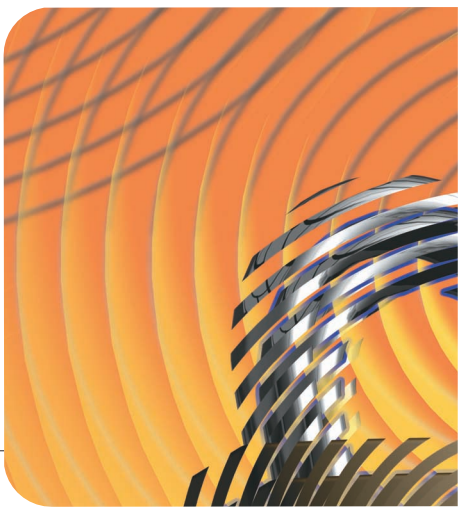
Autonomic 802.11 Wireless LAN Security Auditing

The authors describe their Distributed Wireless Security Auditor (DWSA), which works toward finding unauthorized wireless access points in large-scale wireless environments while providing an autonomic and unobtrusive layer of network protection.



Joel W. Branch
Rensselaer Polytechnic Institute

Nick L. Petroni Jr.
University of Maryland

Leendert Van Doorn and David Safford
IBM T.J. Watson Research Center

Securing 802.11b wireless local area networks (WLANs) is a daunting task. A large collection of configuration options and initial security design problems have resulted in numerous clever techniques for strengthening WLAN security. These methods range from media access control (MAC) address authentication to the full-blown use of virtual private networks (VPNs). The main problem with these techniques is that they solve only part of the problem and typically do not scale well for large organizations. It was this stalemate that prompted the early release of an interim wireless security standard, Wi-Fi Protected Access (WPA), as part of the long-awaited proposed IEEE 802.11i standard (for more on WPA, see wifi.org). While 802.11i promises to solve many of the inherent security design problems in WLANs, these networks present continual challenges to large corporate facilities.

One key design problem is the introduction into the organization of unauthorized wireless access points (APs), improperly configured APs, or ad hoc networks. With the price of APs falling to less than US$100 (well below most corporations' expense limits), it's easy for an employee to install his or her own office AP and plug it into the corporate wired network. However, such APs could provide corporate network access well beyond the building's perimeter, such as out in the parking lot or even farther. Another source of wireless leaks stems from incorrectly configured APs. This situation arises when APs are not centrally managed or frequently verified. A third source of leaks occurs when machines configured for ad hoc wireless networks are connected to a wired network, providing an open gateway to the wired network for unauthorized users.

These security anomalies can compromise any range of confidential corporate assets, creating valuable opportunities for targeted attackers and being physically difficult to locate in large buildings or campuses. Typically, administrators can address these threats by conducting "walk arounds" with wireless sniffing equipment, but this corporate version of *war-driving* provides only casual protection against the threats we just outlined and depends heavily on the audits' frequency and coverage.

In response to this, we designed the Distributed Wireless Security Auditor (DWSA). Our Linux- and Windows-based (functional on both 2000 and XP) implementations provide continuous wireless network assessments by harnessing the power of available, trusted wireless clients as distributed anomaly sensors throughout a company's network infrastructure. Using periodic security reports, a back-end server detects rogue and misconfigured APs and subsequently locates them via 3D trilateration, a location-finding algorithm used in systems such as GPS. The result is an autonomic, unobtrusive, real-time application for providing large-scale WLAN security auditing around the clock. In this article, we describe the methodology and architecture behind our prototype implementation.

The wireless challenge

Over the past few years, the IEEE 802.11 standard has been the focus of a large amount of research with respect to its security architecture and mechanisms. Our own research has shown a huge deficiency in the 802.11 standard with regard to security, as well as deficiencies in 802.11 network implementation and deployment. Fur-

Tools of the trade

To deal with protocol and management issues, managers often use a set of security tools to better understand the state of their networks. Here, we briefly describe the available tools including some of their advantages and disadvantages. While many of these tools overlap in functionality, we generally separate them into four categories: sniffers, auditors, intrusion detection systems, and honeypots/honeynets.

Sniffers

Sniffers, or protocol analyzers, are the most basic tools available to administrators of any type of network. They provide access to raw protocol packets as user stations (in this case 802.11 wireless cards and access points) transmit and receive them. "Sniffing" is the foundation for most auditors, intrusion detection systems (IDSs), attack tools, and several other applications. Network protocol analyzers are most useful when an administrator is interested in a particular session of network traffic, looking for a specific piece of information, or simply wants a complete record of what transpired. It is difficult for administrators to use a sniffer for large-scale security analysis of their networks because of the vast amounts of highly detailed information they would need to parse. Instead, more logic in the code is usually built on top of a sniffer to produce a higher-layer tool such as an IDS. There are several commercial and freely available open-source sniffers for IEEE 802.11.

Auditors

Network managers use security auditors to analyze their network's current state with respect to a corporation's security policy. Unlike sniffers, which provide information about raw traffic, auditors can perform passively or actively to identify which aspects of the system are accessible and under what conditions. Auditors can run once or be set to check system state periodically or continuously. Tools such as Netstumbler (www.netstumbler.com) and Kismet (www.kismetwireless.net), both extremely common among wireless enthusiasts, are classic examples of wireless auditors. Our own DWSA is a distributed auditor with several advanced features.

Intrusion detection systems

While auditors focus on system state, IDSs aim to identify aspects of its use. In particular, they seek to determine when the system has been used in a way that violates a particular security policy—or at least when an attacker has tried to do so. In practice, many IDSs provide overlapping functionality with security auditors (especially for wireless networks). Most wireless IDSs are based on online wireless sniffers that constantly capture raw data to be interpreted and analyzed. A handful of commercial and open-source wireless IDSs exist on the market today, and a small body of academic research on the subject is beginning to grow.[1]

Honeypots and honeynets

Over the past few years, the concept of allowing attackers to gain access to closely watched systems for research purposes has captured the attention of many in the security community. Systems or networks whose sole purpose is to promote unauthorized access provide administrators of those systems with insight into what an attacker does after compromising a real system. Wireless honeynets have begun to pop up in a few metropolitan areas that let researchers study questions regarding what war drivers do after gaining access to a vulnerable access point.

Reference

1. Y. Zhang and W. Lee, "Intrusion Detection in Wireless Ad-Hoc Networks," *Proc. 6th Ann. Int'l Conf. Mobile Comp. and Networking*, ACM Press, 2000, pp. 275–283.

thermore, while several technologies have been (and continue to be) developed to either augment or replace the standard's flawed portions, the difficulty of managing wireless networks has created a complex situation for network administrators even when they use the latest technologies.[1,2] Let's look at some of the vulnerabilities associated with 802.11 wireless networks, both with the protocol and its management of deployed networks. We do not provide a detailed explanation of any identified protocol flaws, but rather focus on their symptoms and results as related to security auditing.

Protocol issues

The intent of a wireless medium is to provide users ease of access when connecting to the network without the binding requirement of a physical wire (or other medium). However, convenience for valid users is only one side of a double-edged sword—easy access for one is easy access for all users, those with both good and bad intentions. Because of this fundamental characteristic, wireless networking technologies must protect valid users and their traffic from the vast arsenal of capabilities afforded to an attacker with access to the physical medium. Unfortunately, 802.11 fell short of this requirement in several ways.

Weaknesses in Wired Equivalent Privacy (WEP). IEEE 802.11 was initially designed to provide data confidentiality and integrity protection through WEP.[3,4] The algorithm, which uses the RC4 stream cipher to encrypt the entire payload, also requires a CRC-32 checksum (a data field the receiver uses to verify that the packet's payload has not been corrupted in transmission) over the unencrypted data. A 24-bit initialization vector (IV) is sent with

Table 1. Summary of known weaknesses in Wireless Equivalent Privacy (WEP).

WEAKNESS	DESCRIPTION
Passive key recovery	Scott Fluhrer, Itsik Mantin, and Adi Shamir identified a key scheduling attack, known as FMS attack, against the RC4 algorithm that, when used with certain keys, renders the cipher vulnerable to key recovery.[8] Adam Stubblefield, John Ioannidis, and Avi Rubin identified WEP as an example of such a system and later experimentally proved it to be vulnerable.[9] Freely available tools now exist that can execute this attack on commodity hardware.
Small initialization vector space	WEP's initialization vector (IV) space, as Jesse Walker and, Nikita Borisov and colleagues later identified, is much too small to provide an adequate number of distinct crytographic keys, needed for encrypted data.[6,10] IV reuse, or encrypting multiple messages with the same IV and cryptographic key, results in key stream reuse which, among other things, lets an attacker recover portions of the original, unencrypted message.
Replay attacks	Failure to provide message authentication lets an attacker arbitrarily repeat messages without detection.[10]
Message modification attacks	Using CRC-32 for message integrity facilitates several extremely dangerous attacks against encrypted messages without knowledge of the secret key. Attackers can modify any encrypted message and maintain a correct integrity check value.[10]
Keystream (inductive) dictionary attack	Weaknesses in integrity protection and IV space lead to a synchronous dictionary attack for WEP keys whereby an attacker can build an encryption/decryption dictionary without knowledge of any portion of the key. This attack has been demonstrated on real networks and benefits from factors that mitigate the FMS.[11]

each encrypted packet and used as a *nonce*, a specific value inserted into the message to defend against replay attacks. Table 1 summarizes some of WEP's known problems.

Weak authentication and key management. As part of the 802.11 state machine, which controls the operational condition of a wireless network interface, wireless clients always exist in one of three states: unauthenticated and unassociated, authenticated but not associated, or authenticated and associated.[3] While the intention in the original design was to enable APs to require authentication before joining a network, the only form of authentication the standard provides is based on a key shared between station and AP. In most implementations, this shared key is simply a WEP key, because WEP is the algorithm used to verify the key's possession. Unfortunately, well-known attacks against shared-key authentication make it trivial for an attacker to bypass.[5] Furthermore, the lack of a framework for WEP key management has resulted in a predominance of networks with only a single shared key for all users. Using a single key has several limitations, not the least of which is an accentuation of the protocol problems described earlier.[2,6]

No protection for management frames. The 802.11 protocol uses three types of messages (or frames): control, management, and data. Control frames facilitate access to the communication medium, whereas management frames provide a framework for organizing stations into a network (either ad hoc or infrastructure). Data frames are used only to transmit higher-layer protocols. For two stations to communicate, they must first use management frames to set up an association. This is a three-phase process, which can be briefly described as discovery, authentication, and association. Unfortunately, 802.11 does not protect management frames, thereby allowing an attacker to "spoof" valid deauthentication or disassociation frames and isolate a targeted node from the network. The result is a trivial denial-of-service (DoS) attack. Some researchers have suggested schemes to mitigate such attacks, but we know of no broad-scale implementation of these approaches.[7]

Susceptibility to carrier sense attacks. Many networking technologies that use a shared medium to communicate require the transmitter to "carrier sense," or check if the medium is being used by another transmitter, before sending their own traffic. IEEE 802.11 uses an optimization of this technique known as carrier sense multiple access with collision avoidance (CSMA/CD). For example, for longer packets, a station can send request-to-send (RTS) control frames to indicate it wishes to transmit something. Included in the RTS packet is a time-duration field, indicating how long the station will transmit on the channel. As part of the collision-avoidance algorithm, all stations hearing this RTS must remain silent until the time indicated by the duration field has expired. Research has shown that by sending extensive RTS packets with maximum duration, an attacker can monopolize the medium and perform a DoS attack.[7] We should note, however, that on most commodity equipment, this attack is more difficult to implement than the previously discussed attack using management frames.

Deployment and management issues

As previously stated, several solutions either have been or are currently being developed to solve the most egregious problems with 802.11, at least in regard to data frames' confidentiality and integrity.[1] However, even with the

most advanced technology, trying to keep a wireless network (or a wired network with a liberal access policy) secure can be extremely difficult. While the list of challenges is nearly infinite, we highlight a few of the most common problems plaguing network administrators.

AP configuration and management. Administrators must ensure that they have correctly configured all APs according to the company's security policy. While some vendors provide efficient management solutions, most APs come to the consumer preconfigured even without security or with easily defeated default values. Unfortunately, it takes only a single misconfigured AP to cause a complete breach of access control for an entire wireless network. Furthermore, if the overall network architecture does not have sufficient separation from the wireless access, the damage can be even more devastating. Additional problems might arise from AP failures, such as a faulty reboot that leaves the AP set to its default configuration or another unknown state.

Insider threats (rogue APs). One of the most frightening thoughts for a network administrator is the prospect that a user could, intentionally or otherwise, extend access to the network arbitrarily. With the common availability of inexpensive wireless networking equipment, an employee can easily buy an off-the-shelf AP and plug it in at his or her desk. While this action is usually benign in intent, the result is nonetheless the dangerous introduction of a security vulnerability.

Key management, user authentication, and access control. Because 802.11 does not provide a framework for key management, many networks have a single shared key that all users employ when configuring their clients for the first time or simply have no encryption at all. Furthermore, the only form of access control the specification provides is possession of a WEP key or a key for shared authentication, both of which are completely ineffective. For this reason, many vendors provide additional forms of access control such as MAC lists (also ineffective), proprietary higher-layer protocols, or IEEE 802.1X.[5] The latter is the foundation for the IEEE 802.11 Task Group I working group's security solution, which will be part of an update to the existing 802.11 specification. Whichever access-control mechanism administrators employ, managing that mechanism correctly is critical for a successful security policy.[12]

Distributed Wireless Security Auditor: An overview

We designed DWSA in response to 802.11 security vulnerabilities discussed earlier and, in particular, for large-scale wireless deployments in corporate environments. Our goal involved instituting a robust network-auditing solution that provided a passive and unobtrusive layer of network protection in multiple aspects. DWSA's architecture and operational behavior also promote an autonomic security scheme, complementing the inherent complexity and value of rapidly growing networks and protected assets.

Operation theory

The crux of DWSA's function is similar to other 802.11 network auditors: to help regulate access to and behavior of the network with respect to an internally established security policy. One way in which we offer a novel contribution is through our proposed operational theory. We designed DWSA's system operation to complement the arduous task of large-scale auditing by harnessing the power of preexisting wireless clients. To do this, we utilize registered Wi-Fi clients (for example, desktops, laptops, tablet PCs, and PDAs) as widespread security sensors that monitor the presence and security state of observed APs. This creates a beneficial distributed computing architecture, which lets network administrators forego walking tours as a means of auditing large networks. In turn, using available wireless clients provides a cost-effective solution, eliminating the need to purchase specialized 802.11 signal-sensing equipment. Some might raise a caveat concerning operational dependence on preexisting wireless clients for network auditing, and in response, we argue two points: First, 802.11 networks will continue to grow as technology becomes cheaper and more robust, providing an increasing number of available clients for monitoring purposes. Second, our solution applies to the challenge of large-scale deployments, which indicates a widespread presence of both 802.11 clients and APs.

While the clients' function is monitoring and collecting AP information, data aggregation and correlation occurs at a central server. Our motivation is to provide a secure environment for security analysis as well as to limit the burden of client-side processing so as to promote unobtrusive operation for the user. DWSA periodically compares AP data from clients' security reports, received via a secure channel at the central server, against a list of authorized access points and the instituted security policy to determine offensive AP and where they reside. Figure 1 highlights the overall application's architecture as well as operational behavior.

The combination of distributed operation and integration into existing network components forms an autonomic 802.11 auditing solution, as was our intent. Our application design focuses on two aspects of autonomic computing: *self-management* and *self-healing*. DWSA's distributed and automatic reporting and correlation facilities aptly fulfill the self-management behavioral aspect of our application. Continual auditing also promotes self-healing, because results gathered from DWSA can be used for such functions as automatic AP shutdown.

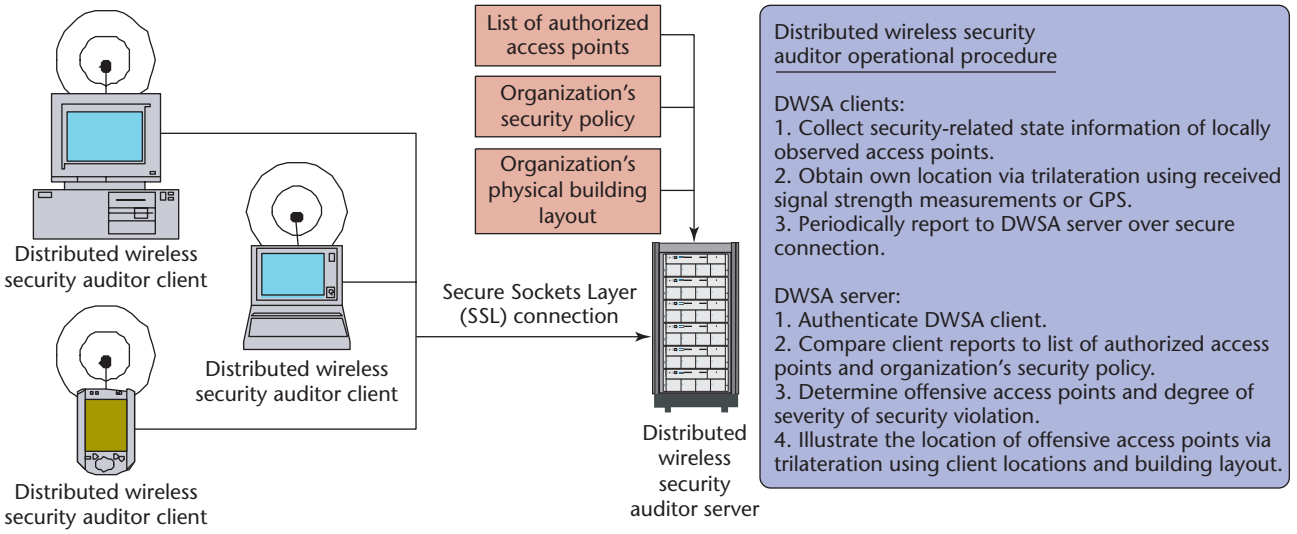


Figure 1. Graphical overview of the DWSA system's architecture and information flow. All decision-making about offensive access points (APs) is carried out on a back-end server, leaving an unobtrusive client application.

DWSA's architecture

As we described earlier, DWSA's fundamental strategy is to use clients as monitoring agents that report to a centralized point of correlation. In this section, we briefly describe the major components of such a system. Toward this end, we consider the following as a set of minimum requirements for the distributed auditor.

- *Lightweight clients*. Because the agent will run on a system that must do real work (for example, that of a company employee), the client must not require resources in a manner that drastically affects the host system's performance. Furthermore, the client is expected to run on low-power, handheld devices that might not have much processing power to begin with.
- *Portability*. The system should run on all common wireless clients, including handheld devices.
- *Secure communication*. The client must be able to report its data back to the correlation point with guarantees of data confidentiality and integrity.
- *Easy-to-use, robust policy specification*. DWSA's decision-making logic must be easily configurable, but should let administrators provide detailed descriptions of what they allow in their systems.
- *Location estimation*. To the degree possible, the system should be descriptive about the geographic location of the information it collects. Even though clients are mobile, they should have some capacity to estimate their own location or provide the correlation system with enough context to make its own estimation.

Let's look at the different pieces of DWSA with a focus on the above requirements.

DWSA client. The DWSA client is based on the principle of using spare processor cycles to gather information about the wireless medium that directly surrounds the client. In its current form, the agent is a small background program that periodically executes and begins sniffing the wireless medium. It passively takes note of all contacts it can hear and creates a brief summary of that information to report back to the DWSA server as soon as it can. Along with MAC addresses and various protocol details (including WEP flags, higher-layer protocol numbers, and FMS weak IVs), the client also sends radio-signal-strength statistics to the server to help it estimate the client's (and, later, other contacts') location. Because all estimation is done on the server, the client need not locally retain any information it has collected; the server might stop or start the process or reboot the system at any time. In addition to its statelessness, the client also minimizes its workload by simply providing data summaries and not performing analysis itself. To meet the portability requirement, we wrote the client so that it runs on recent versions of Windows (XP, 2000, and CE) and Linux. Finally, we achieve secure communication via strong encryption that the Secure Sockets Layer (SSL) protocol provides. Client and root certificates are distributed out of band at the time of software installation.

DWSA server. Unlike the lightweight client, the DWSA server is expected to have a sufficient amount of processing power to correlate the potentially large amounts of information a busy network provides. The server has three primary components in the current implementation: correlation, policy analysis and reporting, and interactive 3D visualization.

Correlation is the process of taking all clients' data reports and summarizing them into a useful view of the network. This data's primary representation is a hash table whose key is the contact's MAC address. As data comes in from various clients, the contact is quickly found or inserted in the hash table, and all protocol information immediately updates for that system. Reporting stations are included in the contact list, and statistics are kept in the table regarding their reporting history. While we do not describe the correlation algorithm's details, the type of information kept in the contact table includes raw protocol details (such as WEP flags and weak IV counts) as well as compound, or calculated, information such as location estimates or the number of connected clients. The resulting hash table is therefore a snapshot of the entire network as a list of all network contacts (or at least those that a reporting station noticed). Figure 2 illustrates this.

DWSA's current approach to *policy analysis* uses a rules language similar to that of many firewalls. Rules are added to one of three chains: valid, warning, or alert. When clients receive information about a particular client, that client's current state is updated and then evaluated with regard to each chain. In this way, each station is constantly assessed as valid, potentially invalid, or convincingly invalid. Administrators are notified of policy changes through two means—a visual notification (color change or pop-up) on the server's graphical user interface and an output to a logging mechanism. Finally, policies can be updated on the fly to help with policy generation and to easily adapt when changing network configurations.

One of the DWSA server's most useful features is its ability to integrate real-world geographical information into its reporting facility. The combination of location estimation with prerecorded locations of objects in the physical world lets DWSA provide administrators with an immediate picture of where different APs, friend or foe, are located. The DWSA server takes as input a set of vector graphics, which it can use to overlay the coordinate system of the physical space being audited. As Figure 3 shows, the result is an extremely detailed visual representation of the network's current state, as it exists in the real world.

Locating vulnerabilities

Correctly determining an offensive wireless AP's presence is only one function of DWSA's back-end server. Locating these APs is an essential part of the auditing process, but as the wireless network grows, this process becomes more problematic. The scope of physical (on foot) human searches for rogue APs begins to parallel that of war-driving (or LAN-jacking), and performing this task on a daily basis decreases security administrators' productivity. Furthermore, any sluggish search process would allow employees enough time to elude detection by either incidental or explicit means. For large enterprises, instantaneous location detection is necessary. Instantaneous locating could translate to additional hardware costs because it requires placement of specialized sensor devices throughout an enterprise's building or campus to detect unregistered APs. In continuing with our purely software-based architecture, though, DWSA's back-end server uses data from clients running the local detection components to locate offensive APs. This allows administrators to be flexible when locating threats with heterogeneous wireless devices. With DWSA, a user can view any AP's physical location in 3D with one click, as Figure 3 shows. Only a small set of WLAN auditors offers similar features.

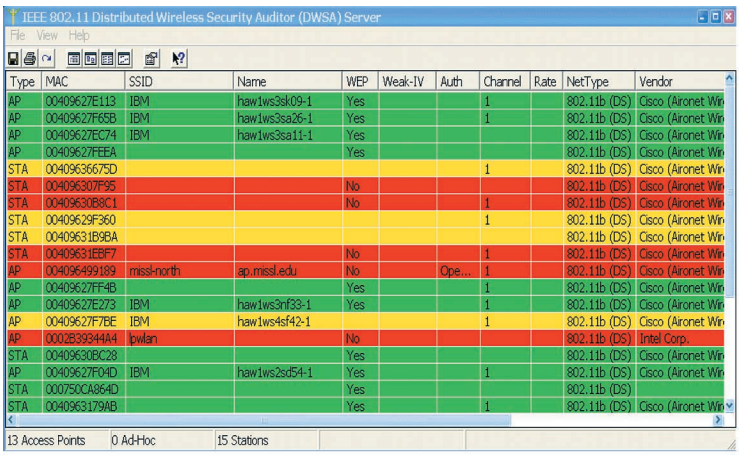


Figure 2. DWSA server console. This screenshot of the DWSA server console application gives the network administrator an informative view of the security state of the company's wireless LAN. DWSA client reports are displayed in an organized fashion, indicating network vulnerabilities at a glance (MAC, SSID) and security states (WEP, weak-IV). Additional information about an AP is returned by selecting its respective row. Threat severity, defined by the administrator's security policy, is indicated by color coding: green is safe, yellow is warning, and red is emergency.

Trilateration: Background and challenges

Location systems find typical applications in WLANs as well as ad hoc wireless networks.[13] We locate APs using *trilateration*, a process heavily used by GPS receivers. Three-dimensional trilateration locates an AP given at least four known fixed points and their respective distances from it. This is analogous to finding the intersection between four spheres, where the known points are the spheres' centers and respective distances are the radii. DWSA uses wireless client locations as the fixed known points; their respective ranges from the targeted AP are

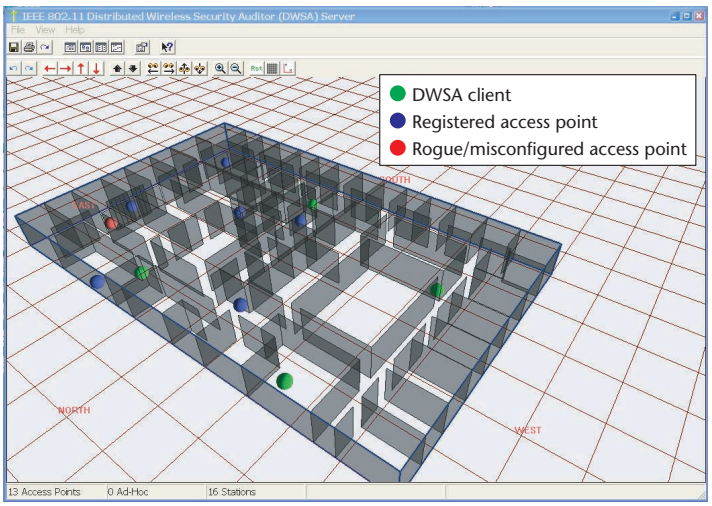


Figure 3. DWSA location visualization. This is a screenshot of the DWSA location visualization program, which runs on the DWSA server. AP locations and classes (for example, registered or rogue) are easily viewed in three dimensions. This particular screenshot shows the location of an illegitimate access point within the confines of a building floor (denoted by the red sphere). The user can manipulate the viewpoint of the environment (rotate, zoom, pan, or tilt). Additionally, multiple floors can be viewed simultaneously.

the radii. Figure 4 illustrates these scenarios. In our prototype, client locations are obtained by trilateration using the fixed points near authorized APs. (If clients are GPS-equipped, then we can use GPS to obtain location instead of trilateration.) The clients' respective distances from the targeted AP are calculated from received signal strength (RSS) measurements—which can be obtained from typical wireless network interfaces.[14] The client reports both sets of information to the back-end server, where the targeted position is calculated. Trilateration using RSS measurements, as well as other radio frequency signal methods (that is, packet arrival delays), have remained strong choices for indoor trilateration techniques. The convenience of using RSS measurements, however, does not come without costs.

Unrefined RSS measurements provide only estimates (of varying degrees of accuracy) of perceived distance. Errors usually arise from several factors such as multipath, fading, shadowing, and differing altitudes of radio antennae. Also, typical office appliances, such as microwave ovens, can distort RF. Propagated throughout a trilateration algorithm, these values can cause intolerable signal errors when determining resultant positions. Some researchers propose attacking this problem by using signal strength maps (representing regional signal propagation behavior) recorded in an offline phase.[15] While sharpening the position estimates' accuracy, these methods require extensive preparation beforehand. For an application such as ours, we accept tolerable errors without the burden of site surveying. A position estimate within six feet of the offensive AP works well for the informed network administrator. Moreover, using RSS measurements as opposed to other RF signal methods requires the least amount of implementation overhead. Nonetheless, technology is constantly changing, and location services using RF signals are slated to improve.

DWSA's method

From a high-level view, the DWSA back-end server locates threats using periodical updates from distributed auditor agents deployed on each enterprise's wireless client. Along with a list of observable APs, the auditor agent reports its distance from each AP and its own current location. Upon detecting an unregistered or misconfigured AP, the back-end server uses the included locations and distances of the nearby clients to locate the vulnerability. A network administrator can then remove or reconfigure the AP in question. From a lower level, we explore the trilateration algorithm's implementation.

Facing the challenge of performing trilateration given estimated distances, the back-end server must still execute a relatively fast algorithm that produces effective results. In seeking such a mathematical solution fitting these requirements, we found a study of ill-conditioned 3D position estimation to fit our needs in another study.[16] We adopted the method described in this study (originally developed for tracking mining equipment) because the problem spaces are similar. Furthermore, the preceding process displays simple programmability and relatively fast execution time. The only main difference is that we use RSS measurements, whereas the chosen study used packet arrival delays to measure distance.

As mentioned earlier, the solution technique for this problem begins with finding the intersection of several spheres. Each sphere is represented by the following equation:

$$(x - x_i)^2 + (y - y_i)^2 + (z - z_i)^2 = r_i^2 \quad (1)$$

Point $p_i(x_i, y_i, z_i)$ represents the location of client i (i = 1, 2 ,…,n), the sphere's center, and r_i represents the distance from point p_i to the offensive AP (the sphere's radius). Using several sphere equations (as in equation 1), we solve for point $p_a(x, y, z)$, representing the offensive AP's location (intersection point of the n spheres). As noted in William Murphy and colleagues' work (see www.mines.edu/fs_home/whereman/papers/trildbl.ps), solving a system of n nonlinear equations simultaneously is not feasible because a high-degree nonlinear equation is produced, which increases the complexity of calculating point p_a. The system of equations could be *linearized* (changing the problem to finding the intersection of several planes), but will still not accommodate approximate distances. Consequently, we use linear and nonlinear least-

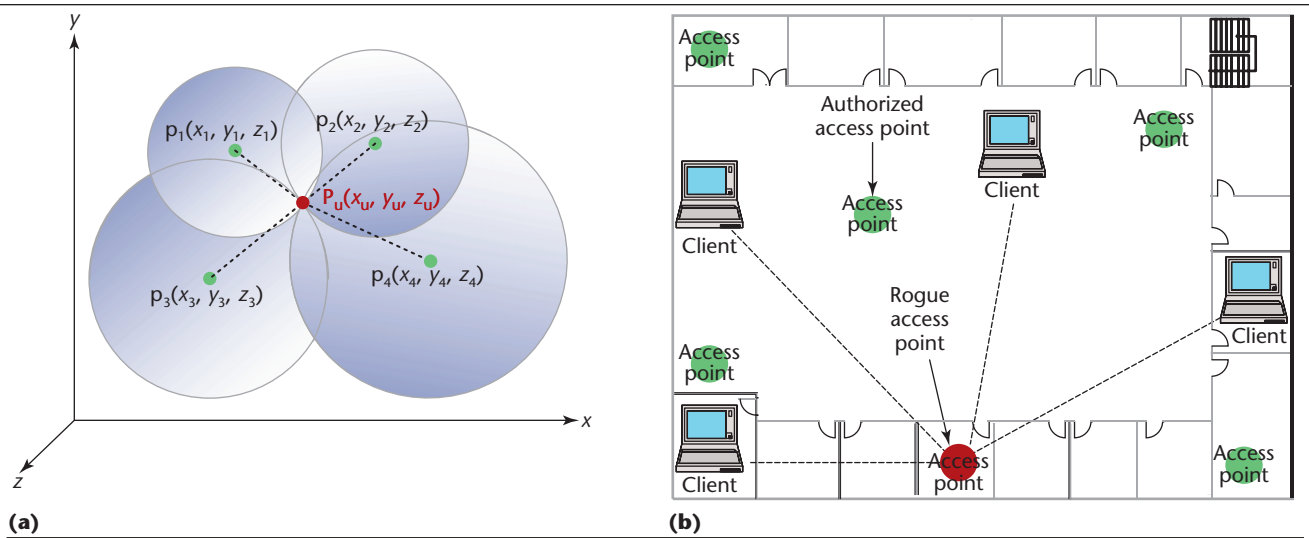


Figure 4. Examples of 3D trilateration. (a). A conceptual drawing illustrates the trilateration process, where the green points represent fixed locations and the red point, where the spheres intersect, is the location in question. (b). A floorplan represents DWSA's comparative view of trilateration where wireless clients represent fixed locations used to find the offensive AP, which is represented by the red circle.

squares methods, which use approximate distances. We briefly describe the process here, but the source work contains more details.[16]

First, the server algebraically manipulates the system of *n* sphere equations to form a linear system of equations. While this initial linear system is easier to solve than the original system of sphere equations, it still does not yield acceptable results in the presence of distance estimations. From here, we apply the linear least squares method to the linear system to increase the calculated position's accuracy. Linear least squares "curve fitting" is a method by which an estimated result by fitting a given model (for example, our initial linear system) to some explanatory data (such as our distance estimates).

We tested DWSA's performance using the linear least-squares method and found the results to be of poor quality. This was generally expected because the signal path loss in our testbed, the IBM T.J. Watson Hawthorne 1 building in Hawthorne, New York, was a critical factor in stressing the trilateration algorithm. This outcome reflected what William Murphy and colleagues' described in their study as well. Additionally, our results were more severe owing to the distance measurements' increased inaccuracy. Following the referenced work, we next applied the nonlinear least squares method. Nonlinear least-squares curve fitting is attractive because it fits a broader class of functions that linear models might not appropriately fit. Our basic data model, RSS falloff as a function of distance, is nonlinear in nature—the RSS measurements will asymptotically level off as the distance approaches infinity. Thus, we received much better results when using the nonlinear least-squares method. Because the method we used is iterative and requires an initial estimation of the desired result, we used the output from the linear least squares execution in initializing the nonlinear least-squares method.

DWSA performance and adjustments

Our testing grounds at the IBM T.J. Watson Research Center modeled a typical large-scale corporate office environment. Four levels of hundreds of partitioned offices, lab equipment, copiers, and various structural materials, all affecting RF signal propagation, presented a rich environment for evaluating the trilateration component's performance.

Through all our trial runs, the result of trilateration was computed in about two seconds. This is not excessive for such an application because we aren't necessarily attempting to track a constantly moving target. Once a threat has been identified and located within a tight geographical region, such as someone's office, a network administrator already has enough information to identify the culprit, even if the AP disappears soon after.

We first performed our evaluation in 2D space: APs and clients from a single floor. Under this condition, we achieved the desired accuracy of the algorithm; we effectively located rogue APs to within 6 feet of their actual locations approximately 90 percent of the time. In this case, we attribute the error source mostly to signal distortion; inherent noise from office and lab machinery is to be expected. Regardless, a 90 percent success rate is excellent because multiple executions of the trilateration algorithm will average to a correct result.

Expectedly, its initial performance in 3D space left room for improvement. Two main sources of error, a

lack of fixed coordinate points and signal distortion, lowered our success rate. The lack of fixed points (authorized APs) was a telescoping problem. While 2D trilateration requires only three fixed points (intersecting circles), 3D trilateration requires four (recall the sphere problem). In our demonstration, each client must first determine its own location before reporting to the DWSA back-end server. In our building, finding three authorized APs to "trilaterate" from was no problem, but finding four was difficult. First, at certain regions of a building floor, only three APs might be visible to a wireless client. This problem, however, could be mitigated—if all three APs resided on the same floor, the client could locate itself using 2D trilateration and adopt its *z*-coordinate from that of the three APs. On the other hand, the APs could belong to as many as three different floors. At this point, the client cannot locate itself; thus, it is of no use when attempting to locate the offensive AP. Therefore, the lack of known authorized AP locations was a telescoping problem because it led to a lack of wireless client locations available to participate in locating the offensive AP. While this is an overall problem, it is not likely to occur as large corporations introduce denser Wi-Fi functionality throughout the workplace.

The complexity of indoor signal propagation provided challenges. Average RF signal path loss confined to a particular floor could be minimized due to less restrictive materials. However, loss from an office building floor (frequently composed of steel-reinforced concrete) as well as additional walls can be significantly higher.[17] This phenomenon created a severe inconsistency in the perceived distances of the wireless clients in our demonstration. Most of the irregularity was caused by clients reporting from floors above or below the targeted AP. As a result, because of the trilateration algorithm's nature, the final coordinate frequently yielded acceptable *x*- and *y*-coordinates, but erroneous *z*-coordinates. We alleviated this source of error by choosing only those clients registering the strongest signals from the offensive AP. When this is infeasible (for example, when only four clients are available to choose from), we can determine the *z*-coordinate by considering which floor the majority of the clients report from—but this is not a robust solution.

These preliminary patches to the 3D trilateration algorithm did improve location performance to about 80 percent accuracy, and GPS-enabled clients might alleviate the effect of some of the previously described errors. We are currently researching self-corrective trilateration algorithms that can accommodate the challenges in using WLAN clients for location-based services. Regardless, in echoing previous observations, the trilateration algorithm's effectiveness depends on the number of registered APs and wireless clients deployed in the network. We addressed this concern earlier, but include the fact that installing additional 802.11 APs not only supports the auditing process but also addresses part of the core problem. Rogue APs are ill-willed, individual attempts at stitching the problems of poor Wi-Fi connectivity. Additional, authorized APs should help mitigate this threat altogether.

Judging from the myriad innovative and profitable devices and services in today's marketplace, the Wi-Fi and WLAN revolutions are far from over. Research and advances in wireless standards indicate that we are still operating on the front-end of "tetherless" ubiquitous computing environments. Just the same, wireless security has not yet reached a fully matured state. As for WLANs, whether they serve as extensions of the core wired network or simply act as access gateways, providing convenient service coverage will always raise security flags. Even in the wake of the anticipated security features of 802.11i, we must take precautions to sharply define WLANs' intended use and structure and continuously audit their behavior. This essential layer of protection remains a pillar among the most rapidly evolving security ideologies in computing today. □

References

1. ISO/IEC 8802-11 ANSI/IEEE Sta. 802.11, *Wireless LAN Medium Access Control and Physical Layer Specifications*, draft amendment, Int'l Org. for Standardization/IEEE, 2003.
2. N. Cam-Winget et al., "Security Flaws in 802.11 Data Link Protocols," *Comm. ACM*, vol. 46, May 2003, pp. 35–39.
3. ISO/IEC 8802-11 ANSI/IEEE Sta. 802.11, *Wireless LAN Medium Access Control and Physical Layer Specification*, Int'l Org. for Standardization/IEEE, 1999.
4. R. Housley and W. Arbaugh, "Security Problems in 802.11-based Networks," *Comm. ACM*, vol. 46, no. 5, 2003, pp. 31–34.
5. W.A. Arbaugh, N. Shankar, and J. Wang, "Your 802.11 Network Has No Clothes," *Proc. 1st IEEE Int'l Conf. Wireless LANs and Home Networks*, IEEE Press, 2001, pp. 131–134.
6. J.R. Walker, *Unsafe at Any Key Size: An Analysis of the WEP Encapsulation*, IEEE 802.11 Task Group E IEEE 802.11/00-362, Oct. 2000, http://grouper.ieee.org/groups/802/11/Documents/DocumentHolder/0-362.zip.
7. J. Bellardo and S. Savage, "802.11 Denial-of-Service Attacks: Real Vulnerabilities and Practical Solutions," *Proc. 12th Usenix Security Symp.*, Usenix Assoc., 2003, pp. 15–28.
8. S. Fluhrer, I. Martin, and A. Shamir, "Weaknesses in the Key Scheduling Algorithm of RC4," *Proc. 8th Ann. Workshop Selected Areas in Cryptography*, Springer-Verlag, 2001, pp. 1–24.
9. A. Stubblefield, J. Ioannidis, and A.D. Rubin, "Using the

Fluhrer, Mantin and Shamir Attack to Break WEP," *Proc. 2002 Network and Distributed Systems Security Symposium*, Internet Society, 2002, pp. 17–22.
10. N. Borisov, I. Goldberg, and D. Wagner, "Intercepting Mobile Communications: The Insecurity of 802.11," *Proc. 7th Ann. Int'l Conf. Mobile Computing and Networking*, ACM Press, 2001, pp. 180–189.
11. N.L. Petroni Jr. and W.A. Arbaugh, "The Dangers of Mitigating Security Design Flaws: A Wireless Case Study," *IEEE Security & Privacy*, vol. 1, no.1, 2003, pp. 28–36.
12. A. Mishra and W.A. Arbaugh, *An Initial Security Analysis of the IEEE 802.1X Standard*, tech. report CS-TR-4328, Dept of Computer Science, Univ. of Maryland, 2002.
13. J. Hightower and G. Borriella, "Location Systems for Ubiquitous Computing," *Computer*, vol. 34, no. 8, 2001, pp. 57–66.
14. Agere Systems, "WaveLAN Chip Set Features," white paper, 2003, www.agere.com/client/docs/PB03044.pdf.
15. M. Youssef, A. Agrawala, and U. Shankar, "WLAN Location Determination via Clustering and Probability Distributions," *Proc. IEEE Int'l Conf. Pervasive Comp. and Comm.*, IEEE CS Press, 2003, pp. 143–150.
16. W. Murphy and W. Hereman, *Determination of a Position in Three Dimensions using Trilateration and Approximate Distances*, tech. report MCS-95-07, Mathematical and Computer Sciences Dept., Colorado School of Mines, 1995; www.mines.edu/fs_home/whereman/papers/trildbl.ps.
17. T. S. Rappaport, *Wireless Communications*, Prentice Hall, 1996.

Joel W. Branch *is a third-year PhD student at Rensselaer Polytechnic Institute. He received a BS in systems and computer science from Howard University and an MS in computer science from Rensselaer Polytechnic Institute. His research interests are wireless network security, privacy, data mining, and wireless ad hoc sensor networks. He is a member of the IEEE and the ACM. Contact him at brancj@cs.rpi.edu.*

Nick L. Petroni *is a third-year PhD student at the University of Maryland, College Park's Department of Computer Science. He received a BS in computer science from the University of Notre Dame and an MS in computer science from the University of Maryland, College Park. His research interests include information security, trustworthy computing, and wireless networks. He is a member of the IEEE, ACM, and Usenix. Contact him at npetroni@cs.umd.edu*

Leendert Van Doorn *is a research staff member at IBM T.J. Watson Research Center where he runs the secure systems department. He received a PhD degree from the Vrije Universiteit in Amsterdam, the Netherlands. His research interests are secure operating systems, secure hypervisors, physical secure coprocessors, and wireless security. He is a member of the IEEE, the ACM, and Usenix. Contact him at leendert@watson.ibm.com.*

David Safford *is manager of the Global Security Analysis Lab at IBM T.J. Watson Research Center. He received a PhD in computer science from Texas A&M University. His research interests include Linux security, trusted computing, autonomic computing, and 802.11 wireless networking. Contact him at safford@watson.ibm.com.*