Detecting and Blocking Unauthorized Access in Wi-Fi Networks

Haidong Xia and José Brustoloni*

University of Pittsburgh, Dept. Computer Science, 210 S. Bouquet St. #6135, Pittsburgh, PA 15260, USA,
{hdxia,jcb}@cs.pitt.edu

Abstract. Academic and commercial 802.11 hotspots often use an SSL-secured captive portal to authenticate clients. Captive portals provide good usability and interoperability, but poor security. After a captive portal has authenticated a client, session hijacking and freeloading allow attackers to capture or use the client's session. Freeloading does not require special tools and, surprisingly, is strengthened by the (widely recommended) use of personal firewalls. We propose and evaluate novel defenses against these attacks, session id checking and MAC sequence number tracking, both of which are transparent to clients and do not require changes in client computers. Experiments demonstrate that the proposed defenses are effective against the mentioned attacks and have little overhead.

1 Introduction

Wi-Fi networks [1,2] provide an unprecedented combination of low cost, high bandwidth, and support for mobility. Consequently, they have become extremely popular, as evidenced by the recent 42%-a-year growth rate of the Wi-Fi chipset market [3]. Wi-Fi networks do have, however, a major weakness: poor security. Wi-Fi's original security scheme, WEP (Wired Equivalent Privacy), has been demonstrated to be easily broken [4,5,6,7,8].

Security has to be balanced with other requirements, such as usability and interoperability. In Wi-Fi networks, the relative importance of these requirements varies with application, as summarized in Table 1. *Enterprise* networks are installed in companies or government offices and therefore require a high level of security, with mutual authentication between mobile stations and network at connection time and encryption and authentication of all user traffic after that. On the other hand, an enterprise usually can provide technical support to users and owns all equipment connected to its network. Consequently, security solutions that end users find difficult to install or configure or that are proprietary can be used in such networks. *Home* networks differ from enterprise networks in

* This project was funded in part by the Pittsburgh Digital Greenhouse through a grant from the Commonwealth of Pennsylvania, Department of Community and Economic Development.

N. Mitrou et al. (Eds.): NETWORKING 2004, LNCS 3042, pp. 795–806, 2004.

Table 1. The relative importance of security, usability, and interoperability in Wi-Fi networks varies according to application. In some cases, users prize usability and interoperability more than security.

Wi-Fi Application	Requirement		
	Security	Usability	Interoperability
Enterprise	High	Medium	Medium
Home	High	High	Medium
Access	Medium	High	High
Open	Low	High	High

that they need to operate with little or no technical support. Therefore, security solutions for such networks have to be easy to install, configure, and use. *Access* networks using Wi-Fi are often deployed in locations such as universities and commercial hotspots. Their primary purpose is to provide Internet connectivity to users who are away from the respective offices or homes. Access networks typically need to block unauthorized users, while interoperating with a wide variety of user-owned equipment and providing little or no on-site technical support. Access networks usually do not attempt to secure user communication. At least in principle, local security would be either redundant, if the user uses end-to-end security protocols, such as IPsec [9] or SSL [10], or insufficient, if the user does not use such protocols (local security cannot prevent attacks from occurring elsewhere on the user's communication paths). *Open* networks need to interoperate with other equipment quickly, with minimal or no configuration or technical support, but do not provide any security. In theory, this mode exists only for initial equipment configuration and testing. However, nearly 70% of existing production Wi-Fi networks actually operate in this mode [11]. This fact is contrary to what a designer might imagine or prefer, and shows that users do, in many cases, prize ease-of-use and interoperability more than security.

There currently isn't a scheme that simultaneously provides high security, usability, and interoperability in Wi-Fi networks. Several solutions exist or are being developed for securing enterprise Wi-Fi networks, including IPsec, WPA (Wi-Fi Protected Access) [2], IEEE 802.11i [12], and a variety of proprietary alternatives. However, these solutions currently involve installation, configuration, or interoperation difficulties that make them poorly suited for home or access networks.

Access Wi-Fi networks typically use MAC address filtering or captive portals to authenticate users, as explained in Section 2. These schemes are easy to understand and use and do not require special client hardware or software. However, they can be defeated by MAC address spoofing, session hijacking, or freeloading attacks, as explained in Sections 2, 3, and 4. The latter attack was previously unreported. It requires no special tools, is strengthened by the (widely recommended) use of personal firewalls, and can easily go undetected.

This paper contributes novel mechanisms for detecting and blocking session hijacking and freeloading attacks in access Wi-Fi networks. The proposed mech-

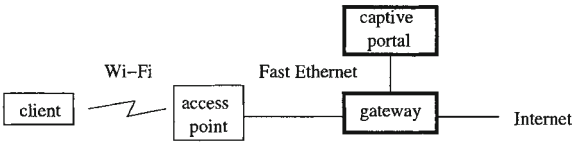

Fig. 1. Captive portals are widely used for user authentication at universities and commercial hotspots, but they are vulnerable to session hijacking and freeloading attacks.

anisms, session id checking and MAC sequence number tracking, are transparent to users and work well with default hardware and software configurations, which makes them well-suited for access applications. Experiments demonstrate that the proposed defenses are effective and impose little overhead.

The rest of this paper is organized as follows. Section 2 discusses previously proposed security schemes for Wi-Fi networks. Section 3 describes session hijacking attacks and a novel defense against them, session id checking. Session 4 characterizes freeloading attacks and presents MAC sequence number tracking, a novel defense against them. The new defenses are evaluated experimentally in Section 5 and discussed in Section 6. Finally, Section 7 concludes.

2 Related Work

MAC address filtering is a simple solution for authenticating users in an access Wi-Fi network. It consists in configuring access points so that they accept association only of computers using certain MAC addresses. Most access points allow this type of configuration, and no changes are necessary in client computers.

Unfortunately, this solution is very insecure. Attackers can simply sniff the network to find the MAC address of an approved computer. Applications for such sniffing are freely available from the Internet (e.g., ethereal). Attackers can then spoof their own MAC address to be that of an approved computer. In Windows, several Wi-Fi drivers support MAC address spoofing from the control panel (e.g., Dell TrueMobile). Even if the driver does not support it, spoofing can be performed using the registry; an application that automates the process (smac) is freely available from the Internet. In Linux, MAC address spoofing is enabled by the built-in command `ifconfig ethXX hw ether XX:XX:XX:XX:XX:XX`.

Many networks combine MAC address filtering with suppression of access points' 802.11 beacon messages. This technique has limited value because user network traffic can still be picked up by sniffers. Some networks combine MAC address filtering with WEP (using static keys). This technique also has limited value, since attackers often can obtain the key by social engineering or by using freely available tools, such as airsnort or WEPcrack.

Captive portals were first proposed in Stanford's SPINACH project [13]. They are widely used for user authentication in access Wi-Fi networks at universities and commercial hotspots. Captive portals require a special gateway between the Wi-Fi network and the rest of the network, as depicted in Figure 1. The

gateway allows unauthorized clients to use DHCP, ARP, and DNS for initial configuration. The gateway also redirects any Web requests from unauthorized clients to the network's captive portal. The captive portal is an SSL-secured Web page where unauthorized users enter their id and password. The captive portal may use a variety of back-ends for user authentication, e.g. Kerberos, RADIUS, or LDAP. The gateway drops any other traffic of unauthorized clients. After authentication, the captive portal authorizes the client to access the rest of the network. Authorization is done by registering the client's MAC and IP addresses in the gateway. The captive portal typically also sends the client a *session management page* that contains a button for terminating the session. This page is usually displayed on a small pop-up window that is not used for browsing. Finally, the captive portal redirects the client to the page that the client had initially requested.

Captive portals do not require special hardware or software configuration in client computers: most contemporary computers already have a Web browser. The Web-based interface is also very intuitive. Captive portals are significantly more secure than is MAC address filtering because they resist simple MAC address spoofing attacks. However, captive portals are still vulnerable to session hijacking and freeloading attacks, as discussed in Sections 3 and 4.

IPsec was proposed for securing access Wi-Fi networks in [14]. However, IPsec configuration and interoperation continue to be problematic. Recent Microsoft operating systems include a VPN (Virtual Private Network) client that implements IPsec in a way that does not support nested secure connections (L2TP over IPsec transport mode). If this VPN client is used for local security, it is not available for secure end-to-end connections, which is unacceptable. Third-party IPsec clients can be used, but IPsec's excessive configuration options can be daunting. Moreover, IPsec currently omits several details that are needed for mobile access, such as methods for obtaining networking configuration parameters from a VPN gateway and support for legacy user authentication methods. These omissions are often supplanted by non-standard drafts or proprietary extensions. IKEv2 is a future version of IPsec's IKE protocol that hopefully will resolve interoperability problems. However, it will not address configuration problems, which continue to hamper the use of IPsec in access networks.

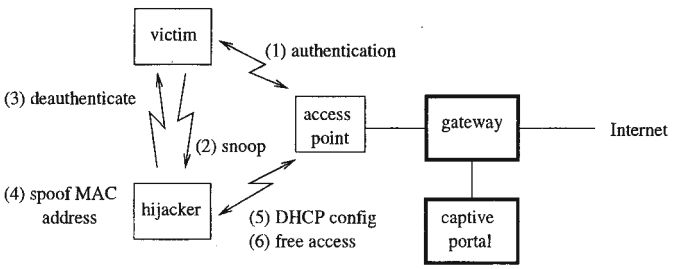

Fig. 2. Low-level 802.11 vulnerabilities allow a hijacker to gain control of a session of an authorized user.

PANS is a security scheme for access Wi-Fi networks that provides high security [15]. However, it requires installation of non-standard software in client computers. Non-standard schemes that provide high security are also available from several vendors, including Cisco (LEAP). Proprietary schemes can be easier to install and configure than is IPsec. However, it is highly desirable that access networks be able to interoperate with a wide range of client equipment; proprietary solutions preclude that, unless they become de facto standards.

IEEE 802.1x [16] is enabling significant improvements in native Wi-Fi security, including WEP with dynamic per-session keys (not a standard, but supported by Microsoft), WPA, and IEEE 802.11i. There are significant difficulties for adopting such schemes in access Wi-Fi networks. First, many details of these solutions have not yet been fully worked out and continue to change. Second, these solutions require the installation of new software, firmware, and possibly hardware in client computers, and these modifications need to implement the same draft versions as those adopted in the network's access points and authentication servers. Third, ongoing improvements are creating a multitude of configuration options that could be confusing for end users. For example, a variety of schemes may be used for initial authentication (e.g., pre-shared keys, EAP-TLS, or PEAP); if PEAP is used, many legacy user authentication schemes are possible (e.g., using passwords, one-time passwords, or tokens); and several packet encryption algorithms will soon be in use (e.g., WEP with dynamic keys, TKIP [2], and CCMP [12]). Resolution of these difficulties requires technical support at a level that typically is not offered in access Wi-Fi networks.

3 Session Hijacking and Session Id Checking

This section describes session hijacking and a new defense against it, session id checking.

Session hijacking is illustrated in Figure 2. The hijacker snoops on the victim's and the access point's MAC addresses. The hijacker then periodically sends to the victim disassociation or deauthentication notifications purported to come from the access point. According to the IEEE 802.11 standard, these notifications are not authenticated and must be obeyed. They cause denial of service to the victim. (The notifications need to be periodically repeated because the victim typically attempts to reauthenticate some time after the last notification.) The hijacker then spoofs his or her MAC address to be the same as the victim's and obtains the victim's networking configuration from the DHCP server. (Alternatively, the hijacker gleans this information by sniffing network traffic beforehand.) Using the victim's MAC and IP addresses and other configuration parameters, the hijacker can then access the rest of the network without being authorized to do so.

Session id checking detects and blocks session hijacking as follows. Captive portals usually send to authorized clients a session management page on a small pop-up window. Session id checking (1) associates with this page a secure non-persistent cookie containing a cryptographically random session id of

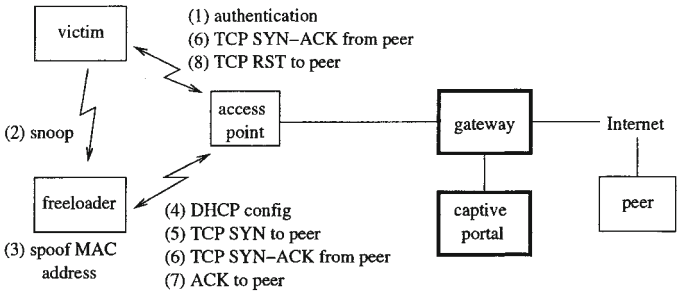

Fig. 3. A freeloader simply uses the MAC and IP addresses of an authorized victim. The victim also receives packets destined to the freeloader. These packets may elicit responses from the victim that disrupt the freeloader's communication, as shown. However, personal firewalls inhibit such responses and enable both freeloader and victim to communicate reliably.

sufficient length (e.g., 128 or more bits); (2) tags the page with the directive `http-equiv="refresh"` and a certain period; and (3) secures the page with SSL.

The refresh directive periodically causes the client's browser to request the captive portal to retransmit the page. Because the page has an associated cookie and is SSL-secured, each such request is accompanied by the cookie containing the client's session id and is also SSL-secured.

The captive portal detects that an authorized client's session has been hijacked when, after an entire period, the captive portal has not received a refresh request with the client's IP address and session id. A hijacker cannot guess the victim's session id because it is cryptographically random, and cannot capture it by eavesdropping because its transmission is SSL-secured. The captive portal then blocks the hijacker's communication by unregistering the victim's MAC and IP addresses in the gateway between the Wi-Fi network and the rest of the network.

4 Freeloading and MAC Sequence Number Tracking

This section describes freeloading and a new defense against it, MAC sequence number tracking.

The freeloading attack consists in spoofing a victim's MAC and IP addresses while the victim remains associated and communicating normally. In principle, freeloading would be expected to perform unreliably, because the victim also receives packets destined to the freeloader and may react in ways that disrupt the freeloader's communication. This problem is illustrated in Figure 3. In this example, the freeloader sniffs the MAC address of an authorized victim and spoofs the freeloader's MAC address to be the same as the victim's. The freeloader then obtains the rest of the victim's networking configuration from the DHCP

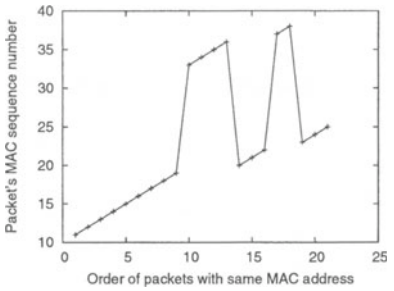

Fig. 4. Freeloading causes interleaving of different senders' MAC sequence number trend lines

server (alternatively, the freeloader gleans this information by sniffing the network beforehand). After that, the freeloader opens a TCP connection with the usual three-way handshake (send a SYN segment, receive a SYN-ACK segment, and send an ACK segment). However, the victim also receives the SYN-ACK segment destined to the freeloader. Given that the victim does not know about this connection, the victim, following the TCP standard, sends a RST segment to the segment's source (i.e., the freeloader's peer). This aborts the freeloader's connection.

If the victim and freeloader use personal firewalls, however, freeloading works reliably. Personal firewalls are widely recommended and quite common. For example, NIST recommends that all U.S. government employees use a personal firewall when on a Wi-Fi network [17]; the trade press makes similar recommendations to business users; and newer operating systems, such as Windows XP, often have a built-in personal firewall. A personal firewall typically allows the respective computer to respond only to packets that the firewall recognizes as part of a session initiated by that computer. Personal firewalls usually interpret other packets as attempts to fingerprint the respective computer's software or find vulnerable ports. Consequently, personal firewalls typically inhibit responses to packets that they do not recognize. In a victim computer, such packets include those that are actually destined to a freeloader. Thus, the victim and freeloader do not respond to or interfere with each other's communication. Because both freeloader and victim get access, freeloading may occur in collusion against the network's owner (e.g., commercial hotspot).

MAC sequence number tracking detects freeloading by observing that IEEE 802.11 frames contain a 12-bit sequence number that should increment by one for each new datagram sent, and remain the same in case of MAC-layer fragmentation or retransmission. Because of the tight timing constraints of MAC-layer acknowledgements and retransmissions, 802.11 sequence numbers are set and verified by network interface card (NIC) hardware or firmware. Host software typically cannot set such numbers.

Consequently, an access point or other device can detect freeloading by noticing that the MAC sequence numbers of successive packets with the same MAC

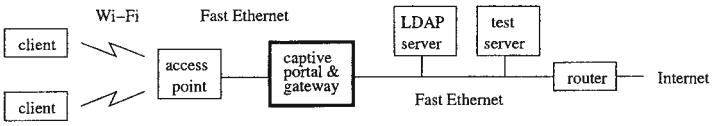

Fig. 5. Testbed used in the experiments

address form more than one trend line, as illustrated in Figure 4. One of the trend lines corresponds to the authorized client, while other trend lines correspond to freeloaders. The detecting device blocks freeloaders' communication by unregistering the victim's MAC address in the gateway between Wi-Fi network and the rest of the network.

Note that a simple jump in MAC sequence number is not a good criterion for detecting freeloading. Simple jumps occur also for other reasons, e.g. because a client has moved out of range and then back in range, or interference has caused several datagrams to be lost. For robustness, we detect freeloading when the MAC sequence number *returns* from one trend line to the previous trend line.

5 Experimental Evaluation

We performed experiments for evaluating the proposed defenses and report in this section the results obtained.

We used the testbed shown in Fig. 5. The access point is an IBM T30 1.8 GHz PC with built-in Intersil Prism 2.5-based 802.11b interface, running Linux 2.4.20 and HostAP driver. We modified HostAP to support MAC sequence number tracking. The captive portal/gateway and LDAP and test servers are Dell Dimension 4550 2.4 GHz PCs running Linux 2.4.20. We implemented session id checking on the captive portal, which also uses the Apache Web server. The LDAP server uses OpenLDAP. As clients, we used IBM T30, Dell, and Sony laptops, 5 Sharp Zaurus PDAs, and 10 Dell Dimension 8300 2.6 GHz PCs with a variety of Wi-Fi network interface cards (NICs), including Linksys, Netgear, D-Link, Proxim Orinoco Gold, and Cisco Aironet 350.

We tested session hijacking using a modified version of the airjack toolset, which is freely available from the Internet. We measured throughput using the ttcp benchmarking application between the test server and a test client. Delay was measured by pinging the test server from a test client and capturing packet times using Ethereal on the Wi-Fi network. Reported results are the average of five tries.

In the first experiment, we had one of the test clients hijack the session of another test client, after the captive portal authorized the latter client's access. We verified that session id checking promptly detects and blocks session hijacking. We varied the refresh period of the session management page between 1 s and 10 s, and verified that this period controls the latency for detecting and blocking a hijacked session.

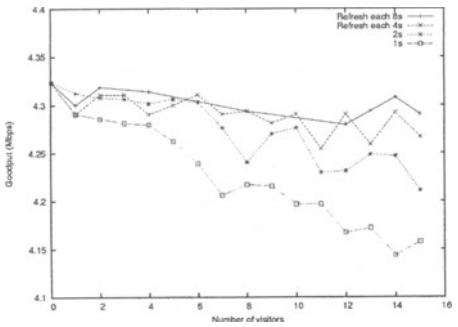

Fig. 6. Session id checking's impact on network throughput is quite low, especially at longer refresh periods

In the second experiment, we measured the throughput between the test server and a test client as the number of authorized clients varied, using session id checking but not MAC sequence number tracking, and for several different refresh periods. The results are shown in Figure 6. The curves show that session id checking decreases the network's throughput roughly in proportion to the number of authorized clients. This result is as expected, since each authorized client adds the same amount of overhead traffic for refreshing the session management page. The curves also show that the overhead increases as the refresh period decreases. This result is also expected, since a smaller refresh period requires the same amount of refresh traffic to occur in a shorter time interval. With 15 authorized clients and 1 s refresh period, the measured network throughput overhead of session id checking was approximately 4%.

In the third experiment, we measured the CPU utilization of the captive portal/gateway under the same conditions as the previous experiment, with a 1 s refresh period. The results are shown in Figure 7. The curve shows that the CPU overhead of session id checking increases roughly in proportion to the number of authorized clients, as expected. With 15 authorized clients and 1 s refresh period, the measured CPU overhead was approximately 5%.

In the fourth experiment, we measured the round-trip time (RTT) between a test client and the test server, under the same conditions as the previous experiment. The RTT remained steady at an average of 2.4 ms with standard deviation of 0.3 ms as the number of associated clients varied between 1 and 22. The same RTT was measured without session id checking. This shows that session id checking introduces negligible delay.

In the fifth experiment, we had one of the test clients freeload on the session of another test client, after the captive portal authorized the latter's access. We verified that MAC sequence number tracking detects and blocks freeloading as soon after the attack starts as the authorized client sends another packet.

In the sixth experiment, we measured the throughput between the test server and a test client, using MAC sequence number tracking but not session id check-

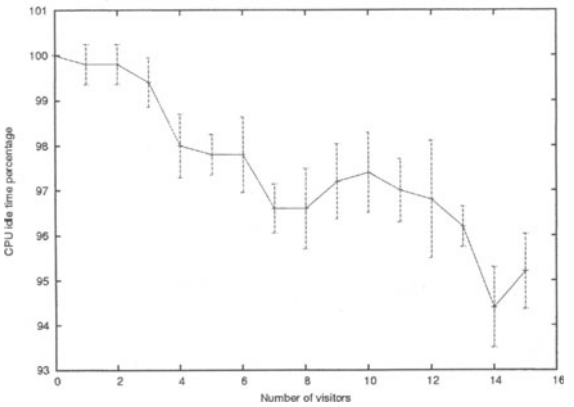

Fig. 7. Session id checking's impact on the captive portal's CPU utilization increases roughly proportionally to the number of authorized users (1 s refresh rate shown)

ing. The throughput remained steady at 4.42 Mbps with a standard deviation of 0.06 Mbps as the number of authorized clients varied between 1 and 8. These measurements show that MAC sequence number tracking has negligible impact on network throughput. This is expected, since this defense is passive and does not add traffic to the Wi-Fi network.

In the seventh experiment, we measured the round-trip time (RTT) between a test client and the test server, under the same conditions as the previous experiment. The RTT remained steady at an average of 2.4 ms with standard deviation of 0.3 ms as the number of associated clients varied between 1 and 8. The same RTT was measured without MAC sequence number tracking. This shows that MAC sequence number tracking introduces negligible delay.

In the eighth and final experiment, we used both session id checking and MAC sequence number tracking. We verified that the defenses interoperate well and continue to be effective against the mentioned attacks when used together. We also verified that they work well with all the Wi-Fi NICs mentioned above.

6 Discussion

It should be noted that neither of the proposed defenses can replace the other. Session id checking does not detect freeloading because the latter attack, unlike session hijacking, allows the victim to continue communicating and refreshing the session management page. Conversely, MAC sequence number tracking does not detect session hijacking because the latter attack, unlike freeloading, causes a simple jump in the MAC sequence number, without clearly delineating different trend lines.

The network throughput and CPU overheads measured in the experiments suggest that session id checking would scale, as shown, up to about 200 authorized users. In larger installations, it would probably be necessary to partition

access points among several captive portals, or migrate session id checking to the access points, in order to handle the session id checking load.

Freeloading is ordinarily an access control violation. MAC sequence number tracking converts it into a denial-of-service attack. If access control is desired, this can be considered a good tradeoff: Much is gained by blocking unauthorized access and, given that Wi-Fi already has numerous other denial-of-service vulnerabilities for which no good defense is known (e.g., jamming Wi-Fi frequencies or broadcasting deauthentication notifications), little is lost by introducing yet another way to achieve denial-of-service. On the other hand, in networks that do not need access control, the proposed defenses probably should be disabled.

An attacker could attempt to thwart the proposed defenses by jumping from one victim to another as soon as the respective attack is detected. It is doubtful that such a strategy would be practical. First, the attacker would quickly run out of potential victims, given Wi-Fi's limited range and the typically low utilization of access Wi-Fi networks. Second, it would be difficult for such an attack to remain unnoticed, given the many victims that would suffer denial of service. Third, the attacker's IP address would keep quickly changing as the attacker jumps to each victim. It would therefore be difficult or impossible for the attacker to maintain connections, especially secure ones.

The proposed defenses do not detect attacks using rogue access points. However, there are several intrusion detection systems that can be used for such purpose and can be expected to interoperate well with the defenses proposed here. Commercially available examples of such systems include AirWave, Wavelink, AirMagnet, and AirDefense, among others.

7 Conclusions

Access Wi-Fi networks provide Internet connectivity to mobile users at locations such as universities and commercial hotspots. They need to block access by unauthorized users and to interoperate with a wide range of user-owned equipment, but usually cannot provide users much technical support. As a result of these usability and interoperability requirements, access Wi-Fi networks often forego security improvements that are being used in other applications (e.g., enterprise networks). Access Wi-Fi networks typically use a captive portal to authenticate users. Captive portals are intuitive and do not require special hardware or software. However, they are vulnerable to session hijacking and freeloading attacks. Freeloading is a previously unreported attack. It does not require special tools and can easily remain undetected. We proposed two novel defenses, session id checking and MAC sequence number tracking, against captive portals' vulnerabilities. These defenses are transparent to users and work with default client hardware and software configurations, and therefore are well-suited for use in access Wi-Fi networks. Our experiments show that the proposed defenses are effective against the mentioned attacks and impose modest overhead.

References

1. IEEE: Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications. 802.11 Std. (1999) [Online] http://standards.ieee.org/getieee802/download/802.11-1999.pdf
2. Wi-Fi Alliance: [Online] http://www.weca.net
3. Mackie, K.: Report Profiles Growth in Wi-Fi IC Shipments. Broadband Wireless Online, Dec. 19 (2002) [Online] http://www.shorecliffcommunications.com/magazine/news.asp?news=1105
4. Borisov, N., Goldberg, I., Wagner, D.: Intercepting Mobile Communications: The Insecurity of 802.11. In: Proc. Seventh Annual International Conference on Mobile Computing and Networking (MOBICOM), ACM (2001) 180–188
5. Fluhrer, S., Mantin, I., Shamir, A.: Weaknesses in the Key Scheduling Algorithm of RC4. In: Eighth Annual Workshop on Selected Areas in Cryptography. (2001)
6. Stubblefield, A., Ioannidis, J., Rubin, A.: Using the Fluhrer, Mantin, and Shamir Attack to Break WEP. Technical Report TD-4ZCPZZ, AT&T Labs (2001)
7. Arbaugh, W., Shankar, N., Wang, J.: Your 802.11 Network Has No Clothes. In: Proc. First IEEE International Conference on Wireless LANs and Home Networks. (2001)
8. Mishra, A., Arbaugh, W.: An Initial Security Analysis of the IEEE 802.1X Standard. Technical Report CS-TR-4328, University of Maryland (2002)
9. Kent, S., Atkinson, R.: Security Architecture for the Internet Protocol. IETF, RFC 2401 (1998) [Online] ftp://ftp.rfc-editor.org/in-notes/rfc2401.txt
10. Freier, A., Karlton, P., Kocher, P.: The SSL Protocol Version 3.0. [Online] http://wp.netscape.com/eng/ssl3/draft302.txt
11. Lemos, R.: Security: Open Networks Pose Dilemma. In: news.com, Feb. 5 (2003) [Online] http://news.com.com/2009-1033-982324.html?tag=rn
12. IEEE: Specification for Enhanced Security. (unapproved draft for 802.11i) [Online] http://standards.ieee.org/getieee802/new.html
13. Appenzeller, G., Roussopoulos, M., Baker, M.: User-Friendly Access Control for Public Network Ports. In: Proc. INFOCOM, IEEE, Mar. (1999) 699–707 [Online] http://mosquitonet.stanford.edu/publications/WebSpinach.ps
14. Brustoloni, J., Garay, J.: MicroISPs: Providing Convenient and Low-Cost High-Bandwidth Internet Access. In: Computer Networks **33** (2000) 789–802 [Online] http://www9.org/w9cdrom/249/249.html
15. Bahl, P., Venkatachary, S., Balachandran, A.: Secure Wireless Internet Access in Public Places. In: Proc. ICC, IEEE, June (2001) [Online] http://www.cs.ucsd.edu/users/abalacha/research/papers/ICC01.pdf
16. IEEE: Port-Based Network Access Control. 802.1x Std. (2001) [Online] http://standards.ieee.org/getieee802/download/802.1X-2001.pdf
17. Karygiannis, T., Owens, L.: Wireless Network Security — 802.11, Bluetooth and Handheld Devices. Special Publication 800-48, NIST (2002) [Online] http://csrc.nist.gov/publications/nistpubs/800-48/NIST_SP_800-48.pdf