SHARKFEST '13

Wireshark Developer and User Conference

Wireless Intrusion Detection

Mike Kershaw, Kismet Wireless

HELLO

my name is

inigo montoya
you killed my father
prepare to die

Wi-Fi & Security

- Everything uses Wi-Fi now
- ***EVERYTHING***
- “How do I get my IV pump on WEP?”
- Set up properly, there's nothing wrong with Wi-Fi security
- Doing it right is ***HARD*** and complicated

Why do we care?

- You need to know something is going on
- Are there rogue APs on your internal network?
- Even if you can't do anything about a DoS attack, you need to know it's happening
- Your LAN might be a WAN if you're not careful

Porous borders

- Physical company networks used to be hard to penetrate
- Not inside the building? Not on the LAN
- No-one brought their own device
- No-one was connecting their work computer to random networks at airport/starbucks/conference

Security goes both ways

- As a user, you (should) care about your own security; credit card, personal information, general exposure
- As a network admin, you (should) care about exfiltration of data / hidden devices on your network, and outside attacks

Users are wily

- If you don't give them what they want, they'll probably do it themselves
- If they do it themselves, they probably won't do a very good job on security
- And you won't even know it's there

* Yes, that's Dr Wily from Megaman

Options?

So what are your options?

Integrated WIDS

- Integrated / Enterprise WIDS
- Build into your AP infrastructure
- Very effective, but usually very expensive, implies enterprise Wi-Fi infrastructure
- Great if you have it, if you don't, or a customer does not, you'll need to find another way

Independent/Overlay WIDS

- Passive monitors distributed throughout the physical area of the wireless network
- Passively monitor wireless data independent of the network core
- Multiple commercial offerings
- Kismet can operate in distributed mode

Wi-Fi Architecture

- Wi-Fi acts both as shared media ***and*** switched media, depending on the configuration!
- When using a **Open** or **WEP** configuration, all traffic is visible to any user
- When using **WPA** each user has a per-association key, so traffic isn't visible (usually)

Monitoring wireless

- Multiple methods of monitoring, not all equal
- “Scanning mode” - same mechanism a client uses to connect, asks “What access points are available”
- “Monitor mode” - Requires support in the driver, such as Linux, or AirPCAP
- “Promsic mode” - Doesn't mean much in WiFi

WIDS can be hard

- Many vulnerabilities in Wi-Fi are not fingerprintable in traditional way
- Protocol violations can often be completely legit packets, just used in a weird way
- Have to be able to monitor trends over time not just single packet events

Who is coming after you

- Lots of problems that may or may not be malicious attackers
- Who is coming after to you?
- Can you assume it's safe?

“oops”

- Non-malicious accidental leakage from an employee bringing in an insecure AP
- Not an attack per se
- But can greatly enable an attacker near you if one is so inclined

General jackasses

- Learned how to do a DoS and likes it
- Prevalent in conferences, public venues, etc
- Not necessarily too prevalent in corporate
- Most interference in enterprise *probably* from misconfigured systems, noisy devices, congestion, etc

Indirect attacks in the wild

- Looking to compromise users in the wild
- Airports, conferences, etc
- Might take advantage of your company, might just be looking for credit card payments

Targetted external attacks

- Someone is trying to get into your company
- Has funding, reasons, and tools
- Is willing to tresspass & directly attack
- Employees leaving the network and going to coffee shops, etc are excellent targets
- You're probably screwed

Targetted internal attacks

- Employee trying to sell secrets
- Willing to bring hardware into the facility specifically to leak data
- May be disguised as an “oops!”
- You're probably screwed, but at least you can prosecute

What gets used?

Types of attacks

- Wi-Fi is vulnerable to many types of attacks
- RF denial of service
- Protocol/L2 denial of service
- Impersonation
- Client Hijacking
- Direct attacks against devices and drivers

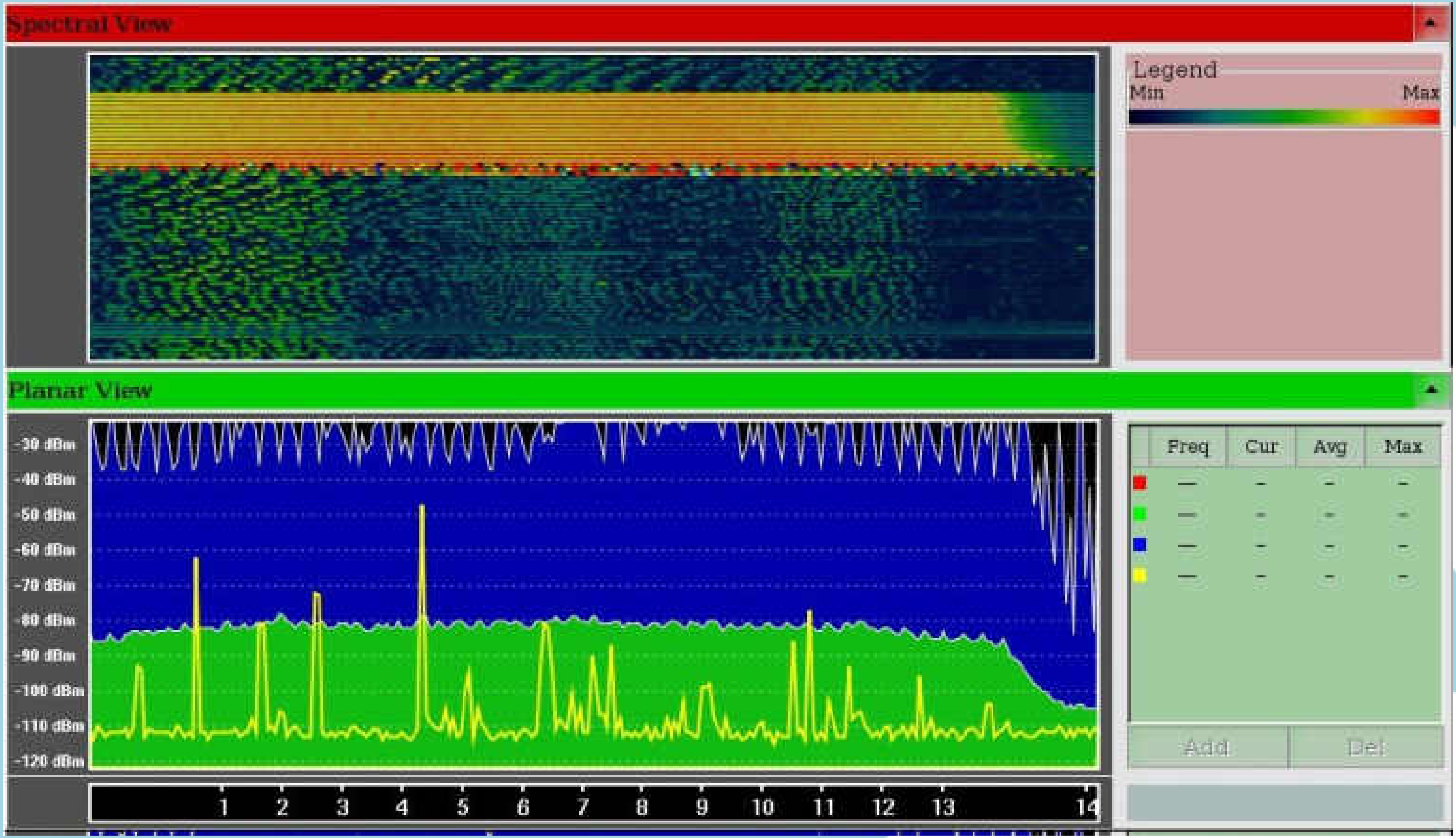
RF Denial of Service

- Wi-Fi operates in FCC ISM (Industrial, Scientific, and Medical) frequencies
- Regulated, but not the same way commercially licensed bands are regulated
- Easy to get transmitters
- Lots of legit devices, too!

RF Jamming

- Licensed “jammers”: Analog devices, security cameras, microwave ovens, baby monitors, cordless phones
- Unlicensed jammers: Wavebubble, modified wireless cards, home-brew devices

Wavebubble jammer

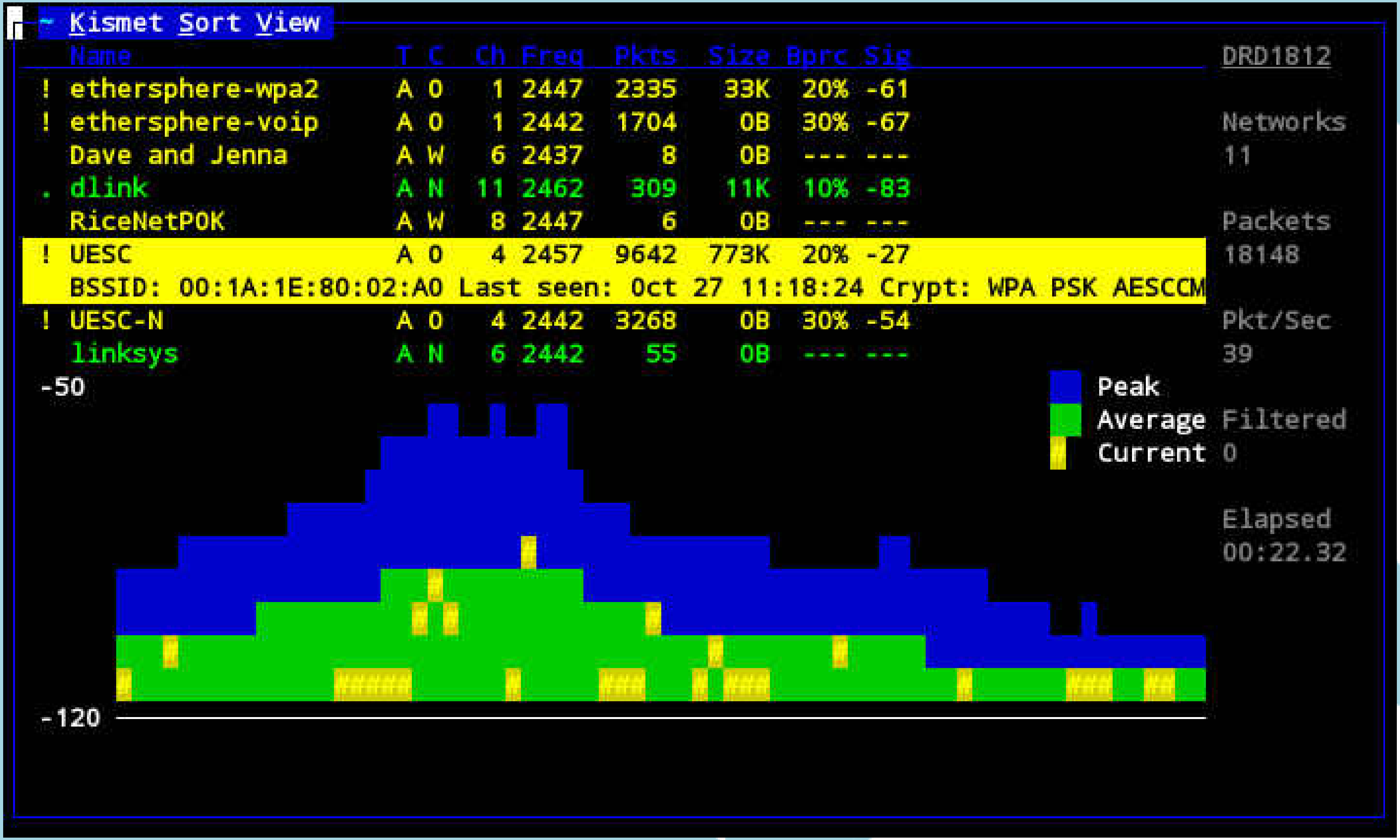


	Freq	Cur	Avg	Max
🟥	—	-	-	-
🟩	—	-	-	-
🟦	—	-	-	-
🟨	—	-	-	-

Detecting jamming

- Using hardware such as a Wi-Spy and the Kismet-Spectools plugin, or EyePA
- No actions can be taken other than “look for the person and hit them with a brick”
- Detecting jamming usually requires dedicated hardware

Detecting jamming



```
 ~ Kismet Sort View
   Name               T C  Ch Freq  Pkts  Size Bprc Sig           DRD1812
 ! ethersphere-wpa2   A O   1 2447  2335   33K  20% -61
 ! ethersphere-voip   A O   1 2442  1704    0B  30% -67           Networks
   Dave and Jenna     A W   6 2437     8    0B  --- ---           11
 . dlink              A N  11 2462   309   11K  10% -83
   RiceNetPOK         A W   8 2447     6    0B  --- ---           Packets
 ! UESC               A O   4 2457  9642  773K  20% -27           18148
   BSSID: 00:1A:1E:80:02:A0 Last seen: Oct 27 11:18:24 Crypt: WPA PSK AESCCM
 ! UESC-N             A O   4 2442  3268    0B  30% -54           Pkt/Sec
   linksys            A N   6 2442    55    0B  --- ---           39

                                                                  Filtered
                                                                  0

                                                                  Elapsed
                                                                  00:22.32
```

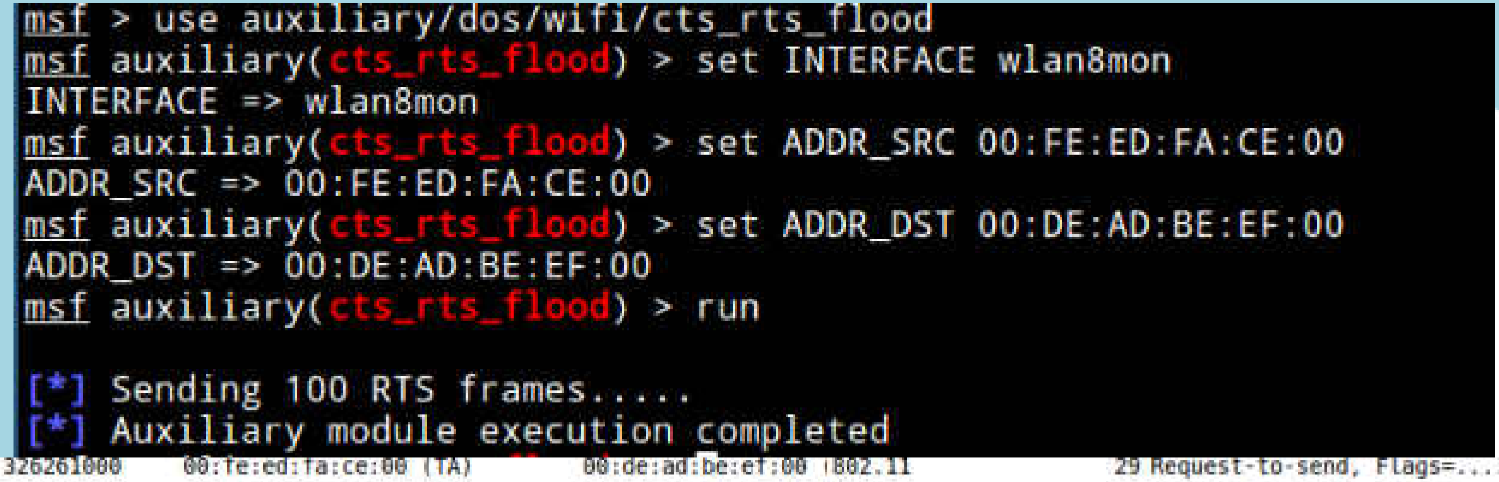
Protocol DoS

- 802.11 is a ***very*** naïve protocol
- Management frames have little to no protection
- 802.11w, 802.11r, 802.11u are finally adding management protection, over a decade later.
- Trivial to mess with 802.11
- Not much you can do about it

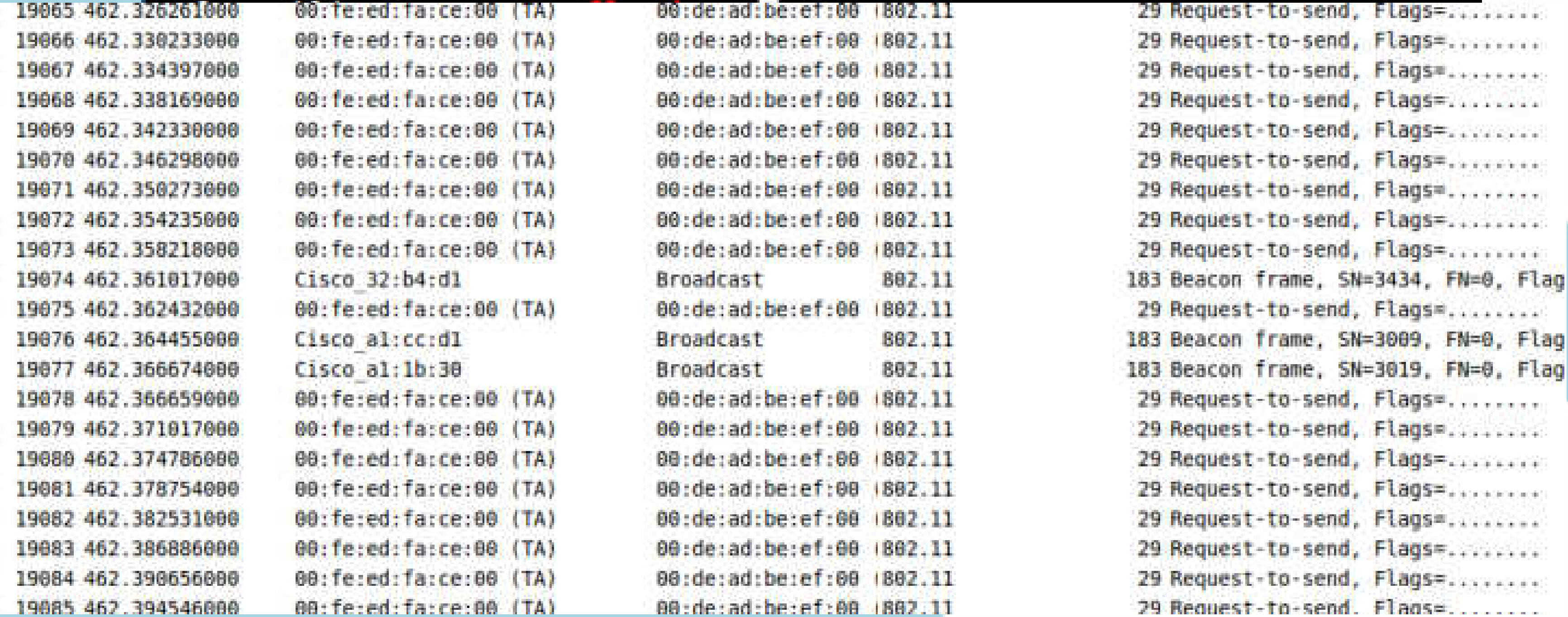
Fake saturation

- 802.11 uses CSMC/CA – unlike shared Ethernet, actively tries to avoid collisions
- “I'm going to transmit for 500ms, everyone else stay quiet”
- Attacker can saturate with spoofed RTS packets
- No-one will talk but channel will be idle!

In action



```
msf > use auxiliary/dos/wifi/cts_rts_flood
msf auxiliary(cts_rts_flood) > set INTERFACE wlan8mon
INTERFACE => wlan8mon
msf auxiliary(cts_rts_flood) > set ADDR_SRC 00:FE:ED:FA:CE:00
ADDR_SRC => 00:FE:ED:FA:CE:00
msf auxiliary(cts_rts_flood) > set ADDR_DST 00:DE:AD:BE:EF:00
ADDR_DST => 00:DE:AD:BE:EF:00
msf auxiliary(cts_rts_flood) > run

[*] Sending 100 RTS frames.....
[*] Auxiliary module execution completed
```



```
19065 462.326261000    00:fe:ed:fa:ce:00 (TA)    00:de:ad:be:ef:00 (802.11          29 Request-to-send, Flags=........
19066 462.330233000    00:fe:ed:fa:ce:00 (TA)    00:de:ad:be:ef:00 (802.11          29 Request-to-send, Flags=........
19067 462.334397000    00:fe:ed:fa:ce:00 (TA)    00:de:ad:be:ef:00 (802.11          29 Request-to-send, Flags=........
19068 462.338169000    00:fe:ed:fa:ce:00 (TA)    00:de:ad:be:ef:00 (802.11          29 Request-to-send, Flags=........
19069 462.342330000    00:fe:ed:fa:ce:00 (TA)    00:de:ad:be:ef:00 (802.11          29 Request-to-send, Flags=........
19070 462.346298000    00:fe:ed:fa:ce:00 (TA)    00:de:ad:be:ef:00 (802.11          29 Request-to-send, Flags=........
19071 462.350273000    00:fe:ed:fa:ce:00 (TA)    00:de:ad:be:ef:00 (802.11          29 Request-to-send, Flags=........
19072 462.354235000    00:fe:ed:fa:ce:00 (TA)    00:de:ad:be:ef:00 (802.11          29 Request-to-send, Flags=........
19073 462.358218000    00:fe:ed:fa:ce:00 (TA)    00:de:ad:be:ef:00 (802.11          29 Request-to-send, Flags=........
19074 462.361017000    Cisco_32:b4:d1            Broadcast          802.11         183 Beacon frame, SN=3434, FN=0, Flag
19075 462.362432000    00:fe:ed:fa:ce:00 (TA)    00:de:ad:be:ef:00 (802.11          29 Request-to-send, Flags=........
19076 462.364455000    Cisco_a1:cc:d1            Broadcast          802.11         183 Beacon frame, SN=3009, FN=0, Flag
19077 462.366674000    Cisco_a1:1b:30            Broadcast          802.11         183 Beacon frame, SN=3019, FN=0, Flag
19078 462.366659000    00:fe:ed:fa:ce:00 (TA)    00:de:ad:be:ef:00 (802.11          29 Request-to-send, Flags=........
19079 462.371017000    00:fe:ed:fa:ce:00 (TA)    00:de:ad:be:ef:00 (802.11          29 Request-to-send, Flags=........
19080 462.374786000    00:fe:ed:fa:ce:00 (TA)    00:de:ad:be:ef:00 (802.11          29 Request-to-send, Flags=........
19081 462.378754000    00:fe:ed:fa:ce:00 (TA)    00:de:ad:be:ef:00 (802.11          29 Request-to-send, Flags=........
19082 462.382531000    00:fe:ed:fa:ce:00 (TA)    00:de:ad:be:ef:00 (802.11          29 Request-to-send, Flags=........
19083 462.386886000    00:fe:ed:fa:ce:00 (TA)    00:de:ad:be:ef:00 (802.11          29 Request-to-send, Flags=........
19084 462.390656000    00:fe:ed:fa:ce:00 (TA)    00:de:ad:be:ef:00 (802.11          29 Request-to-send, Flags=........
19085 462.394546000    00:fe:ed:fa:ce:00 (TA)    00:de:ad:be:ef:00 (802.11          29 Request-to-send, Flags=........
```

Detecting saturation attacks

- Can look for absurdly long CTS/RTS durations
- Can look for CTS/RTS without corresponding data
- Both vulnerable to false positives, especially if your monitoring hardware can't see all data
- 11g seeing 11n will see control frames but not data, for example

Get off my lawn: Deauth/disassoc

- Network tells clients when they're allowed in, and when they're being disconnected
- Of course this is unencrypted...
- Deauthentiction or disassociation packets both cause the client to leave
- All you need is the BSSID and client MAC

Detecting deauth/disassoc

- Easy to detect broadcast attacks – AP will rarely send them legitimately
- Can try to fingerprint based on deauth rates, some degree of false positive

WPS Reaver

- WPS meant to make it “easy” to connect to “secure” networks
- Supposed to protect settings with an 8-digit PIN
- 10^8 = 100,000,000 possible PINs
- Except...

When is 100m = 11k?

- Handshake broken into 2 messages, one 4 digits and one 3... The last character is a checksum!
- Each message validated independently
- Errors reported for each half
- So pin is really 10^4 + 10^3, or 11,000.
- Oooooops.

What do you get?

- WPS is meant to configure clients with the security settings of the network
- Break WPS, get everything you need to join the network
- … Or become the network

Detecting Reaver attacks

- Legitimate WPS traffic should be very irregular
- Only new users joining the network for the first time
- 11,000 is still a lot of requests
- Floods = suspicion
- … But why are you using WPS!?
- Many consumer routers can't turn it off!

Impersonation attacks

- What identifies an access point?
- The network name / SSID, and encryption options
- What identifies an open access point?
- The network name … that's it.

Extremely vulnerable

- Roaming happens by looking for other networks with the same name
- Clients will happily stick to the strongest AP
- Only unique identification as far as Wi-Fi is concerned is SSID and encryption

Beacons

- Network sends a beacon ~10 times a second
- Beacon contains SSID
- What prevents someone from bringing up a network with the same name?
- Absolutely nothing

Two main ways to impersonate

- Method 1: Bring up an another AP with the same network name – Noisy but effective
- Method 2: Hijack clients as they try to connect via Karma attack – Less noisy, still effective

Client connection

- Client sends probe request to access point asking to join network "SomeNet"
- Access point responds with a probe response allowing or rejecting the client
- Client now knows the MAC address of the AP and completes the association

Spoofing the network name

- 802.11 roaming works by multiple networks with the same name, client picks the strongest one
- It's “normal” to have multiple access points with the same SSID
- Almost impossible for a user in the “connect to wireless” window to determine something is wrong

Karma attack

- Client sends probe request for an existing AP
- Karma device responds with probe response with its own MAC address
- Client is now connected to hostile AP
- But the hostile AP isn't beaconing and won't show up in normal scanning lists!

Strengthening the system

- WPA-PSK is OK if you don't share the PSK and it's reasonably strong
- As soon as you share the PSK, the two unique pieces of information, the SSID and the PSK, are public
- No good solution for public networks

WPA-EAP

- WPA-EAP methods provide secure TLS backed authentication
- PEAP, TTLS about the only ones supported on a wide range of devices / operating systems
- Require a SSL signing chain and populating every client system with them... a big pain

Impersonation impact

- Once you control the clients view of the network, you ***ARE*** the network
- I don't have to own ***the*** Internet, I just have to own ***your*** Internet

Impersonation impact

- If you're the gateway, you control all traffic
- DNS, HTTP, IP ranges
- Monitor traffic, inject into streams
- Block encryption and hope user falls back to open
- Can't decode SSL/SSH but CAN present bogus certs if user is dumb

Stream hijacking

- Unencrypted networks are basically 1980s style shared media Ethernet
- All the old-school attacks are back again!
- TCP stream hijacking trivially easy
- Both clients and network infrastructure are vulnerable

TCP hijacking

- TCP streams secure from tampering only because sequence numbers unknown
- When you can see those and can race the remote host, can hijack the TCP stream
- Allows browser injection attacks, other application stream attacks
- Allows hijacking streams from clients into servers as well

Extremely pernicious

- Targets your users in the field
- No easy way to know it's happening
- Turns zero-threat actions (going to Twitter, CNN, whatever) into high-risk high-threat actions
- Exposes persistent attacks via browser cache
- Exposes DNS hijacking vulnerabilities

Detecting stream hijacking

- Very difficult
- Hijacker can use extremely low power antenna to target a specific user in an area
- Requires more knowledge than most sniffers can have
- May not trigger IP level IDS systems either

Direct attacks against drivers

- Drivers have been better, lately, but still a vector of attack
- Packets are complex and difficult to parse, and driver authors get it wrong
- Vulnerability in driver can lead to kernel-level compromise, extremely bad

Example driver attacks

- Prism2/Orinoco firmware vulnerable to a probe response with SSID length of 0
- Broadcom windows drivers vulnerable to buffer overflow
- Dlink windows drivers vulnerable to support rates buffer overflow

Easy to detect... sort of

- Driver attacks at least are easy to detect...
- … If you're watching for them
- … In the right place at the right time
- … And you know about them

Client spoofing

- Spoofing a client MAC is easy
- Can duplicate an authenticated client
- Bypasses login requirements on open networks

Detecting client spoofing

- Different operating systems from the same MAC in DHCP requests
- Different operating systems reported by browser traffic
- Lots of weird tcp errors when different stacks get bogons

Application attacks

- Ultimately Wi-Fi just carries data
- Same attacks against systems on wired networks don't care if it's on wireless
- Border IDS can still help – so long as the user is within your border

Application attacks

- Border IDS can be placed where wireless bridges to wired network, treating wireless as a hostile external network
- Overlay WIDS can feed data to traditional IDS
- Kismet can feed Snort via virtual network interface (tun/tap in Linux)

How easy is it to perform attacks?

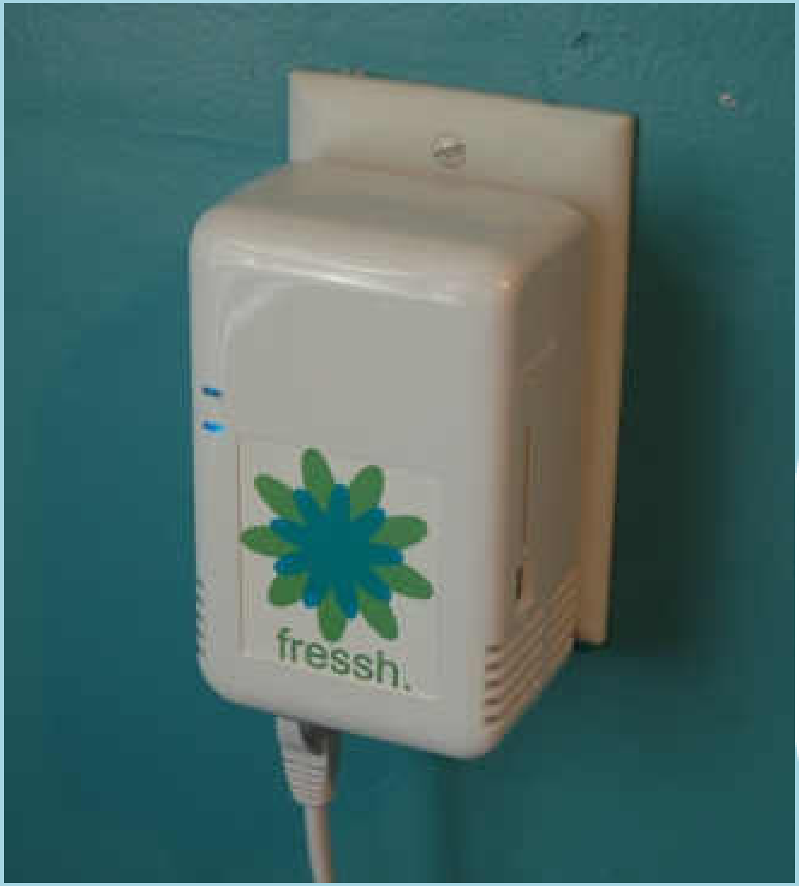
- Aircrack-NG + Linux
- Wi-Fi Pineapple
- PwnPad
- PwnPlug
- Metasploit + LORCON + Linux

Wi-Fi Pineapple

Pineapple

- Karma, Aircrack, Kismet
- Small box, battery powered
- Capable of 3G/LTE backhaul

PwnPlug



- Looks like power adapter
- Would you notice it in your office?

Attack mitigation

- DoS attacks are more or less impossible to defend against, even if we solve the protocol vulns
- WPA2 CCMP (***NOT*** WEP, ***NOT*** TKIP)
- WPA2-PSK is only as secure as the PSK – know the PSK, can spoof the AP
- WPA-EAP good, hard to set up and enforce

How bad is WEP, really?

- ***HORRIBLE***
- So bad even slow-moving standards groups like PCI have finally said “Don't use WEP”
- Trivially easy to crack a WEP network
- In seconds.

WEP is so bad...

- How bad is it?
- It's so bad that thanks to AircrackNG code in a plugin, Kismet can try to automatically crack it
- Every 5,000 packets
- Just because it's there.

Where WIDS falls down

- We can protect a single network pretty well
- WPA+EAP is very secure but hard to config
- Once users leave the secure network, all bets are off
- You can't out-engineer stupid. “Free public wifi!?”
- Users want Internet, not security

See no evil

- If you can't see what's going on you can't do anything
- 802.11n – harder to see multi-stream, increased data stream to process
- 802.11ac – will be even harder
- Super-fast tech pushing towards central AP WIDS

Things we can't currently fix

- Open networks are insecureable
- There is no way to maintain trust – no unique information in an open network
- WPA-PSK only provides trust when the PSK is unknown, no good for public networks
- WPA-EAP needs cert chain, difficult and dangerous

Active defense

- Actively defend via injection of packets
- Use the same attacks
- Difficult to enforce in shared airspace, unless you're the only occupant in a building...
- Kismet doesn't, but could with plugins

Corralling clients

- Can attempt to fence clients in
- Once you know they're legit try to keep them from connecting to illegitimate Aps
- Can try to prevent specific clients from roaming or shut down hostile AP entirely
- Requires very good overlay coverage

Things you CAN do

- Policy enforcement on company hardware
- “You can't plug that in, you can't use that work laptop at Starbucks”
- Passive interference – cages, metal walls, etc
- … Of course your users will hate you and try to find a way around you, and probably will

Things you CAN'T do

- Run jammers – the FCC will get very mad, even on Part15 networks
- Interfere with cell phones – again, the FCC will be mad
- Try to hack-back – well, I guess you CAN, but it's a REALLY bad idea

Kismet stuff!

Kismet

- Started as purely a network discovery tool
- Evolved into trend tracking, WIDS, etc
- Extensible via plugins and clients
- Interfaces with existing IDS tools
- Wireshark is concerned with packets, Kismet is concerned with devices and trends

Kismet basic operation

- Places one or more WiFi cards into monitor mode
- Listens to all the packets in the air
- Infers wireless networks, clients, crypto settings, etc from raw packets
- Discovers clients, hidden nodes, so on
- Can measure data patterns, channel usage, etc

Kismet IDS

- Both signature and trend based IDS
- Can tie into traditional IDS like Snort via tun/tap
- Can tie into other IDS/Alert systems via Kismet client protocol

Supported Kismet platforms

- Linux is still the best supported platform
- Some OSX support, but Apple likes to break drivers
- Some Windows support, with AirPCAP
- Some BSDs will work, depends on the variant and drivers

Getting the latest version

- Your distribution probably lies to you
- Latest release is 2013-03
- Debian and Ubuntu have been shipping a 2008 version. This is bad.
- Website always has the latest

Selecting hardware

- *Nearly* any wireless card can do monitor mode now (in Linux)
- Generally “best” cards are Atheros based
- External antenna jacks are almost always better
- To capture on multiple channels simultaneous, you need multiple cards

Host hardware

- Kismet is not particularly CPU expensive
- … but it IS fairly RAM hungry
- The more RAM the better – for long-term capture in busy environments, 512M+ is best, more would be better
- Drones use nearly no CPU *or* RAM since they don't need to track devices

Simpler than before

- Used to have to know what chipset & driver
- Thanks to a unified driver architecture nearly everything on Linux can be auto-detected
- Provide an interface (-c wlan0) and Kismet figures out the rest automatically
- Out-of-kernel drivers still suck

WIDS to Syslog

- Two ways to get from Kismet alerts to syslog
- Syslog plugin directly logs from Kismet to the localhost syslog, can be directed from there to central
- Syslog ruby example can be run on any system and connects to the Kismet server to get alerts and log

Kismet to Snort

- Tuntap export allows virtual 802.11 device on Linux
- Can be opened/closed repeatedly w/out disrupting Kismet
- Can point TCPDump / Wireshark / Snort at the tuntap interface
- Works just like a normal network interface

Expanding Kismet - Distributed Capture

- Kismet supports remote capture via “Kismet Drones”
- Remote capture can run on very limited hardware
- Captures packets and shoves raw data through the pipe, no packet processing overhead beyond network transmit

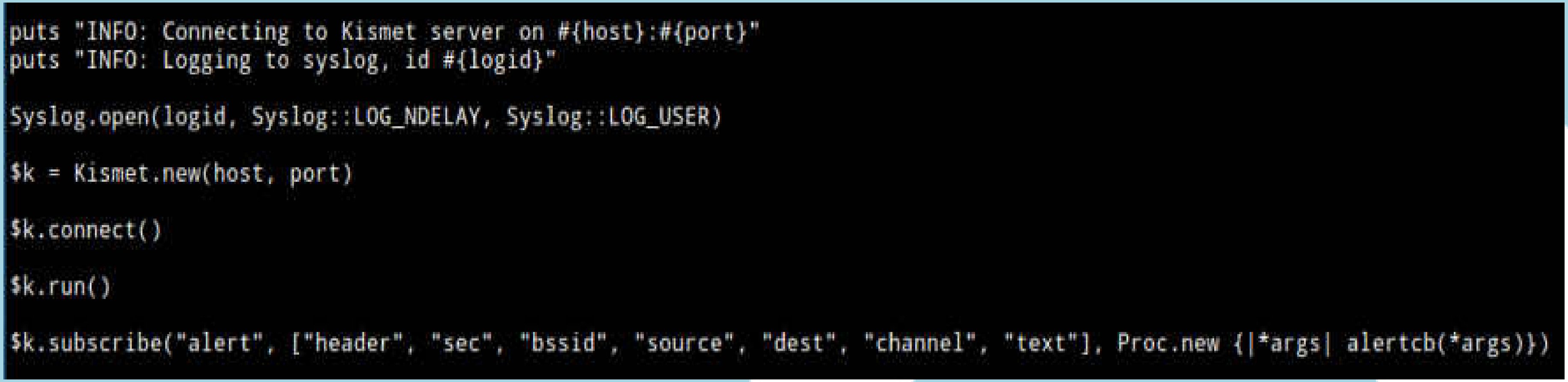
Expanding Kismet - Clients

- TCP Server/Client protocol
- Kismet UI just a network client
- Can talk to Kismet with Telnet if you're determined
- Many tasks can be completed without a plugin – just write a client!
- Example Ruby code for clients in < 100 lines

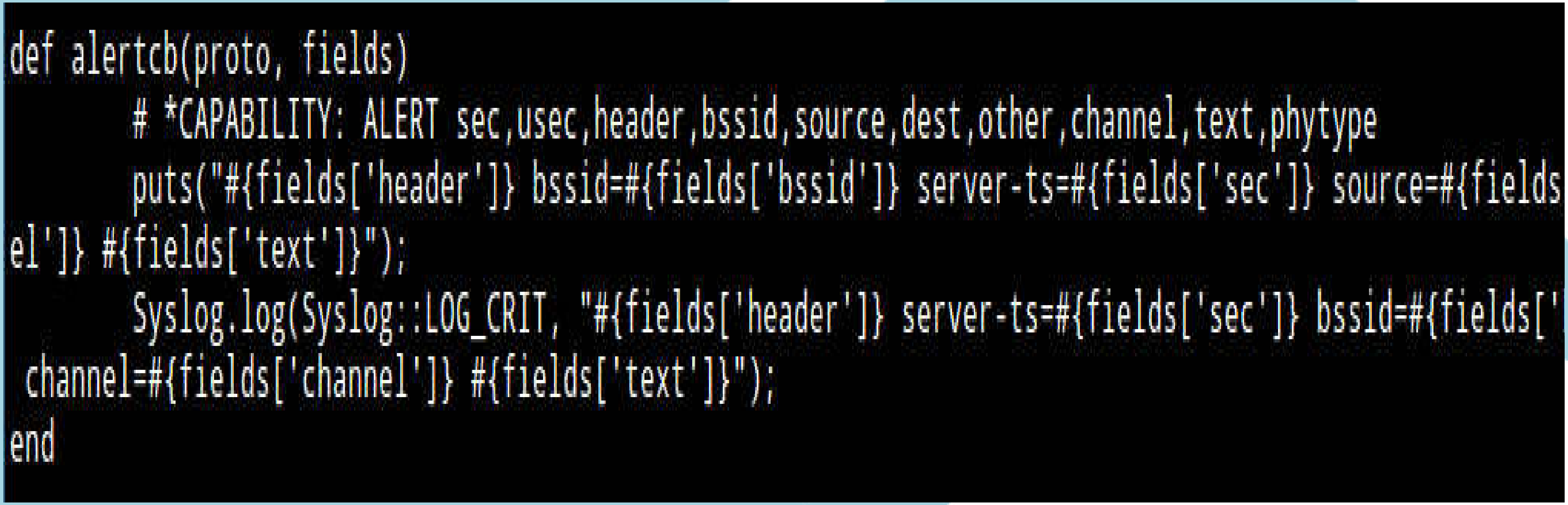
Kismet protocol

- Similar to IMAP
- Multiple sentences, can enable specific fields
- Anything displayed in the Kismet UI can be gotten from the client
- Raw packets not transmitted for sake of bandwidth

Kismet protocol



```
puts "INFO: Connecting to Kismet server on #{host}:#{port}"
puts "INFO: Logging to syslog, id #{logid}"

Syslog.open(logid, Syslog::LOG_NDELAY, Syslog::LOG_USER)

$k = Kismet.new(host, port)

$k.connect()

$k.run()

$k.subscribe("alert", ["header", "sec", "bssid", "source", "dest", "channel", "text"], Proc.new {|*args| alertcb(*args)})
```



```
def alertcb(proto, fields)
        # *CAPABILITY: ALERT sec,usec,header,bssid,source,dest,other,channel,text,phytype
        puts("#{fields['header']} bssid=#{fields['bssid']} server-ts=#{fields['sec']} source=#{fields
el']} #{fields['text']}");
        Syslog.log(Syslog::LOG_CRIT, "#{fields['header']} server-ts=#{fields['sec']} bssid=#{fields['
 channel=#{fields['channel']} #{fields['text']}");
end
```

Expanding Kismet - Plugins

- Plugins written in C++
- Directly interface with Kismet internals
- Can be for the server or client
- Harder to write but as powerful as Kismet itself
- Internal architecture all basically statically compiled plugins

Server plugins

- Able to define new capture source types
- Able to define new PHY layers (ie Ubertooth, etc)
- Able to create new log types
- Able to create new network protocols, or entirely new network servers

Client plugins

- Able to interface to server sentences
- Able to create new ncurses widgets in the UI
- Able to modify menus, etc to add preference options and such

Wi-Fi – One of Many

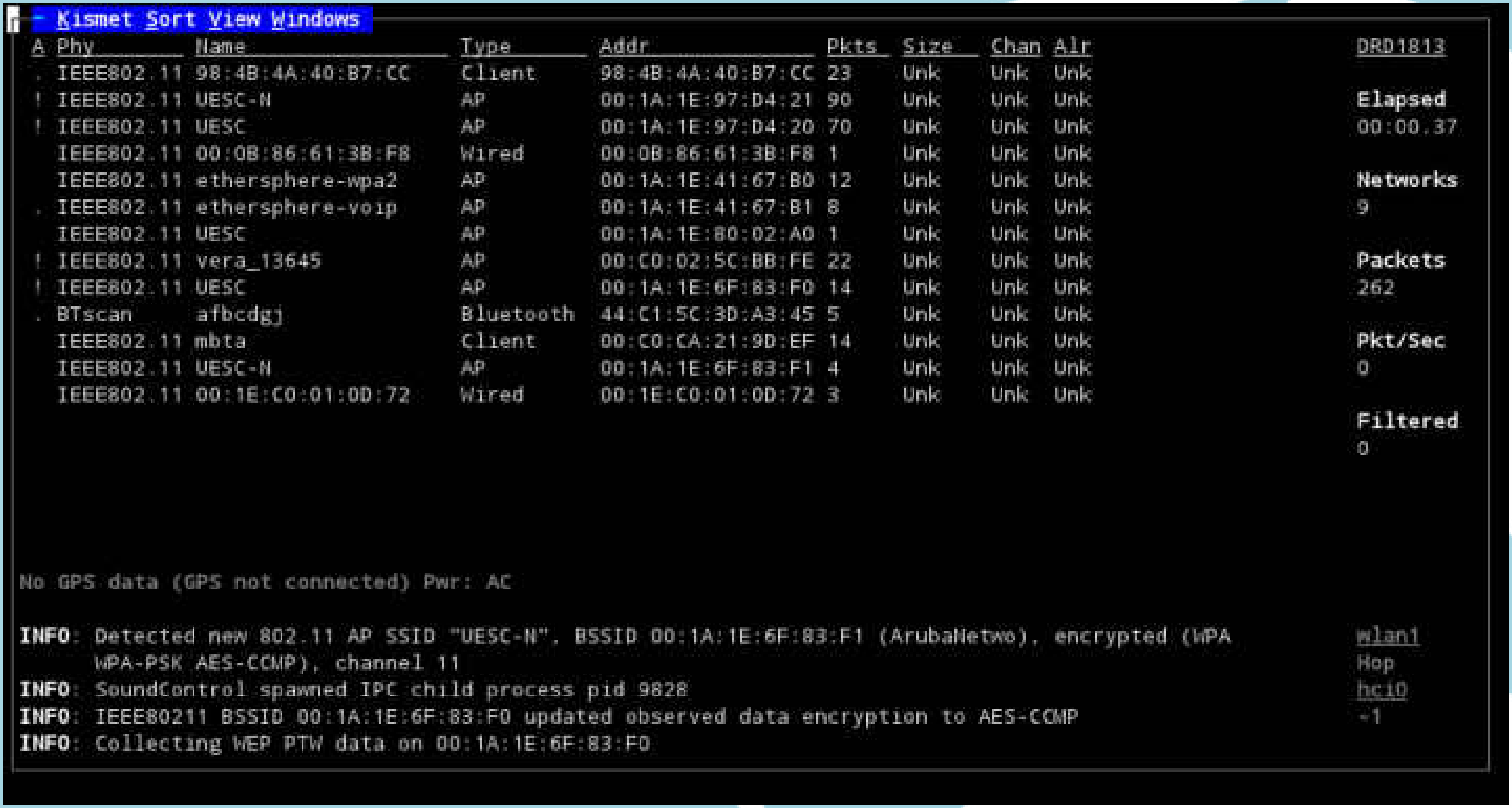
Going beyond Wi-Fi

- What about other protocols?
- Attackers can definitely use alternate networking standards once on your network
- Do you know what devices are bridged to your network?
- What about SCADA, inventory, etc systems?

Kismet Phy-Neutral

- Significant rewrite of Kismet core tracker
- Instead of being 802.11 centric, will be able to take plugins for any packetized PHY type
- Will also be able to take plugins for non-packetized device detection (some SDR, etc)
- Common device list across all phy types

Kismet Phy-Neutral



```
- Kismet Sort View Windows
 A Phy         Name                  Type        Addr              Pkts  Size   Chan Alr                  DRD1813
 . IEEE802.11 98:4B:4A:40:B7:CC      Client      98:4B:4A:40:B7:CC 23    Unk    Unk  Unk
 ! IEEE802.11 UESC-N                 AP          00:1A:1E:97:D4:21 90    Unk    Unk  Unk                  Elapsed
 ! IEEE802.11 UESC                   AP          00:1A:1E:97:D4:20 70    Unk    Unk  Unk                  00:00.37
   IEEE802.11 00:0B:86:61:3B:F8      Wired       00:0B:86:61:3B:F8 1     Unk    Unk  Unk
   IEEE802.11 ethersphere-wpa2       AP          00:1A:1E:41:67:B0 12    Unk    Unk  Unk                  Networks
 . IEEE802.11 ethersphere-voip       AP          00:1A:1E:41:67:B1 8     Unk    Unk  Unk                  9
   IEEE802.11 UESC                   AP          00:1A:1E:80:02:A0 1     Unk    Unk  Unk
 ! IEEE802.11 vera_13645             AP          00:C0:02:5C:BB:FE 22    Unk    Unk  Unk                  Packets
 ! IEEE802.11 UESC                   AP          00:1A:1E:6F:83:F0 14    Unk    Unk  Unk                  262
 . BTscan     afbcdgj                Bluetooth   44:C1:5C:3D:A3:45 5     Unk    Unk  Unk
   IEEE802.11 mbta                   Client      00:C0:CA:21:9D:EF 14    Unk    Unk  Unk                  Pkt/Sec
   IEEE802.11 UESC-N                 AP          00:1A:1E:6F:83:F1 4     Unk    Unk  Unk                  0
   IEEE802.11 00:1E:C0:01:0D:72      Wired       00:1E:C0:01:0D:72 3     Unk    Unk  Unk
                                                                                                          Filtered
                                                                                                          0



No GPS data (GPS not connected) Pwr: AC

INFO: Detected new 802.11 AP SSID "UESC-N", BSSID 00:1A:1E:6F:83:F1 (ArubaNetwo), encrypted (WPA          wlan1
      WPA-PSK AES-CCMP), channel 11                                                                       Hop
INFO: SoundControl spawned IPC child process pid 9828                                                     hci0
INFO: IEEE80211 BSSID 00:1A:1E:6F:83:F0 updated observed data encryption to AES-CCMP                      -1
INFO: Collecting WEP PTW data on 00:1A:1E:6F:83:F0
```

PHY-N Advantages

- Much simpler plugins – tracking, logging, basic display handled by Kismet
- Designed to produce usable consistent logs from any set of input types
- Kismet becomes central data gatherer for any wireless data

PHY-N support in progress or planned

- Ubertooth and Ubertooth BTLE – in progress, supported in Git
- Kisbee – 802.15.4 capture, supported but hard to classify networks
- RFCat / FSK – Planning classification still
- SDR – HackRF, etc can in theory talk any protocol

Writing for PHY-N

- Each device record has a common component
- Additional information is attached as tagged blobs of data
- Phy-N plugin can define any additional data for any device it needs

So what else do we care about?

- Other protocols used for *important* things
- “Internet of Things” is already here – it may be your inventory or factory control
- “Active” inventory tags use things like Zigbee
- Zigbee often used on sensor networks and physical devices

The value of data

- Can you trust your sensor network?
- What can go wrong if someone can spoof it?
- Is it connected to your company Intranet?
- Does it control security or safety responses?
- Can it leak internal processes? (Bake @ 1500F for 20 minutes, then...)

Heist of the century

- When used for inventory control, how is it checked?
- Can someone spoof it and try to remove the original tagged item?
- Place original tag in faraday cage, bridge to spoof tag over cell, replicate packets at original location
- Silly? What if tag is on semi truck of components?

Loss of control

- What does your sensor network control?
- How much money would you lose if your factory crashed?
- Can your network be seen from outside your perimeter?
- If you're not looking, you can't know

Ninja-level problems

- Attackers may not even need to be w/in wireless range
- If a packet format is FF1245678 and someone sends a wired packet of FF123FF45678
- If something causes the beginning of the packet to corrupt...

Go away PIP nobody likes you

- Then the second part of the packet may be detected as a complete frame and handled as if it was the original data!
- Forge wireless from the wire!
- Zigbee is especially vulnerable b/c of simplicity, see work by Travis Goodspeed

Different != better

- Custom protocols haven't been exampled as closely
- SDR is now really cheap
- Simpler protocols may have less or no protection against injection/replay/packet in packet
- Do you know every wireless device on your net?

Other thoughts on wireless data leakage

- Do you still use pagers to communicate to staff?
- Bridge your email directly to them?
- Did you know those are unencrypted?
- And you can pick them up w/ a $20 USB SDR?
- Of course, that's illegal.
- And a criminal would never break the law...

Things you probably send to pagers

- Router interface names
- Alarm system updates
- Webserver failures... with internal names
- Internal server names
- When someone is on duty
- What monitoring SW you run
- What email servers you run
- ... Just sayin'...

Some folk'll never commit a felony to break into your company...

- ... But then again, some folks'll
- Just because it's illegal to monitor or attack something doesn't mean someone won't do it
- They're a criminal, after all.
- Be aware of as many vectors as you can and try to be capable of monitoring, etc

Recap

- If you don't know to look you can't know how bad it is
- Look in unexpected places
- Everything has security problems; arm yourself with more info
- More wireless tech = more things to monitor

Q&A

Questions? Anyone? Bueller?