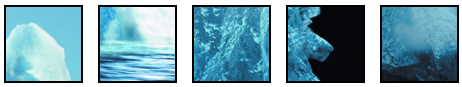

THE TELECOM VISION SUPPLIER

Informative article February, 2004.

Increasing security awareness: visualizing WEP insecurity.

Stijn HUYGHE (stijn.huyghe@thti.telindus.be).

The latest version of this article can be found on http://thti.telindus.be, section "spotted by thti".
Current version: 0304Ra – Telindus High-Tech Institute © 2004.

Introduction.

It is a simple fact; security awareness is a very important aspect of every security solution.
It is confirmed every single day – just take a look at the latest MyDoom infection rate.

Failure to give attention to the area of security (awareness) training puts an enterprise at risk because the security of the enterprise resources is as much a human issue as it is a technological issue.

Note: see NIST-800-500 [6].

Goal.

This article describes how one can setup and perform a small wireless demonstration that is quick and easy to perform with a good visual result to trigger the attention of your co-workers. The goal of the setup is to demonstrate a well-known WEP vulnerability (802.11b).

Note: in order to demonstrate the WEP vulnerability, we will use the OpenBSD [2] operating system. You don't need any prior OpenBSD knowledge, all the information you need to execute the demonstration is in here.

Note: this article will not describe the WEP vulnerabilities (again), if you want more information regarding this topic, see some added references at the end of the article [3].

Note: for this demonstration, you do not need hours of traffic capturing. The entire demonstration can be done within 30 minutes.

Equipment.

Material used during our example test:
- One 802.11b access point: ELSA ViaNect WLAN access point.
- One OpenBSD 3.4 host ("attacker"):
    - Compaq Armada 7790DMT (120Mhz CPU, 150 MB RAM).
    - Linksys WPC11 PCMCIA wireless network adaptor (PRISM3, firmware 1.1.0).
- One Compaq E500 with an ELSA WiFi wireless network adaptor, used to generate traffic on the wireless network. You can choose the operating system of this host, as long as it is able to join your WEP encrypted wireless network.

Note: for the demonstration it is advised to use equipment that sticks to the WiFi standard as some vendors have added (or are offering on their devices) some additional enhancements to increase the security of the 802.11b communication.

Note: this setup requires a PRISM based WiFi card in order to work with the “bsd-airtools” that we are going to use. See [1].

Lab setup.

As you are going to demonstrate WEP insecurity, you need to have a functional 802.11b network using WEP encryption. Make sure that your wireless network is fully operational (64 bit or 128 bit encryption).

The OpenBSD host is going to audit the wireless network, to capture interesting WEP encrypted packets and is going to perform a brute force attack on the smallest captured WEP encrypted packet.

The other member(s) of the wireless network are just going generate encrypted traffic that the OpenBSD host can capture and audit. To make the traffic generation easy; make sure that you have some (large) files that you can send over the wireless network.

Configuring and operating the OpenBSD host.

We suppose that you have already a working default installation of OpenBSD 3.4 on the attacking laptop. Installing OpenBSD takes about 15 minutes on a network with a fast internet connection and is quite easy. For a detailed step-by-step description of an OpenBSD installation see the OpenBSD website [2] or my other article on performing a basic OpenBSD installation [4].

Note: use the “man” pages to find more information about the utilities used.

Log in to your OpenBSD 3.4 system as root. Plug in your wireless card in the OpenBSD host. When everything goes ok, it should become available on “wi0”. Make sure that the card is up by using the command “ifconfig wi0 up”.
To verify use “ifconfig –a” and take a look in “/var/log/messages” - check if it is there.

“Bsd-airtools” from Dachb0den Labs is the package that we are going to use during the demonstration. It can be installed very easily as a package.

Example 1: to install the package from cdrom (you can buy it on http://www.openbsd.org) to support the project):

1. Create mount point: “mkdir /mnt/cdrom”.
2. Mount your disk: “mount /dev/cd0a /mnt/cdrom”.
3. Install the package: “pkg_add /mnt/cdrom/3.4/packages/i386/bsd-airtools-0.2p2.tgz”.

Example 2: to install the package from a FTP mirror:

1. Locate an OpenBSD FTP mirror, check http://www.openbsd.org/ftp.html.
2. Install the package: “pkg_add ftp://open.bsd.mirror/path/to/package.tgz”.

After the installation of the package, you should find “dstumbler”, “prism2ctl”, “dwepkeygen”, “dwepdump”, “prism2dump” and “dwepcrack” - that are all part of the bsd-airtools package - in “/usr/local/sbin” directory.

First we are going to enumerate available wireless access points in the neighborhood using “dstumbler” (a comparable tool for the Microsoft Windows platform is “netstumbler”) [5].
Go to “/usr/local/sbin” and launch dstumbler with the command “./dstumbler wi0” (with wi0 the wireless device that you are currently using).
For more dstumbler options, do simply “./dstumbler”.

When you have launched dstumbler, your card will start to enumerate wireless access points. Your access point should also light up. Information returned contains information like the SSID, channel number, signal/noise ratio etc. Make sure that you write down the channel number (channel “x”). Exit dstumbler (use “q” and confirm).

In order to sniff, we need to configure our wireless device (“wi0”) to operate in monitor mode.
To put the wireless card into monitor mode, do “./prism2ctl wi0 –m”.
To put the wireless card on channel “x”, do “wicontrol –F 10”.
Verify if your wireless card is in monitor mode, do “./prism2ctl wi0”. The response should contain a line: “Monitor mode: [on]”.

Try to sniff some wireless traffic to verify the monitor mode. Use “./prism2dump wi0 –v 2” which will start listening for wireless traffic and print them out in the console (“-v 2”: high verbose mode). Make sure that you are generating some traffic as well with the other wireless clients. You should see some traffic. You can see basic information like the SSID, rates and the channel number.

Now that we can sniff, we need to capture enough packets that we can feed later on into our WEP cracker (in fact: “dwepcrack” only performs a brute force crack on the smallest packet).
For the capturing of packets, we are going to use the “dwepdump” utility.
Launch “./dwepdump wi0 mycapturefile.dump” to start capturing packets.
Start generating wireless traffic using the other wireless clients. Capture around 10.000 packets (you can also execute the attack with much less packets – check the references).

Crack the WEP keys using the “dwepcrack” utility. We are going to use “dwepcrack” in brute force mode on the smallest packet - start dwepcrack with “./dwepcrack –b mycapturefile.dump”.
After a small minute, you should see the four WEP keys on your screen.

Verify these four WEP keys with the configured WEP keys on the encrypted wireless network!

Screenshots.

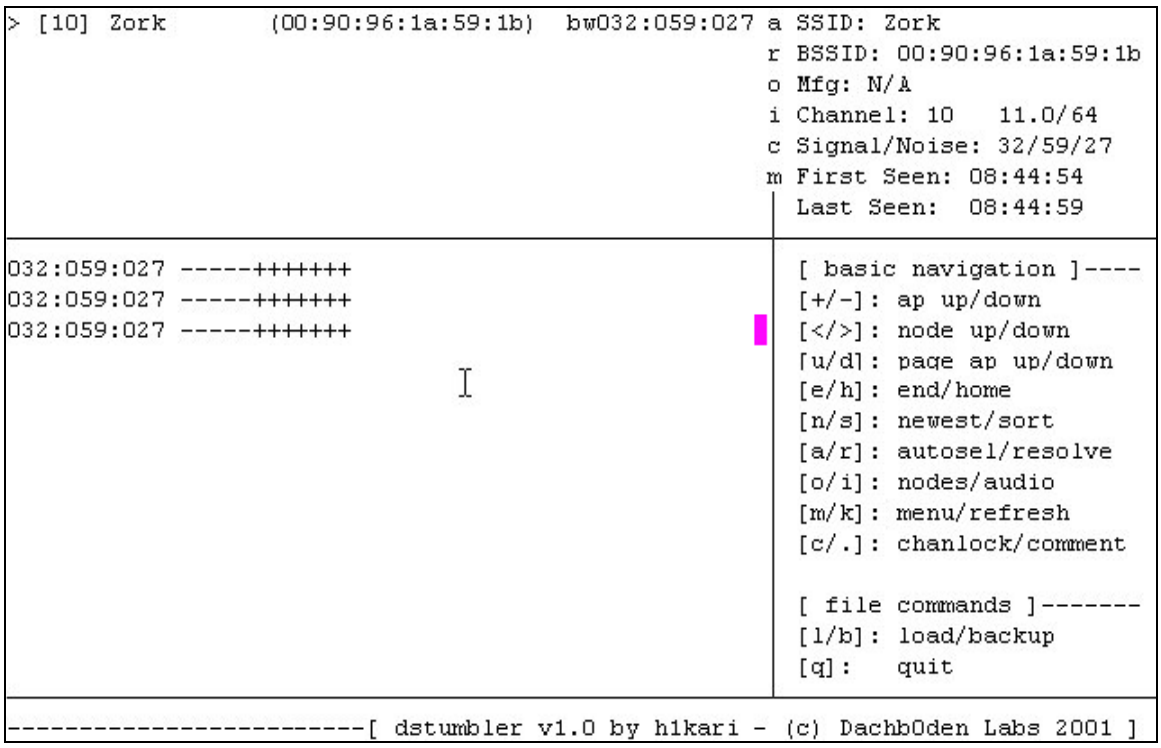

Figure 1 Dstumbler action.

```
- fctl: rtry sn: 7312 (31:d9:b3:73:4e:e5) len: 27
  - ** mgmt-proberesp ** ts: 2003.612784 int: 64 capinfo: ess priv
    + ssid: [Zork]
    + rates: 1.0 2.0 5.5 11.0
    + ds ch: 10

- [0:6:25:2a:56:b5 <- 0:90:96:1a:59:1B <- 0:90:96:1a:59:1b]
- port: 7 ts: 76.890018 12:53 20:0
- fctl: rtry sn: 7312 (b9:b6:11:ad:79:1f) len: 27
  - ** mgmt-proberesp ** ts: 2003.617563 int: 64 capinfo: ess priv
    + ssid: [Zork]
    + rates: 1.0 2.0 5.5 11.0
    + ds ch: 10

- [ff:ff:ff:ff:ff:ff <- 0:90:96:1a:59:1B <- 0:90:96:1a:59:1b]
- port: 7 ts: 76.905171 11:52 20:0
- sn: 7328 (f9:32:3e:c:80:58) len: 33
  - ** mgmt-beacon ** ts: 2003.632715 int: 64 capinfo: ess priv
    + ssid: [Zork]
    + rates: 1.0 2.0 5.5 11.0
    + ds ch: 10
    + dtim c: 0 p: 3 bc: 0 pvb: cfbef36d
```

Figure 2 Basic sniffing.

```
* dwepcrack v0.4 by h1kari <h1kari@dachb0den.com> *
* Copyright (c) Dachb0den Labs 2002 [http://dachb0den.com] *

starting brute force crack on smallest packet:
 packet length: 44
 init vector: 43:1d:00
 default tx key: 0

progress: ....

wep keys successfully cracked!
 0: b9:e4:4a:17:a4 *
 1: 53:b0:08:a6:0a
 2: 9b:5f:9b:83:66
 3: 9c:4c:74:c1:3c
done.
```

Figure 3 Dwepcrack in action.

References.

[1]. Bsd-airtools, Dachb0den Labs, http://www.dachb0den.com.
- Bsd-airtools supports the following *BSD operating systems: NetBSD 1.5.1+ (http://www.netbsd.org), OpenBSD 2.9+ (http://www.openbsd.org) and FreeBSD 4.4 (http://www.freebsd.org).
- Monitor mode supports currently only PRISM2 based cards. Some cards that have been reported to work with bsd-airtools: Addtron AWP-100, Bromax Freeport, Compaq WL100, D-Link DWL-650, GemTek (Taiwan) WL-211, Linksys WPC11, Samsung 2632W, Z-Com XI300, Zoom Telephonics ZoomAir 4100 and LeArtery Solutions SyncbyAir LN101.

[2]. OpenBSD, http://www.openbsd.org. A free, multi-platform 4.4 BSD-based UNIX-like operating system.

[3]. Some WiFi/WEP references:
- “Using the Fluhrer, Mantin, and Shamir Attack to Break WEP, revision 2”, AT&T Labs Technical Report TD-4ZCPZZ.
- “Security of the WEP algorithm”, Nikita Borisov, Ian Goldberg, David Wagner, http://www.isaac.cs.berkeley.edu/isaac/wep-faq.html.
- “Unsafe at Any Key Size: An Analysis of WEP Encapsulation”, Jesse Walker.
- “Practical Exploitation of RC4 Weaknesses in WEP Environments”, David Hulton, http://www.dachb0den.com/users/h1kari/work/docs/wepexp.txt.

[4]. “Informative article: installation of OpenBSD 3.3”, Huyghe Stijn, http://thti.telindus.be/news_thti/~en/news_thti.asp#04.

[5]. Netstumbler, http://www.netstumbler.com. A wardriving utility for Microsoft Windows.

[6]. NIST Special Publication 800-50, “Building an Information Technology Security Awareness and Training Program”, Mark Wilson and Joan Hash.