Layer 2 Analysis of WLAN Discovery Applications for Intrusion Detection

Joshua Wright, GCIH, CCNA
11/8/2002

Abstract

Wireless LAN discovery through the use of applications such as NetStumbler, DStumbler, Wellenreiter and others is an increasingly popular technique for network penetration. The discovery of a wireless LAN might be used for seemingly innocuous Internet access, or to be used as a "backdoor" into a network to stage an attack. This paper reviews some of the tactics used in wireless LAN netwo rk discovery and attempts to identify some of the fingerprints left by wireless LAN discovery applications, focusing on the MAC and LLC layers. This fingerprint information can then be incorporated into intrusion detection tools capable of analyzing data-link layer traffic.

Introduction

The growth of 802.11 networks has been met with the development of several wireless local area network (WLAN) discovery applications. These applications are designed to identify WLAN activity and network characteristics, providing enough information for an unauthorized user to gain access to the target network. For obvious reasons, WLAN administrators should be concerned about unauthorized access to their networks and therefore should attempt to identify the applications used to discover their networks.

WLAN intrusion analysis is not entirely unlike traditional intrusion analysis; we are primarily concerned about the identification of traffic signatures or fingerprints that are unique to the applications we want to detect. Unlike traditional intrusion analysis however, we have additional challenges that are unique to wireless networks:

1. *Location of traffic capture station*
   Where traditional intrusion detection systems can be location in a functional area (DMZ, inside a firewall, outside a firewall, etc), a data collection agent (agent) capturing 802.11 frames must be installed in the same service area of each wireless LAN we wish to monitor. The improper location of a wireless LAN agent will inevitably lead to false positive results. If the receive sensitivity of the agent exceeds that of the monitored network, traffic may be characterized as WLAN discovery while being outside the cell range of the monitored network. Another interesting challenge is monitoring "hidden node" IBSS stations where the last wireless station to generate a beacon is responsible to reply to probe requests (ANSI/IEEE, 126). In these cases, the wireless LAN agent may not be within the coverage area necessary to collect responses or further solicitation of management information from the responding “hidden node” station.

2. *Identifying anomalous traffic*
   In order for wireless clients to locate a network to join, the IEEE 802.11 specification made an accommodation for clients to broadcast requests for available networks. Applications

like NetStumbler and DStumbler utilize this and other scanning techniques to discover wireless LANs. Since these techniques are part of the 802.11 specification, we should expect to see their use for legitimate network discovery. Fortunately, the scanning techniques used by NetStumbler and DStumbler that are presented here are anomalous when compared to the 802.11 specification and can be detected among otherwise legitimate network scanning traffic.

Characteristics for Analysis

Through lab experimentation and the analysis of "hostile" network activity, I have found several fields of interest that are pertinent to the analysis and discovery of WLAN discovery applications.

1. *Sequence number*
   A sequence control field is used to accommodate fragmentation of 802.11 frames. Two bytes in length, the first 4 bits (low order) are used to uniquely identify each fragment with the remaining 12 bits used to uniquely identify the frame based on a modulo 4096 counter (Gast, 41). The sequence number portion of the sequence control field is analogous to its upper-layer counterpart, the IPID field.

2. *Control Type and Subtype*
   The control type and control subtype fields utilize two bytes of the 802.11 frame control structure to identify data, management and control frames used to support the robust operation of 802.11 networks. Common type/subtype combinations include 00/0100 (probe request), 00/0101 (probe response), 00/1000 (beacon) and 10/0000 (data). As many of the remaining combinations of type and subtype are reserved for future use, we will likely see a good deal of experimentation to elicit unique responses to unexpected frame types.

3. *Destination MAC*
   A client seeking to discover available wireless networks through active scanning will transmit a probe request frame to the MAC layer broadcast address (“FF:FF:FF:FF:FF”).

4. *Service Set Identifier (SSID)*
   In addition to setting the destination MAC address to the broadcast address, the SSID is set to a value of “0x00” in probe request frames. The last station to communicate in an IBSS network is responsible for responding with the network SSID to probe requests. In infrastructure networks, the access point (AP) is responsible for responding to probe requests with its configured SSID, unless otherwise directed.

5. *Organizationally Unique Identifier (OUI)*
   Part of the 802.2 LLC frame encapsulation, the OUI field consists of three bytes used to uniquely identify a vendor as part of a convention for locally administered protocol identifiers. This value will commonly be set to all 0's for encapsulated Ethernet frames.

6. *Data Payload*
   The data payload of LLC frames may contain unique information for use in identifying the network discovery application generating the traffic.

7. *LLC Protocol Type Field*
   The LLC protocol type field specifies the upper-layer protocol type. Common examples include 0x0800 for IP traffic, 0x0806 for ARP and 0x888e for 802.1x authentication.

8. *LLC Protocol ID*
   Part of the SNAP header, the LLC protocol ID number is used to specify the protocol type used with the particular OUI specified in the frame.

Research Techniques and Data Collection

Ethereal version 0.9.5 was the primary tool for data collection using, Libpcap version 0.7 libraries. A Cisco Aironet 352 card was used with Cisco Aironet version 1.8 drivers. The capture host ran Slackware 8.1 Linux and a custom 2.4.18 kernel.

Where noted, various hosts were used to generate sample traffic that was later analyzed for pattern matching signatures. Each test set was performed in multiple permutations: against WEP-protected and open networks; with and without client power management enabled; and with an AP using beaconing SSID advertisements and with an AP configured to not beacon its ESSID (a cloaked ESSID configuration).

After identifying unique characteristics of the application data, Ethereal was used on a Sun Microsystems Sun Fire 280R to analyze datasets collected by the author from various network security conferences including SANSFIRE 2002 and DEFCON X, as well as against packet captures from various production wireless networks maintained by the author. For the sake of confidentiality, the traces provided as examples are from lab-generated traffic, although each traffic pattern has been identified in multiple dissimilar environments.

Active Probing vs RF Monitoring

Network discovery applications will typically employ one of two different techniques when discovering wireless LANs. Each has their advantages and disadvantages, as explained below.

Active Probing

Described in the IEEE 802.11 specifications (ANSI/IEEE, 126), the active probing method uses probe request frames on each channel where it is able to detect wireless activity (to avoid blocking the discovery process on frequencies that are not in use). When an AP comes within range of the client and receives a probe request frame it will typically respond with a probe response frame containing the network ESSID.

The active scanning method for network discovery is the easiest to implement and is presently the only network discovery method used on Windows systems. This method has the distinct disadvantage of being unable to discover wireless networks that are configured not to advertise their ESSID using a cloaked ESSID configuration. Furthermore, the active scanning method requires the client to communicate actively with the AP, giving the intrusion analyst the opportunity to discover the intrusion.

RF Monitoring

An increasing number of WLAN discovery applications utilize a completely passive method of WLAN discovery known as radio frequency monitoring (RFMON). A client with a wireless card that is configured in RFMON mode will be able to capture all RF signals on the channels to which it is configured to listen.

WLAN discovery applications using the RFMON discovery technique will interpret and display information about the networks they are able to receive; it will do so without the limitation of presenting only information discovered in probe request and probe response frames. As a completely passive technique, it is very difficult for the intrusion analyst to discover this activity at the MAC layer.

This passive discovery technique does present some distinct disadvantages to the attacker. At the time of this writing, free network discovery applications using RFMON scanning are only available on Linux and BSD-based operating systems, with limited chipset support on BSD hosts. Through the work of ghandi (ghandi@dopesquad.net), Mac OS X clients are now able to utilize Apple AirPort cards for passive monitoring (Ghandi). Some commercial products are available for Windows, but at significant cost.

While in RFMON mode, wireless clients are unable to transmit any frames; their cards are only able to receive, and therefore capture traffic. This limits the client to reporting only current or recorded network traffic. For instance, a client using passive monitoring would be able to report on the MAC addresses and number of associations to a discovered AP, but would be unable to probe the discovered AP for SNMP MIB information. The client would be able to report the results of an SNMP GET request, however, if the request initiated from a different card or a different client in the same frequency and in proximity to the attacker.

A small window of opportunity does exist, however, to fool the attacker with *falsified network management* traffic. Since the RFMON client captures *all* traffic on its listening frequency, it is only a matter of protocol decoding for the tool author to present information to the user about traffic seen on the network. Some of the more advanced tools provide full DHCP packet dissection and reporting, as well as reporting Cisco Discovery Protocol (CDP), BOOTP and SNMP traffic. Since the client cannot transmit while the wireless card is in RFMON mode, they have little opportunity to validate the source of the generated traffic as legitimate.

The intrusion analyst could generate falsified network management frames in the form of DHCP address allocation, CDP advertisements, or SNMP MIB information to and from non-existent hosts on the network. The WLAN discovery application would interpret this information and dutifully report the results to the attacker. The intrusion analyst could then monitor network traffic with traditional layer 3+ intrusion detection tools, looking for probes based on this falsified data (for example, watching for an SNMP string that was used to collect false MIB information). Should the falsified network management traffic be "interesting" to the attacker, they may opt to remove their card from RFMON mode in order to gain the ability to transmit frames and associate with the discovered network.

A practical example of this technique is as follows: An administrator generates DHCP response frames that appear as if addresses in the range 172.16.1.0/24 are being allocated to clients (this type of traffic could be generated with a short Perl or C program). A passive WLAN discovery application would report to its user, the attacker, that it is seeing a DHCP server issue addresses in the noted range. An attacker, curious to discover what services these hosts are offering, might try to obtain a valid network address by requesting a DHCP address or by specifying an "unallocated" static address. It is likely that the attacker would then attempt to perform port scanning and/or host fingerprinting against all the hosts that were noted by the network discovery application as having received DHCP leases. An intrusion analyst would only need to monitor their network for signs of activity to the falsified network address advertisements and then capture traffic and generic RF traffic statistics about the attacker. Through the use of triangulation techniques, the intrusion analyst might even be able to establish the location of the assailant.

This approach may not be well suited to all intrusion analysts, resembling more closely a honeypot than a traditional intrusion analysis technique. As a potential alternative, there is ongoing research in attempting to discovery passive network discovery tactics through the use of physical layer RF characteristics that may be more applicable to high-security environments (reference).

Detecting NetStumbler

- Name: NetStumbler, MiniStumbler
- Download: http://www.netstumbler.com/
- Source available: no
- Discovery method: Active Scanning/Probing
- Features: Simple install, GPS support, probes discovered AP for additional information
- Supported chipsets: Lucent
- Supported platforms: Windows, Pocket PC
- Author: Marius Milner

NetStumber is a very popular tool for Windows users, utilizing active scanning through the use of probe requests sent to a broadcast address with a broadcast BSSID and an unspecified ESSID (length of 0). The probe requests are difficult to identify definitively as NetStumbler activity since NetStumbler utilizes the active scanning method described in the IEEE 802.11 specification without anomalous characteristics. Once an AP is discovered however, NetStumbler will probe a discovered AP for its "nickname" information, often the same information stored in the SNMP MIB system.sysName.0 parameter. This LLC/SNAP frame contains unique characteristics that allow us to uniquely identify NetStumbler activity.

In a posting to the Kismet Wireless mailing list on March 28 2002, Mike Craik first identified a unique pattern that can be used to identify NetStumbler traffic (Craik). LLC-encapsulated frames generated by NetStumbler will use an organizationally unique identifier (OID) of 0x00601d and protocol identifier (PID) of 0x0001. NetStumber also uses a data payload size of 58 bytes containing a unique string that can be used to identify the version of NetStumbler:

NetStumbler Version	Payload String
3.2.0	Flurble gronk bloopit, bnip Frundletrune
3.2.3	All your 802.11b are belong to us
3.3.0	intentionally blank [1]

[1] Note leading spaces.

To identify NetStumbler traffic we can use the following Ethereal display filter to detect any of the data string patterns that match the OUI and PID criteria:

```
        (wlan.fc.type_subtype eq 32 and llc.oui eq 0x00601d and llc.pid eq 0x0001) and
                (data[4:4] eq 41:6c:6c:20 or data[4:4] eq 6c:46:72:75 or data[4:4] eq 20:20:20:20)
```

The following detect is taken from NetStumbler version 3.2.3 using a Windows 2000 host for discovery.

```
IEEE 802.11
        Type/Subtype: Data (32)
        Frame Control: 0x0908
                Version: 0
                Type: Data frame (2)
                Subtype: 0
                Flags: 0x9
                        DS status: Frame is entering DS (To DS: 1 From DS: 0) (0x01)
                        .... .0.. = Fragments: No fragments
                        .... 1... = Retry: Frame is being retransmitted
                        ...0 .... = PWR MGT: STA will stay up
                        ..0. .... = More Data: No data buffered
                        .0.. .... = WEP flag: WEP is disabled
                        0... .... = Order flag: Not strictly ordered
        Duration: 258
        BSS Id: 00:50:18:07:13:92 (ADVANCED_07:13:92)
        Source address: 00:02:2d:52:cb:27 (Agere_52:cb:27)
        Destination address: 00:50:18:07:13:92 (ADVANCED_07:13:92)
        Fragment number: 0
        Sequence number: 3057
Logical-Link Control
        DSAP: SNAP (0xaa)
        IG Bit: Individual
        SSAP: SNAP (0xaa)
        CR Bit: Command
        Control field: U, func = UI (0x03)
                000. 00.. = Unnumbered Information
                .... ..11 = Unnumbered frame
        Organization Code: Unknown (0x00601d)
        Protocol ID: 0x0001
Data (58 bytes)

0000  00 00 00 00 41 6c 6c 20 79 6f 75 72 20 38 30 32   ....All your 802
0010  2e 31 31 62 20 61 72 65 20 62 65 6c 6f 6e 67 20   .11b are belong 
0020  74 6f 20 75 73 2e 20 20 20 20 20 20 fe ca ba ab   to us.      ....
0030  ad de 0f d0 4f 45 43 45 46 46                     ....OECEFF
```

MiniStumbler is a similar tool compiled to run on the Microsoft Pocket PC platform. Although the binary “ministumbler.exe” program contains the string “All your 802.11b are belong to us”, this version of NetStumbler does not exhibit the same characteristics as its Win32 counterpart. MiniStumbler will not send a data probe to a discovered AP.

Recent trends in the NetStumler online discussion forums indicate that NetStumbler users are well aware of the intrusion detection possibilities that exist with NetStumbler. Several posts have recommended using a binary file editor to change the "All your 802.11b are belong to us." string in the netstumbler.exe program file to avoid detection.

Detecting DStumbler

- Name: DStumbler, bsd-airtools
- Download: http://www.dachb0den.com/projects/dstumbler.html
- Source available: yes
- Discovery method: Active Scanning/Probing or Passive RF Monitoring
- Features: Reports APs with default configuration, GPS support, reports additional information about discovered networks (WEP enabled, beacon interval, node detection)
- Supported chipsets: Any that are supported by host OS, Prism II for Passive RF Monitoring
- Supported platforms: NetBSD, FreeBSD, OpenBSD
- Author: h1kari

DStumbler is an open-source curses-based application for use on BSD-based systems. DStumbler offers two methods of scanning for wireless networks: active probe/response scanning (similar to NetStumbler) and passive (RFMON) scanning. Several additional features are also available including support for additional card chipsets; additional reporting on discovered networks (number active nodes, beacon intervals, default SSIDs, supported data rates); and reports a partial detect for WEP key lengths.

Unlike NetStumbler, DStumbler does not actively probe discovered any APs for additional information, instead presenting additional detail derived solely from the data it discovers. When in active scanning mode, DStumbler does have unique qualities that make it possible for us to detect its probe request frames.

In a posting to the Kismet Wireless mailing list on March 28 2002, Mike Craik first identified a pattern for identifying DStumbler traffic using only WLAN sequence numbers in the pattern 0, 3, 6, 9, 11 (Craik2). I was able to reproduce this pattern, but discovered that it is inconsistent when capturing DStumbler activity on different channels.

When DStumbler starts in active scan mode, it will generate multiple probe request frames (frame control 0x0040) using low-numbered, modulo 12 sequence number values. After receiving a probe response from an AP, DStumbler will attempt to authenticate, and then associate with, the AP. The authenticate frame uses a consistent sequence value of 11 (0x0b), and the following association request uses a sequence value of 12 (0x0c). This pattern will repeat until DStumbler does not receive probe response frames from AP.

Unfortunately, a simple per-packet pattern-matching algorithm will not be able to identify DStumbler activity reliably; instead we need to identify a repetitive pattern of activity. We can use the following Ethereal display filter to create a data extract which we can then manually inspect for the repeating authentication frames with a sequence of 11 and associate frames with a sequence of 12:

```
        (wlan.seq eq 11 and wlan.fc.subtype eq 11)
               or (wlan.seq eq 12 and wlan.fc.subtype eq 00)
```

The following detect was generated using DStumbler 1.0 on a FreeBSD 4.6.2 client.

```
IEEE 802.11
    Type/Subtype: Authentication (11)
    Frame Control: 0x00B0
        Version: 0
        Type: Management frame (0)
        Subtype: 11
        Flags: 0x0
            DS status: Not leaving DS or network is operating in AD-HOC mode (To DS: 0  From DS: 0)
(0x00)
            .... .0.. = More Fragments: This is the last fragment
            .... 0... = Retry: Frame is not being retransmitted
            ...0 .... = PWR MGT: STA will stay up
            ..0. .... = More Data: No data buffered
            .0.. .... = WEP flag: WEP is disabled
            0... .... = Order flag: Not strictly ordered
    Duration: 258
    Destination address: 00:e0:63:82:19:c6 (00:e0:63:82:19:c6)
    Source address: 00:02:2d:0a:01:de (00:02:2d:0a:01:de)
    BSS Id: 00:e0:63:82:19:c6 (00:e0:63:82:19:c6)
    Fragment number: 0
    Sequence number: 11
IEEE 802.11 wireless LAN management frame
    Fixed parameters (6 bytes)
        Authentication Algorithm: Open System (0)
        Authentication SEQ: 0x0001
        Status code: Successful (0x0000)

IEEE 802.11
    Type/Subtype: Association Request (0)
    Frame Control: 0x0000
        Version: 0
        Type: Management frame (0)
        Subtype: 0
        Flags: 0x0
            DS status: Not leaving DS or network is operating in AD-HOC mode (To DS: 0  From DS: 0)
(0x00)
            .... .0.. = More Fragments: This is the last fragment
            .... 0... = Retry: Frame is not being retransmitted
            ...0 .... = PWR MGT: STA will stay up
            ..0. .... = More Data: No data buffered
            .0.. .... = WEP flag: WEP is disabled
            0... .... = Order flag: Not strictly ordered
    Duration: 258
    Destination address: 00:e0:63:82:19:c6 (00:e0:63:82:19:c6)
    Source address: 00:02:2d:0a:01:de (00:02:2d:0a:01:de)
    BSS Id: 00:e0:63:82:19:c6 (00:e0:63:82:19:c6)
    Fragment number: 0
    Sequence number: 12
IEEE 802.11 wireless LAN management frame
```

```
    Fixed parameters (4 bytes)
        Capability Information: 0x0011
            .... ...1 = ESS capabilities: Transmitter is an AP
            .... ..0. = IBSS status: Transmitter belongs to a BSS
            ...1 .... = Privacy: AP/STA can support WEP
            ..0. .... = Short Preamble: Short preamble not allowed
            .0.. .... = PBCC: PBCC modulation not allowed
            0... .... = Channel Agility: Channel agility not in use
            CFP participation capabilities: No point coordinator at AP (0x0000)
        Listen Interval: 0x0001
    Tagged parameters (9 bytes)
        Tag Number: 0 (SSID parameter set)
        Tag length: 1
        Tag interpretation:
        Tag Number: 1 (Supported Rates)
        Tag length: 4
        Tag interpretation: Supported rates: 1.0 2.0 5.5 11.0 [Mbit/sec]
```

DStumbler supports the ability to passively detect networks using Prism II cards in RFMON mode, which does not leave a noticeable fingerprint.

Detecting Wellenreiter

- Name: Wellenreiter
- Download: http://www.remote-exploit.org/
- Source available: yes
- Discovery method: Passive RF Monitoring
- Features: GPS support, reports default ESSIDs, embedded statistics engine for collection of signal strength, packet counters, and other related information. Wellenreiter is also the only tool that employs an ESSID brute-force attack script.
- Supported chipsets: Lucent, Cisco, Prism II
- Supported platforms: Linux, experimental BSD
- Author: Max Moser

Wellenreiter is a Perl/Gtk+ application for use on Linux systems. Wellenreiter only supports network discovery through RFMON packet capture, but includes some additional features using a second 802.11 network card, including ESSID brute-forcing and automatic network association. It is through these features that we are able to identify Wellenreiter-generated traffic.

When an ESSID brute-force attack is initiated, Wellenreiter will use the Linux "iwconfig" program to temporarily set its ESSID to "this_is_used_for_wellenreiter". The MAC address will also be set to a random value (with a consistent leading “00” to avoid generating MAC addresses that might be confused as multicast traffic). The following code fragment is taken from Wellenreiter version 1.6:

```
system("$fromconf{iwpath} $fromconf{interface} essid
   'this_is_used_for_wellenreiter'");
system("$fromconf{ifconfig} $fromconf{interface} down");
my $brutessid = shift (@g_wordlist);
my $mactouse = build_a_fakemac;
system("$fromconf{ifpath} $fromconf{interface} hw ether $mactouse");
print STDOUT "\nI test now the essid: $brutessid";
system("$fromconf{iwpath} $fromconf{interface} essid $brutessid");
system("$fromconf{ifpath} $fromconf{interface} up");
return ($true);
```

This code will generate probe request frames using the fixed ESSID between attempts to guess the ESSID of a selected AP. We can identify this traffic with the follow Ethereal display filter:

```
wlan.fc eq 0x0040 and wlan_mgt.tag.number eq 0 and wlan_mgt.tag.length eq 29 and
       wlan_mgt.tag.interpretation eq “this_is_used_for_Wellenreiter”
```

The following detect was generated using Wellenreiter v1.6 on a Slackware Linux workstation using a stock 2.4.18 kernel:

```
IEEE 802.11
    Type/Subtype: Probe Request (4)
    Frame Control: 0x0040
        Version: 0
        Type: Management frame (0)
        Subtype: 4
        Flags: 0x0
            DS status: Not leaving DS or network is operating in AD-HOC mode (To DS: 0  From DS: 0)
(0x00)
            .... .0.. = Fragments: No fragments
            .... 0... = Retry: Frame is not being retransmitted
            ...0 .... = PWR MGT: STA will stay up
            ..0. .... = More Data: No data buffered
            .0.. .... = WEP flag: WEP is disabled
            0... .... = Order flag: Not strictly ordered
    Duration: 0
    Destination address: ff:ff:ff:ff:ff:ff (ff:ff:ff:ff:ff:ff)
    Source address: 00:40:96:47:e2:7d (00:40:96:47:e2:7d)
    BSS Id: ff:ff:ff:ff:ff:ff (ff:ff:ff:ff:ff:ff)
    Fragment number: 0
    Sequence number: 3849
IEEE 802.11 wireless LAN management frame
    Tagged parameters (37 bytes)
        Tag Number: 0 (SSID parameter set)
        Tag length: 29
        Tag interpretation: this_is_used_for_wellenreiter
        Tag Number: 1 (Supported Rates)
        Tag length: 4
        Tag interpretation: Supported rates: 1.0 2.0 5.5 11.0 [Mbit/sec]

0000  40 00 00 00 ff ff ff ff ff ff 00 40 96 47 e2 7d   @..........@.G.}
0010  ff ff ff ff ff ff 90 f0 00 1d 74 68 69 73 5f 69   ..........this_i
0020  73 5f 75 73 65 64 5f 66 6f 72 5f 77 65 6c 6c 65   s_used_for_welle
0030  6e 72 65 69 74 65 72 01 04 02 04 0b 16 ff ff ff   nreiter.........
0040  ff                                                .
```

Wellenreiter also uses randomized MAC addresses, in an effort to increase the level of anonymity for the attacker. Using a random MAC address does give the intrusion analyst an opportunity to detect “anomalous” MAC addresses, that is those addresses that utilize organizationally unique identifiers for the first three octets of the MAC address that are unallocated by IEEE standards body, or MAC addresses that are allocated, but not used for 802.11 network cards. In order to support this effort, the intrusion analyst needs a database of MAC OUI prefixes that are expected to be received on the wireless interface of an AP. Supporting this effort, Colin Grady has established a database at http://www.unbolted.net/ where users can submit their MAC address prefixes, manufacturer name and card type.

Since Wellenreiter network discovery is performed passively through the use of RFMON listening, we are unable to detect that portion of the application through layer 2 traffic analysis.

Detecting Windows XP

- Name: Windows XP, Microsoft Corp.
- Information: http://www.mirosoft.com/windowsxp/
- Source available: no
- Discovery method: Active Scanning/Probing
- Features: Support for wireless networking built into operating system, simple procedure for network scanning.
- Supported chipsets: Lucent, Cisco, Prism

Windows XP is the first Microsoft operating system to include built-in support for 802.11 wireless networking. One of the wireless service extensions in Windows XP is a network scanning service that a user can run to obtain a list of available SSIDs. Access to this service is available through the wireless control panel applet. A detection method for identifying Windows XP WLAN scanning is presented here for completeness.

Windows XP utilizes the active scanning method to discover available access points through the use of probe request frames with a broadcast SSID and a second unique SSID value. It is through this second SSID value that we can identify Windows XP network scanning.

In probe request frames, Windows XP will set a tagged parameter as a portion of the management frame using a length of 32 bytes. The tagged parameter is part of the "SSID Parameter Set" type, using a string of non-printable characters. This data string is presented in hexadecimal below:

```
0x14 0x09 0x03 0x11 0x04 0x11 0x09 0x0e
0x0d 0x0a 0x0e 0x19 0x02 0x17 0x19 0x02
0x14 0x1f 0x07 0x04 0x05 0x13 0x12 0x16
0x16 0x0a 0x01 0x0a 0x0e 0x1f 0x1c 0x12
```

As 32 bytes is the maximum size of the SSID field and due to the randomness of the string, I believe this traffic pattern is the likely result of a bug in the implementation of wireless networking drivers supplied with Windows XP. For this reason, I believe Microsoft will likely remove this unique signature from future Windows operating systems, possibly "fixing" this bug in later patches and releases of the Windows XP operating system.

We can use the following Ethereal display filter to identify Windows XP workstations scanning for wireless networks:

```
wlan.fc eq 0x0040 and wlan_mgt.tag.number eq 0 and wlan_mgt.tag.length eq 32 and
       wlan_mgt.tag.interpretation[0:4] eq 0c:15:0f:03
```

The following detect was generated using Windows XP, service pack 1:

```
IEEE 802.11
    Type/Subtype: Probe Request (4)
    Frame Control: 0x0040
        Version: 0
        Type: Management frame (0)
        Subtype: 4
        Flags: 0x0
            DS status: Not leaving DS or network is operating in AD-HOC mode (To DS: 0  From DS: 0)
(0x00)
            .... .0.. = More Fragments: This is the last fragment
            .... 0... = Retry: Frame is not being retransmitted
            ...0 .... = PWR MGT: STA will stay up
            ..0. .... = More Data: No data buffered
            .0.. .... = WEP flag: WEP is disabled
            0... .... = Order flag: Not strictly ordered
    Duration: 0
    Destination address: ff:ff:ff:ff:ff:ff (ff:ff:ff:ff:ff:ff)
    Source address: 00:60:1d:f0:91:68 (00:60:1d:f0:91:68)
    BSS Id: ff:ff:ff:ff:ff:ff (ff:ff:ff:ff:ff:ff)
    Fragment number: 0
    Sequence number: 20
IEEE 802.11 wireless LAN management frame
    Tagged parameters (40 bytes)
        Tag Number: 0 (SSID parameter set)
        Tag length: 32
        Tag interpretation:
\024\t\003\021\004\021\t\016\r\n\016\031\002\027\031\002\024\037\a\004\005\023\022\026\026\n\001\n\0
16\037\034\022
        Tag Number: 1 (Supported Rates)
        Tag length: 4
        Tag interpretation: Supported rates: 1.0 2.0 5.5 11.0 [Mbit/sec]
```

Conclusion

While it is possible to detect some wireless network discovery tools, intrusion detection monitoring on wireless networks is a difficult task. Few tools presently exist that are capable of analyzing layer 2 802.11 frames and even fewer that provide the free-form pattern matching that is a critical component to intrusion detection engines. The complexity of 802.11 intrusion detection does not reduce the need for such tools however; new methods of attack are rapidly introduced while the feature list of WLAN discovery application grows.

Establishing intrusion detection probes to cover all the serviceable areas of anything larger than the simplest wireless LAN will likely prove to be cost-prohibitive. A much more elegant solution would be to provide some intrusion analysis and reporting features in today's enterprise access points. With presumably simple software, the access point would be able to detect and report denial-of-service attacks such as (deassociate and associate floods), the presence of previously undetected access points (including fake advertisements generated by tools such as FakeAP), and the evidence of traffic signatures, such as those presented in this paper. It is the author's opinion that the adoption of such features would provide a significant advantage over competing products in the enterprise 802.11 marketplace.

Works Cited

ANSI/IEEE Standard, 802.11. *"Part 11: Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications."* 1999.

Craik. Craik, Mike. "All Your 802.11b Are Belong To Us (Netstumbler Signature)." Online Posting. 28 Mar 2002 02:07:48 +0000. Kismet Wireless list. URL: http://www.kismetwireless.net/cgi-bin/ezmlm-cgi?mss:366:eafojgdoalggkiopbclf (14 Oct. 2002).

Craik2. Craik, Mike. "RE: All Your 802.11b Are Belong To Us (Netstumbler Signature)." Online Posting. 28 Mar 2002 18:42:32 +0000. Kismet Wireless list. URL: http://www.kismetwireless.net/cgi-bin/ezmlm-cgi?mss:371:200203:eafojgdoalggkiopbclf (14 Oct. 2002).

Gast, Matthew S. *"802.11 Wireless Networks, The Definitive Guide."* Sebastopol, CA: O'Reilly & Associates, Inc., April 2002.

Ghandi. "Dope Squad Security¸ Viha Mac OS X Wireless Tools." URL: http://www.dopesquad.net/security/ (14 Oct 2002).