ```
   # ifconfig eth0:1 192.168.9.11/24
   ```

7. Confirm connectivity to the appliance by pinging it:

   ```
   $ ping 192.168.9.10
   ```

8. You can now access the appliance through the Pwnix UI: proceed to “Accessing the Pwnix UI” below.

Tip: You can also now connect to the appliance via SSH as show below:

```
$ ssh pwnie@192.168.9.10
```

Using the Pwnix UI

Accessing Pwnix UI

1. Open a web browser and access the UI: https://*[appliance_ip_address]*:8443

   Tip: If accessing for the first time, the default URL is https://192.168.9.10:8443

2. The UI is SSL-enabled, but you will receive a warning as the certificate is self-signed.
3. You are then prompted for login/password (default is **pwnie : pwnplug8000**)
4. The "Setup" page appears.

   Important: Please change the default Pwnix password as soon as possible! Proceed to "Change Pwnix Password" below.

Setup tab

Change Pwnix Password

1. Click "Setup" on the top menu.
2. Click "User Interface"
3. Enter a new password for the "pwnie" user into both fields and click "Update password".

   Note: This will change the password for the 'pwnie' UI user and the 'pwnie' system (Linux/SSH) account. Pwnix UI authentication is integrated with Linux PAM, allowing the UI and system passwords to be synced for the "pwnie" user.

4. The Pwnix UI will restart then prompt you for the new credentials.

   Tip: You can also set the "pwnie" user's password via the command line, as shown:

```
# passwd pwnie
```

Reverse Shell Key

1. Click “Setup” on the top menu.
2. Click “Reverse Shell Key”
3. This section shows the current root user SSH key used to establish the reverse shells.
4. (Optional) To generate a new key pair for the reverse shells, click “Generate”.

   Tip: If a key pair doesn’t already exist, a new one will be automated generated after enabling one or more reverse shells on the “Reverse Shells” page.

Network Config

1. Click “Setup” on the top menu.
2. Click “Network Config”.
3. The current network settings for the appliance’s onboard Ethernet interfaces are displayed under “Current Network Settings”.

   By default, the appliance ships with the following interfaces:

   1. eth0 - The onboard Ethernet interface, configured for DHCP
   2. eth0:1 - The virtual default interface for initial access, set to 192.168.9.10/24
   3. wlan0 - The onboard 802.11a/b/n/g wireless adapter (DOWN by default)
   4. virbr0 - The default bridging interface for virtual machines

4. To change the appliance’s host name, enter a new hostname and click “Change hostname”.

   Tip: After changing the hostname, log out of any active terminal sessions to update your terminal prompt.

5. To change the IP configuration for eth0, click the “eth0” link.

   1. To set a static IP for eth0, enter a new IP address, network mask, default gateway, and primary DNS server and click “Apply static IP settings”.

      Note: After the appliance’s IP address is changed, reconnect to the UI using the newly assigned IP address.

   1. To acquire the eth0 network settings from a DHCP server instead (recommended), click “Switch to DHCP”.

      Note: After switching to DHCP, you’ll need to access the appliance via the virtual default interface (192.168.9.10) or via local keyboard/monitor to determine the new IP address assigned by DHCP. Once the new DHCP-assigned IP address is known, reconnect to the UI using the newly assigned IP address.

   Note: The virtual default interface (192.168.9.10) can be shut down by running the following command:

```
# ifdown eth0:1
```

Clear History & Logs

1. Click “Setup” on the top menu.
2. Under “Clean up Pwnix History and Logs”, click the “Cleanup now” button.
3. This clears the root user’s bash history, UI logs, and all logs in /var/log.

   Note: The bash history for any currently active root user sessions will be cleared at next logout.

   Tip: The cleanup script can also be invoked from the command line as follows:

   ```
   # /opt/pwnix/pwnix-scripts/cleanup.sh
   ```

Appliance Updates

1. Click “Setup” on the top menu.
2. Under "Update Device", click the "Update Now" button.

   Note: The appliance must have Internet access via ports 80 and 443 for the update to succeed.

3. The latest stable Pwnix release is downloaded and applied (typically 3-5 minutes). You will be redirected to the update log.
4. The Pwnix version can be viewed under the “System Details” tab
5. The Pwnix Update log can be view under the “System Details” tab

Appliance Reboot

1. Click “Setup” on the top menu.
2. Under "Restart Device", click the "Reboot Now" button.
3. The device will reboot immediately.

Services tab

Metasploit RPC

1. Click “Services” on the top menu.
2. Click “Metasploit RPC”, then “Enable Metasploit RPC”.
3. When enabled the, Metasploit RPC service can be accessed through the RPC listener

Tip: The Metasploit RPC service can also be enabled/disabled from the command line as follows:

To enable:
```
# service pwnix_msfrpcd start
# update-rc.d pwnix_msfrpcd defaults
```

To disable:
```
# service pwnix_msfrpcd stop
# update-rc.d -f pwnix_msfrpcd remove
```

Tip: The Metasploit RPC service can configured in /opt/pwnix/pwnix-config/services/msfrpcd.conf

LISTEN - This option configures the listen address. Defaults to 127.0.0.1
USERNAME - This option configures the username. Defaults to pwnie.
PASSWORD - This option configures the msfrpcd password. Defaults to a random string.

Passive Recon

1. Click "Services" on the top menu.
2. Click "Passive Recon".
3. Click "Enable" to start the passive recon service.
4. While enabled, the Enterprise Appliance will passively listen on eth0, recording HTTP requests, user-agents, cookies, OS guesses, and clear-text passwords to the following logs:

1. **HTTP requests:** /var/log/pwnix/passive_recon/http.log
2. **OS guesses:** /var/log/pwnix/passive_recon/p0f.log
3. **Clear-text passwords:** /var/log/pwnix/passive_recon/dsniff.log

Tip: Passive Recon is most effective when the Enterprise Appliance is in NAC Bypass / transparent bridging mode, or when connected to a switch monitor port or network tap.

Tip: The Passive recon service can also be enabled/disabled from the command line as follows:

To enable:
```
# service pwnix_passive_recon start
# update-rc.d pwnix_passive_recon defaults
```

To disable:
```
# service pwnix_passive_recon stop
# update-rc.d -f pwnix_passive_recon remove
```

Evil AP

1. Ensure the wireless antenna is connected to the appliance

2. Click "Services" on the top menu.
3. Click "Evil AP".
4. Enter an SSID for your Evil AP, then click "Start Evil AP".
5. Wireless clients will begin connecting to the AP, either automatically via preferred network lists or by direct AP association.

   Tip: To view realtime Evil AP activity from the command line:

   ```
   # tail -f /var/log/pwnix/evilap.log
   ```

6. By default the device will function as a standard AP, transparently routing all client Internet requests through the wired interface (eth0).

   Tip: The Evil AP service can also be enabled/disabled from the command line as follows:

   To enable:
   ```
   # service pwnix_evil_ap start
   # update-rc.d pwnix_evil_ap defaults
   ```

   To disable:
   ```
   # service pwnix_evil_ap stop
   # update-rc.d -f pwnix_evil_ap remove
   ```

   Tip: The Metasploit RPC service can configured in /opt/pwnix/pwnix-config/services/evil_ap.conf

Reverse Shells tab

See section "Using the reverse shells" for details on this feature.

System Details tab

This section displays the appliance's software release level,system logs, disk usage, etc.

Using the reverse shells

Reverse shell overview

1. All Pwnix devices include aggressive reverse tunneling capabilities for persistent remote SSH access.

2. SSH over HTTP, DNS, ICMP, and other covert tunneling options are available for traversing strict firewall rules, web filters, & application-aware IPS.
3. All tunnels are encrypted via SSH and will maintain access wherever the device has an Internet connection - including wired, wireless, and 3G/GSM where available.

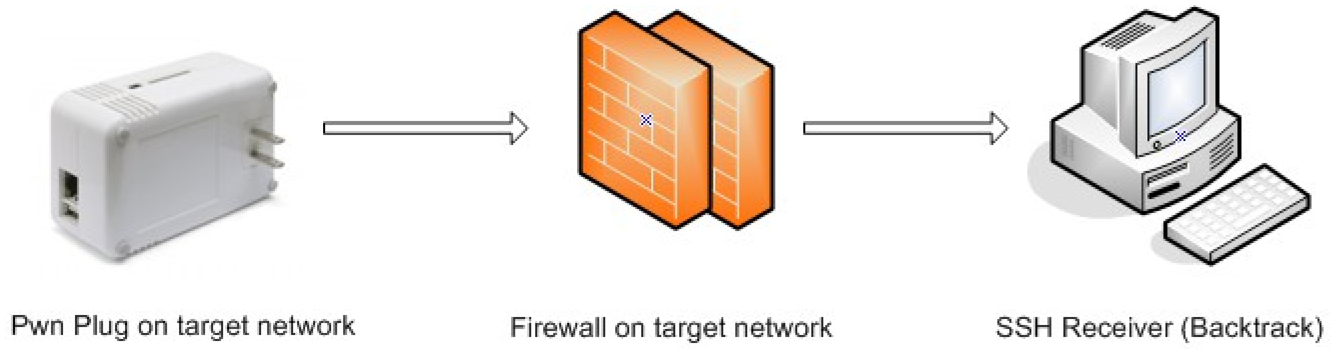
Typical deployment scenario

1. On your staging/lab network, enable the desired reverse shells (see "Activating the reverse shells")
2. Select the platform you'd like to use to receive the reverse shells (the "shell receiver"):

1. **Option 1 (recommended):** The Pwnix Shell Receiver virtual machine. This is a locked-down Ubuntu 12.04 system preconfigured to receive all reverse shells (see "Configuring the Pwnix Shell Receiver VM to receive the reverse shells")

1. **Option 2:** Use Backtrack 5 to receive the reverse shells ("Configuring Backtrack to receive the reverse shells")

3. Test the reverse shells in a lab / local LAN to confirm all shells are working as expected (see "Connecting to the reverse shells")
4. [Optional] Enable Stealth Mode (see "Maintaining the Appliance -> Stealth Mode")
5. Deploy the Enterprise Appliance to your target network and watch your SSH receiver for incoming shells (see "Deploying to target network")



Activating the reverse shells

1. Log into the Pwnix UI.
2. Click "Reverse Shells" on the top menu.
3. Use the checkboxes to indicate the reverse shells you'd like to enable.

   Tip: To best maintain persistent remote access, enable all of the reverse shells.

4. Enter the SSH shell receiver IP address or DNS name for each selected reverse shell. The appliance will connect to this shell receiver system to establish the reverse shell connections.

5. Choose how often each reverse shell connection should be attempted. By default, a shell connection will be attempted every minute (recommended).
6. To use an HTTP proxy for the "SSH over HTTP Tunnel", enable the "Use HTTP Proxy" checkbox and enter the proxy server address and port (and optionally, proxy server credentials).

   Note: The HTTP proxy auth password is stored in clear text in /opt/pwnix/pwnix-scripts/script_configs

7. Click "Configure all shells" at the bottom of the page to apply your changes.

   Note: The following SSH client config directives (/etc/ssh/ssh_config) are set on all devices to allow for automation of reverse shell connections. Be sure you understand the security implications of these settings before connecting to other SSH servers from the device.

   ```
   StrictHostKeyChecking no
   UserKnownHostsFile /dev/null
   ```

8. Proceed to configure your shell receiver.

Configuring the Pwnix Shell Receiver VM to receive the reverse shells

The Pwnix Shell Receiver virtual machine is the recommended SSH shell receiver. This is a locked-down Ubuntu 12.04 system pre-configured to receive all reverse shells. The Enterprise Appliance will connect to this system when initiating the reverse shell connections.

1. Start up the Pwnix Shell Receiver virtual machine if it's not already running.
2. From your local workstation, open a browser and connect to the Enterprise Appliance UI: https://*[appliance_ip_address]*:8443
3. Login to the UI when prompted.
4. Click "Reverse Shells" on the top menu.
5. Click the "Generate Pwnix Shell Reciever config" link at the top of the page (under step 5) to download the "pwnix_receiver.sh" script to your local workstation.
6. Copy the script file (pwnix_receiver.sh) to the Pwnix Shell Receiver virtual machine via scp:

   ```
   $ scp pwnix_receiver.sh pwnie@[Pwnix_Shell_Receiver_IP_address]:
   ```

7. Login to the Pwnix Shell Receiver virtual machine:

   ```
   $ ssh pwnie@[Pwnix_Shell_Receiver_IP_address]
   ```

8. Enter the following commands:

   ```
   $ chmod +x pwnix_receiver.sh

   $ sudo ./pwnix_receiver.sh
   ```

9. The script auto-configures and starts the reverse shell listeners on the Pwnix Shell Receiver.
10. If prompted, enter the desired certificate information for the stunnel SSL certificate (or just press ENTER to accept the defaults)
11. Once the auto-config script completes you will see:

    ```
    [+] Setup Complete.
    [+] Press ENTER to listen for incoming connections..
    ```

12. Press ENTER to watch for incoming Enterprise Appliance connections. Each reverse shell will attempt to connect using the interval you specified in the UI.

    Tip: You can list all active device connections at any time by typing:

    ```
    $ sudo netstat -lntup4 | grep 333
    ```

13. Proceed to "Connecting to the reverse shells".

Configuring Backtrack to receive the reverse shells (optional)

A Backtrack 5 system can also serve as the SSH tunnel "receiver". The Enterprise Appliance will connect to this system when initiating the reverse shell connections.

Note: These steps assume you're using Backtrack 5 as your SSH receiver. Older Backtrack distributions may be used, but different steps may apply.

1. Place the Enterprise Appliance and the Backtrack system on the same local network/subnet
2. Login to the Backtrack system and open Firefox
3. Connect to the UI: https://*[appliance_ip_address]*:8443
4. Login to the UI when prompted.
5. Click "Reverse Shells" on the top menu.
6. Click the "Generate Backtrack config" link at the top of the page (under step 5) to download the "backtrack_receiver.sh" script.
7. Save the script file (backtrack_receiver.sh) into the root user's home directory (selected by default)
8. Open a terminal window and enter the following commands:

    ```
    # cd

    # chmod +x backtrack_receiver.sh

    # ./backtrack_receiver.sh
    ```

9. The script auto-configures and starts the reverse shell listeners on Backtrack.
10. When prompted, enter the desired certificate information for the stunnel SSL certificate (or just press ENTER to accept the defaults)
11. Once the auto-config script completes you will see:

```
[+] Setup Complete.
[+] Press ENTER to listen for incoming connections..
```

12. Press ENTER to watch for incoming Enterprise Appliance connections. Each reverse shell will attempt to connect using the interval you specified in the UI.

    Tip: You can list all active device connections at any time by typing:

    ```
    # netstat -lntup4 | grep 333
    ```

13. Proceed to “Connecting to the reverse shells”.

Connecting to the reverse shells

1. Open a terminal window on your shell receiver system and connect to any available “listening” Enterprise Appliance shell as follows:

    1. **Standard SSH:** ssh pwnie@localhost -p 3333
    2. **SSH Egress Buster:** ssh pwnie@localhost -p 3334
    3. **SSH over DNS:** ssh pwnie@localhost -p 3335
    4. **SSH over SSL:** ssh pwnie@localhost -p 3336
    5. **SSH over 4G/GSM:** ssh pwnie@localhost -p 3337
    6. **SSH over HTTP:** ssh pwnie@localhost -p 3338
    7. **SSH over ICMP:** ssh pwnie@localhost -p 3339

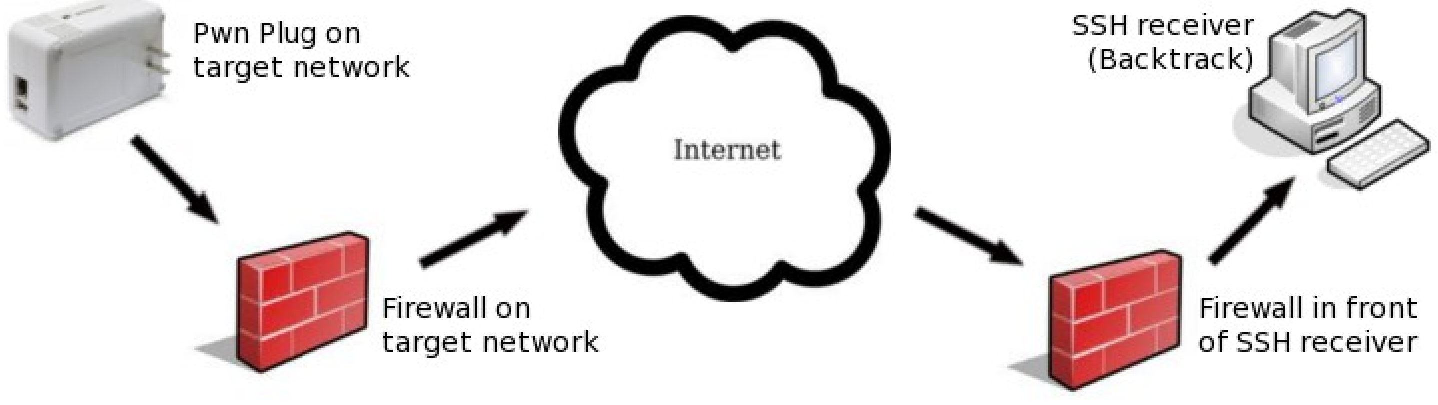
1. Enter your Enterprise Appliance pwnie SSH user password and voila! You're now remotely connected to the appliance through the reverse shell.
2. Proceed to “Deploying to target network”

    Standard SSH / SSH Egress Buster Note: If there’s no firewall between the Enterprise Appliance and your shell receiver system, be sure the shell receiver system SSH server is listening on the ports you selected for the Standard Reverse SSH and SSH Egress Buster shells in the UI. For example, if you set port 31337 for Standard Reverse SSH, add the line “Port 31337” to /etc/ssh/sshd_config, then restart SSHd (/etc/init.d/ssh restart).

    Tip: The SSH receiver address can be anonymized using the “Tor Hidden Service” feature as described here http://www.securitygeneration.com/security/reverse-ssh-over-tor-on-the-pwnie-express/

    Special thanks to Sebastien J. of Security Generation for streamlining the SSH receiver setup process, and to Lance Honer for his resilient autossh script improvements.

Deploying to target network


Pwn Plug on target network
Firewall on target network
Internet
SSH receiver (Backtrack)
Firewall in front of SSH receiver

1. Place your shell receiver system behind a public-facing firewall.
2. Configure the appropriate port forwarders on your firewall:

    1. **Standard Reverse SSH:**
    Forward the port selected in the UI to port 22 of your shell receiver.

    2. **SSH over HTTP:**
    Forward port 80 to port 80 of your shell receiver system

    3. **SSH over SSL:**
    Forward port 443 to port 443 of your shell receiver system

    4. **SSH over DNS:**
    Forward <u>UDP</u> port 53 to <u>UDP</u> port 53 of your shell receiver system

    5. **SSH over ICMP:**
    Requires your shell receiver system to be directly connected to the Internet (no firewall).

    6. **SSH over 3G:**
    Forward the port selected in the UI to port 22 of your shell receiver system

    7. **SSH Egress Buster:**
    Forward all ports selected in the UI to port 22 of your shell receiver system

1. In the Pwnix UI ("Reverse Shells" page), configure the reverse shells to connect to your firewall's public IP address (or DNS name if available).
2. (Optional) Enable Stealth Mode
3. You can now deploy the Enterprise Appliance to your target network. The Enterprise Appliance will automatically "phone home" to your shell receiver system, providing encrypted remote access to your target network.

    Tip: In some environments you may wish to schedule a nightly reboot of the device to re-initiate all connections from the device side. This way, if some part of the connection process crashes on the device side (for example, sshd), the connection process will start "fresh" again after the reboot.

Using SSH port forwarders on Backtrack

Example 1: Connecting to remote RDP servers

1. On Backtrack:

   ```
   # ssh pwnie@localhost -p XXXX -NL 3389:xxx.xxx.xxx.xxx:3389
   ```

   .. where "*XXXX*" is the local listening port of an active reverse shell (such as 3333 for standard reverse SSH), and where "*xxx.xxx.xxx.xxx*" is the IP address of an RDP target system on the remote network the Enterprise Appliance is physically connected to.

2. Login to the Enterprise Appliance when prompted.
3. Connect to the remote RDP server through the SSH tunnel by using "localhost":

   ```
   # rdesktop localhost
   ```

Example 2: Connecting to remote web servers

1. On Backtrack:

   ```
   # ssh pwnie@localhost -p XXXX -ND 8080
   ```

   .. where "*XXXX*" is the local listening port of an active reverse shell (such as 3333 for standard reverse SSH).

2. Login to the Enterprise Appliance when prompted.
3. Open Firefox and configure it to use localhost as an HTTP proxy on port 8080.
4. You can now connect to any web server on the remote network by entering the IP address or URL into Firefox.

Creating an SSH VPN

The OpenSSH server on the Enterprise Appliance supports SSH-based VPN tunnelling through any active reverse shell, allowing transparent (albeit slow) access to your target network from your Backtrack machine. This is mainly useful when the need arises for a GUI-based or third-party pentesting tool, such as BurpSuite, Nessus, Remote Desktop client, etc.

Sample environment

The steps below assumes the following IP addresses/ranges. Substitute the addresses/ranges for your target and local (Backtrack) networks where appropriate.

1. Target network (where the Enterprise Appliance is deployed): 172.16.1.0/24
2. Local network (where Backtrack SSH receiver is located): 192.168.1.0/24
3. VPN network: 10.1.1.0/30
4. Backtrack VPN address (tun0 interface): 10.1.1.1
5. Enterprise Appliance VPN address (tun0 interface): 10.1.1.2
6. Assumes a reverse shell is currently established and listening on localhost:3333 (Standard Reverse SSH). Any active reverse shell can be used to carry the VPN tunnel (change "3333" where appropriate).

Activating the SSH VPN tunnel

1. On Backtrack (VPN client):

   ```
   # ssh -f -w 0:0 localhost -p 3333 true
   ```

   (Login to the Enterprise Appliance as root when prompted)

   ```
   # ifconfig tun0 10.1.1.1 10.1.1.2 netmask 255.255.255.252
   # route add -net 172.16.1.0/24 gw 10.1.1.2
   ```

2. On the Enterprise Appliance (VPN server):

   ```
   # /opt/pwnix/pwnix-scripts/Enable_SSH_VPN.sh
   ```

3. The SSH VPN tunnel should now be active.
4. On Backtrack, test connectivity to target network through the VPN tunnel:

   ```
   # ping 10.1.1.2
   # ping 172.16.1.1 (or any remote machine on the target network)
   # nmap -sP 172.16.1.*
   ```

5. To disable the VPN tunnel on the Backtrack side:

   ```
   # ifconfig tun0 down
   ```

6. To disable the VPN tunnel on the Enterprise Appliance side:

   ```
   # /opt/pwnix/pwnix-scripts/Disable_SSH_VPN.sh
   ```

Using the wireless hardware

802.11 wireless

Connecting to an open wifi network

1. Set the wireless interface to managed mode:

   ```
   # iwconfig wlan0 mode managed
   ```

2. Bring up the interface:

   ```
   # ifconfig wlan0 up
   ```

3. Scan for access points in the area:

   ```
   # iwlist scan
   ```

4. Associate with an access point with SSID “example” on channel 6:

   ```
   # iwconfig wlan0 essid “example”
   # iwconfig wlan0 channel 6
   ```

5. Restart the interface:

   ```
   # ifconfig wlan0 down
   # ifconfig wlan0 up
   ```

6. Acquire a DHCP address:

   ```
   # dhclient wlan0
   ```

Running Airodump-ng & Kismet

1. Bring down the interface:

   ```
   # ifconfig wlan0 down
   ```

2. To launch airodump-ng:

   ```
   # airodump-ng wlan0
   ```

   Note: The output of airodump-ng can only be viewed within an SSH session (no via serial console).

3. When finished, press CTRL+C to exit
4. To launch Kismet:

   ```
   # kismet
   ```

5. Press ENTER 3 times, then TAB, then ENTER

6. When finished, press CTRL+C to exit

   Tip: Certain wireless tools may leave the wireless adapter in a mode that's not compatible with other wireless tools. It's generally recommended to set the interface to a "down" state before running most wireless tools:

   ```
   # ifconfig wlan0 down
   ```

Packet injection & WEP cracking

1. To run a simple packet injection test, execute the following commands. This example assumes a WEP-enabled access point on channel 6 with SSID "example" is within range of the appliance.

   ```
   # ifconfig wlan0 up
   # iwconfig wlan0 channel 6
   # ifconfig wlan0 down
   # aireplay-ng -e example --test wlan0
   ```

2. Look for the following output:

   ```
   17:19:45  Waiting for beacon frame (ESSID: example) on channel 6
   Found BSSID "00:13:10:9E:52:3D" to given ESSID "example".
   17:19:45  Trying broadcast probe requests...
   17:19:45  Injection is working!
   17:19:46  Found 1 AP
   ```

3. To auto-crack all WEP-enabled access points on channel 6 using wepbuster:

   ```
   # ifconfig wlan0 down
   # wepbuster 6
   ```

   Tip: WEP cracking performance is very dependant on the amount of wireless client traffic being generated on the target wifi network. The more traffic on the wireless network, the faster the cracking process.

Wireless client de-authentication

1. This example assumes the target access point is on channel 6:

   ```
   # iwconfig wlan0 channel 6
   ```

2. In one terminal, start airodump-ng:

   ```
   # airodump-ng --bssid [MAC of target AP] -c 6 wlan0
   ```

3. Then, in a second terminal, start the client de-authentication:

```
# aireplay-ng -0 0 -a [MAC of target AP] -c [MAC of target client]
wlan0
```

Karmetasploit

Once an Evil AP is running, Karmetasploit can be invoked as follows.

1. CD to the Metasploit directory:

   ```
   # cd /opt/metasploit/msf3/
   ```

2. Confirm the following variables in karma.rc:

   ```
   setg AUTOPWN_HOST 192.168.7.1
   set LHOST 192.168.7.1

   use auxiliary/server/capture/http
   set SRVPORT 9443
   set SSL true
   run
   ```

1. Start Metasploit with the karma script:

   ```
   # ./msfconsole -r karma.rc
   ```

   Note: The module loading is CPU intensive and can take 1+ minutes to complete.

   Tip: To redirect all DNS queries to the local Metasploit FakeDNS listener:
   iptables -t nat -A PREROUTING -p udp --destination-port 53 -j REDIRECT --to-port 53

Bluetooth

Special thanks to JP Ronin (hackfromacave.com) for getting all of this working for us!

Using the Bluetooth adapter

1. Confirm the output of the following commands:

   ```
   # lsusb
   Bus 001 Device 002: ID 0a12:0001 Cambridge Silicon Radio, Ltd Bluetooth Dongle (HCI mode)
   ```

   ```
   # hciconfig
   hci0:   Type: BR/EDR  Bus: USB
           BD Address: XX:XX:XX:XX:XX:XX  ACL MTU: 310:10  SCO MTU: 64:8
   ```

```
DOWN
RX bytes:466 acl:0 sco:0 events:18 errors:0
TX bytes:73 acl:0 sco:0 commands:17 errors:0
```

1. Enable the Bluetooth interface and set it to “Non-Discoverable”:

   ```
   # hciconfig hci0 up
   # hciconfig hci0 noscan
   ```

2. To scan for local Bluetooth devices

   ```
   # hcitool -i hci0 scan --flush --info --class
   ```

1. To ping the address of a local Bluetooth device

   ```
   # l2ping -i hci0 XX:XX:XX:XX:XX:XX
   ```

2. To dump Bluetooth packets:

   ```
   # hcidump -i hci0 -t -X
   ```

3. To pair with a local Bluetooth device:

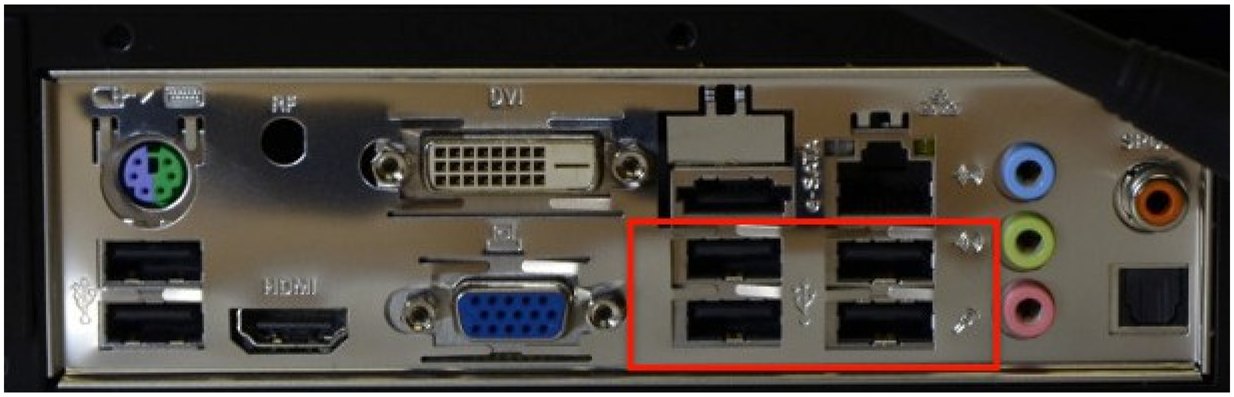
   ```
   # bluez-simple-agent hci0 XX:XX:XX:XX:XX:XX
   ```

Accessing additional Bluetooth tools

```
# l2test
# bss
# bluebugger
# bluelog -h
# psm_scan
# rfcomm_scan
# carwhisperer
# redfang -h
# ussp-push
# sobexsrv -h
```

4G/GSM cellular

IMPORTANT: Make sure to place the cellular adapter in the back group of 4 USB ports (when looking at the device from the back). The other USB ports may not provide enough power for the 4G card. :


Place the 4G adapter in these ports.

Using the unlocked GSM adapter

The unlocked GSM adapter supports five GSM cell bands (HSDPA / GSM / UMTS / EDGE / GPRS) and is compatible with AT&T, Vodafone, Orange, and GSM carriers in over 160 countries.

GSM carriers in the Americas:
http://en.wikipedia.org/wiki/List_of_mobile_network_operators_of_the_Americas

GSM carriers in Europe:
http://en.wikipedia.org/wiki/List_of_mobile_network_operators_of_Europe

Note: Verizon, Sprint, Virgin Mobile, and other CDMA carrier SIMs will not work in the unlocked GSM adapter.

1. First, obtain a SIM card from the GSM cell provider of your choice. In the US, SIM cards from AT&T devices (including iPhones) are supported.

   Note: The mobile service attached to the SIM card must have mobile broadband data service. Verify you can access the Internet from your phone using the SIM card before proceeding.

2. Slide open the the plastic cover on the GSM adapter.
3. Insert your SIM card into the adapter with the notch positioned as shown by the line drawing on the SIM slot, with the SIM card contacts facing down.

   Note: Some GSM phones, including the iPhone4, use a micro-SIM instead of a standard-sized SIM card. To fit these SIM cards into the GSM adapter, use the included micro-SIM card adapter.

4. Slide the plastic cover back onto the adapter.
5. Connect the GSM adapter to the appliance's USB port.
6. Confirm the GSM adapter is detected properly (note adapter detection may take 15-20 seconds):

```
# lsusb
Bus 001 Device 003: ID 12d1:1436 Huawei Technologies Co., Ltd.
```

1. To query the GSM modem for adapter details:

```
# gsmctl -d /dev/ttyUSB0 me
```

Note: If the command returns “SIM failure”, the SIM card is either missing or not inserted properly.

Tip: If the modem does not respond on /dev/ttyUSB0 after 10 seconds,
try /dev/ttyUSB1, /dev/ttyUSB2, or /dev/ttyUSB3

2. To list cellular operators in range:

```
# gsmctl -d /dev/ttyUSB0 op
```

3. To show currently attached operator:

```
# gsmctl -d /dev/ttyUSB0 currop
```

4. To show signal strength of current operator connection:

```
# gsmctl -d /dev/ttyUSB0 sig
```

5. To check PIN status (READY = No PIN set):

```
# gsmctl -d /dev/ttyUSB0 pin
```

6. To send a text message:

```
# gsmsendsms -d /dev/ttyUSB0 [destination 11-digit cell number]
"Test"
```

7. To make an outbound phone call:

```
# gsmctl -d /dev/ttyUSB0 -o dial [11-digit phone number]
```

Activating the Virgin Mobile / Verizon adapters

1. The Virgin Mobile and Verizon CDMA adapters must be activated with a mobile broadband plan before they can connect the the Internet. This one-time activation must be completed on Windows.

   Virgin Mobile mobile broadband plans:
   http://www.virginmobileusa.com/mobile-broadband/

   Verizon prepaid mobile broadband plans:
   http://www.verizonwireless.com/b2c/mobilebroadband/?page=products_prepaidmb

1. Insert the adapter into a Windows computer (XP recommended). The adapter will load a virtual CD-ROM device; open this device through “My Computer” and launch the Broadband2Go (Virgin Mobile) or VZaccess (Verizon) installer.

2. Once the installer completes, launch Broadband2Go (Virgin Mobile) or VZaccess Manager (Verizon) and complete USB device detection.
3. Verify the USB adapter is detected and a 1x data signal is available, then click “Connect”.
4. You will be prompted to activate the device and sign up for new service. Complete the activation process by following the prompts.
5. Once activated, confirm you are able to access the Internet using the 3G adapter on Windows.
6. Connect the adapter to the Enterprise Appliance’s USB port and wait 30 seconds for the adapter driver to load.

Connecting to the Internet via 3G

1. Call the appropriate pppd dialup script:

   For the unlocked GSM adapter:

   ```
   # pppd nodetach call e160 &
   ```

   For the Verizon / Virgin Mobile adapters:

   ```
   # pppd nodetach call 1xevdo &
   ```

   For the T-mobile Rocket 4G adapter:

   ```
   # pppd nodetach call tmobile &
   ```

2. Assuming a 3G cellular data signal is available, the adapter will establish an Internet connection within 10-20 seconds. Once connected, you will see a solid LED on the top of the adapter.
3. [Optional] Reset the default route to use the 3G interface (ppp0):

   ```
   # route del default
   # route add default ppp0
   ```

4. Test 3G Internet connectivity:

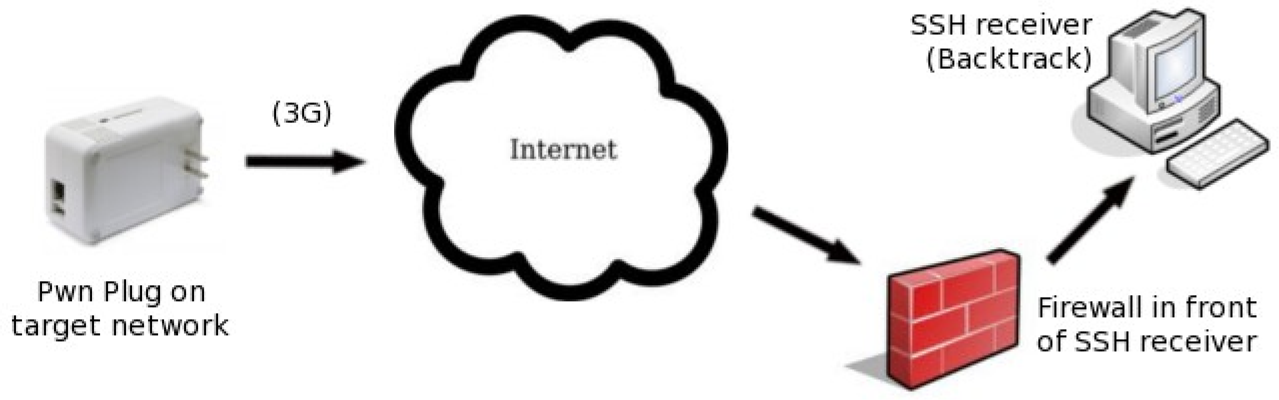
   ```
   # ping google.com
   # traceroute google.com
   ```

5. To close the 3G connection and restore Internet connectivity on eth0:

   ```
   # killall -s SIGHUP pppd
   # ifdown eth0 && ifup eth0
   ```

Using the SSH-over-3G shell

The SSH-over-3G reverse shell provides secure, out-of-band access to your Enterprise Appliance wherever a 3G cellular data signal is available. While this bypasses your target network's perimeter, a reverse shell is still recommended; many cell carriers do not assign public IP addresses to 3G data access devices.



1. If you haven't done so already, complete the reverse shell setup steps (see "Activating the reverse shells" and "Configuring the SSH receiver")
2. In the "Reverse Shells" page in UI, enable the "SSH over 3G/GSM" shell.
3. Configure the shell to connect to your firewall's public IP address (or DNS name if available).
4. Enter the destination port you'd like the Enterprise Appliance to use for the SSH connection.
5. Select your 3G adapter from the drop-down list.
6. Click the "Configure all shells" button.
7. Configure your firewall to forward the port selected in the UI to port 22 on your Backtrack machine.
8. On the Backtrack SSH receiver, watch for the inbound SSH-over-3G connection:

```
# watch -d "netstat -lntup4 | grep 3337"
```

9. Once the connection appears, connect to the Enterprise Appliance as shown:

```
# ssh pwnie@localhost -p 3337
```

10. Enter your Enterprise Appliance pwnie SSH user password and voila! You're now remotely connected to the appliance through the reverse shell.. over 3G!

    Note: The 3G connection will be released and reconnected at the selected retry interval until a reverse SSH tunnel is established.

Zigbee

Using the Goodfet utilities

Travis Goodspeed's Zigbee utilities are located in /pentest/goodfet. Simply connect a compatible Zigbee hardware radio running the Goodfet firmware to the Enterprise Appliance via USB and use these utilities for Zigbee wireless auditing.

For more information, see: http://goodfet.sourceforge.net

Network penetration testing with the PX-EPA

Footprinting

Use whois to identify companies, domains, ips and other target information:

```
# whois <company name>
# whois <ip address>
```

Use traceroute to identify hosts and path information:

```
# traceroute <host>
```

Use dig to find DNS information:

```
# dig <DNS domain name>
```

Use Fierce to identify additional entities via DNS:

```
# cd /pentest/fierce
# ./fierce.pl -dns <domain> -wide -ztstop –wildcstop
```

Discovery & Scanning

Use arp-scan to identify nearby hosts:

```
# arp-scan --local-net
```

Use nmap to identify hosts and services:

```
# nmap -A <ip address or range>
```

Use the nmap scripts to identify additional information

```
# nmap -sC <ip address or range>
```

Pentesting tools on the PX-EPA

Configuring and Running Metasploit

Using Metasploit via msfconsole

The Metasploit binaries (msfconsole, msfcli, etc.) can be run from any directory. Simply type 'msfconsole' run the local Metasploit Console. The Metasploit binaries can be found in /opt/metasploit/msf3.

Using Metasploit via msfrpcd

The metasploit service can by editing the file /opt/pwnix/pwnix-config/services/msfprcd.conf

```
# nano /opt/pwnix/pwnix-config/services/msfrpcd.conf
```

The metasploit service can be started by running:

```
# service pwnix_msfrpc start
```

To use metasploit from a remote host, you will need a “front-end” application. We suggest using Backtrack for this. Log into Backtrack with your credentials, and run the following from the commandline:

```
# startx
# cd /opt/metasploit
# ./diagnostic_shell
# ./msf3/msfgui
```

Using the credentials provided on the https://[epa]:8443/services/msfrpcd page, or in /opt/pwnix/pwnix-config/services/msfprcd.conf, log into MsfGui.

Tip: You may have to open msfgui more than once to get it to prompt you for a server. By default it starts an msfrpcd instance on the local Backtrack system.

Running The Social Engineering Toolkit (SET)

1. To launch SET, type:

```
# cd /pentest/set && ./set
```

Running FastTrack

1. To launch Fasttrack, type:

```
# cd /pentest/fasttrack && ./fast-track.py -i
```

Note: Fasttrack’s autopwn is incompatible with Metasploit 3.7+ due to removal of sqlite support: http://dev.metasploit.com/redmine/issues/4399

Running the tools in /pentest

```
# perl /pentest/asp-auditor/asp-audit.pl
# perl /pentest/bed/bed.pl
# cd /pentest/cisco-auditing-tool && ./CAT
# perl /pentest/cisco-global-exploiter/cge.pl
# perl /pentest/cms-explorer/cms-explorer.pl
# python /pentest/darkmysqli/DarkMySQLi.py
# perl /pentest/dnsenum/dnsenum.pl
# /pentest/easy-creds/easy-creds.sh
# perl /pentest/fierce/fierce.pl
# python /pentest/fimap/fimap.py
# /pentest/goohost/goohost.sh
# python /pentest/grabber/grabber.py
# /pentest/lbd/lbd.sh
# python /pentest/metagoofil/metagoofil.py
# python /pentest/miranda/miranda.py -h
# python /pentest/plecost/plecost-0.2.2-9-beta.py
# python /pentest/sickfuzz/sickfuzz.py
# python /pentest/sipvicious/svmap.py -h
# perl /pentest/smtp-user-enum/smtp-user-enum.pl
# perl /pentest/snmpcheck/snmpcheck-1.8.pl
# perl /pentest/snmpenum/snmpenum.pl
# python /pentest/sqlbrute/sqlbrute.py
# perl /pentest/sqlninja/sqlninja
# cd /pentest/sslstrip && ./sslstrip.py
# python /pentest/theharvester/theHarvester.py
# python /pentest/ua-tester/UAtester.py
# cd /pentest/voiper && python fuzzer.py
# python /pentest/waffit/wafw00f.py
# cd /pentest/weevely && python weevely.py
# python /pentest/wifitap/wifitap.py -h
# python /pentest/wifite/wifite.py
# python /pentest/wifizoo/wifizoo.py
```

Running tools installed via aptitude

```
# arp-scan
# ettercap -h
# dsniff -h
# hping3 -h
# john
# nbtscan
# nc -h
# ftp -h
# telnet -h
# nikto -Help
# openssl
# scapy -h
# xprobe2 -h
# iodine
# openvpn
# cryptcat -h
# sipsak
# miredo -h
# sslsniff
# tcptraceroute
# netdiscover
# udptunnel -h
# dnstracer
# sslscan
# ipcalc
# socat -h
# onesixtyone
# tinyproxy -h
# dmitry
# ssldump -h
# fping -h
# gpsd -h
# darkstat
# arping
# sipcrack
# proxychains
# proxytunnel --help
# wapiti
# skipfish -h
# nmap
# hydra
# alive6
# amap6
# denial6
# detect-new-ip6
# dnsdict6
# dos-new-ip6
# exploit6
# fake_advertise6
# fake_dhcps6
# fake_dnsupdate6
# fake_mipv6
# fake_mld26
# fake_mld6
# fake_mldrouter6
# fake_router6
# flood_advertise6
# flood_dhcpc6
# flood_mld26
# flood_mld6
# flood_mldrouter6
# flood_router6
# flood_solicitate6
# fragmentation6
# fuzz_ip6
# implementation6
# kill_router6
# ndpexhaust6
# parasite6
# randicmp6
# redir6
# rsmurf6
# sendpees6
```

# sendpeesmp6
# smurf6
# thcping6
# toobig6
# trace6

Additional features

Installing Tenable Nessus Server

1. Download the "Ubuntu 12.04 (64 bits)" Nessus deb package from: http://www.tenable.com/products/nessus/nessus-download-agreement
2. Install the Nessus deb package and start the Nessus server:

```
# dpkg -i Nessus-5.0.1-ubuntu1110_amd64.deb

# service nessusd start
```

3. Connect to the Nessus server web UI: https://*[appliance_IP]*:8834
4. Proceed through the Nessus server setup wizard

Deploying virtual machines

Creating a Backtrack 5 VM

1. Download the Backtrack 5 ISO to /opt/pwnix/virtual-machines/
2. For this example, we'll assume the ISO file name is "BT5R1-GNOME-32.iso"
3. Run the following commands to create a new virtual machine that boots from the BT5 ISO:

```
# chmod 644 /opt/pwnix/virtual-machines/BT5R1-GNOME-32.iso

# virt-install -n bt5r1 -r 512 --disk path=/opt/pwnix/virtual-
machines/bt5.img,bus=virtio,size=6 -c /opt/pwnix/virtual-
machines/BT5R1-GNOME-32.iso --accelerate --network
network=default,model=virtio --connect=qemu:///system --vnc
--noautoconsole -v
```

4. View the current state of the new VM:

```
# virsh -c qemu:///system list
```

5. From a remote system with a GUI, install virt-viewer (for example, via apt-get or yum)
6. Remotely connect to the Backtrack VM GUI console:

   ```
   $ virt-viewer -c qemu+ssh://pwnie@[appliance_IP]/system bt5r1
   ```

   Note: Remote GUI connections require key-based SSH access to the appliance

Setting up an encrypted partition

1. Run the following script to create an encrypted partition at /opt/pwnix/crypt-store

   ```
   # /opt/pwnix/pwnix-scripts/encrypted_volume_setup.sh
   ```

2. Enter a desired passphrase for the encrypted partition
3. Answer "yes" when prompted (these prompts are expected when first encrypting a partition)
4. Copy a test file to the encrypted partition:

   ```
   # echo test > /opt/pwnix/crypt-store/test-file
   ```

5. Unmount the encrypted partition and very the contents of your test file is unreadable:

   ```
   # umount /opt/pwnix/crypt-store
   # cat /opt/pwnix/crypt-store/test-file
   ```

6. To re-mount the encrypted partition (enter your passphrase when prompted):

   ```
   # mount -t ecryptfs /opt/pwnix/crypt-store/ /opt/pwnix/crypt-store/
   ```

Using Stealth Mode

Important: Enabling stealth mode will prevent access direct access to the Enterprise Appliance's SSH server and UI. Once stealth mode is enabled and the device is rebooted, access to the device can only be obtained through a reverse shell or via locally-attached keyboard/monitor.

1. To enable stealth mode:

   ```
   # service pwnix_stealth start
   # update-rc.d pwnix_stealth defaults
   ```

2. While enabled, stealth mode does the following:

1. Disables IPv6 support (prevents noisy IPv6 broadcasting)

2. Disables ICMP replies (won't respond to ping requests)
3. Disables the UI (closes port 8443)
4. Sets the local SSH server to listen on the loopback address only (closes port 22 to the outside)
5. Still allows ALL reverse shells to function as expected

1. For additional stealthiness:

1. If using DHCP, kill the dhclient process (closes listening UDP port 68):

```
# killall dhclient
```

2. Randomize your MAC address:

```
# macchanger -r eth0
```

3. Disable ARP replies (careful! this may affect network connectivity):

```
# ifconfig eth0 -arp
```

1. To disable stealth mode:

```
# service pwnix_stealth stop
# update-rc.d -f pwnix_stealth remove
```

Adding unsupported software from the Ubuntu repositories

The EPA has been designed as a network appliance and packages and software not installed by a Pwnie Express update are not supported on the device. However, root access to the device is provided, and you can install additional software as needed during testing. To do so, you will need to enable the Ubuntu repositories. Copy the appropriate sources.list to the /etc/apt folder with the following command:

```
# cp /opt/pwnix/chef/cookbooks/pwnix-base-cookbook/files/default/config_files/sources.list.all-enabled /etc/apt/sources.list
# apt-get update
# apt-get install <software-package>
```

Note that you should copy the appropriate configuration file back when your installation is completed, or the Pwnix update process could break. To do so:

```
# cp /opt/pwnix/chef/cookbooks/pwnix-base-cookbook/files/default/config_files/sources.list.all-disabled /etc/apt/sources.list
# apt-get update
```

Reviewing the OS environment

1. Show Enterprise Appliance software revision:

```
# grep Release /etc/motd
```

1. Show kernel version:

```
# uname -r
```

1. Show date/time:

```
# date
```

1. Show filesystem disk usage (note your disk usage may vary):

```
# df -h
```

1. Show CPU details:

```
# cat /proc/cpuinfo
```

1. Show total memory:

```
# grep MemTotal /proc/meminfo
```

1. Show current eth0 config:

```
# ifconfig eth0
```

1. Show currently listening TCP/UDP services (note dhclient won’t be present if not using DHCP):

```
# netstat -lntup
```

1. Check syslog for errors, warnings, etc:

```
# egrep -i "warn|fail|crit|error|bad|unable" /var/log/messages
```

1. Show Ruby version:

```
# ruby -v
```

1. Show Perl version:

```
# perl -v
```

1. Show Python version:

```
# python -V
```

Additional Pentesting Resources

1. PTES: http://www.pentest-standard.org/index.php/PTES_Technical_Guidelines
2. Analysis of Metasploit relative to PTES: http://www.tinyurl.com/msf-ptes

How to get support

1. Pwnie Express Support Portal: http://www.pwnieexpress.com/support.html
2. Pwnie Express Community Support Forum: http://forum.pwnieexpress.com
3. Pwnie Express support e-mail: support@pwnieexpress.com