ECE/CS 4984: Wireless Networks and Mobile Systems
Laboratory 11Pre-lab and In-class Exercise

Part I – Objectives and Lab Materials

Objective:

- Observe security vulnerabilities in wireless local area networks (WLANs).
- Observe Denial of Service (DoS) attacks that target IEEE 802.11 WLANs.
- Observe the operation of an 802.11 WLAN detector, sniffer, and Intrusion Detection System (IDS).

Hardware to be used in this lab assignment:

- Dell notebook with 802.11b card (with a fully charged battery).
- Compaq iPAQ with a dual card sleeve and 802.11b card (with a fully charged battery).
- The GTA's Intel WLAN gateway.

Software to be used in this lab assignment:

- Kismet
- Xircom link status meter

Overview:

The purpose of this lab is to introduce some of the vulnerabilities of WLANs. WLANs face a number of challenges with respect to security. Among these challenges are the lack of a clear line of defense or any traffic concentration points, the broadcast nature of the transmission medium, the dynamically changing topology, the reliance on node collaboration as a key factor of network survivability, and the severely constrained computational and energy resources of some hosts. Such challenges leave such networks susceptible to a number of attacks ranging from brute-force authentication key cracking attacks to denial of service (DoS) attacks.

Part II – Pre-lab Assignment

This portion of the assignment should be completed *prior* to the in-class lab session.

- Make sure *ncurses* packages are installed on your notebook under Linux. You can test that by:

  > rpm -qa | grep ncurses

  If both packages were installed, you should get:

  ncurses-5.2-26
  ncurses-devel-5.2-26

- If the *ncurses-5.2-26* package is not installed proceed as follows:

a. Download the following file from Blackboard:

ncurses-5.2-26.i386.rpm

b. Test the rpm file before installing to make sure no dependencies or conflicts exist:

> rpm -i --test ncurses-5.2-26.i386.rpm

If any dependencies exist, contact your GTA immediately.

c. Install the file:

> rpm -i ncurses-5.2-26.i386.rpm

- If the *ncurses-devel-5.2-26* package is not installed proceed as follows:

a. Download the following file from Blackboard:

ncurses-devel-5.2-26.i386.rpm

b. Test the rpm file before installing to make sure no dependencies or conflicts exist:

> rpm -i --test ncurses-devel-5.2-26.i386.rpm

If any dependencies exist, contact your GTA immediately.

c. Install the file:

> rpm -i ncurses-devel-5.2-26.i386.rpm

- Download latest version of kismet from Blackboard
- Installing the kismet files:

> tar -xzvf kismet-feb.04.01.tar.gz
> ./configure
> make
> make suidinstall

- Read the man page for *ifconfig*. Note how to set the MAC address of an interface card.

Part III – In-class Lab Assignment

Part A: Spoofing the Intel gateway's IP address (ARP cache poisoning)

The goal of this experiment is to observe the effect of spoofing the IP address of a WLAN gateway. Upon spoofing the IP address of the Intel gateway by an attacker, WAN traffic will be maliciously forwarded to the attacker leading to failure of all communication attempts with nodes on the WAN.

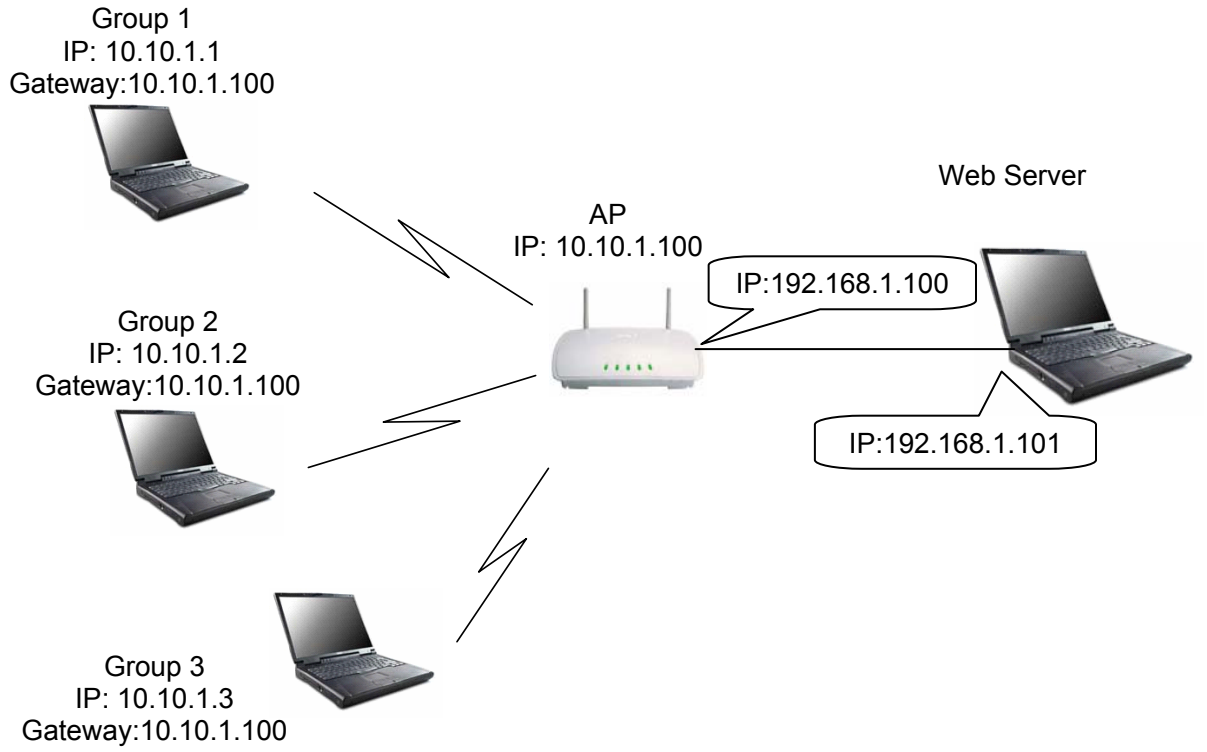

Figure 1. Network setup for experiment 1

Procedure:

- Boot your notebook in Linux with the 802.11b card in a PCMCIA slot. NVC groups should use *"WNMS"* as the ESSID. Groups in Blacksburg will be divided evenly into two networks. Groups in Network A should use *"WNMSA"* as the ESSID and groups in Network B should use *"WNMSB"* as the ESSID.
- Manually assign an IP address *10.10.1.{group number}* to your notebook. Make sure the subnet mask is *255.255.255.0* and the broadcast address is *10.10.1.255* using the following command:

  > *ifconfig eth1 netmask 255.255.255.0 broadcast 10.10.1.255*.

  Verify your connectivity to the network by pinging the Intel gateway at *10.10.1.100*.
- Make the access point (acting as a gateway) the default router by executing the following command:

> *route add default gw 10.10.1.100*

Test your connectivity to the WAN by accessing the web server at *192.168.1.101* through your web browser.

- All groups should start Ethereal and use menu "Capture->Start" to open the "Capture Options" dialog box. Select the 802.11b wireless interface (eth1). Disable the "**capture packets in promiscuous mode**", "**MAC name resolution**", and "**transport name resolution**" options. Enable the "**update list of packets in realtime**" option. Click "**OK**" to start tracing packets.
- Check the entries in your routing table by running the command:

  > *route –n*

  Capture a screenshot of the table and notice the entry for the default route.
- Check the entries in your ARP table by running the command:

  > *arp –n –a*

  Capture a screenshot of the table. Observe the MAC address of the default route noted above.
- The group with the lowest group id from each network (group 1 in *WNMS*, group 1 in *WNMSA*, and group 8 in *WNMSB*) will change their IP address to the IP address of the Intel gateway noted above. This can be done using the command:

  > *ifconfig eth1 10.10.1.100*

  This group will then ping the group with the next lowest id from the same network (group 2 in *WNMS*, group 2 in *WNMSA*, and group 9 in *WNMSB*).
- Check the entries in you ARP table again and capture a screenshot of the table. Observe the MAC address of the default route.
- Observe that the web server is not reachable and that any attempts to ping it or access it will timeout.
- Stop the Ethereal capture and save it.

Part B: Network sniffing, detection, and intrusion detection

Kismet is an 802.11 layer 2 wireless network detector, sniffer, and intrusion detection system. Kismet will work with any wireless card that supports raw monitoring (rfmon) mode, and can sniff 802.11b, 802.11a, and 802.11g traffic. Kismet identifies networks by passively collecting packets and detecting standard named networks, detecting hidden networks, and inferring the presence of nonbeaconing networks via data traffic. We will explore how kismet detects a number of intrusion attempts.

Procedure:

- This section assumes the completion of the pre-lab, which involves installing and running kismet in client mode. If you have any trouble installing Kismet contact your GTA as soon as possible, preferably before the lab.

- Update the following parameters in the kismet config files:

    a. kismet.conf located in */usr/local/etc*

    Open the file kismet.conf in a text editor like vi or kedit, and make sure the value of the active *source* parameter reflects the driver used. In our case, there should be a line that says something like:

    source=cisco,eth1,ciscosource

    which identifies the driver, the interface, and the name of the source.

    b. kismet_ui.conf located in */usr/local/etc*

    Open the file kismet_ui.conf in a text editor like vi or kedit, and change the value of *host* to the IP address and port number given to you by the GTA

    host=IP_ADDRESS:PORT_NUMBER

- Set up an infrastructure network consisting of notebooks and the Intel WLAN gateway setup by the GTA (ESSID: *WNMS* in NOVA and *WNMSA* in Blacksburg). Manually assign an IP address *10.10.1.{group number}* to your notebook. Make sure the subnet mask is *255.255.255.0* and the broadcast address is *10.10.1.255* using the following command:

    > *ifconfig eth1 netmask 255.255.255.0 broadcast 10.10.1.255*.

    Verify your connectivity to the network by pinging the Intel gateway at *10.10.1.100*.

- Groups 1 and 2 will initiate an *iperf* session. One notebook should run *iperf* as a server and the other should run *iperf* as a client. Configure and run the *iperf* client and the *iperf* server to transmit and receive UDP datagrams for 10 minutes (600 seconds).

- All other groups should run *kismet_client*:

    > *kismet_client*

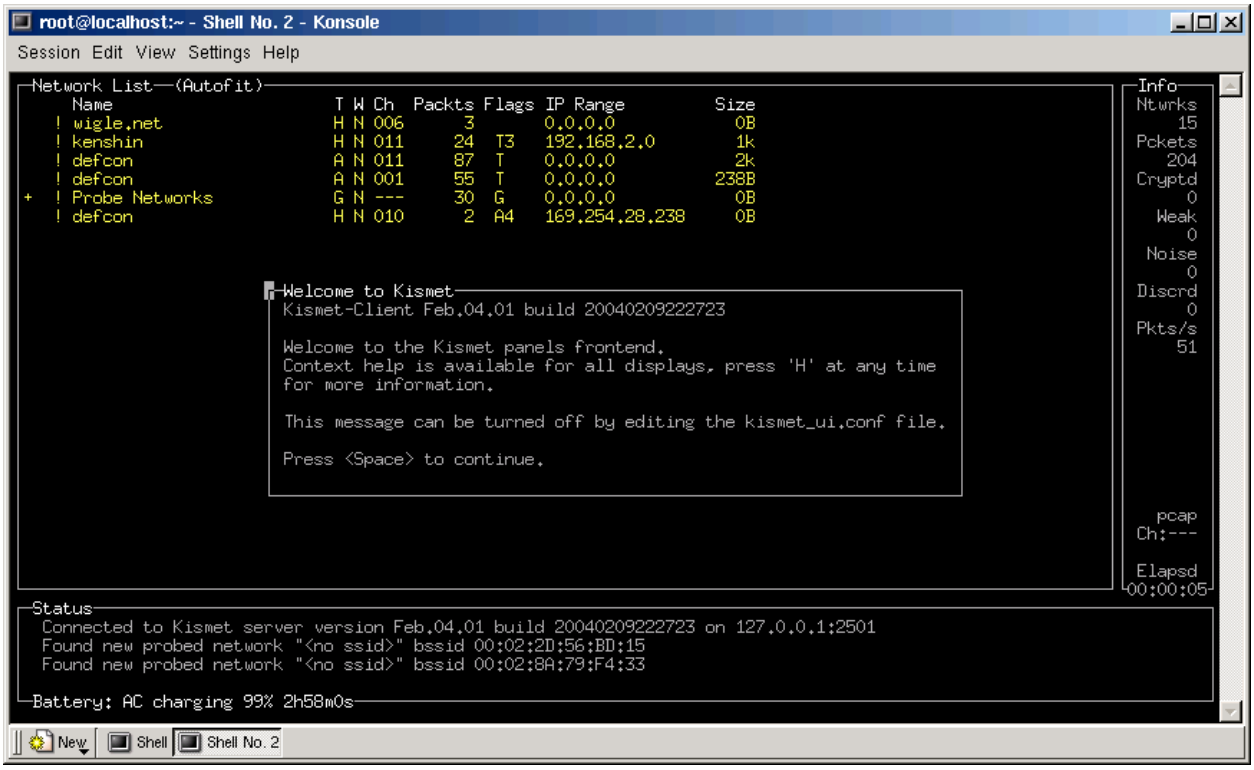

Figure 1. Kismet_client running.

- This is a brief list of commands supported by kismet:

  “h” pops up help menu

  “q” closes the current pop up menu

  “Q” quits kismet_client

  “s” pops up sort menu (choose anything other than Auto-fit)

  “w” pops up the alert menu (IDS)

  “L” locks the sniffer to the current network channel

  “H” hops the sniffer to different channels

- Change the sort value:

  a. Press the “s” key to pop up the sort menu

  b. Press the “s” key again to sort by SSID

  c. Press the “q” key to close the pop up menu

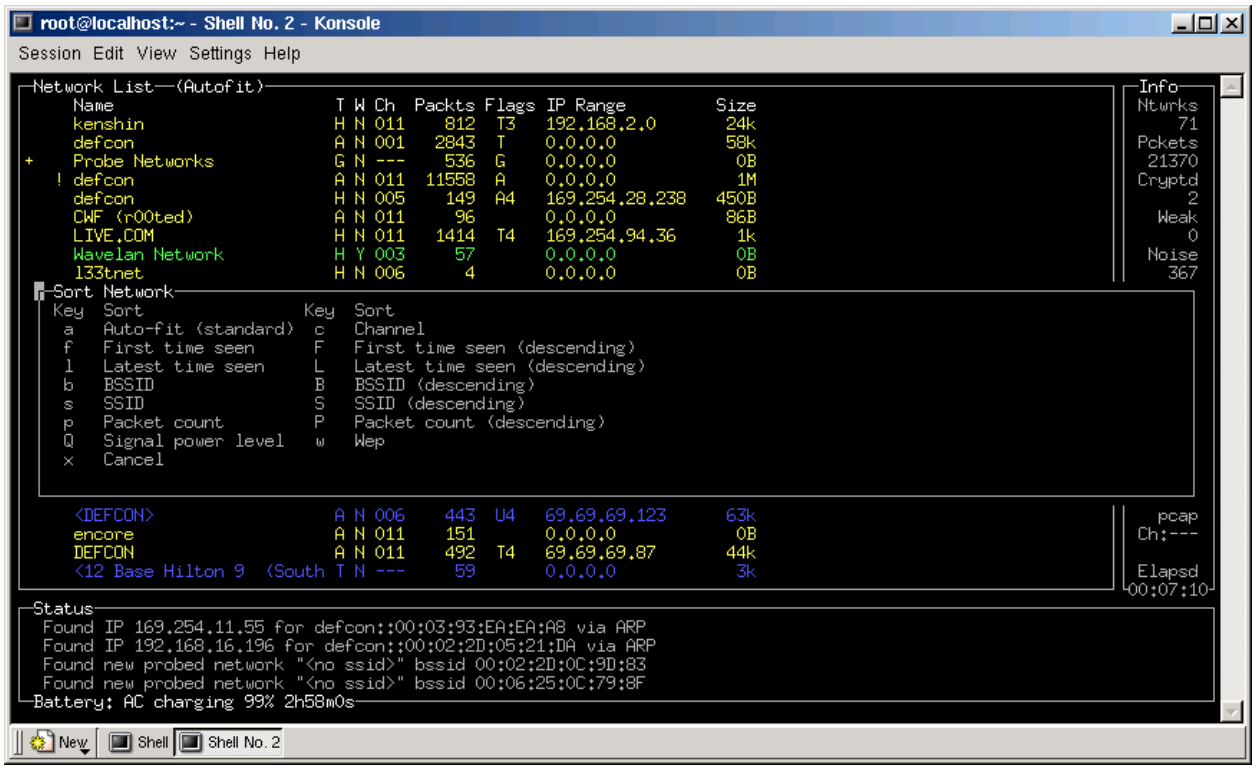

Figure 2. Sorting networks display on *kismet_client*.

- Kismet will detect the network set up by the GTA and any other WLANs within range. Packets captured by kismet are saved periodically to the directory where kismet_client was executed. Examine these files (*.dump) by opening one or two using Ethereal.
- Notice that the GTA's notebook, which is running *kismet_server*, is able to capture traffic even though it is not connected to any WLAN.

Part C: Impersonating an access point

The GTA will run *hostAP* on his notebook, which is a driver that enables the notebook to run as an access point. The GTA will impersonate the Intel WLAN gateway by assigning the notebook the same BSSID (MAC Address), ESSID, IP Address, and the same channel of operation of the gateway. The attack takes advantage of the unauthenticated management frames vulnerability in 802.11 by sending disassociation frames to nodes in the network. To these nodes, the GTA's notebook looks no different from the access point. Thus, nodes will believe they were disassociated from the network, but will then attempt to reassociate.

Procedure:

- Connect iPAQs to the infrastructure network that was setup in the previous experiment. You are to configure your iPAQ to associate with the Intel WLAN gateway and manually assign an IP address 10.10.1.{*groupnumber + 20*}. On the iPAQs, the IP address obtained can be determined by using vxUtil. Ping the Intel WLAN gateway *{10.10.1.100}* to make sure you are connected. Keep kismet_client running on the notebooks.

- Observe the signal strength and the status of the iPAQ through the Xircom Link Status meter in Start>Programs>Xircom.
- The GTA will perform a DoS attack on all nodes by impersonating the access point with a notebook and sending disassociation frames. Observe the link status meter falling to zero and the status of a node changing to "unassociated."
- The IDS of kismet should raise an alert about "Broadcast Deauthentication" and "Broadcast Deauthentication/Disassociation flooding" originating from the rogue access point.
- Capture a screenshot of the Kismet client interface running on your notebook showing the alerts.