Network Security

Professor Gaurav S. Kasbekar
Department of Electrical Engineering
Indian Institute of Technology, Bombay

Week - 05
Lecture - 28
Secure Email: Part 1

Hello, in the previous several lectures, we discussed different mechanisms for achieving security, such as cryptography, message integrity, authentication, and so on. In this lecture and the next several lectures, we will discuss several practical systems that are used on the internet. So, these will use the mechanisms that we have been discussing, such as cryptography, message integrity, authentication, and so on. We start with secure email. In this lecture and the next lecture, we will discuss secure email.

So, recall that some desirable properties of secure communication are confidentiality, message integrity, and end-point authentication. And in the last several lectures, we have discussed several mechanisms that can be used to achieve secure communication. Some examples of mechanisms that we discussed are symmetric and public key cryptography, cryptographic hash functions, digital signatures, public key infrastructure, and nonces. So, these are some mechanisms that we discussed, and these mechanisms can be used to achieve secure communication, including confidentiality, message integrity, and end-point authentication. In this lecture and the next several lectures, we'll study several systems that use all these mechanisms to provide security on the internet.

First, we'll discuss the following systems: Pretty Good Privacy (PGP) and Secure Multipurpose Internet Mail Extension (S/MIME). So, these are two systems for securing email. So, we'll study PGP and S/MIME for securing email. Then, we'll discuss securing TCP connections. So, recall that TCP is a transport layer protocol, and by default, it doesn't have any security mechanisms.

But we'll study protocols, namely Secure Sockets Layer (SSL) and its standardized version, Transport Layer Security (TLS), for securing TCP connections. So, we'll study SSL and TLS for securing TCP connections. Then, we'll discuss network layer security. We'll discuss IPsec and Virtual Private Networks (VPNs) for network layer security. So, many of us use virtual private networks to connect to the local area networks of campuses, such as universities and companies, when we are outside those campuses.

So, these virtual private networks are extensively used by ordinary users as well. So, we'll discuss IPsec and VPNs for achieving network layer security. And Wi-Fi or wireless LANs are ubiquitous. We'll discuss mechanisms, namely the protocols 802.11i and 802.11w, for securing wireless LANs or Wi-Fi. And we'll discuss several systems for securing wireless cellular networks, including 2G, 3G, 4G, and 5G cellular networks.

So, these systems provide security at different layers. PGP and S/MIME provide security at the application layer. SSL and TLS provide security at the transport layer. IPSec and VPNs provide security at the network layer, and 802.11i and 802.11w provide security at the link layer. Similarly, systems for securing wireless cellular networks also provide security at the link layer.

So, these systems listed over here provide security at the application layer, transport layer, network layer, and link layer, respectively. Now, what is the reason for providing security at multiple layers? Why not just provide security at one layer? So, the reason is that different kinds of attacks can be made by malicious users, and security at different layers is required to defend against them. So, let's look at some examples of attacks and the layer at which we need to secure the network to defend against those attacks.

One example of an attack is a laptop user connecting wirelessly to a Wi-Fi router. Suppose you want to defend against sniffing on the wireless channel by intruders. An intruder can use the packet sniffer and listen to the wireless channel and intercept the communication that is being exchanged between the user and the Wi-Fi router. So, if you want to defend against such sniffing, then it is sufficient to secure the wireless link, that is to provide link layer security. So, this is a very common case where a laptop user connects wirelessly to a Wi-Fi router.

So, to defend against eavesdropping by an intruder, for example, we require to secure the wireless link. So, here we require link layer security. As another example, suppose a user wants to connect to a bank server for an online payment. And the user wants to check that the website is indeed the bank's website. The bank should also be able to check that the user is legitimate.

The user also wants confidentiality from internet service provider employees who access the data. So, the user wants to check that the bank server is legitimate and also wants confidentiality from ISP employees who access the data that is passing through the ISP network. In this case, we need to secure the transport layer. So, in this case, there is a TCP connection between the user's computer and the bank server. We need to secure this TCP connection.

Once we secure the TCP connection, it will provide authentication. That is, the user will be able to check that the bank server is legitimate and vice versa. And also, all communication that is sent over the TCP connection will be encrypted, and message integrity will be added to it. So, this connection between the user's computer and the bank server will pass through many internet service provider networks. The employees of these ISPs won't be able to sniff the communication and obtain the actual data that is being exchanged because the communication will be encrypted.

As another example, suppose the company has offices at multiple locations, for example, one office in Mumbai, one office in Bangalore, and so on. And the company wants to establish a virtual private network to securely connect together all the machines in all the offices. So, there may be, for example, 100 computers in the Mumbai network, 100 computers in the Bangalore network, and so on and so forth. And the company wants to establish a virtual private network to securely connect together all the machines in all the offices. One additional security mechanism that the company wants is this: the company does not want intruders on the public internet to find out the amount of traffic that flows between any pair of machines.

So, as we said, there are 100 computers in the Mumbai network and 100 computers in the Bangalore network. So, how much traffic is flowing between an individual computer in the Mumbai network and an individual computer in the Bangalore network? So, an attacker should not be able to find out how much information flows between a given pair of systems in different offices. This kind of attack is known as a traffic analysis attack. We discussed that during the lecture on different attacks on networks.

So, an intruder who sniffs the traffic that is exchanged between different offices of the same company may analyze the traffic to find out how much traffic is flowing between different pairs of computers in different offices. And this traffic analysis may result in some useful information being known to the attacker. So, to defend against this attack, we need to secure the network layer. So, in summary, these are three different attacks, and we require security mechanisms at different layers to defend against these three attacks. So, in the first case of a laptop user connecting wirelessly to a Wi-Fi router, we need to secure the link layer.

In the second case of the user connecting to a bank server, we need to secure the transport layer. And in the third case of a company wanting to set up a virtual private network to connect its offices at multiple locations, we need to secure the network layer. So, this shows that we need security mechanisms at different layers of the Internet protocol stack to defend against different kinds of attacks. So, we want to discuss email security, but before that, we will provide an overview of email in the Internet, and then we'll discuss where email security mechanisms fit in this architecture of email in the Internet. So, this shows the architecture of email in the Internet.

There are three major components in this architecture. One is user agents. User agents are used by all of us. Examples are Microsoft Outlook, Pine, Gmail, SquirrelMail web interfaces. These user agents are shown here.

This is a user agent. These are user agents, and these are user agents, and so on. So, these user agents allow users to read, reply to, forward, compose emails, and so on. So, for example, we can open Gmail on a browser and we can read emails, reply to emails, forward emails, compose emails, and so on. So, user agents are programs like Gmail and Microsoft Outlook, and so on, which allow users to perform all these functions.

Then, the next component is mail servers, which are shown over here. This is a mail server, and this is a mail server, this is a mail server, and so on. So, mail servers contain user mailboxes and the outgoing message queues. So, the different folders of a user, such as inbox, sent mail, trash, and so on, are stored in the mail server. They are shown here; the user mailboxes are shown over here by this symbol.

And also, when a user composes an email, it is first placed in the outgoing message queue that is shown here. This is the outgoing message queue. And the message is first placed in the outgoing message queue and then it is transferred to the mail server of the recipient. So, this is the second component of email on the internet: mail servers. And the third component is a protocol known as SMTP, or Simple Mail Transfer Protocol.

This is an application layer protocol that uses TCP to transfer email reliably from the sender's mail server to the receiver's mail server. So, at every mail server, there is a client SMTP process and there is a server SMTP process. So, when a user wants to send an email to another user, for example, suppose this user wants to send an email to this user, in that case, this user's email is initially stored in the outgoing message queue. Then, the client SMTP process at this mail server opens a TCP connection with the server SMTP process at this mail server. Then, the email is transferred from the outgoing message queue to the mailbox of the recipient, and the recipient can read the email subsequently.

So, this is the third component of email: SMTP. So, in summary, there are three major components in the email infrastructure. One is user agents, then mail servers, and SMTP. So, here's an example of an email being sent from a sender to a receiver. Suppose Alice wants to send an email to Bob.

The process is shown here of Alice sending an email to Bob. First, Alice invokes her user agent. It might be something like Gmail or Microsoft Outlook and so on. And then, Alice composes and sends an email to the recipient, who is bob@someschool.edu. So, this is the first step, where Alice invokes her user agent like Microsoft Outlook or Gmail, composes, and sends an email to the recipient.

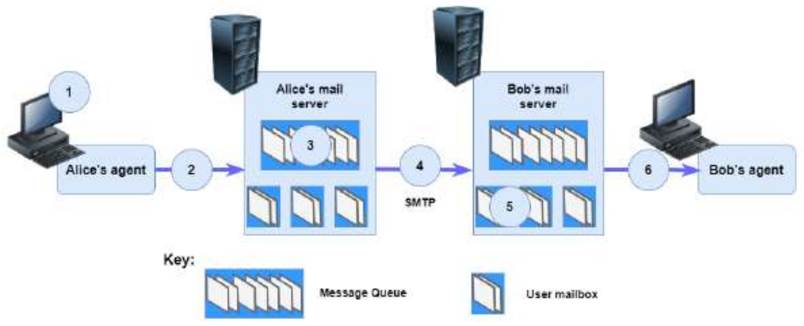
Then, Alice's user agent sends the message to her mail server. So, this transfer is shown in this step, step 2. So, the user agent sends the message to the mail server, where it is placed in the outgoing message queue. So, this is Alice's mail server, and the message is stored in the outgoing message queue of the mail server. So, this is the second step.

Then, in the third step, the client side of SMTP running on Alice's mail server sees the message in the queue. So, the message is in this message queue here. And the client side of SMTP opens a TCP connection to an SMTP server running on Bob's mail server. So, this is Bob's mail server, and there is an SMTP server process running on Bob's mail server. So, the client side of SMTP running on Alice's mail server opens a connection with the server process on Bob's mail server, and this connection will be used to transfer the email.

After some initial SMTP handshaking between the SMTP client process at Alice's mail server and the SMTP server process at Bob's mail server, the SMTP client sends Alice's message into the TCP connection. So, this transfer is shown in step 4. The message is sent into the TCP connection. Then, in step 5 at Bob's mail server, the server side of SMTP receives the message and places it in Bob's mailbox. This is Bob's mailbox, and the server side of SMTP receives the message and places it in Bob's mailbox.

So, at this point, the message is in the recipient's mailbox, that is, in Bob's mailbox. Then, later on, whenever Bob checks his email, he invokes his user agent and sees the email from

Alice. That’s the final step, where Bob has received the email. So, this shows the process by which an email is sent from a sender, Alice, to a receiver, Bob. So, these are the steps that are involved.



So, now, how do we securely send email? So, that part we’ll discuss next. So, before that, we note that SMTP has two sides. One is the client side, and the second is the server side. Both the client and server sides run on every mail server.

So, for example, in Alice’s mail server, there is a client-side SMTP and a server-side SMTP. And likewise, in Bob’s mail server, there is a client-side SMTP and a server-side SMTP. When an email is sent, the client-side of the SMTP process on the sender’s mail server opens a TCP connection to the server-side of SMTP on the receiver’s mail server and transfers the email. So, in the previous example, the client-side of SMTP on Alice’s mail server opened a TCP connection to the server-side of SMTP on Bob’s mail server and transferred the email. It can happen that the sender’s server is not able to deliver an email to the receiver’s server.

This can, for example, happen because of a power failure at the receiver’s server. In that case, what should the sender’s server do? It should not drop the email. Instead, the sender’s server holds the message in the message queue, that is, in the outgoing message queue, and attempts to transfer the message later. For example, reattempts may be done every 30 minutes or so.

So, the sender’s server attempts to transfer the message a fixed number of times, and if the message still is not transferred, in that case, the sender’s server sends a notification to the sender saying that the message could not be sent. So, the sender’s server, in general, tries several times to transfer the email to the recipient’s mail server. So, this concludes our discussion of the email architecture on the internet. We now discuss how to secure email. So, we’ll discuss two systems for securing email.

One is PGP, or Pretty Good Privacy, and the other is S/MIME. So, we start with PGP. PGP is an email security package that provides the following security mechanisms: confidentiality, message integrity, compression, and key management. So, the email is encrypted before it is sent to the other side. A digital signature is used to provide message integrity; that is, if anyone tampers with the message during transit or someone pretending to be the sender and sends a message, in that case, the receiver can know that either the wrong sender sent it or it was tampered with during transit.

So, message integrity is provided in this package, and compression is also included. So, the email message can be long, and it is compressed to save space. It is decompressed at the receiver, and key management is performed. So, there is a mechanism included in PGP to distribute public keys of users. So, we will discuss how PGP accomplishes these security mechanisms.

PGP is available free of charge on the internet for various platforms, including Linux, Windows, and Mac OS. The components of PGP are as follows. Email data encryption is done using a block cipher called IDEA. IDEA stands for International Data Encryption Algorithm. It is a symmetric key-based cipher that uses 128-bit keys, and it is similar to the well-known ciphers DES and AES.

So, IDEA is also a symmetric key cipher similar to DES and AES. The reason that IDEA was used instead of DES or AES was that at the time when PGP was invented, DES was shown to have weaknesses, so DES was not used, and AES was not as standardized, so hence AES was not used. So, for this reason, PGP includes a block cipher called IDEA. Then another component of PGP is that a digital signature is used for message integrity. So, to create a digital signature, an MD5 hash of the message is found and then it is encrypted.

So, the cryptographic hash function used is MD5. So, now we have seen that MD5 has been found to have security weaknesses, so it is not a secure cryptographic hash function, but at the time of the invention of PGP, weaknesses were not discovered yet in MD5, so hence MD5 was the hash function in PGP. RSA is used for securely sharing the 128-bit IDEA key and generating a digital signature for message integrity. So, we discussed earlier that a combination of public key cryptography and symmetric key cryptography can be used to efficiently send long messages from the sender to the receiver. So, this is an instance where this scheme is used.

RSA, which is a public key scheme, is used just for securely sharing the 128-bit IDEA key. Then this key serves as the secret symmetric key, and subsequently, the symmetric block cipher IDEA is used for sending the actual email message. So, we use a combination of public key cryptography and symmetric key cryptography to achieve encryption. RSA is also used for generating a digital signature for message integrity. So, we saw that a digital signature is an encrypted MD5 hash.

This encryption is done using RSA. Also, we discussed that one of the components of PGP is compression. The Lempel-Ziv algorithm, which is a popular algorithm for compression, is used for compression in PGP. Checking whether a public key indeed belongs to a specific user may be done using certification authorities or a 'Web of Trust'. We discussed certification authorities in the public key infrastructure part of the course.

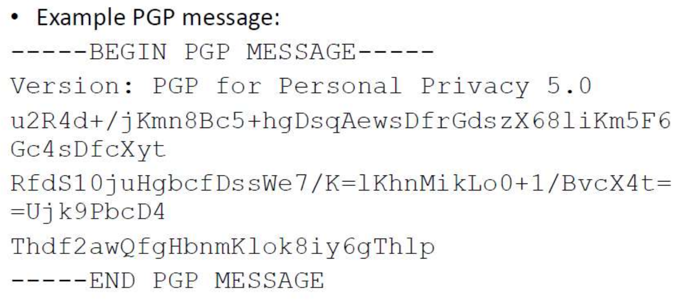
Another mechanism for distributing public keys is the web of trust, which is used in PGP. We'll discuss the details of the web of trust later. So, we discussed the architecture of email on the internet. Where does PGP fit in that architecture? PGP is like a preprocessor that takes plaintext as input and produces signed ciphertext as the output.

So, it encrypts the plaintext as well as adds a digital signature for message integrity. And this output, namely the signed ciphertext, can then be emailed using a user agent. So, PGP is like a preprocessor in that architecture. So, if we just go back to that email architecture; this is the transfer of an email from Alice to Bob. So, in this case, if PGP is used, then at this point 1, PGP will process the message to produce the signed ciphertext, and then the signed ciphertext will be transferred to Bob's agent, and then the PGP at Bob's side will decrypt the message, and it can be read by Bob.

So, PGP is a preprocessor. So, it just operates on the sender side and the receiver side. On the sender's side, it encrypts the message and adds message integrity, and so on. On the receiver's side, it decrypts the message and verifies the integrity of the message. So, that's where PGP fits into this email architecture.

Here's an example of a PGP message. So, it says: 'BEGIN PGP MESSAGE'. Version: PGP for personal privacy 5.0. So, the version is 5.0, and this is the encrypted message. So, we see that this message does not make much sense because it is ciphertext.

So, the plaintext cannot be deduced from this ciphertext. And then, 'END PGP MESSAGE'. So, if the message is sent in the clear, then it can be read by anyone who intercepts the channel. But if PGP is used, then it encrypts that message and produces such a ciphertext, which is then sent using the mechanisms that we discussed. So, PGP creates a message like this one, which is then sent over the internet using the process we discussed earlier.

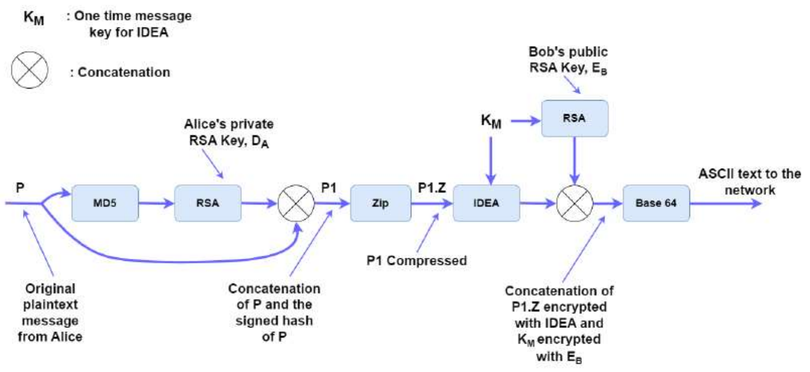


- Example PGP message:

```
-----BEGIN PGP MESSAGE-----
Version: PGP for Personal Privacy 5.0
u2R4d+/jKmn8Bc5+hgDsqAewsDfrGdszX68liKm5F6
Gc4sDfcXyt
RfdS10juHgbcfDssWe7/K=lKhnMikLo0+1/BvcX4t=
=Ujk9PbcD4
Thdf2awQfgHbnmKlok8iy6gThlp
-----END PGP MESSAGE
```

Browser plugins are available, which provide interfaces for PGP encryption and decryption for user agents, such as Gmail. So, we can use a standard user agent, such as Gmail, and use a browser plugin along with it, which provides interfaces for PGP encryption and decryption. So, this plugin will perform the functions of PGP, such as providing confidentiality, message integrity, compression, and so on. So, we can use this plugin along with a user agent to get access to PGP. We summarize the operation of PGP, and we'll discuss it in detail in the next lecture.

So, suppose Alice wants to send an email to Bob. Let D_A denote Alice's private key, and DB denote Bob's private key. Similarly, E_A and E_B denote Alice's and Bob's public keys. So, E stands for encryption, and D stands for decryption. So, recall that a public key is used for encryption, and a private key is used for decryption when using public key cryptography to send a message encrypted to the receiver.



K_M : One time message key for IDEA
: Concatenation
Bob's public RSA Key, E_B
K_M
RSA
Alice's private RSA Key, D_A
P
MD5
RSA
P1
Zip
P1.Z
IDEA
Base 64
ASCII text to the network
P1 Compressed
Original plaintext message from Alice
Concatenation of P and the signed hash of P
Concatenation of P1.Z encrypted with IDEA and K_M encrypted with E_B

So, we use this notation. Let P denote the plaintext message, which is the original email message that the sender wants to send to the receiver. This shows the processing of the plaintext message P. So, this P is the original plaintext message from Alice. So, for message integrity purposes, PGP first finds the hash of P. So, this is the hash of P. And then Alice's private RSA key D_A is applied to the hash of P to get the digital signature of this message, created using Alice's private key.

For a message P, the digital signature is $K^{-}H(M)$. The H function is MD5, and this is the digital signature, that is, $K^{-}H(P)$. So, this is the digital signature, and the original message P is concatenated with the digital signature. This P1 denotes the concatenation of P and the signed hash of P. So, P1 is the message along with its digital signature. Then it is compressed to get P1.Z, which is the compressed version of P1. And then a random key is generated. So, this K_M is a one-time message key for IDEA.

It is a random key. So, it is different for different email messages. And this K_M is used as the secret key in the symmetric block cipher IDEA for encrypting the message P1.Z. This is the encrypted message. And then the secret key K_M itself has to be shared with the receiver Bob.

So, the secret key K_M is encrypted using Bob's public RSA key E_B, and this is the encrypted version of the secret key K_M. This is the concatenation of P1.Z encrypted with IDEA and K_M encrypted with E_B. This is converted to a format called Base 64 format. We'll discuss the reasons for conversion to Base 64 format. This uses a limited number of characters, namely the capital letters, small letters, digits, and a couple of other characters.

So, after conversion to Base 64, this is the text that is actually sent over the email infrastructure. So, at the sender's side, all this processing is done. First, the digital signature is found, then compression is done, then it is encrypted using the symmetric key cipher, and then it is converted to Base 64 format. The secret key used for encryption is included along with the message that is to be sent, and then the message is sent over the network. So, this summarizes the operation of PGP at the sender, and at the receiver, the operations are essentially the reverse of this. The receiver converts from Base 64 to ordinary ASCII format and then reverses all these operations to get the original message and verifies its integrity.

We'll discuss the detailed operations of PGP in the next lecture. So, to summarize various security systems, practical security systems which employ the security mechanisms that we have been discussing in the last several classes, we provided an overview of email in the internet and then we started our discussion of the secure email system PGP. We will continue our discussion of PGP in the next lecture. Thank you.