Secure Open Wireless Access

Tom Cross
IBM X-Force
tcross@us.ibm.com

Takehiro Takahashi
takehiro.takahashi@gmail.com

ABSTRACT

One of the primary disadvantages of public Wi-Fi hotspots today is that most do not afford their users with any data privacy, due to the inconvenience of using pre-shared passwords. Worse yet, users of open hotspots are left without any means to verify the authenticity of wireless network providers. The consequences for security and privacy are disastrous. We propose Secure Open Wireless Access, a technology that enables Wi-Fi users to make secure, encrypted connections to wireless access points without a pre-shared password or other credential. Secure Open Wireless Access works by reserving particular Service Set Identifiers (SSIDs) for the exclusive use of the Wi-Fi network operators they are associated with. These SSIDs are tied to the digital certificates used in the anonymous form of the 802.1X EAP-TLS authentication process. Hence an anonymous user can validate the identity of the wireless network operator by comparing the SSID of a wireless network with the digital certificate presented during authentication. We propose two methods to institute exclusive SSIDs; a global trusted SSID database and the use of domain names as SSIDs. The latter is equivalent to the well known security model of HTTPS used by web browsers. We implemented and tested Secure Open Wireless Access with popular implementations, and verified that our proposal is a viable solution which provides an anonymous yet secure wireless connection.

General Terms

Design, Reliability, Security, and Standardization.

Keywords

802.11, wireless security, 802.1X, EAP, RADIUS, TLS

1. INTRODUCTION

Open 802.11 [16] wireless access points are everywhere. You can find them in hotels and airports, in living rooms and coffee shops, in public parks and business conference centers, in fast food restaurants and libraries. In some cities, municipal wireless networks have been strung up by local governments, providing Internet access all over town. Millions of people use these access points every day to check their email, surf the web, and chat online. Unfortunately, these people face security threats. Their communications are broadcast by their wireless cards a fair distance in cleartext and can be surveilled by anyone within range. They may also encounter rogue access points that masquerade under the same SSID as legitimate networks and launch man in the middle attacks against them to steal payment credentials or other sensitive information [13, 14].

While numerous attack tools that take advantage of this vulnerability have been available for years, few have provided as dramatic an illustration of the risks as Firesheep [6]. Firesheep is an extension for the Firefox web browser that sniffs insecure networks for potential victims accessing popular websites like Twitter and Facebook. When Firesheep observes a potential victim connecting to one of these websites, it collects the victim's authentication cookie and displays their name and photograph to its user. (See Figure 1) The user can click on the victim and be instantly logged into their account using the stolen access credentials.

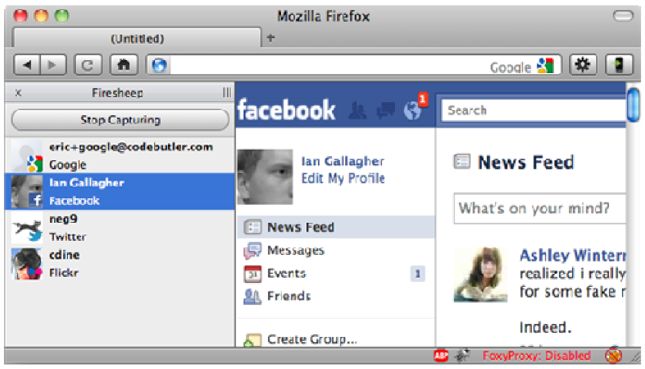


Figure 1. Firesheep

Of course, standards exist for encrypting wireless communications with 802.11 access points, but they require that a password or other credential be shared with the user before they can associate. This is clearly a problem since most wireless service providers have no way to establish a secret password with users before they connect. Therefore wireless service providers typically allow unencrypted connections, relying on captive web gateways as a user friendly method of authentication and access control if it is required. Unfortunately, this choice leaves user's traffic unencrypted and exposed at the link layer.

The bottom line is that Wi-Fi users often face a hobson's choice between using cumbersome pre-shared credentials or forgoing link layer privacy and authentication all together. We present a third way, Secure Open Wireless Access, in which a wireless access provider presents a digital certificate tied to its Service Set Identifier (SSID) [16] to provide for authentication of the service provider and to generate keys for privacy encryption, without requiring an individual pre-shared credential. Our approach borrows from the well known security models of Secure Sockets Layer (SSL) and Secure Shell (SSH) to allow trust to be established between access points and anonymous users. The result is that any unsolicited user can connect to an access point without a pre-shared secret, but encryption is used, and the user can be fairly certain that the access point is operated by a trusted party.

In Section 2, we explain the motivation for Secure Open Wireless Access, and briefly discuss prior work. Section 3 discusses our threat model. Section 4 explains how digital certificates and explicit SSIDs can address the problem. Section 5 explains how Secure Open Wireless Access works at the protocol level. Section 6 suggests some desired changes to the 802.11 Management Frame format to support Secure Open Wireless Access. Section 7 describes our prototype for Secure Open Wireless Access.

2. MOTIVATION AND PREVIOUS WORK

While the privacy concerns associated with attack tools like Firesheep ought to be obvious, what may not be obvious is whether the appropriate solution to this problem is to encrypt the link layer of wireless communications. In our efforts to promote Secure Open Wireless Access we have encountered a wide array of counter arguments. Some people consider end to end encryption of all Internet communications to be a more desirable goal and argue that focusing on the wireless link layer is not necessary. Demonstrations like Firesheep have prompted the adoption of HTTPS by a number of major websites, and some people believe that this effort is sufficient to protect wireless end users from harm. Others believe that point to point VPN solutions are the right approach to wireless security. We will address each of these perspectives in turn.

First, let us consider the question of full end to end encryption on the Internet. On some level, all Internet communications are subject to surveillance. Untrustworthy network operators can see Internet traffic regardless of whether or not it started out on a wireless network, as can interlopers who've taken control over poorly secured infrastructure routers and switches. That being the case, there is a reasonable argument that all sensitive communications across the Internet ought to be encrypted [18].

While we are sympathetic with that point of view, end-to-end encryption of all Internet traffic seems a long way off, and the wireless link layer deserves particular attention, because wireless communications can be easier for attackers to intercept and interfere with. While ISP infrastructure certainly can fall into the hands of unscrupulous employees and system intruders, this happens rarely. However, cleartext wireless traffic can be readily intercepted by anyone who happens to be staying at the same hotel, or sitting in the same public park, or coffee shop, or airport terminal. In dense urban areas the number of people who have access to a given unencrypted wifi transmission can be quite large. This exact sort of consideration has led to cellular phones that encrypt audio over the wireless link while the backbone network still carries traffic in the clear to its destination. There is a greater risk that wireless communications will be intercepted.

In response to public concern about the interception of wireless communications, raised in some cases by demonstrations like Firesheep, many large websites have adopted HTTPS for user access and authentication. We think these efforts are important. They can significantly reduce the risk that access credentials for those websites will be compromised. However, we don't think this effort goes far enough toward addressing the risks that users of wireless networks face. Most websites have not adopted HTTPS, leaving much of wireless users' traffic open to surveillance and manipulation, and unencrypted link layers can create opportunities for other kinds of attack scenarios.

Figure 2 is a photograph of a commercially available rogue wifi access point called a Wifi Pineapple, which at the time of this writing was being sold for $99 at hakshop.com. This device allows the user to masquerade as a legitimate wireless access point that clients are expecting to connect to. These clients connect to the Wifi Pineapple instead, where, according to hakshop.com, their traffic can be "easily viewed or even modified by the pineapple holder."

A tool called SSLStrip [5] provides an example of one attack that a rogue access point might make. SSLStrip enables a rogue access point to convert HTTPS links in HTML pages being transmitted to the user into HTTP links that the attacker can observe. This allows the attacker to see inside of connections made to HTTPS websites by users who aren't paying careful attention.

Figure 2. HakShop Wifi Pineapple

Users of a suspicious access point might decide to play it safe by not accessing sensitive websites and limiting their browsing to supposedly safe activities such as reading the news. However, IBM's Rational Application Security Research Group recently published a paper illuminating the dangers of that approach [22]. An active attacker can incorporate references to sensitive domains into "safe" content. The victim's browser might be tricked into serving up an authentication cookie for a website the victim did not intend to access, or the victim's browser cache could be poisoned with javascript of the attacker's choosing that will run in the context of a sensitive site the next time the victim accesses it.

A rogue access point could also enable an attacker to create a malicious captive gateway that uses network access fees as a pretext to steal the victim's credit card numbers. Although the security model afforded by HTTPS certificates allows users to know that they are, in fact, communicating with the website that their browser says they are communicating with, this does not help much if the user is unfamiliar with that website. Often, wireless networks operated by well known companies will direct users to HTTPS based captive payment gateways that are operated by third parties with strange domain names that the user is unlikely to be familiar with. Users have no choice but to enter their payment credentials and hope for the best. They could easily be connecting to a rogue access point with a safe sounding SSID whose captive gateway site hosts a legitimate SSL certificate. Although this concern could be addressed by better practices on the part of wireless network operators, the fact is that it is the SSID, and not the domain name of the captive gateway, that establishes the identity of the network operator in the user's mind and creates the foundation for trust.

Furthermore, service provider captive gateways cannot trust authenticated users on unencrypted wireless networks. Gateways prevent unauthorized access by checking the source IP and MAC address of packets as they pass through. In an unencrypted network, it is trivially easy for an attacker to assume the MAC and/or IP of another legitimate user and begin communicating without providing payment information or other credentials. While encryption is not sufficient to solve this problem on wireless networks, it is a necessary prerequisite, and can be effective when coupled with firewall capabilities in the access point that prevent users from intercommunicating or assuming IP addresses that they were not assigned by DHCP.

There are VPN services that offer to solve some of these problems by allowing users of suspicious access points to encrypt all of their information and send it to a trusted party on the Internet, who will decrypt it and send it on to its destination. However, this approach is inefficient. Maintaining inbound and outbound bandwidth for receiving and retransmitting user's traffic is expensive and these services have to charge their users a monthly fee. Furthermore, users have to be savvy enough to go to the trouble of signing up and downloading the client software. Casual Wi-Fi users need not apply. It seems clear to us that a more fundamental solution is called for.

A few years ago George Ou, a technology journalist and IT consultant, proposed a solution for encrypting anonymous Wi-Fi connections using Protected Extensible Authentication Protocol (PEAP) along with a digital certificate and a default username and password of "guest" [19]. Mr. Ou's proposal comes close to solving this problem, but it suffers from one important weakness beyond the need to somehow inform users about the guest account - it does not protect users from rogue access points. Anyone can obtain a signed digital certificate associated with any domain that they have registered, and setup an access point with the same Service Set Identifier (SSID) as Mr Ou's access point. The way to solve this problem is to take the next step, and tie the SSID to that digital certificate. Although we have learned of one previous author who has publicly suggested this approach, this paper is the first time that the implications of this idea have been completely evaluated [23].

3. THREAT MODEL

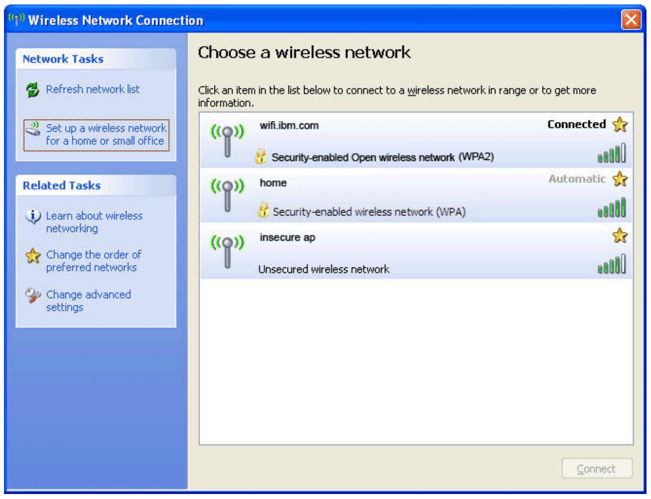
The attacker's goal is to eavesdrop or tamper with victim's wireless traffic. We assume that the attacker is capable of observing unprotected or weakly encrypted messages in the vicinity. We consider WEP [16] and WPA-PSK [16] with a short passphrase as weakly encrypted. We also assume that the attacker can set up a rogue access point with an arbitrary SSID such as "Free CoffeeShop WiFi", and can trick users into using the attacker's access point. As a result of this, the attacker can observe and manipulate some of victim's network traffic such as HTTP, DNS, etc. An attacker running a rogue access point may be able to set up a phoney captive gateway to collect victim's access and payment credentials. The attacker cannot tamper with or eavesdrop on strongly encrypted traffic such as HTTPS or SSH.

4. SECURE OPEN WIRELESS ACCESS

The SSID is the primary means that Wi-Fi users have to identify the networks that they use. If a digital certificate verified that the operator of a Wi-Fi network was the exclusive, rightful user of the SSID associated with that network, users could connect without worrying that the network might be operated by someone else. Figure 3 provides an example of how this would look from a user's perspective. It shows a window with multiple wireless networks in the user's vicinity that the user can choose to connect to. There is an open wireless network with the SSID "insecure ap" which does not provide secure connections. There is a second wireless network with the SSID "home" which does offer secure connections, but only to those who know the correct password. And, there is a third wireless network which supports Secure Open Wireless Access.

The SSID of the secure open wireless network in our example is "wifi.ibm.com." If the user were to connect to this network, his or her wireless client software would establish an encrypted connection, verifying that the digital certificate presented by the network during the connection setup process is signed by a trusted certificate authority and is tied to the domain name "wifi.ibm.com." As IBM is supposed to be the only organization that can obtain valid digital certificates that are tied to domain names within the "ibm.com" domain, the user can be relatively sure that they are connecting to a wireless network that is operated by IBM.

This process can help protect the user against the threat model described in Section 3 of this paper without requiring a pre-shared access credential. The user's connection is encrypted, so it cannot be sniffed or manipulated, and the user is assured that the network is operated by IBM and not an attacker. Hopefully the user trusts IBM not to spy on or manipulate their Internet traffic. If the network operator desires further client authentication, they can use a captive portal without any of the risks currently associated with the use of captive portals on unencrypted wireless networks today.


Wireless Network Connection
Network Tasks
Refresh network list
Set up a wireless network for a home or small office
Related Tasks
Learn about wireless networking
Change the order of preferred networks
Change advanced settings
Connect

Choose a wireless network

Click an item in the list below to connect to a wireless network in range or to get more information.

wifi.ibm.com Connected
Security-enabled Open wireless network (WPA2)

home Automatic
Security-enabled wireless network (WPA)

insecure ap
Unsecured wireless network

Figure 3. Secure Open Wireless Access GUI Mockup

Secure Open Wireless Access protects users from rogue access points by providing an assertion about the identity of the network operator. This assertion will only protect users if they are willing to exercise some discretion regarding which wireless networks they are willing to connect to. Anyone can go out and get a digital certificate and anyone can run a wireless network. Some of the people who do so are going to be unscrupulous. If a user is willing to connect to any network, regardless of what the SSID of that network is, and regardless whether or not it is actually protected by a digital certificate, then the existence of Secure Open Wireless Access isn't going to do that user any good.

On the other hand, if a user is willing to be selective about which wireless networks he or she connects to, Secure Open Wireless Access can provide a number of interface cues that allow that user to make informed choices. The first cues are the lock icon associated with the access point and the lack of a warning about encryption while connecting. These cues tell the user that they are establishing an encrypted connection to the access point.

One advantage of this approach is that connecting to an unencrypted wireless network can always result in a warning message. Users should only be able to connect to a network without seeing a warning message if that network connection is encrypted. Compare this situation to that of HTTP, where unencrypted websites are the default case and don't produce a warning, and the signs that HTTPS is in use can be subtle. This can lead to situations where users aren't aware of the differences between protected and unprotected websites. [5] With Secure Open Wireless Access, the differences between encrypted and unencrypted wireless networks would be clear to the user.

Another interface cue is the network SSID. Users can choose to only connect to networks with SSIDs that they are familiar with and comfortable with, knowing that criminals are unable to set up secure networks with the same SSIDs. Although criminals could attempt to register misleading SSIDs that are similar to those of trusted network operators, registered SSIDs would be published in a global database (such as the DNS) that network operators can monitor for attempts at fraud.

We believe that this would be a vast improvement upon the present status quo. Today, there is absolutely no way for a user to know who is operating an open wireless network. Anyone can use any SSID, and networks operated by criminals are totally indistinguishable from legitimate networks. Accessing these networks is always a bit of a "crap shoot." Users who are both capable and willing to exercise some discretion have no means with which to do so. Users who would prefer an encrypted connection have no means to establish one. In dense urban environments, where there are many wireless networks operating, and one's traffic is exposed to many potential eavesdroppers, the ability to ensure that you are communicating securely with a network operator that you know and trust would be tremendously valuable.

One barrier to the acceptance of this proposal is that it involves a change in perspective about SSIDs. While Wi-Fi service providers often choose unique SSIDs that identify the company operating the network, the idea of globally exclusive SSIDs has never been proposed before. Nevertheless, we see no reason why these unique SSIDs cannot be considered the exclusive property of the network operators that use them. This would not interfere with legitimate reuse of generic sounding SSIDs (like "FreeWifi") by insecure wireless networks.

Another objection that has been raised to this approach is that adoption would spark a land rush by wireless network operators to register generic sounding SSIDs like "Free Wifi" with certificate authorities for exclusive use in secure wireless networking. The whole point of associating SSIDs with certificates is to give users an indication of the identity of the company or organization operating a wireless network. Discerning users should be suspicious of any supposedly secure access point with a generic sounding SSID because those SSIDs don't convey any of that identity information, which can form the basis for trust. Therefore, generic sounding SSIDs have less value than unique names.

In addition to protecting open wireless networks from rogue access points it is also worth considering whether certified SSIDs might also improve the security of closed wireless networks. It is currently possible to protect users of closed networks from attacks by rogue access points through the use of mutual authentication, but in order for this to work client computers must be configured with specific radius server information associated with the SSID. If closed networks were protected by certified SSIDs, users could connect to them and present usernames and passwords without preconfiguration of their computers and without fear that they might be speaking with a rogue access point.

5. IMPLEMENTING SECURE OPEN WIRELESS ACCESS

In order to explain how Secure Open Wireless Access is implemented it is necessary to provide a little bit of background on security in wireless networks. The 802.11 wireless networking standard addresses authentication by incorporating IEEE 802.1X, an authentication mechanism for devices wishing to attach to a network. Figure 4 illustrates how the 802.1X authentication works in a wireless network. First, a workstation informs a wireless access point that it wants to connect to the network. If the access point accepts this connection request, the workstation will then engage in an authentication transaction with a network authentication server, via the access point. If the authentication is successful, the authentication server informs the access point, and the access point establishes an encrypted connection with the workstation, ideally using a strong encryption algorithm such as AES with a secure protocol such as WPA2. Once that secure connection is established, the workstation can begin using the network.

The access point typically communicates with the authentication server via the Remote Authentication Dial In User Service (RADIUS) protocol [17]. RADIUS is a networking protocol that offers authentication, authorization, and accounting management for users, and is typically used by Internet Service Providers. The access point uses the RADIUS protocol to encapsulate the authentication transaction between the workstation and the authentication server. The workstation authenticates to the server using a form of Extensible Authentication Protocol (EAP) [17], which is a framework for supporting a wide range of authentication protocols over RADIUS.

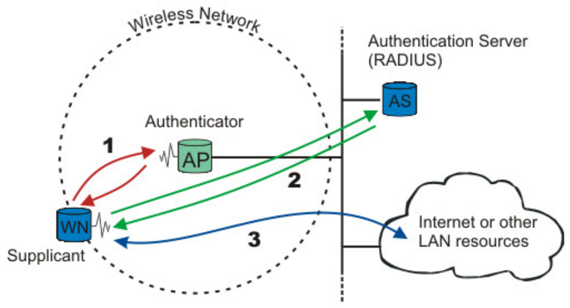

Wireless Network
Authentication Server (RADIUS)
AS
Authenticator
AP
1
2
3
WN
Supplicant
Internet or other LAN resources

Figure 4. 802.1X Authentication Process

EAP-TLS is the form of EAP used in Secure Open Wireless Access. While it would be technically possible to implement our proposal using other forms of EAP, such as PEAP or EAP-TTLS, those protocols require client authentication, which is not desired by an open access point. The IETF standard for EAP-TLS [21] states in section 2.1.1 that under some circumstances, EAP-TLS may be used in a context where no credential is provided by the user, and only the identity of the network is really being authenticated. The RFC states that "while the EAP server SHOULD require peer authentication, this is not mandatory, since there are circumstances in which peer authentication will not be needed (e.g., emergency services, as described in [UNAUTH]), or where the peer will authenticate via some other means" [21]. Following this observation in the RFC, authentication servers can implement anonymous EAP-TLS by omitting a certificate_request from the TLS server_certificate handshake message sent during the EAP-TLS negotiation.

Our proposal consists of using this anonymous form of EAP-TLS with exclusive SSIDs. When a workstation associates with a network that supports Secure Open Wireless Access, the client application on that workstation verifies the authenticity of the access point by matching the SSID of the network with the SSID in the digital certificate used by the authentication server in the EAP-TLS negotiation. The client application also verifies that the certificate was signed by a trusted, third party certificate authority. If these checks fail, the application prompts the user with a warning message indicating that the network may not be safe. This process is analogous to the way that HTTPS connections currently authenticate using SSL certificates, but in this case the security is tied to SSIDs rather than domain names.

The private encryption keys for the digital certificate used in the EAP-TLS transaction would be stored on the network authentication server. Therefore Secure Open Wireless Access would be relatively easy for network providers to deploy. A single authentication server on the Internet could provide services to a large number of wireless access points at multiple physical locations. The authentication server could be hosted in an environment with the high level of physical and network security required to protect the integrity of private keys, even if the wireless access points themselves are sometimes placed in open environments. If an access point were physically stolen, the network provider could easily revoke its RADIUS access.

There are two approaches to making SSIDs exclusive; utilizing domain names or preparing a global, trusted SSID database. The first method requires service providers to set their SSIDs to be the same as their Internet domain names. This would be convenient, since it is easy for SSL certificate authorities to validate that a requesting entity controls a particular domain name, and domain names are unique identifying strings that many consumers are already familiar with. However, since the length of an SSID is limited to 32 characters, this will only work for short domain names without changes to the 802.11 protocol Moreover, although the IETF 802.11 specification does not specify any restrictions on the types of characters that can be used in SSIDs, many wireless access points currently available do not allow punctuations such as periods or hyphens. We discuss possible 802.11 protocol extensions to overcome some of these problems in section 6.

If certificate authorities want to issue certificates for SSIDs that are not domain names, they must communicate with each other through a common database to ensure the following conditions are true - (1) only one certificate is valid at any given time for a particular SSID, and (2) SSID requests are globally unique and related to the actual name of the organization requesting them, which the certificate authorities will have to carefully verify. It is possible that the existing whois database system could be used for this purpose. However, it is also likely that such certificates will be more expensive than modern SSL certificates, and would require more hands on effort from certificate authorities in order to issue them.

It is possible that a hybrid approach is desirable. Domain names have a lot of value as identifiers because we already have an infrastructure for issuing certificates tied to them. However, network operators will need to prevent unauthorized parties from obtaining useful certificates tied to names that are substantially similar to their own SSIDs, in order to prevent fraud. Domain names are tools that can be used in a variety of different contexts, so the mere fact that another organization has registered a domain

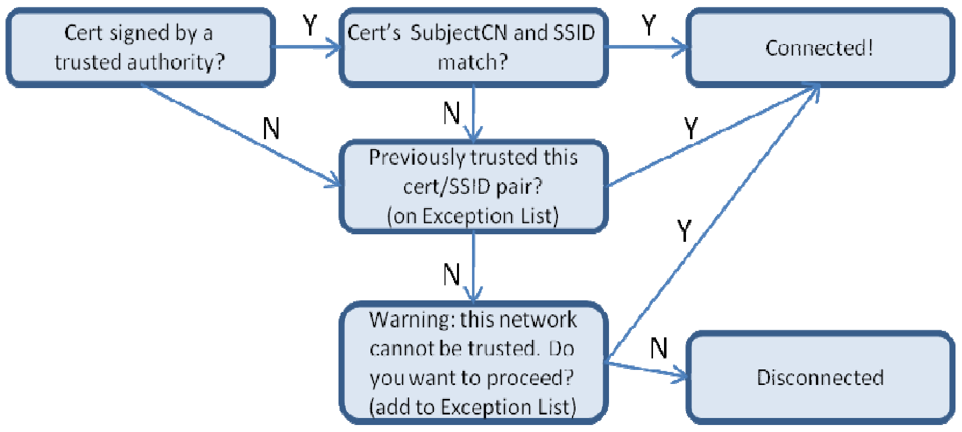

Cert signed by a trusted authority?
Y
Cert's SubjectCN and SSID match?
Y
Connected!
N
N
Y
Previously trusted this cert/SSID pair? (on Exception List)
Y
N
Warning: this network cannot be trusted. Do you want to proceed? (add to Exception List)
N
Disconnected

Figure 5. Authentication State Diagram

name that is substantially similar to the SSID used by a network operator may not be sufficient for that operator to establish a trademark violation and have that domain name revoked. It would help if names used for secure wireless access were published in a global database specifically intended for that purpose so that certificate authorities could avoid issuing certificates for secure wireless access where there is a potential for confusion and trademark claims could be made in the specific context of providing wireless access. In order for this to work, digital certificates intended specifically for secure wireless access would need to be different from certificates used in other contexts, and client software would need to be able to distinguish the two.

There may be a desire by some small network operators and home users to be able to support Secure Open Wireless Access using digital certificates that have not been signed by a certificate authority. If a wireless client encounters a certificate that is not signed by a known certificate authority, that client could engage in an optional process that allows users to develop trust relationships with uncertified wireless networks. The process we propose is similar to the way that Secure Shell (SSH) allows users to develop trust relationships regarding the keys used by SSH servers. Figure 5 provides an overview of this process in the form of a state diagram.

When the client first encounters a certificate that was not signed by a trusted certificate authority or does not match the SSID associated with that network, the client would confirm whether the user has previously decided to trust this SSID and certificate keyid combination when presented by a network. If so, the client proceeds with the network authentication as if the certificate had been signed correctly. Otherwise, the client checks to see if the user has previously indicated implicit trust of this SSID with a different certificate keyid or if this SSID has been used in a context where the key was signed by a trusted certificate authority. In these cases a stern warning is presented to the user, indicating a possible attack in progress, and the user is allowed to proceed only if they insist that they are aware of possible unwanted consequences. On the other hand, if the user has not seen this SSID before, the client presents the user with the fact that this is an untrusted network, and asks whether the user would like to implicitly trust this particular SSID-certificate combination in the future.

It may also be helpful for clients that support Secure Open Wireless Access to verify, when connecting to an insecure access point, that the SSID associated with that access point has not been seen in conjunction with a secure network in the past. If the SSID has been seen before, a stern warning is also warranted. That warning may help protect users from insecure networks masquerading as trusted service providers.

Finally, we argue that certificate revocation should be handled through Online Certificate Status Protocol (OCSP) [16] and OCSP stapling [16]. OCSP is a standard protocol used for verifying the revocation status of digital certificates, and it allows a client to query a certificate issuer about a certificate's status. Since the certificate issuer's response is signed by issuer's private key, the status information cannot be tampered with. OSCP stapling is a supplement protocol to OCSP, and it allows the presenter of the certificate to cache the CA's time-stamped OSCP response and forward it to the client. In the context of Secure Open Wireless Access, the RADIUS server can implement OSCP stapling so that a TLS handshake during EAP-TLS includes the OCSP information as described in RFC4366 [16]. The client would need to verify this information.

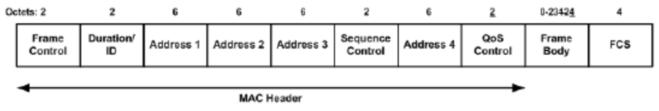


Figure 6. 802.11 Frame Header

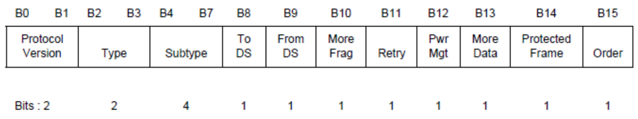


Figure 7. Frame Control Field in 802.11 Frame Header

6. EXTENSIONS TO THE 802.11 MANAGEMENT FRAME

Our proposal for Secure Open Wireless Access could be implemented today, with existing wireless access points and wireless cards. Changes would be needed for wireless client applications to support the certificate validation, SSID checking, and other processes described in Section 5. RAIDUS servers may also need to be modified to support the anonymous form of EAP-TLS, which is not universally implemented as of this writing. Wireless access points might need firmware updates to support punctuation marks in SSIDs. Digital certificate authorities would need to establish a process for issuing certificates for secure open wireless access. However, the basic hardware devices in use today for wireless networking can support this proposal without modification.

Users of Secure Open Wireless Access would want some way to differentiate these networks from closed, password protected networks in the list of nearby access points presented by their wireless clients. This could be achieved in the short term through the use of a standard SSID naming convention. For example, if domain names were used for certified SSIDs, users might learn to associate the use of domain names with secure open wireless access, and attempt to connect to networks with such SSIDs, in-spite of the fact that those networks appear "locked" in their wireless client.

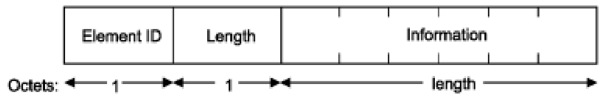


Figure 8. Information Element

Nevertheless, there are some changes that could be made to the 802.11 Management Frame specifications in order to make Secure Open Wireless Access more user-friendly. First, it would be helpful if wireless networks could advertise their support for Secure Open Wireless Access in a way that clients could easily detect. Second, SSIDs are currently limited by the 802.11 standard to 32 characters in length but domain names can be longer. To fully support the use of domain names as SSIDs we need to expand the SSID specification to support longer names as well as internationalized domain names.

The basic format of 802.11 frames is depicted in Figure 6 and Figure 7, and each frame consists of the following sections:

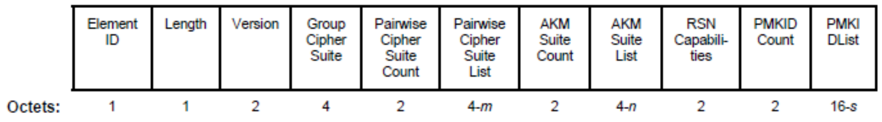


Figure 9. RSN Information Element

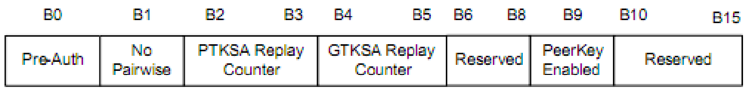


Figure 10. RSN Capabilities Field Format

1. A MAC header, which comprises frame control, duration, address, and sequence control information, and, for QoS data frames, QoS control information;
2. A variable length frame body, which contains information specific to the frame type and subtype;
3. A FCS, which contains an IEEE 32-bit CRC.

The 3rd and 4th bits in MAC header's frame control field represent 802.11 frame's type whose value can be one of the following: (1) management, (2) control, and (3) data. Management frames allow wireless hosts to establish and maintain connections. Control frames control device's access to the wireless medium. CTS (clear-to-send), RTS (request-to-send), and ACK frames are good examples of the control frame. Finally, data frames carry user payloads. The 5-8th bits in the header represent each frame's subtypes. The frame body consists of an array of information elements specific to each frame subtype. Each information element is assigned a unique element ID, and stores data in the Information section. Figure 8 illustrates the information element format. In Secure Open Wireless Access, the following information element types are extended: (1) RSN, and (2) XSSID.

6.1 RSN (Robust Security Network)

The RSN information element describes the security features that a wireless station supports. Specifically, it contains the cipher suite information for authentication, pairwise key management, and group key management, and is also a part of beacon and probe response frames. Therefore, a wireless station can learn about an access point's support for Open Secure Wireless Access if we extend the RSN information element.

The RSN information element, as illustrated in Figure 9, contains the following fields: Element ID, Length, Version, Group Cipher Suite, Pairwise Cipher Suite List, Authentication and Key Management (AKM) Suite List, RSN Capabilities, and PMKID List. The Group Cipher Suite field describes the cipher suite selector used to protect broadcast/multicast traffic. Likewise, the Pairwise Cipher Suite List field has a series of cipher suite selectors for unicast traffic. These cipher suites types can be WEP-40, TKIP, CCMP, WEP-104, etc. The AKM Suite List field contains a series of AKM suite selectors supported by the station. Each AKM suite selector indicates station's authentication and key management capabilities. The RSN Capabilities field indicates requested or advertised capabilities. The PMKID (Pairwise Master Key ID) List field is used to transmit pairwise master key IDs used for re-association.

We propose the following changes to the RSN information element: (1) a new RSN capability bit field to indicate a support for anonymous 802.1X authentication and (2) a new RSN capability bit field to indicate support for certificate validation against network's SSID. We can support these features by using 2 of the 9 unused bits in the RSN capabilities field as shown in Figure 10. We recommend using two bit fields to indicate these two pieces of information separately since it allows for greater flexibility. For instance, an operator of closed wireless networks who wishes to use certified SSIDs to provide rogue access point protection could use PEAP authentication with certificate validation.

Wireless client software can display a wireless network as supporting Secure Open Wireless Access if the RSN capability field of a received beacon frame has the anonymous 802.1X and certification validation bits set.

6.2 XSSID

In addition, we propose adding another information element type, XSSID (eXtended SSID), to support full length, internationalized domain names in SSIDs. Figure 11 illustrates the format for XSSID. The first two fields specify its type and length, and the third field holds the extended SSID value whose size can be up to 253 octets. A reserved element ID number such as 51 may be good for the element ID.

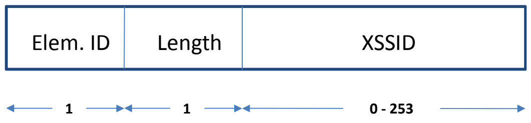


Figure 11. XSSID Information Element

Any management frames which support the SSID information element, such as Beacon or Association frames, should support the XSSID element. Secure Wireless Access ready clients should display the XSSID value instead of the SSID whenever it is available.

7. IMPLEMENTATION

We have implemented a proof of concept using several open source projects such as FreeRADIUS[7], wpa_supplicant[8] and wicd[10] under Ubuntu Linux. FreeRADIUS is a RADIUS server, and wicd and wpa_supplicant are Linux network components which work together to establish 802.11 connections. In this section, we explain how these applications work together in Secure Open Wireless Access, and describe the extensions we made to their source code. Figure 12 provides a rough illustration of their interaction.

When our Secure Open Wireless Access client establishes a connection to a wireless network, the following steps are taken:

1. The user chooses a network, and configures his workstation to use Secure Open Wireless Access from wicd's GUI.
2. Wicd signals wpa_supplicant on behalf of the user to make a connection to the wireless access point using anonymous EAP-TLS (Secure Open Wireless Access).
3. The access point understands the 802.1X request, and forwards traffic to the FreeRADIUS server.
4. Wpa_supplicant and FreeRAIDUS initiate a TLS handshake. Wpa_supplicant's certificate verification code then checks the SSID against the server certificate.
5. If the authentication was successful, wpa_supplicant proceeds to derive session keys with the access point.
6. Using the session keys, the workstation can now encrypt traffic to the access point, and communicate with the rest of the network.

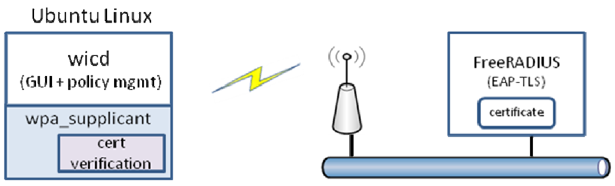
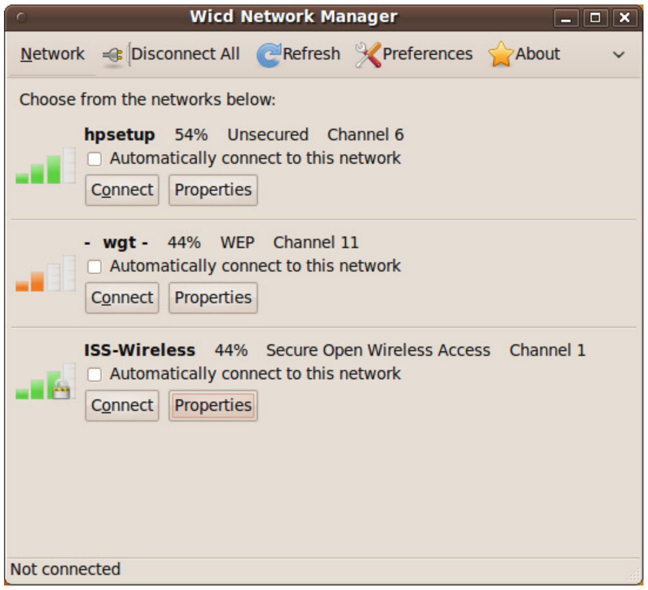


Figure 12. Secure Open Wireless Access Interaction

The FreeRADIUS project offers a RADIUS server daemon, a client, and several supplementing libraries. Their RADIUS also supports a rich set of authentication protocols including EAP-TLS, PEAP, EAP-TTLS, etc. For Secure Open Wireless Access, we had to modify their EAP-TLS implementation to make anonymous TLS possible. Specifically, we changed the source code in rlm_eap_tls.c to use the SSL_VERIFY_NONE flag for the OpenSSL [11] function "SSL_CTX_set_verify". This flag prevents the server from sending a certificate_request message to the client. With this small change, the FreeRAIDUS server successfully handled requests from Secure Open Wireless Access clients. Although further work such as policy management and configuration parsing is needed to fully support Secure Open Wireless Access, our FreeRADIUS implementation indicates that other existing RADIUS server implementations are likely to require little to no modification.

Wpa_supplicant is a supplicant tool for WPA, WPA2 as well as 802.1X, and it controls client's roaming/association with another IEEE 802.11 station, and negotiates keys with a WPA authenticator. Wpa_supplicant supports popular operating systems such as Windows, Mac OSX, Linux and BSD, and is designed to function as a daemon program which interacts with wireless drivers and higher level user controls. We implemented our proposed certificate verification process by registering a call back function in wpa_supplicant's EAP-TLS component. Specifically, we modified its OpenSSL library wrapper "tls_openssl.c", and registered our custom certificate-SSID verification function using SSL_set_verify. SSL_set_verify, like SSL_CTX_set_verify of FreeRADIUS, allows the user to specify the certificate verification flags as well as a verification callback function during



Choose from the networks below:

hpsetup 54% Unsecured Channel 6
Automatically connect to this network

- wgt - 44% WEP Channel 11
Automatically connect to this network

ISS-Wireless 44% Secure Open Wireless Access Channel 1
Automatically connect to this network

Not connected

Figure 13. GUI Mockup

a TLS handshake. Hence, when our wpa_supplicant's EAP-TLS module initiates a TLS connection to the RADIUS server, our verification function determines whether this wireless network can be trusted by checking to see if the SSID of the network matches the Common Name of the certificate. We plan to implement the authentication process illustrated in Figure 5 in the next version.

Wicd is a wired and wireless network manager for Linux, and it provides a simple interface to configure and enforce a wide range of network settings. Since wicd uses wpa_supplicant to control wireless connections, it replaces wpa_supplicant's built-in GUI frontend. We have extended wicd's GUI template written in Python to support Secure Open Wireless Access.

In our future work, we plan to implement the 802.11 frame extensions in the wireless client as well as the access point. We may use an open source wireless driver like hostap [9] which can be used for both clients and access points. Figure 13 illustrates how the wicd GUI would look when the proposed extensions are fully implemented in wpa_supplicant and wicd. A key lock icon next to ISS-Wireless's signal strength indicates that the network uses Open Secure Wireless Access.

8. CONCLUSION

With relatively minor changes to existing software and standards, our proposal effectively solves a serious security problem which has plagued wireless networks for many years. We enable users to make encrypted connections to open wireless access points without first establishing a username, password, or other authentication credential and users can finally rest assured that the wireless networks they are connecting to are, in fact, really operated by the companies implied by their SSIDs. Our proposal does not disrupt the existing use of SSIDs with insecure networks and provides small network operators and home users the ability to take advantage of the security properties offered by anonymous encrypted connections without having to pay for a digital certificate. It is our view that this proposal represents a significant step forward for the security of wireless Internet infrastructure.

9. ACKNOWLEDGMENTS

We would like to thank Evgenv Cherkashin, David Dewey, and Andrew vonNagy for their input and suggestions in developing these ideas. David was the first person to suggest to us that traditional domain names could be used as SSIDs. Andrew suggested that we investigate certificate revocation lists more carefully, the utility of certified SSIDs in closed networks and that we might use the Information Element of the RSN rather than the AKM Suite to convey support for this feature.

Note: IBM has a patent pending on Secure Open Wireless Access.

10. REFERENCES

[1] *Kismet*. http://www.kismetwireless.net

[2] *Etherpeg*. http://www.etherpeg.org

[3] *Driftnet*. http://www.ex-parrot.com/~chris/driftnet/

[4] *Hamster Sidejacking Tool*. http://hamster.erratasec.com

[5] *SSLStrip*. http://www.thoughtcrime.org/software/sslstrip

[6] *FireSheep*. http://codebutler.com/firesheep

[7] *FreeRADIUS*. http://www.freeradius.org

[8] *wpa_supplicant*. http://hostap.epitest.fi/wpa_supplicant/

[9] *hostap*. http://hostap.epitest.fi

[10] *wicd*. http://wicd.sourceforge.net

[11] *OpenSSL*. http://openssl.org

[12] R. Beyah, S. Kangude, G. Yu, B. Strickland, and J. Copeland. Rogue access point detection using temporal traffic characteristics. In Proc. IEEE GLOBECOM, Dec 2004.

[13] H. Yin, G. Chen, and J. Wang. Detecting Protected Layer-3 Rogue APs. In Proceedings of the Fourth IEEE International Conference on Broadband Communications, Networks, and Systems (BROADNETS), Raleigh, NC, September 2007.

[14] S. Jana and S. K. Kasera. On fast and accurate detection of unauthorized access points using clock skews. In ACM MOBICOM Conference, Sept. 2008.

[15] Scott R. Fluhrer , Itsik Mantin , Adi Shamir, Weaknesses in the Key Scheduling Algorithm of RC4, Revised Papers from the 8th Annual International Workshop on Selected Areas in Cryptography, p.1-24, August 16-17, 2001

[16] IEEE Standard 802.11-2007. Information technology. Telecommunications and information exchange between systems. 2007.

[17] IEEE Standard 802.1X-2001. IEEE Standard for Local and metropolitan area networks . Port-Based Network Access Control. June, 2001.

[18] Network Working Group IAB. IAB and IESG Statement on Cryptographic Technology and the Internet. RFC1984, August 1996.

[19] Ou, G. A secure Wireless LAN hotspot for anonymous users. http://blogs.zdnet.com/Ou/?p=587. July 2007.

[20] S. Blake-Wilson, M. Nystrom, D. Hopwood, J. Mikkelsen, and T. Wright. RFC 4366: Transport layer security (TLS) extensions, April 2006.

[21] D. Simon,B. Aboba, R. Hurst, RFC 5216 The EAP-TLS Authentication Protocol, March 2008.

[22] Saltzman, R. and Sharabani, Adi. Active Man in the Middle Attacks. OWASP AU. February 2009.

[23] Byrd, C. Open Secure Wireless. http://riosec.com/files/Open-Secure-Wireless.pdf. May 2010.