WLAN and Stealth Issues

Oudot Laurent
oudot@rstack.org
http://rstack.org/oudot/

Agenda

- WLAN security & some related physical issues
- Problems with open services
  - Classical solutions for TCP/IP open services
  - Another solution for TCP/IP : Port Knocking
  - Port knocking in real life ?
- Thoughts about WLAN
  - About WLAN closed mode
- New concept : WLAN Knocking ?
  - The WKnock project
- White Hat point of view / Black Hat point of view

WLAN security and physical issues

- Power : adjust AP coverage in order to radiate out toward windows but not beyond
- Directional antenna to control & contain the radio frequency array (prevent unauthorized access)
- It won’t protect against internal illicit end users or temporary visitors
  - The Wlan weapons get smaller and smaller
    - e.g : SL-C3000 + Prism2 + Internal HDD of 4Go (!)
- Classical advice : track down unknown internal AP (and clients), they could have been connected to internal networks
  - Red Teams with Wardriving stuff (not too expensive)

Problems with open services

- Let’s forget WLAN and focus on open services issues
- Some TCP/UDP services have to be open all the time
  - e.g: External daemon (sshd server), etc.
- IP addresses of remote users may change
  - No filtering based on IP source (FW,TCP wrapper,etc)
- If a service is open, blackhat may :
  - know it exists and it’s open (trivial)
  - use it for fingerprinting (OS, application)
  - try to attack the service
    - Patches must be applied on (+hardened)
    - 0day may imply intrusion (containment ?)

Solutions for TCP/IP open services

- In order to o limit and to control the access to open services :
  - Authentication through VPN solutions, etc
- But then, VPN services remain open and blackhat may still try to :
  - Scan (database for future attack when an exploit is released)
  - Fingerprint (OS, Application)
    - Example for IPSEC : IKE-SCAN
      - http://www.nta-monitor.com/ike-scan/
  - Use 0day (if any...)
- WLAN networks thus use cryptographic solutions such as WEP, WPA, etc
  - Some security risks remains (WEP is broken, etc)

Another solution : Port Knocking

- Port knocking solutions open services only when they have to be open (hidden services)
  - Principle of the least priviledge
- The concept is quite simple :
  - 1) Listening for secret sequence
    - Kind of cover channel
      - e.g : Catch a SYN to port 79, then a FIN|ACK to port 81, then ...
  - 2) Secret sequence caught => open the service
  - 3) End of session => goto 1)
- Is it security through obscurity ?
- Such techniques might be used by BH to hide trojans

Port knocking in real life ?

- Old school example : cd00r, FX
  - http://www.phenoelit.de/stuff/cd00rdescr.html
- COK, Cryptographic Port-Knocking, David Worth, Black Hat Las Vegas 2004,
  - http://blackhat.com/html/bh-usa-04/bh-usa-04-speakers.html#worth
- More implementations ?
  - http://www.portknocking.org/view/implementations

Thoughts about WLAN

- What about threats over WLAN services ?
  - Opportunistic attackers, etc
- Blackhats first need to discover WLAN services
  - Wardriving, etc
  - Identify Wlan networks by listening for Beacon, etc
- AP infrastructure and Ad-hoc modes send 802.11 Beacon frames (default average 1/100ms)
- Even non used WLAN networks might be seen
  - Okay for public hot-spots, but what about private AP ?
  - Where is the respect of the least priviledge ?

About WLAN “closed mode”

- Some AP support a feature called “closed mode”, etc
  - Beacon are still broadcasted but the SSID is hidden
  - The AP won’t send probe responses to probe requests without a correct SSID
  - It helps at avoiding illegal clients (AP browsing, etc)
- Breaking the closed mode (even with WEP)
  - Broadcast deauthentication frame with AP @MAC and clients will send the SSID in probe/association requests
- Would it be possible to avoid sending beacons ?
  - No : beacons are needed (power saving mode, etc)
  - It would stop most clients (test with XP SP2 : blocked)

Closed mode example

- Example on a Linksys WRT54GS
  - GPL firmware used “OpenWRT” [http://openwrt.org]
    - Use eth1 for WRT54G v2, eth2 for WRT54G v1&v1.1
  - Default : beacon sent every 100ms

```
root@OpenWrt:~# nvram get wl0_closed
0
root@OpenWrt:~# nvram get wl0_bcn
100
```

“closed mode” off

Delay between Beacons

  - Changes : 1 beacon per minute (problems with XP, etc)

```
root@OpenWrt:~# nvram set wl0_closed=1
root@OpenWrt:~# nvram set wl0_bcn=60000
root@OpenWrt:~# wlconf eth1 down
root@OpenWrt:~# wlconf eth1 up
root@OpenWrt:~# nvram commit
```

Demo

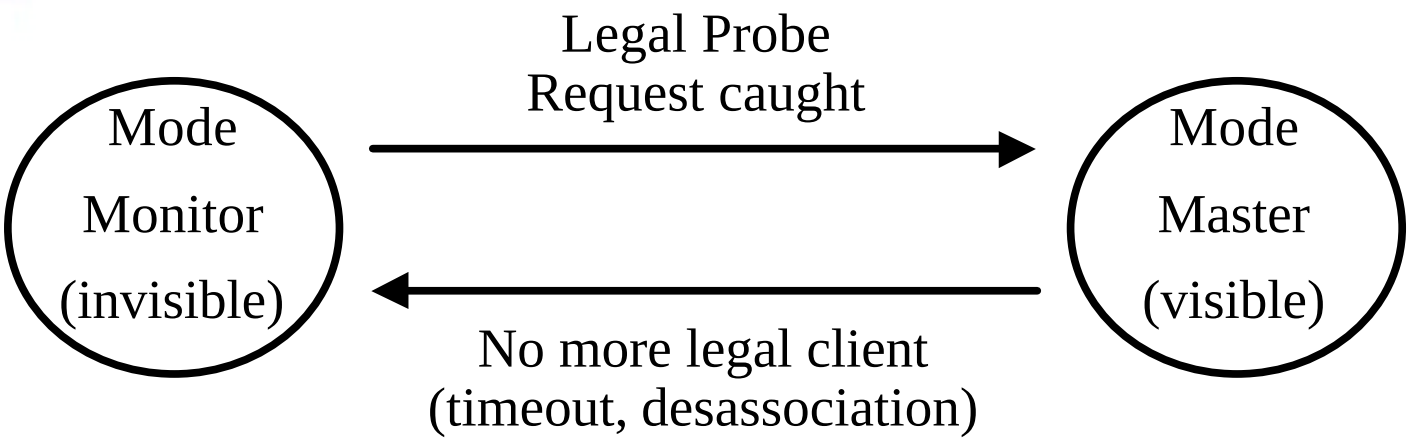
New concept : WLAN Knocking ?

- Knocking exists for TCP/IP services to avoid opportunistic attackers, etc
- Could it be possible to open a WLAN service only if it is needed ?
  - Call it “WLAN Knocking”, “Hidden Mode”, “Stealth Mode”...

WLAN Knocking

- Here is the concept :
  - 1) AP listens to the 802.11 traffic (Mode monitor (or auto))
  - 2) AP recognizes specific 802.11 frames
    - Option : AP may send specific frames too in order to exchange a WEP key, a password, etc (Diffie Hellman)
  - 3) AP switches to “open” and accept sessions (master || ad-hoc)
    - Option : MAC address filtering for trusted only clients...
  - 4) End of session => AP returns to the 1)
- Phase 2 bonus : this technique would allow to dynamically change some parameters for each session : WEP, ESSID...

Simple state machine



Wardrivers won't see the AP

This will avoid opportunistic attackers (e.g. when you're not at home)

Wardrivers will see the AP

Classical threats

Potential issues

- The biggest problem is to handle multiple clients
- At first, this solution aims at protecting one single client trying to get a hardened session with its AP
- It could be extended to multiple clients :
  - Would need 2 wireless cards on the AP
    - 1st card used for the cover channel with the clients trying to reach the service (WEP key exchange, ESSID, channel, etc)
    - 2nd card is down (no client) by default, and comes up (monitor mode) when clients have negociated an access
      - No more clients => ifdown [wl assoclist]
  - Global timeout (if nobody is connected => stealth)

The WKnock project

- WKnock is a GPL based tool (oudot@rstack.org)
  - http://rstack.org/oudot/wknock/
- This is a proof concept to show how Port Knocking might be used over WLAN networks
- Version for Linux prism54 and wireless-tools
- Version for Cisco Linksys WRT54GS (openwrt)
  - Compiled for MIPS architecture
  - Configure your network with WEP and just launch wknock :
    - The wireless card will wait for a probe request with the real essid before accepting any session
    - The AP will remain in stealth mode while there is no legal client

Inside WKnock

- AP goes to stealth mode (monitor) waiting for someone who would know the SSID
  - The field used could be something else than the SSID...
- An official client (C) tries to reach the AP
  - Found new probed network "Wanadoo_1337”
- AP goes through mode master and accept client C
- Every 60 seconds, AP goes through mode monitor
  - If the client is still there : associated probe network, so that the AP come back to mode master.
  - If there is no more client, AP jumps to mode monitor

WKnock : Probe Request analysis

IEEE 802.11
- Type/Subtype: Probe Request (4)
- Frame Control: 0x0040 (Normal)
- Duration: 0
- Destination address: ff:ff:ff:ff:ff:ff (Broadcast)
- Source address: 11:22:33:44:55:66 (11:22:33:44:55:66)
- BSS Id: ff:ff:ff:ff:ff:ff (Broadcast)
- Fragment number: 0
- Sequence number: 68

IEEE 802.11 wireless LAN management frame
- Tagged parameters (29 bytes)
  - Tag Number: 0 (SSID parameter set)
  - Tag length: 7
  - Tag interpretation: BlackHatAP
  - ...

Easy to control this parameter (ESSID)

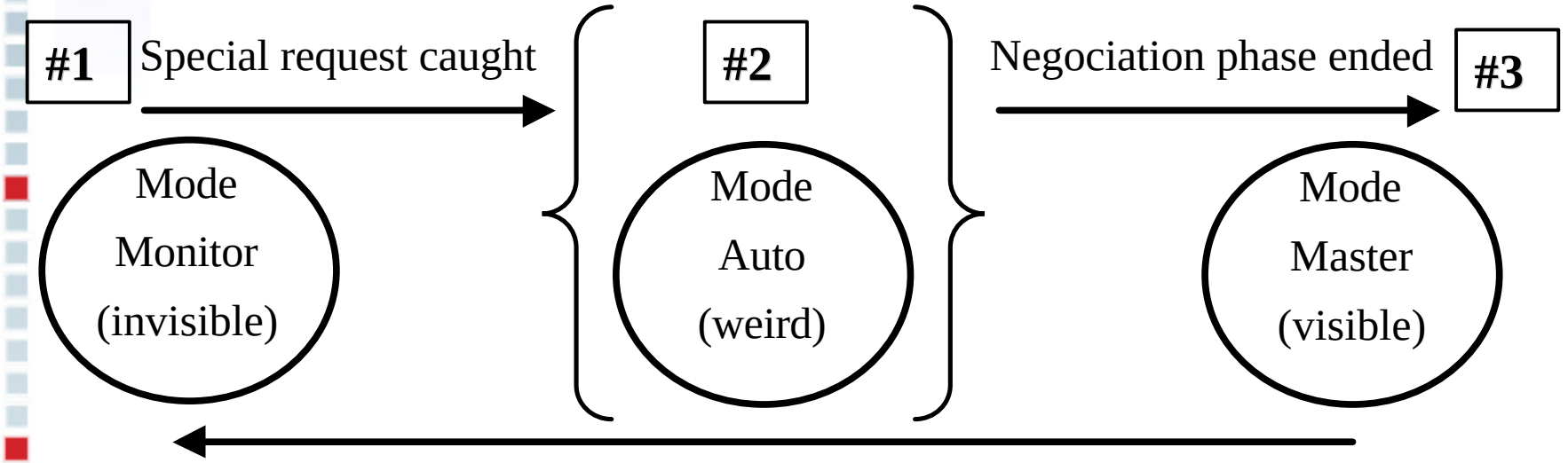
Compliant with Windows XP clients

Demo

Extended ideas

- By using Wknock, you can ask an AP to get-up once it caught a valid Probe-Request
- But what if we play with special 802.11 frames
  - We can exchange private data through this kind of covert channel (WEP Key, etc)
  - Those data may be hidden in forged WLAN packets
    - Unused fields in valide packets,
    - Unusable packets
- Proof of concept : Wknock-ng
  - Wait for a valid ProbeRequest, use the @MAC as part of the WEP key (could be cyphered, etc)
  - Remember that this is just a proof of concept

Advanced state machine



Wardrivers won't see the AP

This will avoid opportunistic attackers

Wardrivers might see weird traffic (bugs ?)

Negociation of keys, parameters, etc.

Covert channel.

Wardrivers will see the AP

Classical threats are mitigated with dynamic parameters (keys for cyphering...)

WKnock-ng : Probe Request analysis

IEEE 802.11
- Type/Subtype: Probe Request (4)
- Frame Control: 0x0040 (Normal)
- Duration: 0
- Destination address: ff:ff:ff:ff:ff:ff (Broadcast)
- Source address: 11:22:33:44:55:66 (11:22:33:44:55:66)
- BSS Id: ff:ff:ff:ff:ff:ff (Broadcast)
- Fragment number: 0
- Sequence number: 68

IEEE 802.11 wireless LAN management frame
- Tagged parameters (29 bytes)
  - Tag Number: 0 (SSID parameter set)
  - Tag length: 7
  - Tag interpretation: BlackHatAP
  - ...

Easy to control those parameters thanks to wireless-tools, etc (@MAC, ESSID)

On a client

- Classical WLAN tools may be used
- On a client
  - Plug your PCMCIA card in (eth1)
  - ifconfig eth1 hw ether 11:22:33:44:55:66
  - iwconfig eth1 essid BlackHatAP
    - Then now, the previous probe request has been sent
  - iwconfig eth1 enc 1122334455
  - ping 192.168.1.1 ...
- On the Cisco Linksys router (Openwrt)

On a WKnock-ng server

- Here, we use a Cisco Linksys router (Openwrt) :

root@OpenWrt:~# ./wknock-ng
Seeking for [mylinksys]
Mode MONITOR (wl monitor 1 && iwconfig eth2 mode managed)
.....Probe Request (mylinksys)
nvram set wl0_key1=1122334455
Mode MASTER (wl monitor 0 && iwconfig eth2 mode master)

Session open & cyphered with the WEP key hidden in the initial Request

Mode MONITOR (wl monitor 1 && iwconfig eth2 mode managed)
.....

Session closed, no beacon sent, Wknock-ng is waiting for a new Probe Request and a new key

Demo

White Hat point of view

- Wardriving and opportunistic attackers limited
  - No AP when the WLAN service is not used
  - Time period of risks is then less than 100% of the time
- This respects the principle of the least priviledge
  - AP is visible only if it’s used (aireplay family attacks slowed)
- Very usefull for private WLAN networks
- Option : create a fake WLAN network for the time your real network is not up (Wireless Honeypot)
  - "Wireless Honeypot Trickery", Oudot, SecurityFocus
    - Feb 2004, http://www.securityfocus.com/infocus/1761

Black Hat point of view

- This solution might help at hiding an AP
  - Very useful ?! It could be possible to hide a rogue AP in a company somewhere on the network
    - Corporate spy : device modified (+wlan) and installed, etc
    - A rogue AP would only reply if a legal (?!) BH come
    - WH would have problems to catch such a rogue AP
- WH (Tiger team...) might look at WLAN traffic :
  - 100% of the time but in some limited places
    - Network of kismet drones in fixed area, Wlan IDS, etc
  - Sometimes (tracking rogue AP) but everywhere (audit)
  - Indepth security and containment is needed

Going further

- Stealthiness, covert channels, WLAN, Knocking...
- Could we see such implementation on real business firmware in the future ?
  - This is Call For Action ! Ask your support...
  - Stealth AP mode (really better than “closed network”)
  - WLAN knocking modules with key exchange, etc
- Could it become a new threat in the future ?
  - Stealth Rogue AP, Opportunistic Evil Rogue AP
    - Get-up only if needed
  - Client trojaned by using WLAN network interface (?) :
    - Most laptops have builtin Wlan support even if unused
    - What about a stealth KeyLogger over WLAN ?

Thanks for your attention,

Any question ?

Greetz: Black Hat organizers and Rstack Team