# CS514: Intermediate Course in Computer Systems

Lecture 19: October 29, 31, 2003

Security (part 1)

## Security is an endlessly huge topic

CS514

- Take CS513 next semester and find out!
  - (Seriously, highly recommended)
- Here, we want to focus on security issues associated with web sites and web services
  - This is still a broad range of problems that goes beyond web…

# What kinds of things concern us?

CS514

- Someone breaks into a web site and steals data (user credit card numbers) or alters contents
- Someone impersonates a web site (and perhaps steals user information)
- Someone impersonates a user
- Someone monitors communications between a user and a web site, and gathers sensitive information
- Someone overwhelms a web site with requests or traffic and makes it unusable by others (denial of service)

# What kinds of things concern us?

CS514

- Someone breaks into a web site and steals data (user credit card numbers) or alters contents
- Someone imp… onates a web site (and perhaps steals us… ation)
- So…
- So… etween a user an… nformation
- So… requests or tra… (denial of se…

Firewall (to protect against port scanning and other intrusions, and make life harder for the attacker)
Access Control and Authentication (to prevent attacker from getting admin privileges
Intrusion detection (to discover suspicious activity)

# What kinds of things concern us?

CS514

- Someone breaks into a web site and steals data (user credit card numbers) or alters contents
- Someone impersonates a web site (and perhaps steals user information)
- Someone imper… ates a user
- Someone m… communications between a user … rs sensitive information
- … eb site with requests or … ole by others (denial of

Protect DNS so that attacker can't steer user to the wrong place
Certificates from trusted Certificate Authorities
"Realistic" looking URLs

# What kinds of things concern us?

CS514

- Someone breaks into a web site and steals data (user credit card numbers) or alters contents
- Someone impersonates a web site (and perhaps steals user information)
- Someone impersonates a user
- Someone moni… communications between a user and a web … gathers sensitive information
- … web site with requests or … ble by others (denial of

User Authentication
Encryption of user sessions to protect passwords

# What kinds of things concern us?

CS514

- Someone breaks into a web site and steals data (user credit card
- Someone imper steals user informatio
- Someone imperson a user
- Someone monitors communications between a user and a web site, and gathers sensitive information
- Someone overwhelms a web site with requests or traffic and makes it unusable by others (denial of service)

Encryption of user sessions

# What kinds of things concern us?

CS514

- Someone breaks into a web site and steals data (user credit
- Someone in steals user
- Someone in
- Someone monito unications between a user and a web site, d gathers sensitive information
- Someone overwhelms a web site with requests or traffic and makes it unusable by others (denial of service)

TCP SYN attack prevention
Over provision (same as for dealing with flash crowds)
Load balancers to throttle traffic
Other tricks…

## Classic list of basic security services

CS514

- Access control
- Authentication
- Confidentiality
- Integrity
- Non-repudiation

Following slides borrow heavily from Peter Gutmann's highly recommended tutorial at http://www.cs.auckland.ac.nz/~pgut001/tutorial/

## In a way, everything is built out of two mechanisms

CS514

- Encryption/decryption
  - Which is ultimately about securely keeping and sharing secrets
  - Key distribution
- Hashing (one way)
- But these basic mechanisms are used in many different ways

# Hashing (a.k.a. message digest)

CS514

- Produces an integer when applied to some data
  - Hash(data, len) = I
  - The integer I tends to be uniformly randomly distributed
- But only works in one direction
  - Can't produce the (data,len) from I
- If I is big enough (say, 128 bits), then serves as a unique identifier for the data
- Virtually no other (data,len) will produce the same I
  - And small changes to (data,len) will produce a different I

# What can you do with hashing?

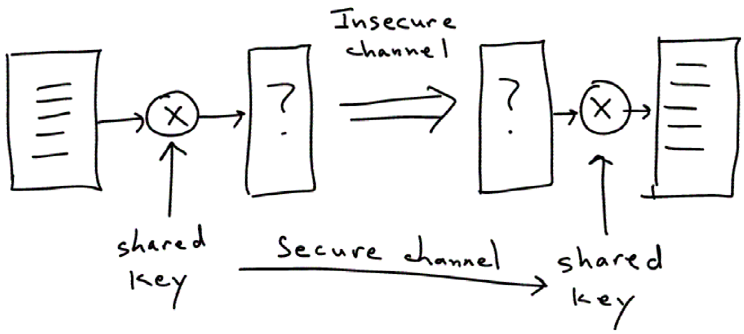
CS514

- If the hash value can be securely conveyed, then can detect tampering
  - I.e. *integrity*
- Used in other ways too (as we'll see)
  - Digital signature

## Conventional encryption (confidentiality)

CS514

Problem of communicating a large message in secret becomes that of communicating a small secret in secret



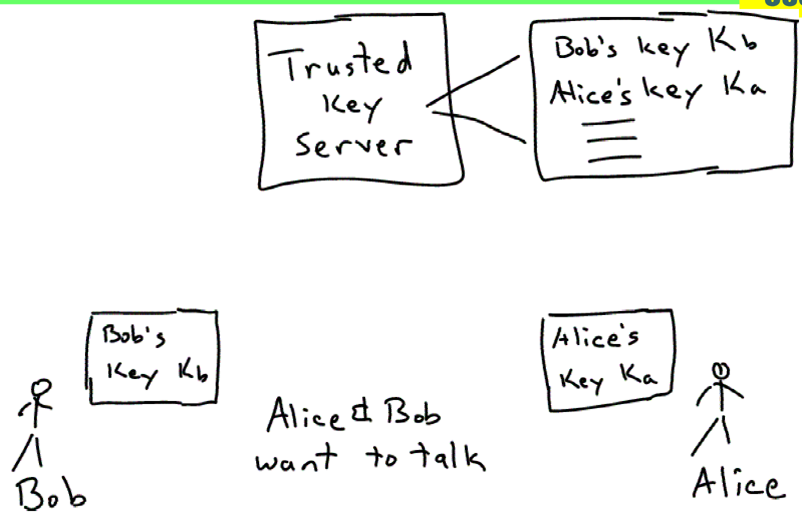

## Difficulties of shared secret encryption

CS514

- Also known as symmetric key encryption
- How do you distribute the keys?
- Need to have a distinct key for every pair of communicators
  - And each needs to be changed periodically ("refreshed") in case it was discovered
- $N^2$ keys!

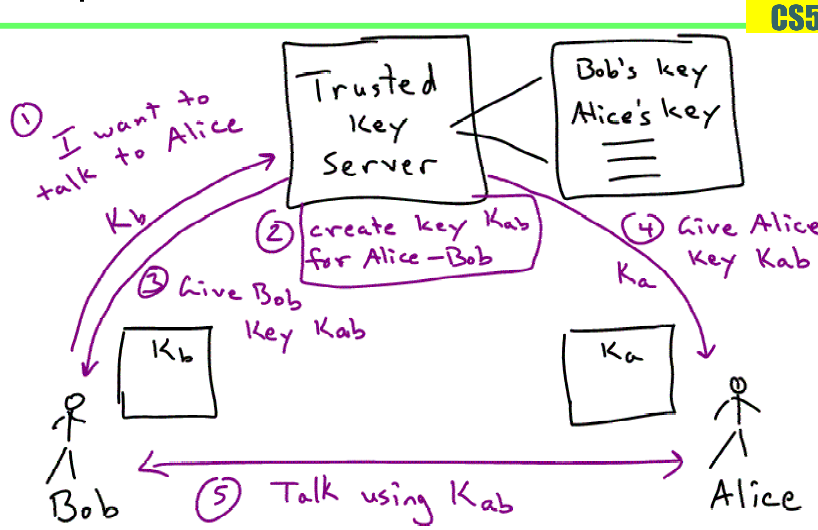
## Trusted third party key distribution

CS514




## Trusted third party key distribution

CS514



## Trusted third party key distribution

CS514

- This is the basis for Kerberos
  - We'll cover this a bit later
- Note that Bob's and Alice's keys (Kb and Ka) have to be refreshed periodically
- The shared key Kab is typically used only once
  - So that an eavesdropper can't, over time, guess the key

## Guessing keys

CS514

- A key is easier to guess when:
  - They are short
  - There is lots of data available that was encrypted by the key
- 48 bits is a short key
- 128 bits is a long key

## Note the single point of failure

CS514

- As a rule, security tends to lead to weakened system reliability
- Simply by virtue of having another box "in the loop"
  - Secure systems typically err on the side of preventing things from happening
- We all have experienced this first hand
  - I.e., can't log into a system, etc…

## Public key encryption

CS514

- Now, what if a given node (say Bob) could use the *same* key with every communicating peer?
  - Instead of a different key for each peer
- Now we have N keys instead of $N^2$ keys
- But now, couldn't every other node decrypt a document?

## Public key encryption



- Actually, each "key" comes as a pair of keys…
  - …a public key and a private key
  - The private key is kept secret
  - Everybody knows the public key
- These things are magic!  Why?
- Something encrypted with the public key and be decrypted with the private key
- And vice versa…something encrypted with the private key and be decrypted with the public key!

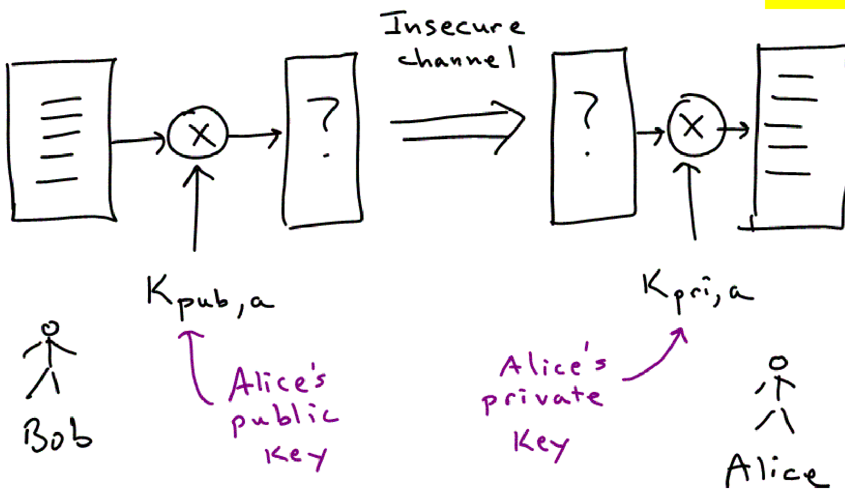
## In other words…



- $E(K_{pub}, Doc) = Doc'$,
  - $D(K_{pri}, Doc') = Doc$
- $E(K_{pri}, Doc) = Doc'$,
  - $D(K_{pub}, Doc') = Doc$
- So what???

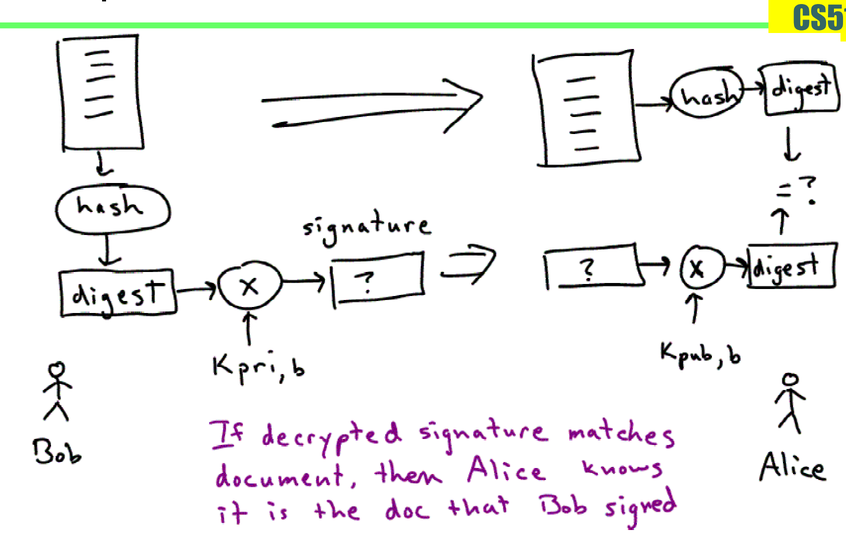
## Confidentiality

CS514



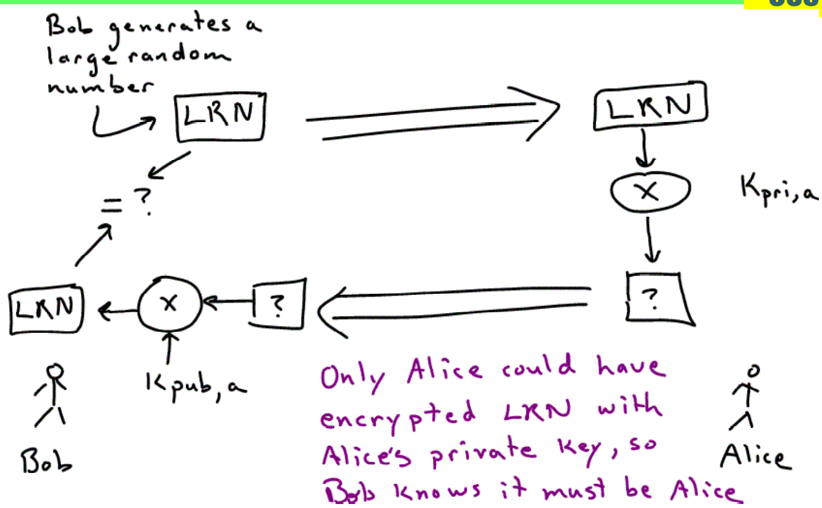

## Digital signature

CS514



# Authentication





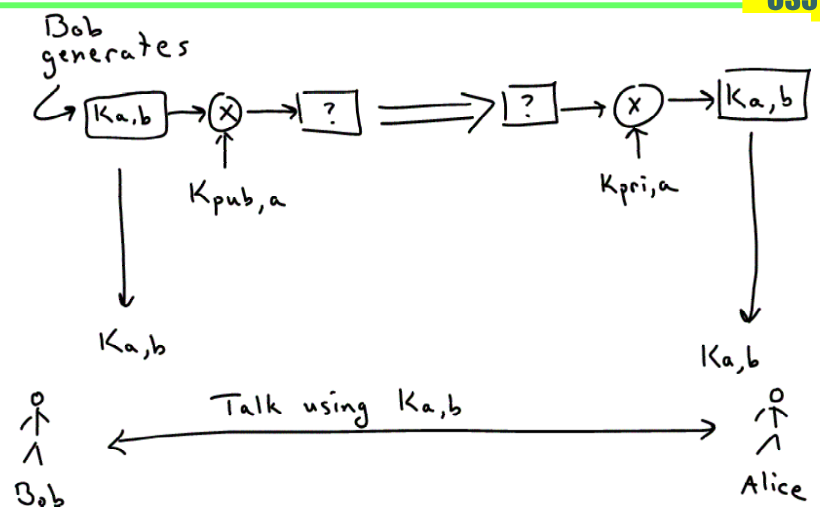

# Problem with public keys



- Used to be, public keys were patented (RSA)
  - That ended in year 2000
  - There were some big parties on that day!
- Public key encryption is expensive!
  - Can't afford to encrypt large data with public key
  - Instead, use public keys to exchange symmetric keys!
- Also, public keys don't eliminate need for trusted third party!

## Symmetric key exchange with public keys

CS514




## Need for trusted 3rd party

CS514

- How do you know that the public key you have for someone is really their public key?
  - I.e., the one that matches their private key?
- Ultimately you still need a trusted 3rd party to give out the public keys
  - But, the job of the trusted 3rd party is much easier
  - Don't need to create and hand out per-connection keys

## Public key certificates

CS514

- The public keys handed out are called certificates
  - Contain the public key, name of the private key holder, and other stuff like expiration date, rigor of the authentication, etc.
- The organization that hands them out is called the Certificate Authority (CA)
- The certs are signed by the CA
- So, you must know the public key of the CA!!!

## Public Key Infrastructure (PKI)

CS514

- A hierarchy of CAs
- The CA handing out a cert could have a cert from a higher level CA
- And so on
- But, everybody has to have a cert for the top level CAs
  - Not unlike DNS, where all resolvers need the IP addresses of the root DNS servers

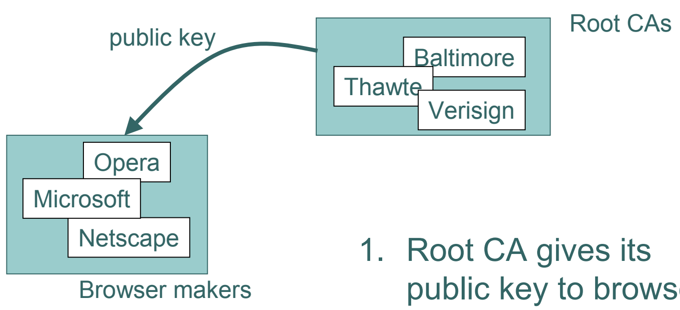
# Web security (HTTPS)

CS514

- Secure web exchanges use CAs and certs
  - HTTPS = HTTP Secure
  - It means HTTP over a secure transport layer (HTTP over SSL over TCP)
- There are a small number of top-level Cas
  - Verisign, Thawte Consulting, Comodo, Baltimore, etc.
- The certs of the top-level CAs are distributed along with the browser software
  - So, ultimately, Microsoft is a top-level CA!

# Public key trust in SSL

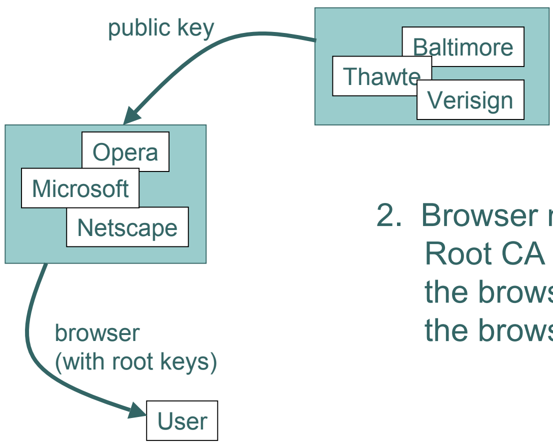
CS514




1. Root CA gives its public key to browser manufacturers

# Public key trust in SSL

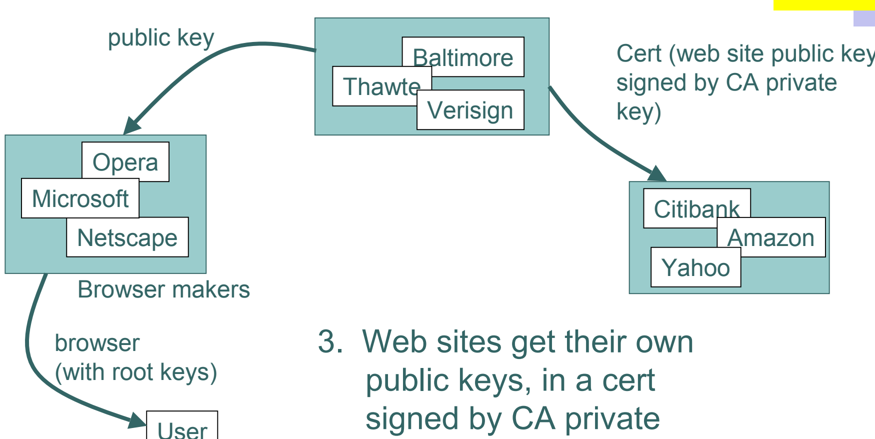
CS514




2. Browser maker puts the Root CA public keys in the browser, and gives the browser to you

# Public key trust in SSL

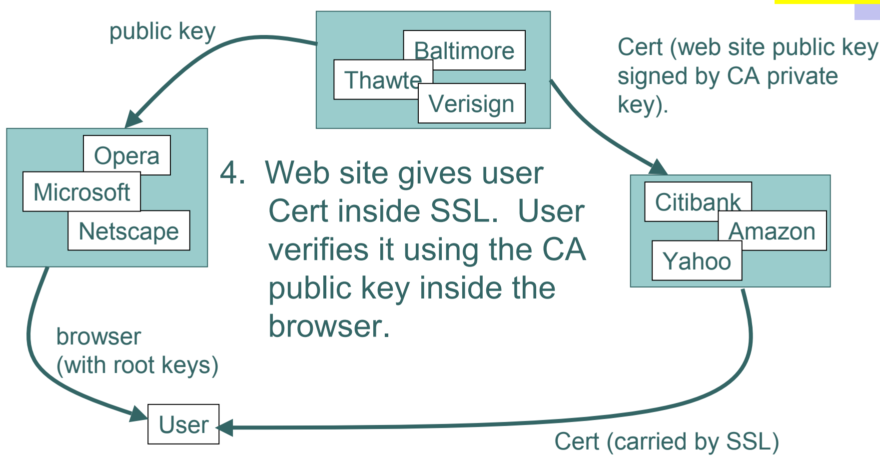
CS514




3. Web sites get their own public keys, in a cert signed by CA private key

# Public key trust in SSL

CS514




4. Web site gives user Cert inside SSL. User verifies it using the CA public key inside the browser.

# The cert is only as good as the person who receives it

CS514

- How many people check the cert when that little "do you trust this certificate" window pops up???
- What if the cert is for amezon.com
  - And amezon.com is pretending to be amazon.com?
  - The cert would still appear legitimate!!
- Note that DNS internationalization makes this kind of DNS spoofing easier
  - Different letters in different encoding schemes may look the same, but DNS sees them as different!

## Core difference between asymmetric and public key distribution

CS514

- Asymmetric:
  - Lots of keys must be distributed, but each key has limited scope
    - I.e. between trusted 3rd party and a single client
- Symmetric:
  - Fewer keys are distributed, but each key has wide scope
    - A client's public key is known by many peers
    - The root CA public key is known by everybody!

## Certificate revocation

CS514

- If a private key is compromised, there must be a way to revoke the certificate containing the matching public key
  - From everybody who got the cert
- Practically speaking this is very hard
  - Because a client can give its certificate to anyone
- In practice, certificates have expiration dates
  - Make expiration period short
  - Revoke by waiting for expiration
  - But short expiration means bigger load on CA

## What if CA private key is compromised?

CS514

- Browsers will have to be patched
- Web sites will need new certs
- A pain, but probably not a complete disaster
  - Not so different from responding to the latest worm or virus…
- Note that Root CA public keys have expired in the past
  - But lifetimes are on the order of ten years
  - Typically not a problem because browser version upgrades contain more recent public keys

## Authentication revisited

CS514

- We saw an authentication example based on public keys
- You can construct a similar example based on symmetric keys
  - Only requirement is that Alice can encrypt something, and Bob can decrypt it
- Problem is, these keys are long!

# Long authentication keys and in-head passwords

CS514

- Authentication often involves a password in someone’s head
- A 160-bit random key looks like this:
  - eIv8%w220M.-wB&`eH7eFI4
    - (23 ASCII characters)
- Nobody could remember this!

# Long authentication keys and in-head passwords

CS514

- In-head passwords are very “mobile”
  - A user can enter his password on any machine
- Private or shared keys are not very mobile
  - You can’t remember them
  - You sure don’t want to write them down
  - Some people use a key-sized USB fob, but this is expensive, can be lost or stolen, etc.
  - So, tend to be tied to individual machines
- And, in-head passwords can be created by the user
  - Which is convenient, but can result in weak passwords

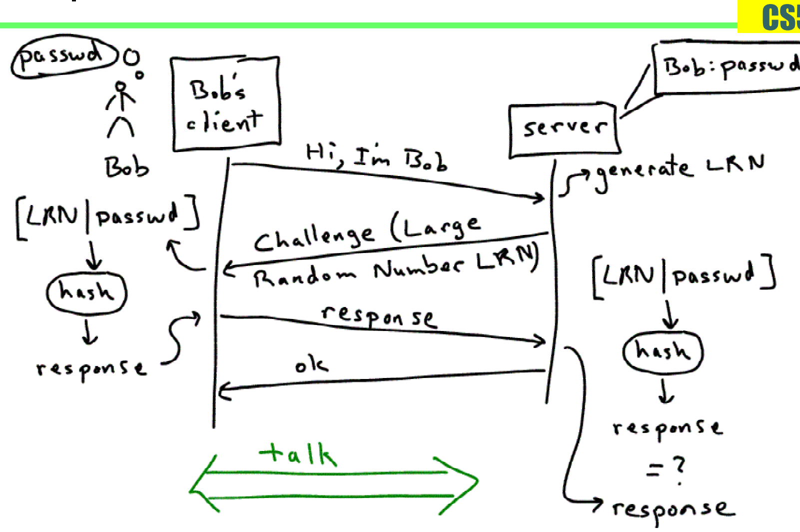
# Authenticating with in-head passwords

CS514

- Obviously don't want to send passwords in the clear
  - Though historically this was done a lot!
- Instead, a random challenge and response are sent in the clear
  - In some protocols anyway (RADIUS)
  - Using hashes

# Authenticating with in-head passwords

CS514



## A couple observations…

CS514

- We see the server authenticating Bob, but not the other way
  - In fact, they could have both challenged each other
- Note that Bob's "hello" message sent his identity in the clear
  - Privacy issue!

## Self-identification paradox

CS514

- Bob wants to introduce himself to the server
- If Bob encrypts his identity, how will the server know how to decrypt it?
  - Doesn't know which password to use to decrypt
  - Can't just "try them all", because may have millions of user passwords
- But if Bob doesn't encrypt his identity, then any eavesdropper can see it!

# Diffie-Hellman (another bit of crypto magic!)

CS514

- Allows two participants, *with no prior private shared knowledge of any kind*, to establish a shared secret *without an eavesdropper knowing the secret*!!
- Its all math:
  - depends on: $(a^x)^y = (a^y)^x$
  - and on the fact that it is hard to figure out x given $g = a^x \bmod p$

# Diffie-Hellman

CS514

- a and p are publicly known values
- Bob creates a secret Sb, and calculates $Gb = a^{Sb} \bmod p$
  - Likewise Alice creates a secret Sa, and calculates $Ga = a^{Sa} \bmod p$
- Alice and Bob exchange Gb and Ga
- Bob calculates $S = Ga^{Sb} \bmod p$
  - Alice calculates $S = Gb^{Sa} \bmod p$

## Summary of basic tools

CS514

- Symmetric key encryption
  - Efficient crypto, but hard key dist problem
- Public (asymmetric) key encryption
  - Easier key dist problem, but inefficient crypto
- One way hash
- Diffie-Hellman key exchange

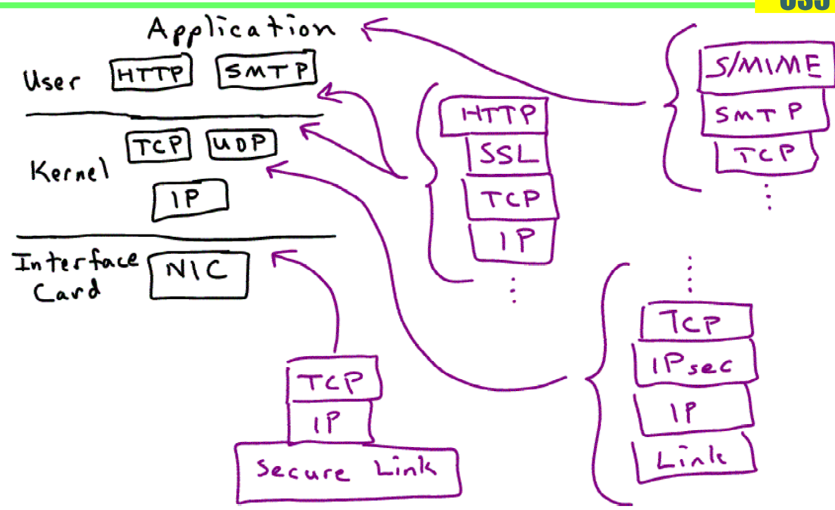
## Four "layers" at which security can take place

CS514

- Link/physical (802.1x)
- Network (IPsec)
- Transport (TLS/SSL)
  - And security protocols that directly use TLS/SSL, such as HTTPS, SSH
- Application
  - S/MIME (e.g. email), XML Encrypt and XML Signature (e.g. Web Services)

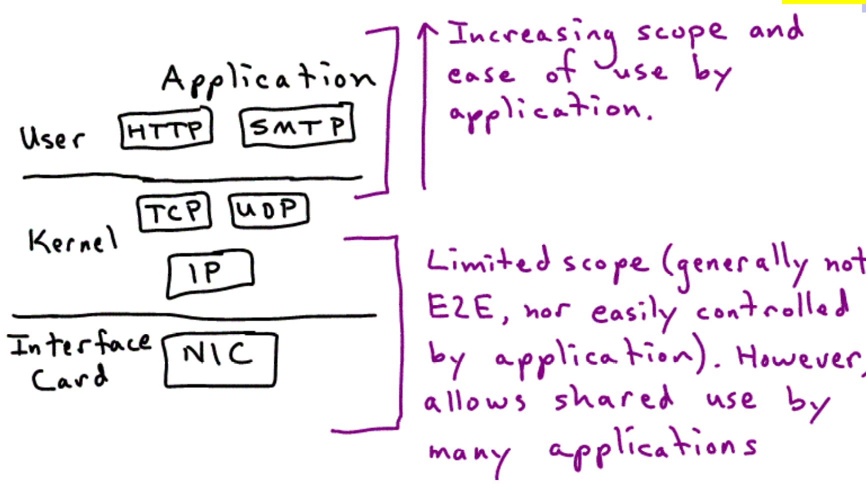
## Four "layers" at which security can take place

CS514




## Four "layers" at which security can take place

CS514



# IPsec E2E by design, but not by use

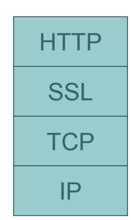
CS514

- IPsec was meant to be a kind of all-purpose E2E security mechanism
- Dream was, IPsec would automatically kick in when two hosts tried to communicate
- In practice, its use is more limited
  - Between VPN routers, or between VPN client and VPN router
    - VPN=Virtual Private Network

# SSL (a.k.a. TLS)

CS514

- Transport-layer security
- Runs above TCP
- Encrypts everything above TCP
- Different applications can run over SSL
  - HTTP, telnet, FTP, LDAP
  - Each requires a separate port number to run over SSL
- TLS (Transport Layer Security) is IETF version of SSL
  - SSL still used in practice



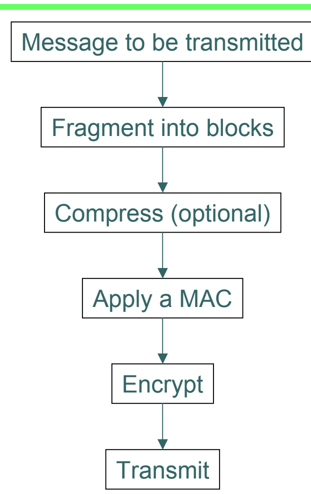
# SSL Overview

CS514

- Establish a session
  - Agree on algorithms
    - DES, 3DES, RC2, RC4, IDEA bulk encryption
    - MAC is SHA-1 or MD5 (Message Authentication Code)
  - Share secrets (RSA or Diffie-Hellman)
  - Perform authentication (Certs)
- Transfer application data
  - Ensure privacy and integrity

# SSL Data Operation

CS514



## SSL "Alternatives"

CS514

- S-HTTP: secure HTTP protocol, shttp://
  - Predates SSL, never caught on
- IPSec: secure IP
  - Wrong features
- SET: Secure Electronic Transaction
  - Protocol and infrastructure for bank card payments
  - More than just a secure pipe
  - Hasn't caught on
- SASL: Simple Authentication and Security Layer (RFC 2222)
  - Framework for selecting authentication and security
  - Encompasses lots of protocols
  - Not sure how much deployed and used

## Why not IPsec (instead of SSL)?

CS514

- Only mutual authentication
  - Server doesn't need to authenticate user until later
- Limitations due to NAT
  - One (or at most a small number of) IPsec session at a time
  - IETF working to fix this
- Can't control by port number
  - HTTPS has a well-known port (443)
  - IPsec would require separate IP address
- Dependent on user IP address
  - Secure session can't span reconnects

## Why not mutual authentication via SSL?

CS514

- In theory it is possible…requires that the user have a Cert and key
- Problem is, Certs are not easily portable across machines
  - Certainly humans can't remember them!
  - As such, cert effectively becomes machine authentication, not user authentication
- If web server needs to authenticate user, this is done at application level over SSL

## Denial of Service (DoS)

CS514

- Various forms
  - Simply overwhelm target system
    - Distributed DoS (DDoS)
    - Smurf attack
  - Consume resources on target system
    - SYN attack
    - Email bomb
  - Exploit bug in target system to crash it (usually some buffer overflow)
    - Ping of Death
    - Code Red
    - SQL Slammer

## Smurf Attack

CS514

- Attacker sends ping to target network
  - Destination address is broadcast
    - Net number + all-ones
  - Spoofed source address is victim host
- Router on target network broadcasts the packet
- All recipients reply to ping, flood victim system
  - Victim need not be on target network

## Smurf countermeasures

CS514

- Configure routers not to forward broadcast packets from off-net
- Configure hosts not to respond to pings to broadcast address
- Still, if you are the victim on some different network, not much you can do but filter incoming ICMP pings

# SYN attack

CS514

- Attacker sends many TCP SYN
  - With spoofed source address
  - So that it looks like lots of different sources
- Victim allocates TCP record for each one
- Eventually exhausts pool of records, legitimate TCP requests are ignored

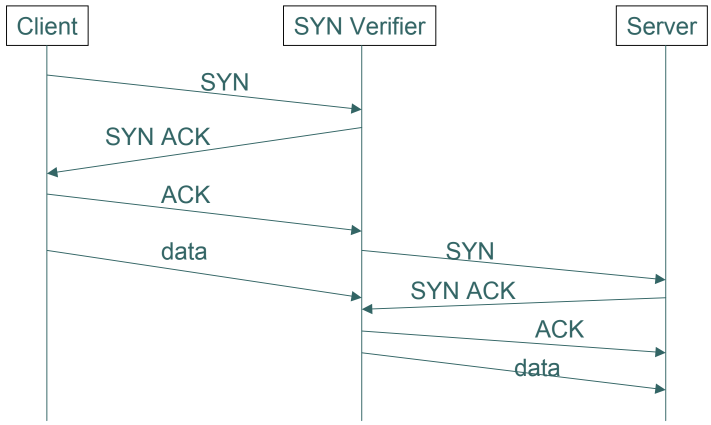
# SYN attack countermeasures

CS514

- Shorter time-outs on half-open connections
  - Or, dynamically shrink time-out when many half-open connections detected
- Put a SYN verifier in front of server
  - SYN verifier responds to SYN, if gets SYN ACK, then knows SYN is legitimate, and bridges connection to server
  - Has lots of buffers

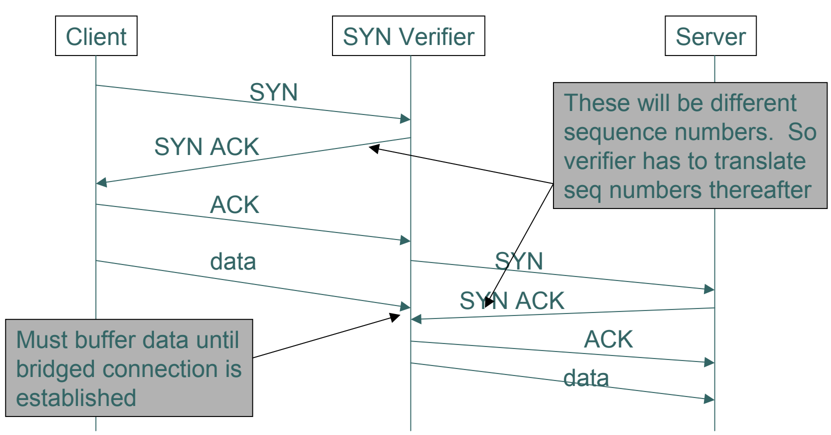
# SYN Verifier

CS514



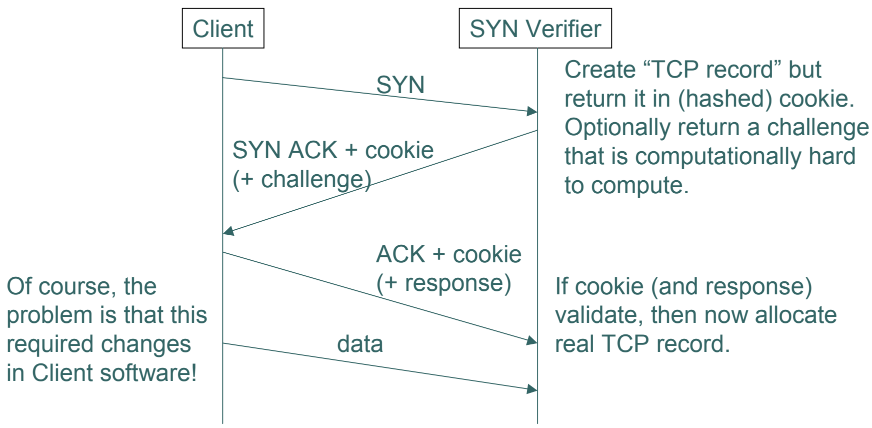

# SYN Verifier difficulties

CS514



# SYN Cookie is cleaner approach

CS514




# Login blocking

CS514

- Some systems will block a login after a few failed attempts
- Attacker simply writes a script that does incorrect logins for every user
  - Can even to lock out root!

## Simple old buffer overflow attacks

CS514

- Ping of Death (overlarge ICMP packet)
  - Crashes victim
- NewTear, Newtear2, Bonk, Boink
  - Exploited bug in Windows 9x/NT
  - Locksup or crashes victim
- These generally fixed now

## Always more buffer overflow attacks…

CS514

- Exploit buffer overflow to insert own code into call stack
  - Code Red worm
  - Recent SQL Slammer worm
- These will always exist
  - Search for “security advisory” on cisco.com generated 1100 hits
    - Casual inspection indicated that many were of this sort
  - Install system patches and firewall filters

## New security systems bring new attack possibilities



- For instance, Blackice Intrusion Detection System (IDS)
- Operates as process that runs on each host
- Had a bug that it allowed it to accept too many TCP connections
- Attacker could consume up to 400MBytes of system memory

## New security systems have their own holes



- When firewall filters for strings, must search for all encodings
  - ASCII, UTF (%xx%xx), or plain hex (%xx)
- Microsoft IIS includes a new encoding that is not an HTTP standard (%u)
- Cisco IDS was not aware of this encoding
- Thus, attacker could bypass IDS by using new encoding
  - Ex: CodeRed worm used the .ida buffer overflow
  - Attacker could encode ".ida" in %u (GET /himom.id%u0061 HTTP/1.0)

## Distributed Denial-of-Service

CS514

- trin00 (WinTrinoo)
- Tribe Flood Netowrk (TFN) (TFN2k)
- Shaft
- stacheldraht
- Mstream

## Trin00

CS514

- Affects Windows and many Unix OS's
- Attacker scans for exploits, gains root, and downloads Trin00 programs.
- Attacker->Master->Daemon hierarchy
  - (One -> More -> Many)
- Attacker can telnet into a Master to initiate commands, which are distributed among its Daemons.

## Trin00

CS514

- Communication between Master->Daemon through a password-protected cleartext UDP-based protocol.
  - In other variants, Internet Relay Chat is used as the means of communicating with Daemons
- Daemons attack the target with a UDP or TCP packet bombardment.
- Used in the February 2000 attacks on eBay, Amazon, CNN, etc.

## Other DDoS

CS514

- TFN(2k)
  - Smurf attack, ICMP flood, SYN flood, UDP flood, simultaneous
- Stackeldraht
  - Smurf attack, ICMP flood, SYN flood, UDP flood
- Shaft
  - ICMP flood, SYN flood, UDP flood, simultaneous

# Intrusion Detection Systems (IDS)

CS514

- Broad range of systems that monitor activity, attempt to flag unusual behavior
  - Changes in volume of traffic
  - Changes in protocols or ports
  - Unusual traffic patterns for a given application
  - Known exploits

# Intrusion Detection Systems (IDS)

CS514

- Broad range of systems that monitor activity, attempt to flag unusual behavior
  - Host based
    - Look through host log files
    - Check integrity of file systems
  - Network based (snoop traffic)
    - Either at host or as network monitor
  - “Honeypots” (pretend to be exploitable systems, attract hackers)