Laureate Online Education
Information Security Engineering

Information Security Engineering Seminar for Week 6

Network and Internetworking security, digital content protection

Internet security is a fashionable and fast moving field with attacks on the Internet often making the headlines of widely distributed newspapers such as The New York Times. Unfortunately, these reports are often misleading. This week we will be studying security issues related to the Internet - concentrating on protection mechanisms. In particular, we will cover the following topics: the most common attacks, distributed denial of service attacks, intrusion detection, firewalls, IETF security protocols, and XML security. For several sections of this lecture, we will assume that you are familiar with network protocols, e.g. you have taken the CC module, or have some prior knowledge of computer communication. Please also re-read Seminar 1 materials related to the OSI/Internet Security Architecture, security services and mechanisms.

The most common attacks

Most reported attacks on the Internet fall into the following three categories: software implementation faults such as buffer overflow, protocols failure, and password guess. The textbook lists the Internet's Top 10 vulnerabilities. Up to now the most important attacks on the Internet have been the Internet Morris worm, SYN flooding, and Distributed Denial of Service attacks (for definitions see Section 18.2.2 in the textbook). In the following, we will briefly describe Distributed Denial of Service Attacks since it is easy to find published information on SYN flooding and the Morris worm in textbooks. For more information on other forms of attack, take a look at the textbook and the CERT homepage.

Computer Viruses

The term *computer virus* is often used to indicate any software that can cause harm to systems or networks. Often, though, people do not include certain malicious software, such as Trojan horses and network worms, in the computer virus family (see Section 18.4 of the textbook). In our discussion, a *computer virus* refers to any code that causes computer or network systems to behave in a different manner to that which is intended. A Trojan horse program is a useful or apparently useful program or a shell script containing hidden code that performs some unwanted function. A simple example of a Trojan horse program might be a *telnet* program. When a user invokes the program, it appears to be performing telneting and nothing more, however it may also be quietly changing file access permissions. Some Trojan horse programs are difficult to detect - for example a compiler on a multi-user system that has been modified to insert additional code into certain programs when they are compiled (this idea was first observed by Thompson). The hidden code of a Trojan horse program is placed there deliberately by the program's author. Generally, the hidden code in a computer virus program is added by another program - itself a computer virus. Thus a typical characteristic of a computer virus is to copy its hidden code to other programs, thereby infecting them.

Generally, a computer virus exhibits three characteristics: a *replication* mechanism, an *activation* mechanism, and an *objective*.

The replication mechanism performs the following functions. It searches for other programs to infect. Then, when it finds a program, *possibly* it determines whether the program has been previously infected, inserts hidden instructions somewhere in the program, modifies the execution sequence of the program's instructions such that the hidden code will be executed whenever the program is invoked, and sets up some flags indicating that the program has been infected. The flag may be necessary because without it, programs could be repeatedly infected and grow noticeably large.

The activation mechanism checks for the occurrence of some event. When the event occurs, the computer virus executes its objective, which is generally some unwanted, harmful action.

Anti-virus tools perform three basic functions: *detecting, identifying*, or *removing viruses*. Detection tools perform proactive detection, active detection, or reactive detection. That is, they detect a virus before it executes, during execution, or after execution. Identification and removal tools are more straightforward in their application.

For the detection of viruses, there are five classes of techniques: signature (note that the meaning of signature here is completely different from those signatures used in public key cryptography - by virus signature, we mean a binary string that is used to identify a specific virus) scanning and algorithmic detection, general purpose monitors, access control shells, checksums for change detection, heuristic binary analysis, and emulation detection. A common class of anti-virus tools employs complementary techniques of signature scanning and algorithmic detection. This class of tool is known as a scanner. Scanners are limited intrinsically to the detection of known viruses. In signature scanning an executable is searched for by a selected binary code sequence, called a *virus signature*.

General-purpose monitors protect systems from the replication of viruses by actively intercepting malicious actions. Access control shells function as part of the operating system, much like monitoring tools. Rather than monitoring for virus-like behavior, the shell attempts to enforce an access control policy for the system. Change detection works on the theory that executables are static objects. Modification of an executable therefore implies a possible virus infection. However, this theory has a basic flaw: some executables are self-modifying. Heuristic binary analysis is a method whereby the analyzer traces through an executable looking for suspicious, virus-like behavior. If a program appears to perform virus-like actions, a warning is displayed. Indeed, the signature scanner is the most commonly used technique for detecting viruses. Note that at the beginning of this section we mentioned that at the end of the replication mechanism, the virus will generally put a flag in the infected program (often incorporating it into the virus signature). Otherwise the program may be repeatedly infected, and the size may increase geometrically. Hence the virus will be detected easily.

The authors of computer viruses always try to design viruses that are immune from those anti-virus software programs on the market. A *polymorphic virus* creates copies during replication that are functionally equivalent but have distinctly different byte streams. To achieve this, the virus may randomly insert superfluous instructions, interchange the order of independent instructions, or choose from a number of different encryption schemes. This variable quality makes the virus difficult to locate, identify, and remove. This kind of virus can be thought as a static signature-free virus. However, to my knowledge, all known polymorphic viruses can be detected using algorithmic detection (or emulation detection) due to their dynamic signatures.

Distributed Denial of Service Attacks (DDoS)

In February 2000, we saw several very successful denial of service attacks on large e-commerce Internet sites such as CNN, Yahoo, and eBay. These attacks were launched from many compromised hosts that acted as daemon or zombie machines. Each zombie carried out a denial of service attack resulting in a large distributed and amplified attack.

The key tools exploited in these attacks were based on tfn or trinoo which are two pieces of code widely distributed in the hackers' world. These tools implement a distributed network denial of service tool capable of waging ICMP flood, SYN flood, UDP flood, and Smurf-style attacks, as well as providing an 'on demand' root shell bound to a TCP port. Indeed, trinoo (also known as trin00) has been well known since the attack on the University of Minnesota's network in August 1999. Generally, a trinoo distributed denial of service attack begins when the attacker compromises one of many master machines. The attacker will put vulnerability scanning tools, root kits (to conceal malicious codes and connections), master and trinoo daemon codes, and a list of vulnerable hosts in the compromised master machine. The master machine then scans for systems that exhibit the vulnerabilities that trinoo can exploit. A list of vulnerable systems is then passed to an exploit script that compromises each of them, sets up and connects a listening shell (with TCP port 1524), and compiles a list of successful 'owned' systems. The list of 'owned' systems is passed to another script that installs the trinoo daemon and a root kit via the open TCP port 1524. This completes the construction of the 'trinoo' network (it is strongly recommended that you take a look at information about this on David Dittrich's web site - links are provided at the end of this lecture). The attacker can then connect to master machines via telnet (at port 27655) and enter a password ('betaalmostdone' for the version examined by Dittrich). Master machines then pass command lines to daemons via UDP port 27444. Daemons respond to masters on UDP port 31335. Master machines form a list of live daemons by listening for the text "**HELLO**" in the data portion of the daemons' UDP packets. The attacker can then send the following commands to master machines: quit, dos IP (to launch a DDoS attack on the address IP), mdos <IP1:IP2:IP3> (to launch multiple DDoS attacks), and bcast (to form a list of started daemons). Master machines can send the following commands to daemons: aaa password IP (DoS attack the address IP by sending UDP packets to random UDP ports 0-65534), bbb password N (period of time in seconds to run DoS attack), rsz N (set size of UDP packets to N bytes), and dle (to shutdown the daemon).

Using the current TCP/IP protocol suite, the following three methods can be used to prevent your machine from launching such attacks. However, it is almost impossible to prevent your network from being DDoS attacked.

1. Secure your server. For example, by installing patches on your operating system if it has vulnerability holes that could be used by these tools for the initial compromise, using firewalls, and monitoring for intruders.
2. At your router, install mechanisms for Ingress (RFC2267) or Egress filtering. That is, ensure that all packages that will go to the Internet have your network address as the source address (this will prevent attacks which forge source addresses).
3. Packets directed at the broadcast address from outside your net should be blocked at the border. The command to do this for Cisco routers is ”no ip directed-broadcast”. This will prevent your network being used for Smurf attacks. Smurf attacks send packets to a ’Smurf amplifier’ network. This is any network that allows in such packets. These packets come from outside the amplifier net, but are directed to its broadcast address. Such packets have a forged source address, to direct all replies from all hosts on the amplifier network to the victim. Each such packet gets repeated by every machine in the net, amplifying the effect of the attack.

Intrusion Detection Systems (IDS)

Intrusion Detection is the art of detecting inappropriate, incorrect, or anomalous activity in a computer system. Generally there are two kinds of Intrusion Detection System (IDS). The first is a host-based IDS that operates on a host to detect malicious activities on that host. The second is a network-based ID system that operates on network data flows. Network-based and host-based IDS have been designed for different purposes, and perhaps it is better to use a combination of both systems.

Host-based IDS

Host-based IDS (HIDS) use log files and the system's auditing agents as sources of data (for example, in UNIX systems, the ’syslog’ file can be used for such a purpose). System administrators are generally responsible for monitoring HIDS. IDS administrators should be trained professionals who are familiar with the host machine, network connections, users and their habits, and all software installed on the machine. In particular, IDS administrators should know the way that the machine is supposed to be running and the programs that are legitimate. Host-based IDS involves looking at the communications traffic in and out of a single computer, checking the integrity of system files, and watching for suspicious processes. There are two kinds of host-based intrusion detection systems: personal firewalls and agent-based software. Personal firewalls can be configured to look at all network packets, connection attempts, or login attempts to the monitored machine. Host-based agents can monitor accesses and changes to critical system files and changes in user privilege. Examples of host-based IDS include:

1. TCPWrappers (http://coast.cs.purdue.edu/pub/tools/unix)
2. GFI LANguard S.E.L.M (http://www.gfi.com/lanselm/?adv=142&loc=35&adclickid=15279873)
3. ELM 3.0 TNT software (http://www.tntsoftware.com/Products/ELMLogManager.aspx)
4. IBM ISS, (http://www.iss.net/)
5. Tripwire (www.tripwiresecurity.com).

Since UNIX has plenty of log and audit files, it is relatively easy to create host-based intrusion detection systems by writing code to automatically analyze log files and alert system administrators.

Network-based IDS
Network-based IDS monitor the traffic on its network segment as a data source, checking the packets on the network as they pass by a sensor. Packets are considered to be of interest if they match a signature (note that 'signature' here means something completely different from signatures used in public key cryptography - here signature means a specific binary string used to identify a package which is similar to the meaning of signature in virus scanning). Here we are mainly interested in string signatures, port signatures, and header condition signatures. String signatures look for a text string that indicates a possible attack. An example string signature for UNIX might be: "cat "+ +" > /.rhosts". Port signatures simply watch for connection attempts to well-known, frequently attacked ports. Examples of these ports include telnet (23), FTP (21/20), and IMAP (TCP port 143). If the site doesn't use any of these ports, then incoming packets to these ports are suspicious. Header signatures watch for dangerous or illogical combinations in packet headers. For example, if both the SYN and FIN flags for a TCP packet are set, then it means the requestor wishes to start and stop a connection at the same time. Examples of network-based ID systems vendors include:

1. Cisco Secure IDS (www.cisco.com)
2. Dragon Enterasys (http://www.enterasys.com/products/advanced-security-apps/dragon-intrusion-detection-protection.aspx)
3. Snort ISS (http://www.snort.org/)

A good IDS capability will use both host- and network-based systems. As we have seen above, there are numerous intrusion detection systems on the market. Since it would of course be advantageous if all intrusion detection systems could share data on attacks, compatibility among them is becoming more and more important. The IETF Intrusion Detection Exchange Format (idwg) working group is working to define such a uniform data format. Their purpose is: "*… to define data formats and exchange procedures for sharing information of interest to intrusion detection and response systems, and to management systems which may need to interact with them*."

Firewalls

As we have noticed in the previous section, firewalls have become one of the most important tools for intrusion detection. In this section we will further our study of network firewalls. An Internet firewall serves the same purpose as firewalls in buildings and cars: to protect a certain area from the spread of fire and a potentially catastrophic explosion. The spread of fire from one part of a building is controlled by putting up retaining walls, which help to contain damage and minimize overall loss and exposure. An Internet firewall is no different. It uses such techniques as examining Internet addresses on packets or ports requested on incoming connections to decide what traffic is allowed into a network.

The most common firewall uses packet filtration, which blocks specified IP services (run on specific port numbers) from crossing the gateway router. Proxies are also a common method of protecting a network while allowing legal users to enter. Proxy services are typically a software solution run on top of a network operating system, such as Unix, Windows NT, or Novell Netware.

In the following we will discuss four firewall design architectures. There are many variations on the four that you may already have seen implemented, and certainly we are omitting several of the most complex and advanced architectures.

Packet restriction or packet filtering routers

Routers and computers that conduct packet filtration choose to send traffic to a network based on a predefined table of rules. The router does not make decisions based on what's inside the packet's payload, but rather on where it is coming from and where it is destined. It only considers that if the packet matches a set of parameters, it should take appropriate action to either allow or to deny the transit. These allow and deny tables are set up to conform to the overall network security policies put in place by the network administrator or security coordinator.

Bastion host

A bastion host or screening host, as it is sometimes called, uses both a packet filtering mechanism provided by the router plus a secured host. A secured host is one that has had its operating system and major services combed by a security expert. The primary security is provided by a packet filtering router, and the secured host is used to stage information flow in each direction. The bastion host is a security-checked machine that is connected to the Internet using the same method as other machines. The gateway allows traffic to pass to it in a less restricted fashion. Bastion hosts are typically used in combination with filtering routers because simple packet filtration systems can't filter on the protocol or the application layer.

DMZ or perimeter zone network

A popular ploy to separate large corporate internal networks from the hostile environment of the Net is to erect a 'routing network' over which all inbound and outbound traffic must travel. Huge installations generally have such networks already set up so that they can effectively separate local traffic from metropolitan and wide-area or worldwide traffic. As you might have guessed, a routing network consists only of routers, including those both internally and externally connected, and usually goes by the term 'backbone'. You might be wondering why the term DMZ is sometimes used interchangeably for a perimeter zone network. DMZ stands for ’demilitarized zone’ and serves the same purpose as it does in areas of geographical conflict: it is a buffer zone between two hostile parties that must coexist in close proximity.

Proxy servers
Proxies act much like bastion hosts. Let’s illustrate the difference with an example. A bastion host is typically set up to act as the ’delivery point’ for email inbound from the Internet. In a mail system, a DNS MX (Domain Name Service Mail eXchanger) refers to a domain name's mail server information. The MX record provides information about which servers handle the domain name's email storage, distribution, or redirection. In particular, MX records are used to deliver mail to users in the domain. When you send mail to someone, your mail typically goes from your email client to an SMTP server. The SMTP server then checks for the MX record of the domain in the email address. For example, with ’joe@mydomain.com’, it would look for the MX record for mydomain.com. A DNS MX is traditionally set up to point traffic to the bastion for delivery. From there, the bastion may re-deliver the mail to an interior mail host (which it can see due to its position in the firewall), or it could hold onto the mail, waiting for the client to read it with a POP mail client. A whole selection of different firewalls can be constructed in this manner. By contrast, a proxy service is more of an ’in-transit’ checkpoint than an information staging area. The proxy pretends to be one end of a connection, but protects the true sender or recipient from unwanted traffic. The service that presents the greatest challenge for security managers is the standard File Transfer Protocol (FTP). In a typical FTP session, the user’s ftp command opens a control channel to the target machine at port 21. The actual data (a file transfer or listing from a directory command) is sent over a separate data channel. The server uses port 20 for this data channel. By default, the client uses the same port number as is used by the control channel. The FTP protocol specification suggests that a single channel be created and kept open for all data transfer during the session. Most common implementations create a new connection for each file. Thus a passive FTP session can be established through proxy servers (using the control and data ports [20 and 21] for actual data transit rather than one greater than 1023).

Defense of your hosts with Freeware

OS hardening
To protect against misconfiguration-based attacks, install the very good hardening utility Bastille (http://sourceforge.net). Bastille essentially closes all the doors left open in a default installation.
Network services access control
Install Wietse Venema’s TCP Wrapper (ftp://ftp.porcupine.org/pub/security/index.html). This is a simple tool, simple to install, simple to configure and simple in operation. It is an access control list for services run under the control of the Internet daemon.
IDS
Get the excellent Intrusion Detection Tool Snort (http://www.snort.org/). There are both Linux version and Windows version. It will let you see what kinds of messages are observed by your network card and let you to write your own rules for IDS. It is almost infinitely configurable.

Firewall

Try Shorewall (http://shorewall.net/), a freeware firewall/gateway based on linux iptables/ipchains. You may also try Astaro's Security Linux (http://astaro.com/), which is a freeware sateful inspection gateway that provides proxy and VPN services.

Secure Remote Access

Never try telnet or ftp. Install OpenSSH (http://www.openssh.com/) for remote access tools (there are both Linux and Windows versions).

Penetration Testing

After your system is set up, now try to break it. Install Nessus (http://www.nessus.org/). It tests each port to determine what sort of listener is active.

File Integrity Utility

Finally, once your security suite is complete, install the freeware version of Tripwire (check free download from Tucows http://www.tucows.com/preview/51673). Tripwire takes a "snapshot" of a large number of critical binaries on your system, and stores that information encrypted and in an obscure place.

IETF Security Working Groups

IPSec

As all of us know, the Internet is based on the TCP/IP protocol suite. In the early 1960s, DARPA funded a project that connected universities and research agencies through a network called ARPANET. In 1983, the TCP/IP protocols replaced the original ARPANET NCP protocols. The TCP/IP protocols running this network were open, simple, and easy to use. This network has since grown considerably into what is now called the 'Internet'. The Internet is a collection of networks running the TCP/IP protocol suite. Since IP packets have no inherent security, it is relatively easy to forge IP packet addresses, modify their contents, replay old packets, and inspect the contents of IP packets in transit. Therefore, there is no guarantee that IP datagrams received are (1) from the claimed sender (the source address in the IP header); (2) that they contain the original data that the sender placed in them; or (3) that the original data was not inspected by a third party while the packet was being sent from source to destination. IPSec is a method recently developed by the IETF for protecting IP datagrams. This protection takes the form of data origin authentication, connectionless data integrity authentication, data content confidentiality, anti-replay protection, and limited traffic flow confidentiality.

IPSec provides a standard, robust, and extensible mechanism for providing security for IP and upper-layer protocols (e.g. UDP or TCP). A default, mandatory-to-implement suite of algorithms is defined to assure interoperability between different implementations, and it is relatively straightforward to add new algorithms without breaking interoperability.

IPSec protects IP datagrams by defining a method of specifying the traffic to protect, how that traffic is to be protected, and to whom the traffic is sent. IPSec can protect packets between hosts, between network security gateways (e.g. routers or firewalls), or between hosts and security gateways. Since an IPSec-protected datagram is, itself, just another IP packet, it is possible to nest security services and provide, for example, end-to-end authentication *between hosts* and send that IPSec-protected data through a tunnel which is, itself, protected by security gateways using IPSec.

This method of protecting IP datagrams or upper-layer protocols uses one of the IPSec protocols, the Encapsulating Security Payload (ESP) or the Authentication Header (AH). AH provides proof-of-data origin on received packets, data integrity, and anti-replay protection. ESP provides all that AH provides - in addition to optional data confidentiality and limited traffic flow confidentiality. One might therefore be tempted to ask, “Why use AH?” That’s a good question, and is a hot topic of debate in the security community. One subtle difference between the two is the scope of coverage of authentication.

It should be noted that the ultimate security provided by AH or ESP is dependent on the cryptographic algorithms applied to them.

The security services that IPSec provides requires shared keys to perform authentication and/or confidentiality. A mechanism to manually add keys for these services is mandatory to implement. This ensures interoperability of base IPSec protocols. Of course, manual addition scales poorly so a standard method of dynamically authenticating IPSec peers, negotiating security services, and generating shared keys is defined. This key management protocol is called IKE - the Internet Key Exchange.

A virtual private network is a way of simulating a private network over a public network, such as the Internet. It called virtual because it depends on the use of virtual connections - that is, temporary connections that have no real physical presence - consisting instead of packets routed over various machines on the Internet on an ad-hoc basis. Secure virtual connections are created between two machines, a machine and a network, or two networks. IPSec protocols are an essential part of Virtual Private Networks (VPN) technologies currently available. You should read the RFC 2401 (IETF), which specifies the architecture of IPSec. The location of IPSec within the TCP/IP suite is demonstrated in Figure 5:

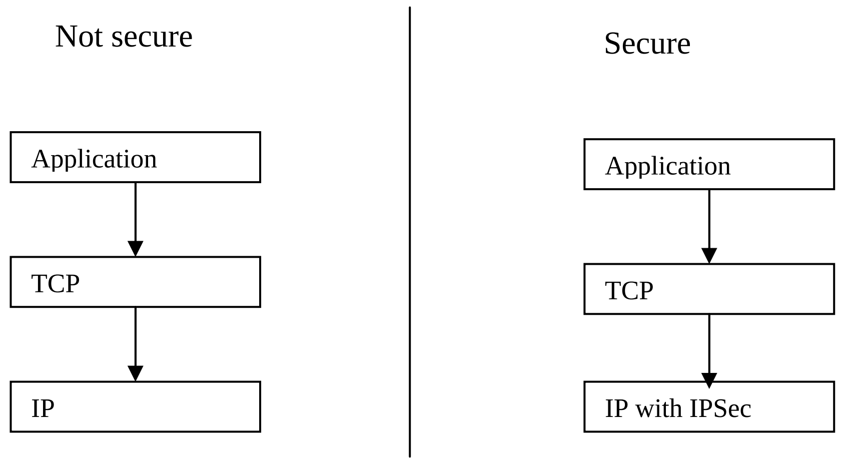

Figure 5

SSL and TLS

Netscape Communications began considering Web security while developing its first Web browser and designed the Secure Sockets Layer protocol (versions 1.0 to 3.0). Starting in May 1996, the Internet Engineering Task Force (IETF) began to take responsibility for SSL development. The IETF renamed SSL as Transport Layer Security (TLS), and released the first official TLS specification in January 1999. Support for SSL is now built in to almost all browsers and Web servers. The designers of SSL chose to create a separate protocol just for security (in addition to Internet protocols that already existed), and inserted it between the HTTP application and TCP. Since it is a new protocol, SSL requires very few changes to the protocols detailed above and below. The most basic function that an SSL client and server can perform is to set up a channel for encrypted communications. The architecture is demonstrated in Figure 6:

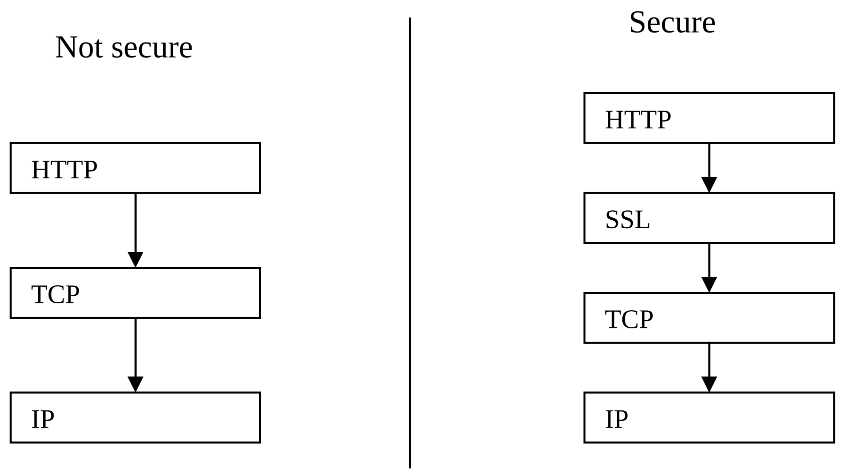

Figure 6

S/MIME

S/MIME is another important protocol suite for content protection. Cryptographic Message Syntax (CMS) is one of the essential protocols in the S/MIME suite. CMS is based on RSA DataSecurity Inc.'s PKCS (public key cryptography standards) standard #7. For those who have further interest in S/MIME, it is referred to as S/MIME RFCs. (We will discuss more details of content protection in XML Digital Signature schemes. There is a close relationship between S/MIME and XML DSIG.)

IETF Security Working Groups

In summary, currently active IETF security working groups include:

Short name	Full WG name
btns	Better-Than-Nothing Security ("http://www.ietf.org/html.charters/btns-charter.html")
dkim	Domain Keys Identified Mail ("http://www.ietf.org/html.charters/dkim-charter.html")

emu	EAP Method Update ("http://www.ietf.org/html.charters/emu-charter.html")
hokey	Handover Keying ("http://www.ietf.org/html.charters/hokey-charter.html")
ipsecme	IP Security Maintenance and Extensions (http://www.ietf.org/html.charters/ipsecme-charter.html)
isms	Integrated Security Model for SNMP ("http://www.ietf.org/html.charters/isms-charter.html")
keyprov	Provisioning of Symmetric Keys (http://www.ietf.org/html.charters/keyprov-charter.html)
kitten	Kitten (GSS-API Next Generation) ("http://www.ietf.org/html.charters/kitten-charter.html")
krb-wg	Kerberos ("http://www.ietf.org/html.charters/krb-wg-charter.html")
ltans	Long-Term Archive and Notary Services ("http://www.ietf.org/html.charters/ltans-charter.html")
msec	Multicast Security ("http://www.ietf.org/html.charters/msec-charter.html")
nea	Network Endpoint Assessment ("http://www.ietf.org/html.charters/nea-charter.html")
pkix	Public-Key Infrastructure (X.509) ("http://www.ietf.org/html.charters/pkix-charter.html")
sasl	Simple Authentication and Security Layer ("http://www.ietf.org/html.charters/sasl-charter.html")
smime	S/MIME Mail Security ("http://www.ietf.org/html.charters/smime-charter.html")
syslog	Security Issues in Network Event Logging ("http://www.ietf.org/html.charters/syslog-charter.html")
tls	Transport Layer Security ("http://www.ietf.org/html.charters/tls-charter.html")
opsec	Operational Security Capabilities for IP Network Infrastructure ("http://www.ietf.org/html.charters/opsec-charter.html")

For WG web pages location see
http://www.ietf.org/html.charters/wg-dir.html#Security%20Area
or http://sec.ietf.org - Security Area Web Page

Public-Key Infrastructure (X.509) Working Group (pkix)
(http://www.ietf.org/html.charters/pkix-charter.html)

The PKIX Working Group is one of the most long lived WG at IETF. It was established in the fall of 1995 with the goal of developing Internet standards to support X.509-based Public Key Infrastructures (PKIs). Initially PKIX pursued this goal by profiling X.509 standards developed by the CCITT (later the ITU-T). Later, PKIX initiated the development of standards that are not profiles of ITU-T work, but rather are independent initiatives designed to address X.509-based PKI needs in the Internet. Over time this latter category of work has become the major focus of PKIX work, i.e., most PKIX-generated RFCs are no longer profiles of ITU-T X.509 documents.

PKIX has produced a number of standards track and informational RFCs. In particular.
RFC 5280 (Certificate and CRL Profile), and RCF 3281 (Attribute Certificate Profile) are recent examples of standards track RFCs that profile ITU-T documents. PKIX will maintain compatibility between ITU-T documents and IETF PKI standards, since the profiling of X.509 standards for use in the Internet remains an important topic for the working group.

PKIX also pursues new work items in the PKI arena which are of sufficient interest for Internet community. Recent developments include defining X.509 Proxy Certificate (RFC3820) and specifying format and protocol for using the trust anchors in the certificate path. Another important group of standards that find their usage in modern highly distributed computing and e-commerce system includes such standards as Online Certificate Status Protocol (OCSP) (RFC 5019), Server-based Certificate Validation Protocol (SCVP) (RFC 5055), Certificate Management Protocol (CMP) (RFC 4210)

It is highly recommend for you to visit the PKIX WG webpage to get impression about the standards and problems in the PKI area.

VPN

As we have discussed in the previous section, a virtual private network (VPN) is a way of simulating a private network over a public network, such as the Internet. It is called 'virtual' because it depends on the use of virtual connections. Secure virtual connections are created between two machines, a machine and a network, or two networks. IPSec is one of the main technologies that is used to implement a VPN. Several standards have been developed for VPN networks, though in all probability, IPSec will become the one universally used. In addition to the IPSec protocols suite, the PPTP (Point-to-Point Tunneling Protocol) has also been used to build VPNs. PPTP is an extension of the standard PPP protocol encapsulating network protocol datagrams within an IP packet. Thus, the tunneling services provided by PPTP are intended to reside on top of the IP layer. In the following, we give a brief example of VPN access via PPTP.

PPTP description

Microsoft Remote Access Services (RAS) allows a network administrator to set up a Windows NT server within a modem bank as a dial-in point for remote users. Authentication for RAS users takes place on the NT server. PPTP was designed to allow users to connect a RAS server from any point on the Internet and still have the same authentication, encryption, and corporate LAN access they would have from dialing directly into it. Instead of dialing into a modem connected to the RAS server, end users dial into their ISPs and use PPTP to set up a call to the server over the Internet. PPTP and RAS use authentication and encryption methods to create a virtual private network.

Building a VPN via ISPs that support PPTP

Now let’s assume that Alice’s ISP uses a remote access switch that supports PPTP and Alice’s corporation network has a PPTP-enabled Windows NT RAS server. Before using PPTP services and building a VPN, Alice needs to configure her profile on her laptop so that it knows the IP address of the RAS server at her corporate office. Then, each time Alice dials into her PPTP-enabled ISP’s POP using Microsoft’s Dial-Up Networking, a PPTP session is started automatically between the ISP’s remote access switch and the corporate office’s NT server. Alice’s PPP session is tunneled through the PPTP stream, and the NT RAS server authenticates her username and password and starts her PPP session. The PPTP session can then tunnel the protocols that dial-up users are allowed to use. Once the PPTP is completed and Alice is authenticated, she has access to the corporate network as if she were on the LAN.

NB: Although most VPN packages themselves don’t implement firewalls directly, firewalls are an integral part of a VPN. The idea is to use the firewall to keep unwanted visitors from entering your network, while allowing VPN users through. If you don’t have a firewall protecting your network, don’t bother with a VPN until you get one - you’re already exposing yourself to considerable risk.

XML Security and Web Services Security

XML has been accepted extensively as the industrial standard for Internet documentation. XML DSIG (XML Digital Signature) and XML ENC (XML Encryption) standards provide authentication and confidentiality syntaxes for XML documents. The XKMS (XML Key Management Scheme) standard is currently in the process of standardization. When this is completed, XKMS will provide a mechanism for processing X.509-based public key certificates (e.g. public key registration, certificate status verification, etc.). In the following, we briefly discuss XML DSIG and XML ENC. However, to understand better these two very important technologies it is recommended to read standards themselves.

The XML security provides a basis for Web Services security to protect SOAP messages that are used in the Web Services interaction. However the Web Services security extends wider that just protecting SOAP based messages and include such standards as WS-Trust, WS-Federation, WS-Policy, and others. Refer to the WS-Security Roadmap web page at http://www.ibm.com/developerworks/library/specification/ws-secmap/

Web Services security architecture has the same philosophy as ISO/Internet security and uses actually the same approach in defining security services interfaces as “orthogonal” to the main services. This is achieved by placing security related attributes and information into the SOAP message header and making security services independent from the main service call which is typically placed into the SOAP message body. In this respect WS-Security services can be also considered as orthogonal to main services and in general arbitrary combined.

XML Signature

We provide here a short overview of the XML Signature format using some simple examples. A full valid XML digital signature is provided in the Appendix of this lecture.

XML Signature has the following structure:

```
<Signature ID?>
      <SignedInfo>
      <CanonicalizationMethod/>
      <SignatureMethod/>
      (<Reference URI? >
            (<Transforms>)?
            <DigestMethod>
            <DigestValue>
      </Reference>)+
</SignedInfo>
      <SignatureValue>
(<KeyInfo>)?
(<Object ID?>)*
</Signature>
```

There are three types of the XML Signature algorithms:

- Enveloped XML Signature.
  An enveloped signature is a signature of either an entire document or a document fragment, where the XML signature will itself be embedded within the signed document. An enveloped signature transform is always used to remove the signature structure from the signing process.
- Enveloping XML Signature
  An enveloping signature is a signature where the signed data is actually embedded within the XML signature structure. The XML signature specification provides the ability for arbitrary XML structures or just simple data object (e.g. code or binary data) to be embedded within a signature, for this express purpose.
- Detached XML Signature
  A detached signature is a signature where the signed entities are separate from the actual signature fragment. The signed entities can be remote XML documents or remote non-XML documents. They can also be XML fragments located elsewhere in the same document as the XML signature, like this takes place when the SOAP message body is signed and the signature is placed in the special field or the message header.

It is important to mention that WS-I Basic Security Profile for Web Services recommends using detached signature and strongly discourages enveloping signature use, enveloped signature can be used but it provides limited functionality for signing the document and managing security components.

XML Signature standard defines a number of transformations required for XML Signature signing and validation:

- Canonicalisation
- Base64

- XPath Filtering
- Envelope Signature Transform
- XSLT Transformation

A canonicalisation is required to ensure the normalised presentation of the XML document in cases when XML format and schema validation allows options, like when using different character encoding or not strict attributes ordering. The canonical form of an XML document is a physical representation of the document produced by the canonicalisation method that implies the following changes. The CanonicalizationMethod element is used to specify what canonicalisation algorithms was used to process the original document:

```
<CanonicalizationMethod Algorithm="http://www.w3.org/TR/2001/REC-xml-
    c14n-20010315" />
```

The SignatureMethod indicates what algorithm was used to calculate the signature. In our case it is RSA with SHA1 as the hash fucntion.

```
<SignatureMethod Algorithm="http://www.w3.org/2000/09/xmldsig#rsa-sha1" />
```

The following Reference element indicates where the to-be-signed (tbs) document is located and what part of the document is signed. The tbs-document could be in the same XML file or in a separate place. For example, the element

```
<Reference URI="http://www.w3.org/TR/xml-stylesheet">
  <DigestMethod Algorithm="http://www.w3.org/2000/09/xmldsig#sha1" />
  <DigestValue>60NvZvtdTB+7UnlLp/H24p7h4bs=</DigestValue>
</Reference>
```

shows that the signature is for tbs-document **http://www.w3.org/TR/xml-stylesheet** (which should be a valid URI address). It also shows that the hash value of this document is **60NvZvtdTB+7UnlLp/H24p7h4bs=** (SHA1 is used to compute the hash value). The URI="" means that If we want to sign the local document to which the signature is attached or jus the element with the attribute Id="subject", the URI attribute will have values URI="" or URI="#subject".

The SignatureValue element holds the signature value:

```
<SignatureValue>juS...=</SignatureValue>
```

It is optional to include the signer's public key and certificate in the KeyInfo element which is a part of the Signature element. For example, the element

```
<KeyInfo>
  <KeyValue>
    <RSAKeyValue>
        <Modulus>uCiu........ ba648=</Modulus>
        <Exponent>AQAB</Exponent>
      </RSAKeyValue>
   </KeyValue>
  </KeyInfo>
```

contains the signer's RSA public key value (the recipient will use this key to verify the signature, and the element

```
      <X509Data>
         <X509SubjectName>CN=Merlin Hughes,O=Baltimore
            Technologies\, Ltd.,ST=Dublin,C=IE</X509SubjectName>
        <X509IssuerSerial>
            <X509IssuerName>CN=Test RSA CA,O=Baltimore
               Technologies\, Ltd.,ST=Dublin,C=IE</X509IssuerName>
            <X509SerialNumber>970849928</X509SerialNumber>
         </X509IssuerSerial>
         <X509Certificate>MIICeDCCAeG</X509Certificate>
       </X509Data>
```

contains the signer's public key certificate.

XML Encryption

XML ENC has the following format.

```
<EncryptedData Id? Type? MimeType? Encoding?>
   <EncryptionMethod/>?
   <ds:KeyInfo>
      <EncryptedKey>?         # extension to XMLSig KeyInfo
      <AgreementMethod>?
      <ds:KeyName>?
      <ds:RetrievalMethod>?
      <ds:*>?        #
   </ds:KeyInfo>?
   <CipherData>               # envelopes or references the raw encrypted data
      <CipherValue>?
      <CipherReference URI?>? # location of the raw encrypted data
   </CipherData>
   <EncryptionProperties>?    # e.g., timestamp
</EncryptedData>
```

where KeyInfo element has the same format as in XML DSIG but instead of the KeyValue has the EncryptedKey element that contains encrypted symmetric key used to encrypt the content of the XML document or its part (please note this difference). It is possible to use the XML ENC standard to encrypt the whole document, a specific element, the content of a specific element, or even the attribute value of an element. It is important to note that the encrypted element in the XML document is replaced with the whole EncryptedData element that may be much bigger than non-encrypted element. If you wish to find out more, it is referred to in the standard specification.

Bibliography

Simson Garfinkel and Gene Spafford. Practical UNIX & Internet Security. 2nd edition, O'Reilly.

Edward Amoroso. Intrusion Detection: An Introduction to Internet Surveillance, Correlation, Trace Back, Traps, and Response. Intrusion.Net Books. ISBN 0-9666700-7-8.

RFC 2401 - Security Architecture for the Internet Protocol. [Online] Available from http://www.ietf.org/rfc/rfc2401.txt

CERT Coordination Center. [Online] Available from http://www.cert.org/
CERT CC Publications. [Online] Available from http://www.cert.org/search_pubs/search.php

CERT Incident Note 99-04. Similar attacks using RPC Services (1999). [Online] Available from http://www.cert.org/incident_notes/IN-99-04.html

Distributed Denial of Service (DDoS) Attacks/tools: [Online] Available from http://staff.washington.edu/dittrich/misc/ddos/

The Evolution of Intrusion Detection Systems. [Online] Available from http://www.securityfocus.com/infocus/1514

Host-Based IDS vs Network-Based IDS
http://www.windowsecurity.com/articles/Hids_vs_Nids_Part1.html
http://www.windowsecurity.com/articles/Hids_vs_Nids_Part2.html

Network mapper: http://www.insecure.org/nmap/

CHIHT - Clearing House for Incident Handling Tools. http://chiht.dfn-cert.de/

The Hacker Quarterly: http://www.2600.com/.

Free antivirus tools: http://www.freeav.com

Cheswick, W., S. Bellovin Firewalls and Internet Security: Repelling the Wily Hacker. First Edition. http://www.wilyhacker.com/1e/

XML Encryption Syntax and Processing. W3C Recommendation 10 December 2002. [Online] Available from http://www.w3.org/TR/2002/REC-xmlenc-core-20021210/

XML Signature Syntax and Processing (Second Edition). W3C Recommendation 10 June 2008. [Online] Available from http://www.w3.org/TR/2008/REC-xmldsig-core-20080610/

RFC3275 - XML-Signature Syntax and Processing. http://www.ietf.org/rfc/rfc3275.txt

Security in a Web Services World: A Proposed Architecture and Roadmap. Whitepaper, 1 April 2002. http://www.ibm.com/developerworks/library/specification/ws-secmap/

WS-I Basic Security Profile Version 1.0. 30 March 2007. http://www.ws-i.org/Profiles/BasicSecurityProfile-1.0.html

Providing Integrity and Confidentiality with the XML Security: XML Digital Signature and XML Encryption overview and usage examples. Work in progress. [Online] Available from http://staff.science.uva.nl/~demch/analytic/draft-xmldsig-xmlenc-01.pdf

Reading requirements
Read the text: Chapter 21, skim RFC2401. Total - 44 pages

Additional reading but very useful for the project assignment in week 7 and 8:

RFC4778 - Current Operational Security Practices in Internet Service Provider Environments. [Online] Available from http://www.ietf.org/rfc/rfc4778.txt

RFC2196 - Site Security Handbook. [Online] Available from http://www.ietf.org/rfc/rfc2196.txt

RFC3013 - Recommended Internet Service Provider Security Services and Procedures. [Online] Available from http://www.ietf.org/rfc/rfc3013.txt

RFC2350 - Expectations for Computer Security Incident Response . [Online] Available from http://www.ietf.org/rfc/rfc2350.txt

Appendix: XML Signature format and example (not mandatory for reading)

```
<?xml version="1.0" encoding="UTF-8" ?>
 <Signature xmlns="http://www.w3.org/2000/09/xmldsig#">
   <SignedInfo>
        <CanonicalizationMethod
           Algorithm="http://www.w3.org/TR/2001/REC-xml-c14n-
           20010315" />
        <SignatureMethod
           Algorithm="http://www.w3.org/2000/09/xmldsig#rsa-sha1" />
      <Reference URI="http://www.w3.org/TR/xml-stylesheet">
           <DigestMethod
              Algorithm="http://www.w3.org/2000/09/xmldsig#sha1" />
           <DigestValue>60NvZvtdTB+7UnlLp/H24p7h4bs=</DigestValue>
        </Reference>
     </SignedInfo>
    <SignatureValue>juS5RhJ884qoFR8flVXd/rbrSDVGn40CapgB7qeQiT+rr0Ne
       kEQ6BHhUA8dT3+BCTBUQI0dBjlml9lwzENXvS83zRECjzXbMRTUtVZiPZ
       G2pqKPnL2YU3A9645UCjTXU+jgFumv7k78hieAGDzNci+PQ9KRmm//ic
       T7JaYztgt4=</SignatureValue>
   <KeyInfo>
```

```
    <KeyValue>
      <RSAKeyValue>
          <Modulus>uCiukpgOaOmrq1fPUTH3CAXxuFmPjsmS4jnTKxrv0
             w1JKcXtJ2M3akaV1d/karvJlmeao20jNy9r+/vKwibjM77F+
             3bIkeMEGmAIUnFciJkR+ihO7b4cTuYnEi8xHtu4iMn6GODB
             oEzqFQYdd8p4vrZBsvs44nTrS8qyyhba648=</Modulus>
          <Exponent>AQAB</Exponent>
        </RSAKeyValue>
      </KeyValue>
    <X509Data>
        <X509SubjectName>CN=Merlin Hughes,O=Baltimore
           Technologies\, Ltd.,ST=Dublin,C=IE</X509SubjectName>
      <X509IssuerSerial>
          <X509IssuerName>CN=Test RSA CA,O=Baltimore
             Technologies\, Ltd.,ST=Dublin,C=IE</X509IssuerName>
          <X509SerialNumber>970849928</X509SerialNumber>
        </X509IssuerSerial>
       <X509Certificate>MIICeDCCAeGgAwIBAgIEOd3+iDANBgkqhkiG9w
           0BAQQFADBbMQswCQYDVQQGEwJJRTEPMA0GA1UECBMGRHVi
           bGluMSUwIwYDVQQKExxCYWx0aW1vcmUgVGVjaG5vbG9naW
           VzLCBMdGQuMRQwEgYDVQQDEwtUZXN0IFJTQSBDQTAeFw0w
           MDEwMDYxNjMyMDdaFw0wMTEwMDYxNjMyMDRaMF0xCzAJB
           gNVBAYTAklFMQ8wDQYDVQQIEwZEdWJsaW4xJTAjBgNVBAoT
           HEJhbHRpbW9yZSBUZWNobm9sb2dpZXMsIEx0ZC4xFjAUBgNV
           BAMTDU1lcmxpbiBIdWdoZXMwgZ8wDQYJKoZIhvcNAQEBBQA
           DgY0AMIGJAoGBALgorpKYDmjpq6tXz1Ex9wgF8bhZj47JkuI50
           ysa79MNSSnF7SdjN2pGldXf5Gq7yZZnmqNtIzcva/v7ysIm4zO
           +xft2yJHjBBpgCFJxXIiZEfooTu2+HE7mJxIvMR7buIjJ+hjgwaB
           M6hUGHXfKeL62QbL7OOJ060vKssoW2uuPAgMBAAGjRzBFMB4
           GA1UdEQQXMBWBE21lcmxpbkBiYWx0aW1vcmUuaWUwDgYD
           VR0PAQH/BAQDAgeAMBMGA1UdIwQMMAqACEngrZIVgu03MA
           0GCSqGSIb3DQEBBAUAA4GBAHJu4JVq/WnXK2oqqfLWqes5vH
           OtfX/ZhCjFyDMhzslI8am62gZedwZ9IIZIwlNRMvEDQB2zds/eE
           BnIAQPl/yRLCLOfZnbA8PXrbFP5igs3qQWScBUjZVjik748HU2s
           UVZOa90c0mJl2vJs/RwyLW7/uCAfC/I/k9xGr7fneoIW</X509C
           ertificate>
      </X509Data>
    </KeyInfo>
</Signature>
```