SECURING WIRELESS COMMUNICATION IN HETEROGENEOUS ENVIRONMENTS

Wu-chun Feng

Research & Development in Advanced Network Technology (RADIANT)
Computer & Computational Sciences Division
Los Alamos National Laboratory
Los Alamos, NM 87545

ABSTRACT

With the convergence of mobile devices and the Internet, ubiquitous computing promises to revolutionize the way that we access services and run applications. However, ubiquitous computing environments, in particular, mobile and wireless environments interfaced with the Internet, currently possess security vulnerabilities that are ripe for attack from cyber-threats. Thus, this paper discusses the limitations of current security mechanisms in ubiquitous (but heterogeneous) computing environments and presents a general-purpose infrastructure that addresses these limitations, thus allowing heterogeneous environments to communicate in a secure manner. These heterogeneous environments of the future will be interoperable and will include environments such as wireless sensor networks, mobile communication, and computational grids.

INTRODUCTION

Rapid advances in mobile devices and wireless networking have converged to enable ubiquitous computing where mobile devices can access services, run programs, utilize resources, and harvest computing power anytime and anywhere. This new generation of ubiquitous and mobile computing enables the delivery of services that are no longer bound by time or location barriers. For the general public, this may provide the ubiquitous delivery of integrated services and multimedia-enabled applications to the home. For the military, it can enable the reconnaissance of enemy movement via wireless sensor networks.

However, while mobile and wireless communication provide greater flexibility and ease of access, they rely on an open and public transmission media over which eavesdropping, unauthorized access, user tracking, and other security threats can be carried out more effectively in comparison with wired networks. Therefore, in this paper, we explore the problems with existing wireless security and propose a general-purpose infrastructure to overcome these problems. We call our infrastructure IRIS, short for Inter-Realm Infrastructure for Security.

RELATED WORK

Kerberos [1] is a security mechanism developed at MIT during the mid-1980's to provide users with a single sign-on to the network and protect authentication information from masquerading. Kerberos, however, only supports symmetric cryptography; hence, it does not scale well to large distributed environments. As a result, a number of Kerberos extensions have surfaced. SESAME (Secure European System for Applications in a Multi-vendor Environment) [2] is an extension to Kerberos that provides additional services such as the use of digital signatures for login and the handling of access control privileges.

Rather than focus at the application layer like Kerberos and similar technologies, SSL (Secure Socket Layer) and TLS (Transport Layer Security) [3] implement security over TCP while IPsec [4] improves security to IP, e.g., authenticating IP packets and providing data confidentiality and authentication to IP's packet payload. Figure 1 shows the relative location of the above security mechanisms (shaded in gray) in the protocol stack.

These aforementioned mechanisms are then used by the commercial sector to provide "secure" communication services. For instance, NTT DoCoMo provides a wireless Internet service called iMode [5,6] based on proprietary protocols that rely on transport layer security (TLS), also known as SSLv4.

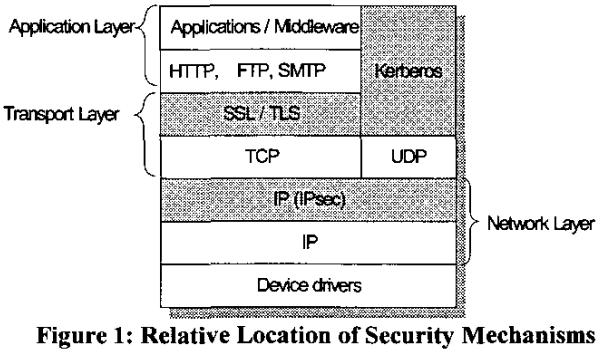

Figure 1: Relative Location of Security Mechanisms in the Protocol Stack

0-7803-7625-0/02/$17.00 ©2002 IEEE.

The 3rd-generation mobile system UMTS (Universal Mobile Telecommunication System) [7] focuses on designing an "all-IP" architecture for voice communication and Internet connectivity, and hence, uses IP security (IPsec) to secure network communication.

The Wireless Application Protocol (WAP) [8] delivers information and services to wireless devices such as mobile phones and handheld devices. WAP defines a suite of protocols at the network, transport, session, and application layers. For example, wireless transport-layer security (WTLS) provides critical security services for mobile devices at the transport layer.

The fundamental problem with all the above systems is that the security mechanisms were primarily devised to retrofit security into the Internet and existing distributed systems.

A. Lack of Reconfigurability

In the coming decade as different generations of mobile communication come and go, mobile devices with significantly different capabilities will co-exist. Imposing a fixed standard or fixed protocol for securing wireless communication leads to systems that are inflexible. Furthermore, such systems can become unusable whenever a security flaw is discovered in the protocol or in any one of the employed cryptographic algorithms, e.g., GSM's A5/1 encryption algorithm [9] and the 128-bit version of WEP, which is employed in wireless LANs [10]. Lastly, many existing approaches offer security as an "all-or-nothing" option.

For the above reasons, ubiquitous computing environments need security services that can be dynamically reconfigured, thus allowing them to adapt to different scenarios, security requirements, and computing resources. Therefore, instead of having a single security protocol that is hardwired into a mobile device, we propose an infrastructure that supports multiple security protocols, thus yielding greater flexibility and compatibility. This infrastructure will provide a thin middleware layer that allows applications and services to invoke the needed security services in a general fashion.

B. Security Gaps

Security gaps appear when a secure session terminates prematurely [5]. Such terminations occur in ubiquitous computing environments due to the multimode nature of the communication link between the mobile device and its final destination, resulting in security gaps that can expose sensitive data. For instance, WAP-enabled devices access Internet services via a WAP gateway. To establish a "secure" connection, the wireless device and the WAP gateway establish a session using the WTLS protocol, as shown in Figure 2. An SSL session is then established between the WAP gateway and the application server providing the requested service. Because of the premature termination and the re-establishment of a secure session, data resides in an insecure state on the WAP gateway.

Similarly, although the specifications of iMode are proprietary, strong speculation exists that iMode also introduces security gaps by establishing two separate TLS sessions, the first between the mobile device and the iMode server and the second between the iMode server and the application server [5].

Even worse, a wireless sensor network that interfaces with the Internet via a base station does *not* employ any security mechanisms, thus leaving its wireless network (similar to the one in Figure 2) and its base station (similar to the default WAP gateway in Figure 2) unsecured and vulnerable to attack. Why does such a situation exist? Today's security mechanisms generally provide heavyweight, "all-or-nothing" solutions, which cannot be run on a resource-constrained sensor in a wireless sensor network.

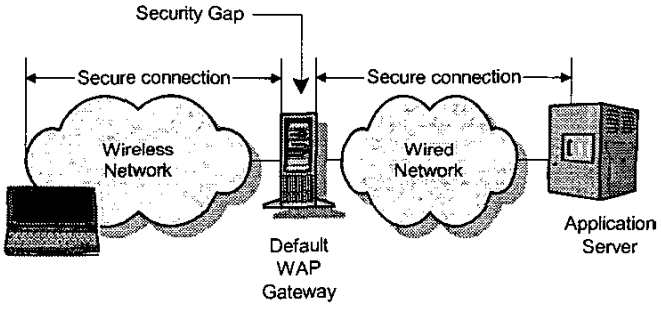

Figure 2. The WAP Security Gap

3G's all-IP network for mobile devices intends to rely on IPsec to secure communications. However, 3G's architecture generally uses IPsec only between different networks and is not truly end-to-end. In fact, in some cases, the deployment of IPsec will only occur between the visited and home networks (Figure 3). We argue that security gaps are introduced whenever data packets have to pass through different network *realms* [11], i.e., heterogeneous environments. Special devices at the realm boundaries exist to handle the diversity between realms. These devices transparently fix packet flows between endpoints, handle data transition between realms, and provide mobility support, address translation, packet

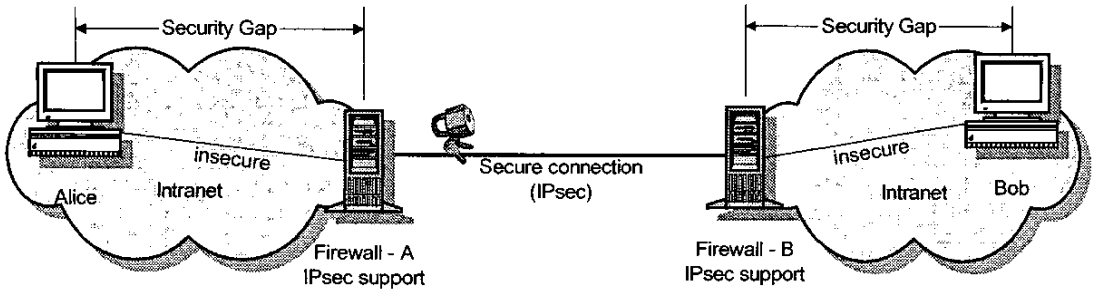

Figure 3. Security Gaps in Current IPsec Implementations

filtering, and data compression. Such special devices are referred to as middleboxes. Examples of middleboxes include WAP gateways, iMode servers, firewalls, proxies, network address translators (NATs), and base stations for wireless sensor networks. Unfortunately, all existing security protocols do *not* take into account the existence of these middleboxes.

These middleboxes are harmful as they often violate the end-to-end nature of conventional Internet applications and hinder the operation of existing and new end-to-end protocols [8]. One common example is the Network Address Translation protocol (NAT) [12,13]. NAT allows a private network of devices with non-global IP addresses to connect to the Internet by sharing a limited number of global IP addresses. The NAT gateways translate between internal IP addresses and corresponding global addresses.

This simple address translation breaks many upper-layer protocols. For example, the control connection for `ftp` transmits the IP address and TCP port to use for the data connection which is made in the opposite direction [14]; therefore, for `ftp` to work, the NAT gateway has to intercept the IP address and TCP port values and modify them. Additionally, since NAT requires modifications to the packet headers, security mechanisms like IPsec's Authenticated Headers (AH) [4] cannot be used in conjunction with NAT.

C. Communication-Protocol Dependence

Many existing security protocols depend on a particular communication protocol. This dependency limits their portability to other networking infrastructures. For example, IPsec is inherently dependent on IP; however, many wireless environments do not use IP for communication, e.g., WAP and wireless sensor networks [15].

IRIS ARCHITECTURE

We advocate a cyber-security infrastructure that not only supports advance negotiation and establishment of a secure session between the endpoints; but also enables the discovery of middleboxes and allows an endpoint to negotiate security requirements and request specific functionality from these middleboxes. Further, it features support for negotiating and establishing end-to-end and/or hop-to-hop security associations. Such a cyber-security infrastructure has applicability to general Internet environments as well as virtual supercomputers and wireless sensor networks. What all these environments have in common is support for multiple access points, thus potentially allowing ubiquitous access to services, services that can be accessed anytime and anywhere using high-end machines, low-end machines, or even mobile devices, like handheld computers and mobile phones.

This puts an additional burden on the cyber-security infrastructure, requiring it to be able to adapt to environments with scarce resources and evolve once more resources become available. Additionally, the infrastructure should not be inherently dependent on IP as there exist many other environments that do not necessarily use IP for communication, e.g., wireless telephony and distributed sensor networking [15].

Figure 4 shows our proposed inter-realm infrastructure for security (IRIS), which incorporates greater flexibility and adaptability as well as the ability to capture the dynamics and agility of mobile environments. IRIS adapts to environments with extreme conditions and scarce resources, e.g., sensor networks, and evolves by providing additional functionality as more resources become available. Consequently, IRIS must be able to support multiple security mechanisms and negotiate security requirements.

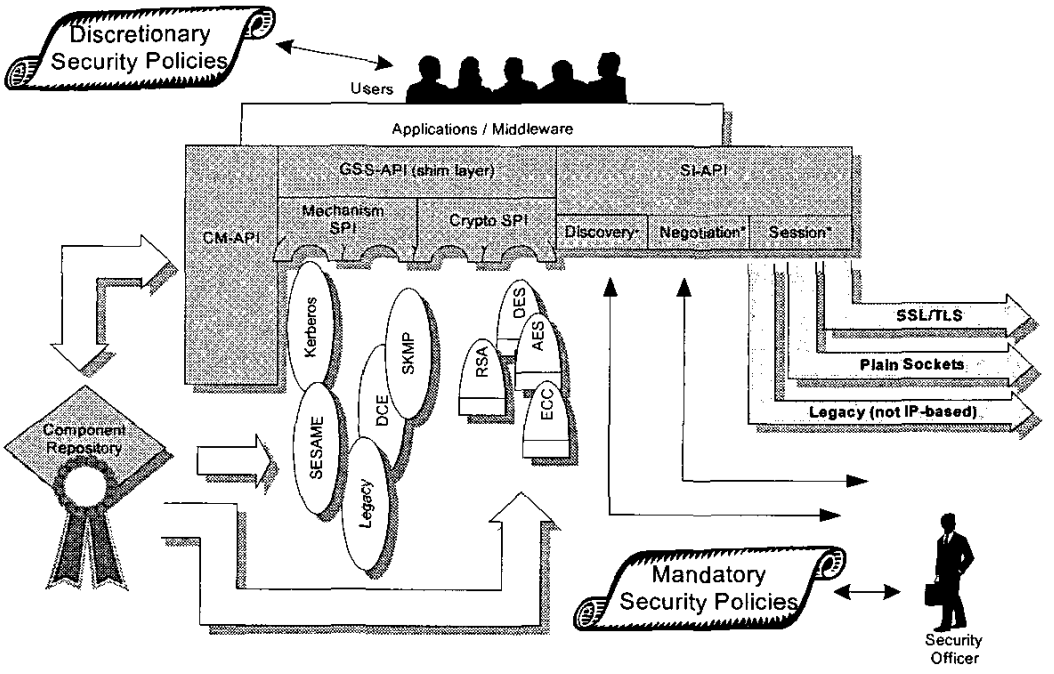

Figure 4. System Architecture

Initially, we base the security services of IRIS on available and proven technologies, thereby granting the IRIS infrastructure with flexibility and secure interoperability with existing systems. We, however, enhance these technologies with a component-based design that includes the discovery of middleboxes and the negotiation of security requirements in order to ensure secure communication across different realms, and hence, eliminate security gaps.

A. Exporting IRIS

As part of a thin middleware layer (or shim) that interfaces with applications, IRIS exports three sets of APIs that are available for an applications' use: GSS-API (Generic Security Services API [16,17]), CM-API (Component Management API), and SI-API (Session Initiation API). These APIs make up the lightweight core of IRIS, which can be pre-loaded into all devices. We designed the core to be small enough to fit into existing mobile devices. Additional functionality can then be loaded on demand and plugged into the core.

B. GSS-API

The GSS-API exports a uniform, generic interface for providing security services that is independent from both the underlying security mechanism and the programming environment. Examples of underlying security mechanisms include SPKM (Simple Public-Key Management Protocol) and Tiny SESAME [18], a lightweight version of a Kerberos extension. The GSS-API abstraction enables security mechanisms to be removed, added, and updated without affecting the applications. Moreover, GSS-API security services are independent from the communication protocol suite being used. This foresight in design will allow IRIS to work across heterogeneous domains, e.g., WAP-based wireless and IP.

The GSS-API can be extended to support an arbitrary list of underlying mechanisms. Such extensions can occur dynamically such that the necessary security protocols and cryptographic functions are loaded on demand. Hence, only the functionality needed at a given time is loaded. This method facilitates the incorporation of new security technologies and bug fixes as they become available.

To simultaneously support multiple security mechanisms and query available mechanisms and load/unload them as necessary, we follow the recommendation of [13] where the GSS implementation consists of two parts: (1) the GSS-API shim layer, which does not provide any security but exports a standardized security interface and (2) underlying security mechanisms and supporting cryptographic profiles, which are added to implement the actual security services. SPIs (Service Provider Interfaces) allow

the GSS to locate and query the different security mechanisms and cryptographic functions.

C. Loadable Components & Component Repository

IRIS implements security mechanisms and cryptographic functions as pluggable components, components that are loaded whenever their functionality is needed. Such a component-based architecture enhances the adaptability of IRIS, allowing it to unload unnecessary functionality to compensate for shortages in resources and re-load that functionality when resources become more available.

IRIS stores the components that implement additional security mechanisms in a *component repository*. To prevent the loading of malicious components, a trusted certificate authority certifies the repository and digitally signs its components to prove their authenticity, and hence, protect them against tampering. Mobile devices can then connect to the repositories to download new functionality or to update their existing security services.

D. CM-API

Because security mechanisms are represented as components, IRIS provides a Component Management API (CM-API) to facilitate the management, loading, unloading and reconfiguration of components as well as the validation of component repositories. The CM-API exposes an interface for the security-aware applications to manage available components.

The CM-API is also utilized by the GSS-API to load and unload functionality on demand. End users and applications then use the CM-API to express discretionary security policies by tagging the loadable components as required, preferred, allowed, or prohibited. This gives end users and applications the flexibility in defining additional security requirements.

E. SI-API

The Session Initiation API (SI-API) provides a session-layer library for applications. It can be used for the discovery of middleboxes and for negotiating security requirements between these middleboxes. This communication session can be based on plain sockets (with GSS-API services to secure transmitted data), or it can use SSL (or SSL-based) protocols for interoperability with existing systems. The implementation of such protocols can be loaded dynamically as well.

The SI-API exports a simple session library for applications' use, as the following pseudo-code illustrates:

```
  //sample application
…
s = session_new();
  // callback function to handle
  // authentication, etc.
session_register_ui_callback(s,
                &my_user_prompt);
session_connect(s, uri, policy, RELIABLE |
                CONGESTION | HOP_ENCRYPTION);
sock = session_fileno(s);
  //if we need to send an address to the
  //other end
t = session_get_my_opaque_addr_token(s);
write(sock, t);
write(sock, …);
…
```

In the pseudo-code above, `session_connect()` triggers the necessary discovery and negotiation and calls the UI callback as needed, for authentication and other purposes.

F. Discovery of Middleboxes

IRIS discovers middleboxes by using a session-signaling protocol [7]. Requirements for the discovery of middleboxes are addressed in [10]. We give an overview of how such a protocol works in IRIS.

If two endpoints (*A* and *B*) decide to communicate securely, the sender *A* initiates the session-layer signaling protocol by sending a special "discover" request along the path to the end destination (see Figure 5). In this scenario, we assume that endpoint *A* supports security mechanism *X*. Middleboxes along the path reply to the "discover" request to indicate their presence and their security requirements, if any. In the example depicted in Figure 5, the first middlebox along the path is a wireless gateway.

The gateway supports mechanisms *Y* and *Z* and requires all outbound traffic to use one of these mechanisms. The gateway responds to the "discover" request announcing its existence and the supported security mechanisms (*Y* and *Z*). Because no common mechanisms exist between endpoint *A* and the gateway, only partial negotiations take place at this time. Proceeding with the discovery process, eventually endpoint *B* will respond and communicate its presence and supported mechanisms (*X* and *Y*).

Now, endpoint *A* can negotiate the establishment of two layers of security: an end-to-end security session

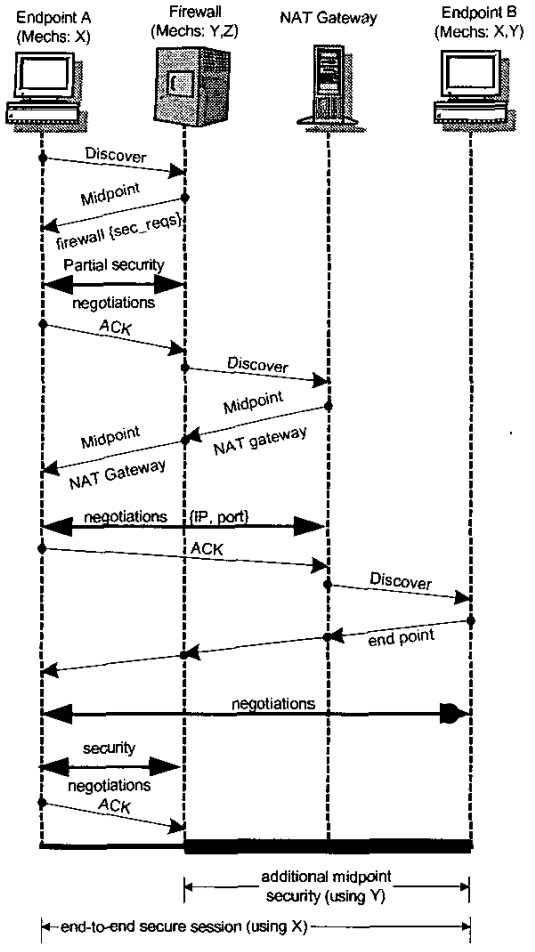

Figure 5. Discovery of Middleboxes

established at endpoint *A*, using mechanism *X*, with the intention to terminate only at endpoint *B*. The other layer is another secure session that is established from the wireless gateway to endpoint *B*, using mechanism *Y*, to meet the additional security requirements imposed by the gateway on outbound traffic. Note that no security gaps are introduced in this scenario.

ACKNOWLEDGMENT

The author gratefully acknowledges discussions with Jalal Al-Muhtadi on wireless security mechanisms and with Mike Fisk on middlebox discovery.

REFERENCES

[1] B. Neumann and T. Tso, "Kerberos: An Authentication Service for Computer Networks," *IEEE Communications Magazine*, 32(9):33-38, September 1994.

[2] P. Kaijser, T. Parker, and D. Pinkas, "SESAME: The Solution to Security for Open Distributed Systems," *Computer Communications*, 17(7): 501-518, July 1994.

[3] T. Dierks and C. Allen, "The TLS Protocol," January 1999, RFC 2246.

[4] S. Kent and R. Atkinson, "Security Architecture for the Internet Protocol," November 1998, RCF 2401.

[5] P. Ashley et al., "Wired versus Wireless Security: The Internet, WAP, and iMode for E-Commerce," Submitted to the *17th Annual Computer Security Applications Conference (ACSAC 2001)*.

[6] iMode, *All About iMode Index*, NTT DoCoMo, http://www.nttdocomo.com/i/index.html.

[7] UMTS, 3rd Generation Partnership Project (3GPP, UMTS Specification), http://www.3gpp.org.

[8] The WAP Forum, http://www.wapforum.org.

[9] A. Biryukov and A. Shamir, "Real-Time Cryptanalysis of the Alleged A5/1 on a PC," Preliminary Draft, December 1999.

[10] A. Stubblefield, J. Loannidis, and A. Rubin, "Using the Fluhrer, Mantin, and Shamir Attack to Break WEP," AT&T Labs Technical Report TD-4ZCPZZ, August 2001.

[11] M. Fisk and W. Feng, "Interactions of Realm Boundaries and End-to-End Network Applications," Los Alamos Unclassified Report (LA-UR) 00-3631.

[12] K. Egevang and P. Francis, "The IP Network Address Translator (NAT)," RFC 1631, May 1994.

[13] P. Srisuresh and M. Holdrege, "IP Network Address Translator (NAT) Terminology and Considerations," RFC 2663, August 1999.

[14] J. Postel and J. K. Reynolds, "File Transfer Protocol," RFC 959, October 1985.

[15] D. Estrin, R. Govindan, J. Heidemann, and S. Kumar, "Next Century Challenges: Scalable Coordination in Sensor Networks," *Proceedings of the 5th International Conference on Mobile Computing and Networks (MobiCom'99)*, August 1999.

[16] J. Linn, "Generic Security Service Application Program Interface," RFC 1508, September 1993.

[17] J. Linn, "Generic Security Service Application Program Interface," Version 2, RFC 2078, January 1997.

[18] J. Al-Muhtadi, D. Mickunas, and R. Campbell, "A Lightweight Reconfigurable Security Mechanism of 3G Mobile Devices," *Proceedings of the International Conference on 3rd Generation Wireless and Beyond (3Gwireless '01)*, May 2001.