COM307000 –
Real - World Protocols

Dr. Anca Jurcut
E-mail: anca.jurcut@ucd.ie

School of Computer Science and Informatics
University College Dublin,
Ireland

Real-World Protocols

- Examples of real protocols
  - SSH — relatively simple & useful protocol
  - SSL — practical security on the Web
  - IPSec — security at the IP layer
  - Kerberos — symmetric key, single sign-on
  - WEP — "Swiss cheese" of security protocols
  - GSM — mobile phone (in)security

Secure Shell (SSH)

SSH

- Creates a “secure tunnel”
- Insecure command sent thru SSH “tunnel” are then secure
- SSH used with things like rlogin
  - Why is rlogin insecure without SSH?
  - Why is rlogin secure with SSH?
- SSH is a relatively simple protocol

SSH

- SSH authentication can be based on:
  - Public keys, or
  - Digital certificates, or
  - Passwords
- Here, we consider ***certificate*** mode
  - Other modes: homework
- We consider slightly simplified SSH…

Simplified SSH

Alice

Alice, CP, Na $\longrightarrow$

$\longleftarrow$ CS, Nb

$g^a \bmod p$ $\longrightarrow$

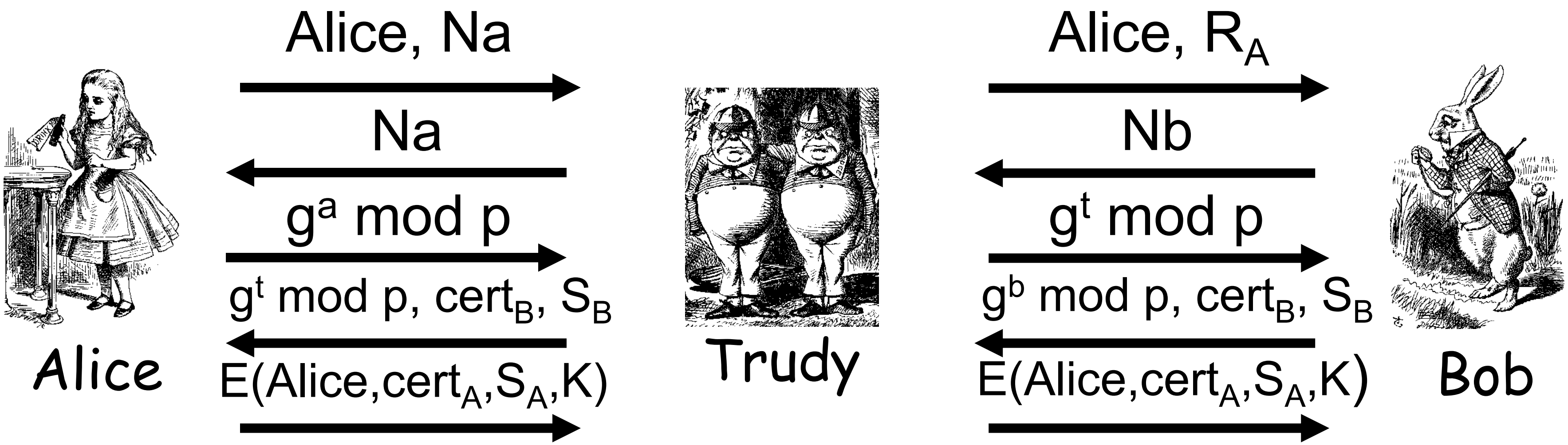
$\longleftarrow$ $g^b \bmod p$, certificate_B, S_B

E(Alice, certificate_A, S_A, K) $\longrightarrow$

Bob

- CP = “crypto proposed”, and CS = “crypto selected”
- $H = h(\text{Alice},\text{Bob},\text{CP},\text{CS},R_A,R_B,g^a \bmod p,g^b \bmod p,g^{ab} \bmod p)$
- $S_B = \{H\}K_{BPriv}$
- $S_A = \{H, \text{Alice}, \text{certificate}_A\}K_{APriv}$
- $K = g^{ab} \bmod p$

MiM Attack on SSH?



- Where does this attack fail?
- Alice computes

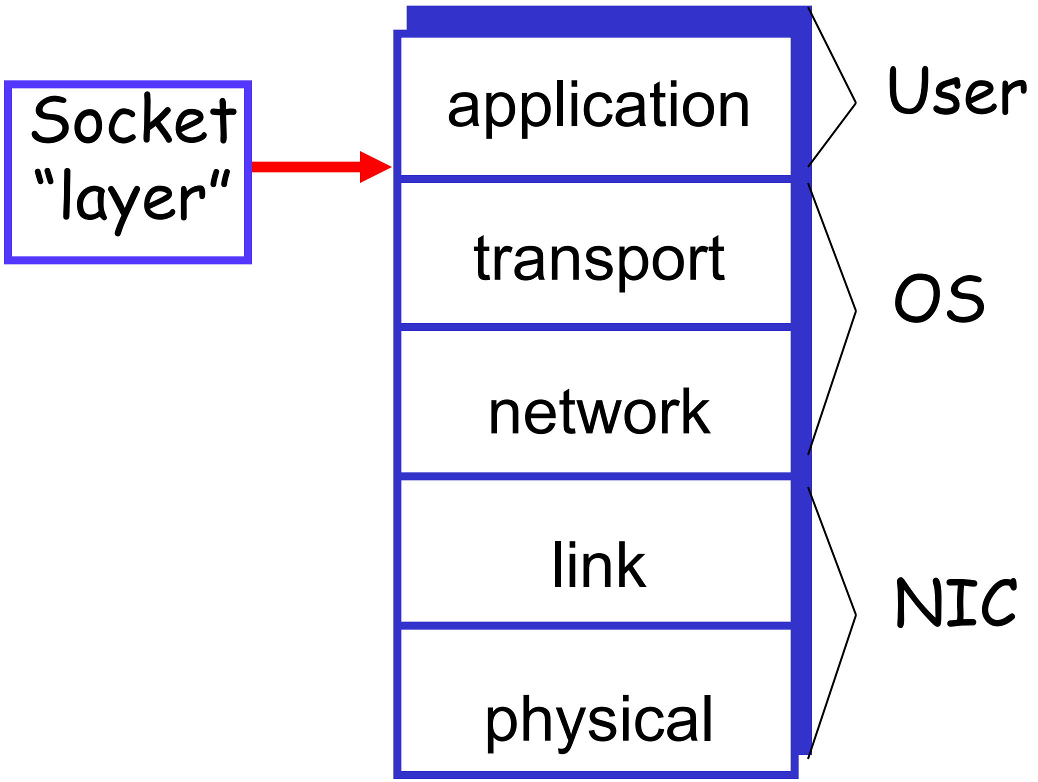
  H_a = h(Alice,Bob,CP,CS,Na,Nb,g^a mod p,g^t mod p,g^{at} mod p)
- But Bob signs

  H_b = h(Alice,Bob,CP,CS,Na,Nb,g^t mod p,g^b mod p,g^{bt} mod p)

Secure Socket Layer (SSL)

Socket layer

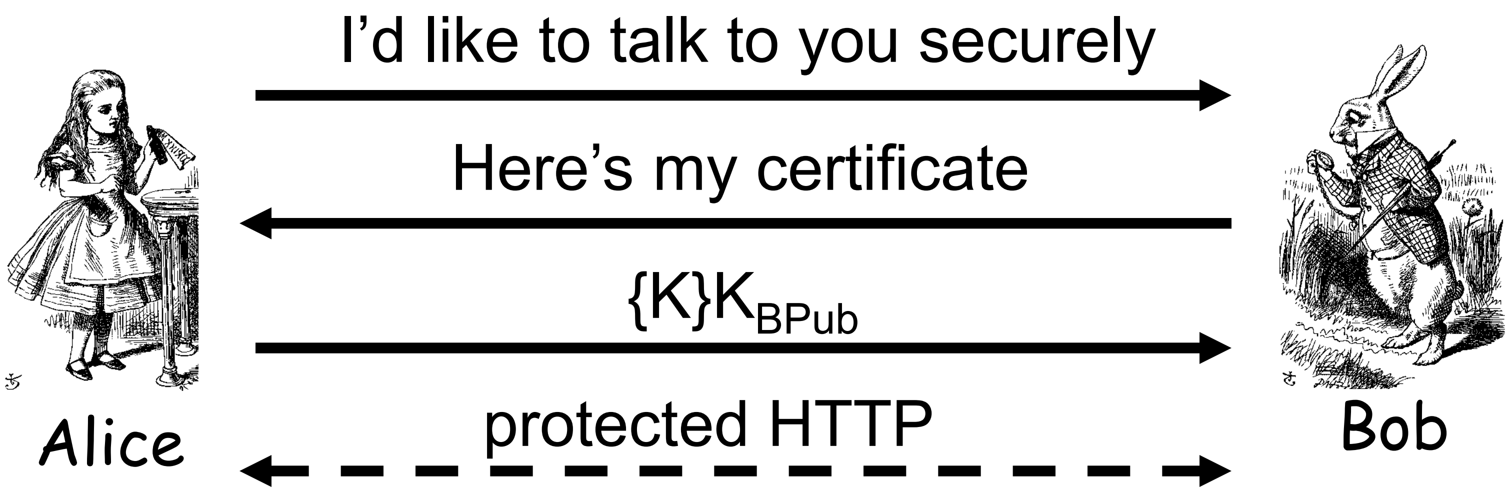
- "Socket layer" lives between application and transport layers
- SSL usually between HTTP and TCP



What is SSL?

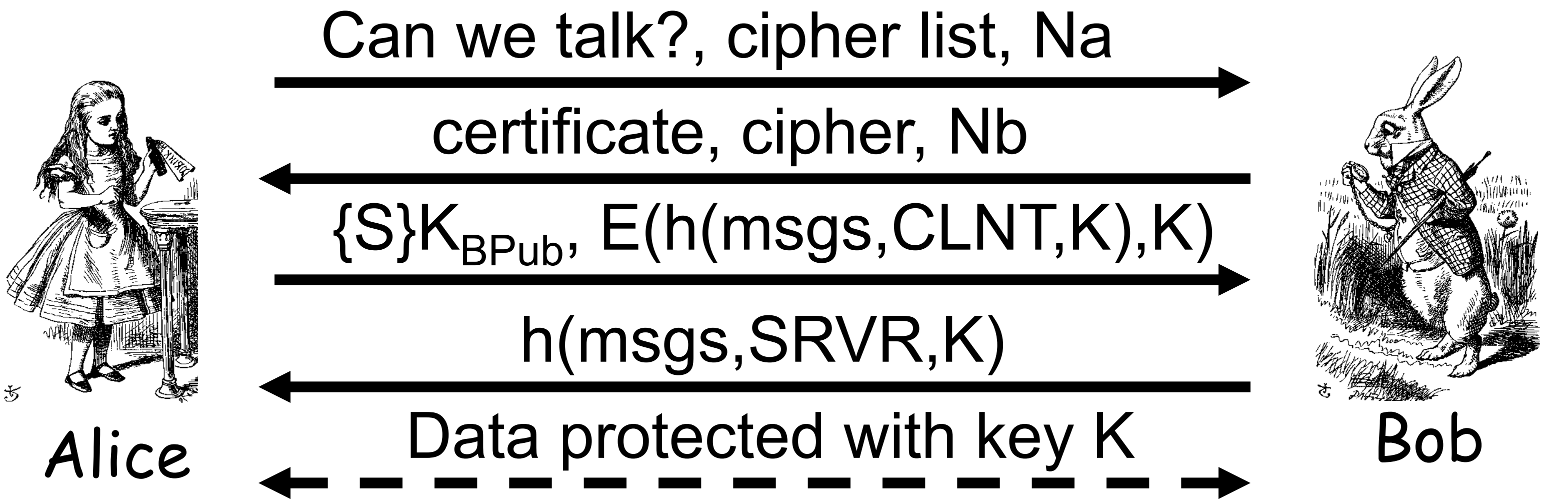
- SSL is the protocol used for majority of secure Internet transactions today
- For example, if you want to buy a book at amazon.com…
  - You want to be sure you are dealing with Amazon (**authentication**)
  - Your credit card information must be protected in transit (**confidentiality** and/or **integrity**)
  - As long as you have money, Amazon does not really care who you are…
  - …so, no need for mutual authentication

Simple SSL-like Protocol



- Is Alice sure she's talking to Bob?
- Is Bob sure he's talking to Alice?

Simplified SSL Protocol



- S is the so-called **pre-master secret**
- K = h(S,Na,Nb)
- "msgs" means all previous messages
- CLNT and SRVR are constants

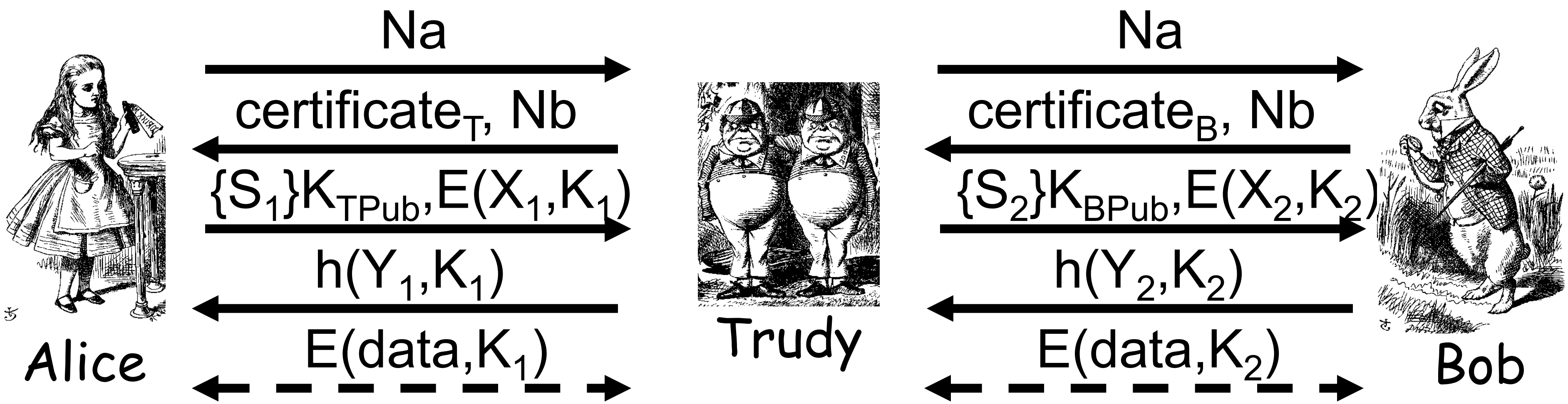
SSL Keys

- 6 "keys" derived from K = h(S,Na,Nb)
  - 2 encryption keys: client and server
  - 2 integrity keys: client and server
  - 2 IVs: client and server
  - Why different keys in each direction?
- **Q:** Why is h(msgs,CLNT,K) encrypted?
- **A:** Apparently, it adds no security…

SSL Authentication

- Alice authenticates Bob, not vice-versa
  - How does client authenticate server?
  - Why would server not authenticate client?
- Mutual authentication is possible: Bob sends **certificate request** in message 2
  - Then client must have a valid certificate
  - But, if server wants to authenticate client, server could instead require password

SSL MiM Attack?

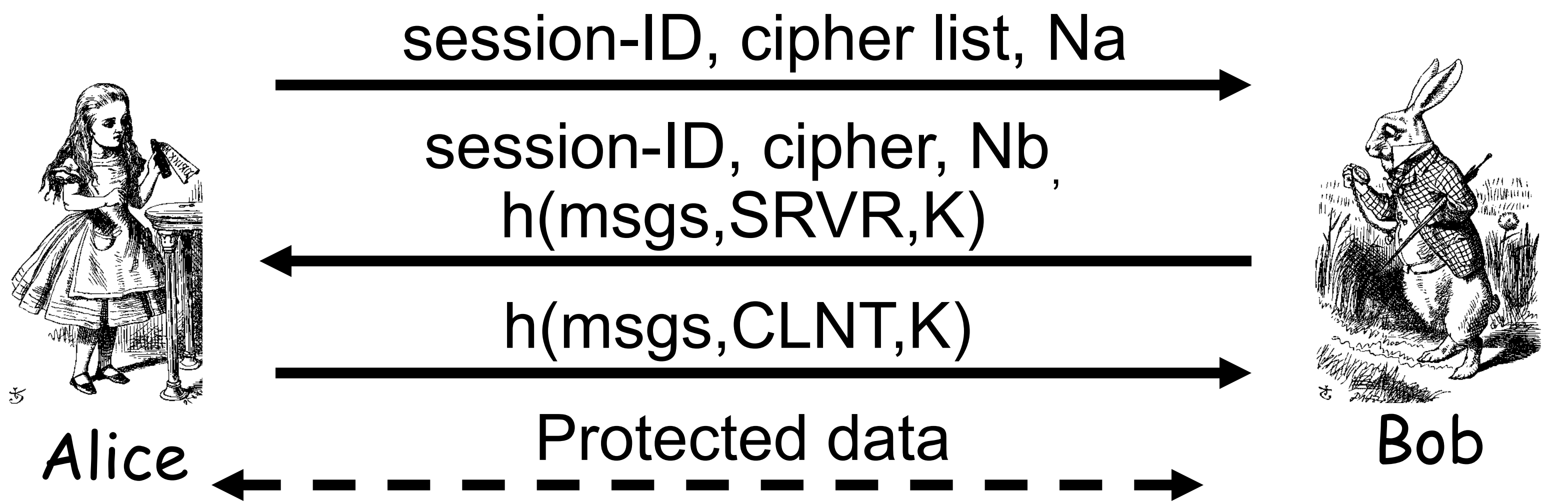


- **Q:** What prevents this MiM "attack"?
- **A:** Bob's certificate must be signed by a certificate authority (CA)
- What does browser do if signature not valid?
- What does user do when browser complains?

SSL Sessions vs Connections

- SSL **session** is established as shown on previous slides
- SSL designed for use with HTTP 1.0
- HTTP 1.0 often opens multiple simultaneous (parallel) **connections**
  - Multiple connections per session
- SSL session is costly, public key operations
- SSL has an efficient protocol for opening new connections ***given an existing session***

SSL Connection



- Assuming SSL **session** exists
- So, S is already known to Alice and Bob
- Both sides must remember session-ID
- Again, K = h(S,Na,Nb)
- **No public key operations!** (relies on known S)

SSL vs IPSec

- IPSec — discussed next
  - Lives at the network layer (part of the OS)
  - Encryption, integrity, authentication, etc.
  - Is overly complex, has some security "issues"
- SSL (and IEEE standard known as TLS)
  - Lives at socket layer (part of user space)
  - Encryption, integrity, authentication, etc.
  - Relatively simple and elegant specification

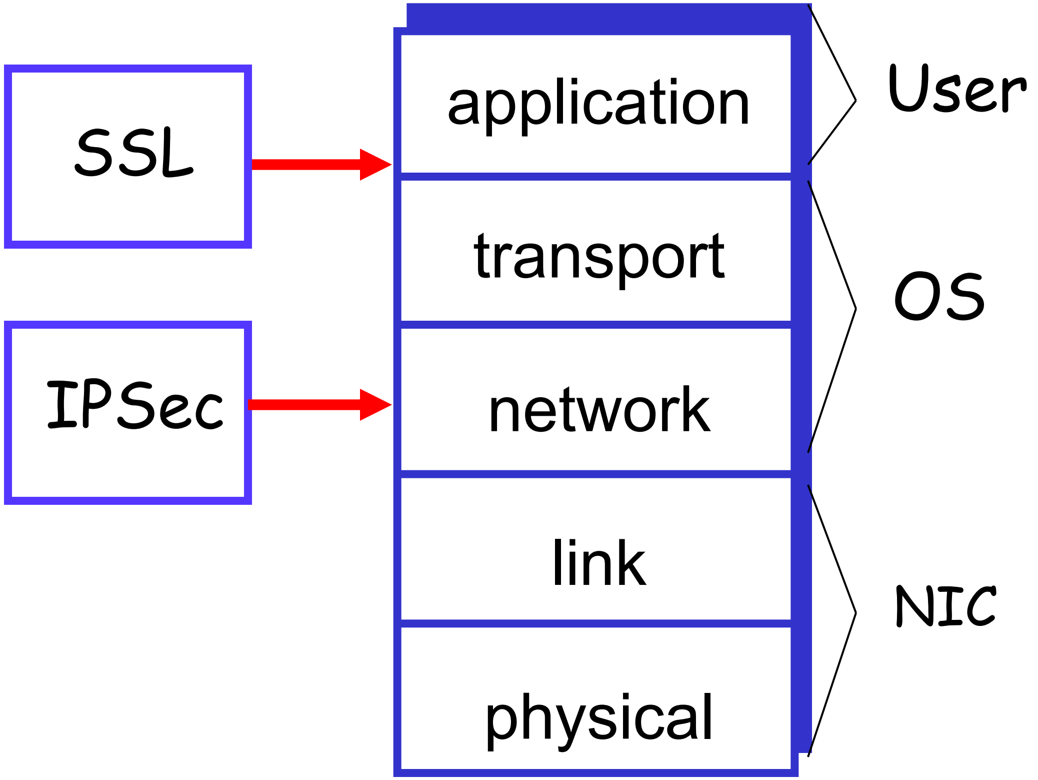
SSL vs IPSec

- IPSec: OS must be aware, but not apps
- SSL: Apps must be aware, but not OS
- SSL built into Web early-on (Netscape)
- IPSec often used in VPNs (secure tunnel)
- Reluctance to retrofit applications for SSL
- IPSec not widely deployed (complexity, etc.)
- The bottom line?
- **Internet less secure than it should be!**

IPSec

IPSec

- IPSec lives at the network layer
- IPSec is transparent to applications



IPSec and Complexity

- IPSec is a complex protocol
- **Over-engineered**
  - Lots of (generally useless) features
- Flawed — Some significant security issues
- Interoperability is serious challenge
  - Defeats the purpose of having a standard!
- Complex
- And, did I mention, it's complex?

IKE and ESP/AH

- Two parts to IPSec…
- **IKE:** Internet Key Exchange
  - Mutual authentication
  - Establish session key
  - Two "phases" — like SSL session/connection
- **ESP/AH**
  - **ESP**: Encapsulating Security Payload — for confidentiality and/or integrity
  - **AH**: Authentication Header — integrity only

IKE

IKE

- IKE has 2 phases
  - Phase 1 — IKE security association (SA)
  - Phase 2 — AH/ESP security association
- Phase 1 is comparable to SSL ***session***
- Phase 2 is comparable to SSL ***connection***
- Not an obvious need for two phases in IKE
  - In the context of IPSec, that is
- If multiple Phase 2's do not occur, then it is **more** costly to have two phases!

IKE Phase 1

- 4 different “key options”
  - Public key encryption (original version)
  - Public key encryption (improved version)
  - Public key signature
  - Symmetric key
- For each of these, 2 different “modes”
  - Main mode and aggressive mode
- **There are 8 versions of IKE Phase 1!**
- Need more evidence it’s over-engineered?

IKE Phase 1

- **Homework:** Read 6 of the 8 Phase 1 variants
  - Public key signatures (main & aggressive modes)
  - Symmetric key (main and aggressive modes)
  - Public key encryption (main and aggressive)
- Why public key encryption and public key signatures?
  - Always know your own private key
  - **May not** (initially) know other side's public key

IKE Phase 1

- Uses ephemeral Diffie-Hellman to establish session key
  - Provides perfect forward secrecy (PFS)
- Let **a** be Alice’s Diffie-Hellman exponent
- Let **b** be Bob’s Diffie-Hellman exponent
- Let **g** be generator and **p** prime
- Recall that p and g are public

IKE Phase 1: Digital Signature (Main Mode)

Alice

IC, CP

IC,RC, CS

IC,RC, g^a mod p, Na

IC,RC, g^b mod p, Nb

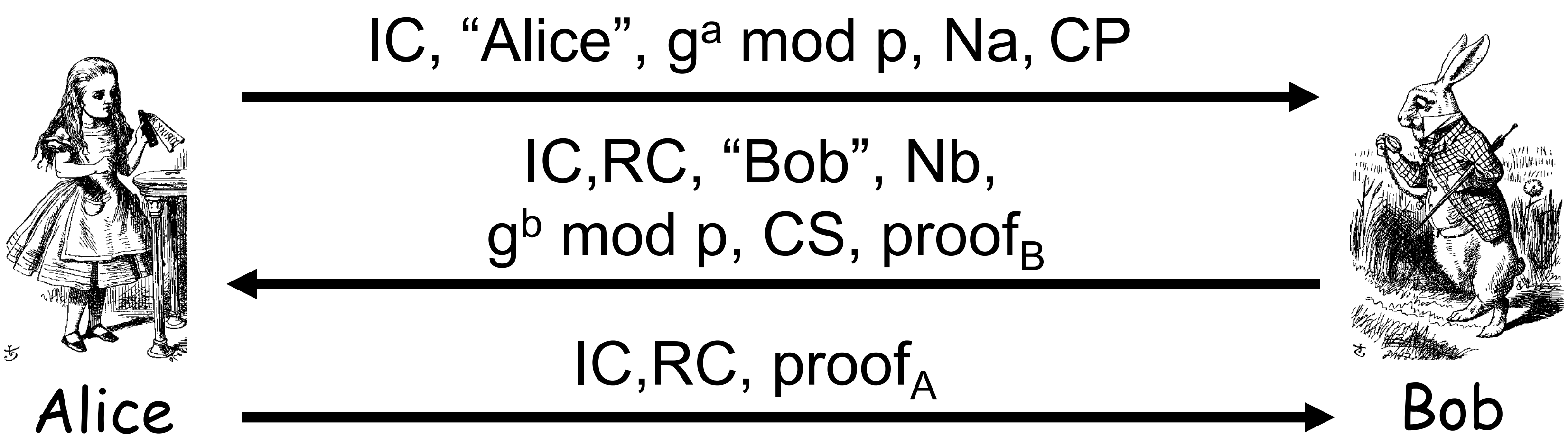
IC,RC, E("Alice", proof_A, K)

IC,RC, E("Bob", proof_B, K)

Bob

- CP = crypto proposed, CS = crypto selected
- IC = initiator "cookie", RC = responder "cookie"
- K = h(IC,RC,g^{ab} mod p,Na,Nb)
- SKEYID = h(Na, Nb, g^{ab} mod p)
- proof_A = {h(SKEYID,g^a mod p,g^b mod p,IC,RC,CP,"Alice")}K_{APriv}

IKE Phase 1: Digital Signature (Aggressive Mode)

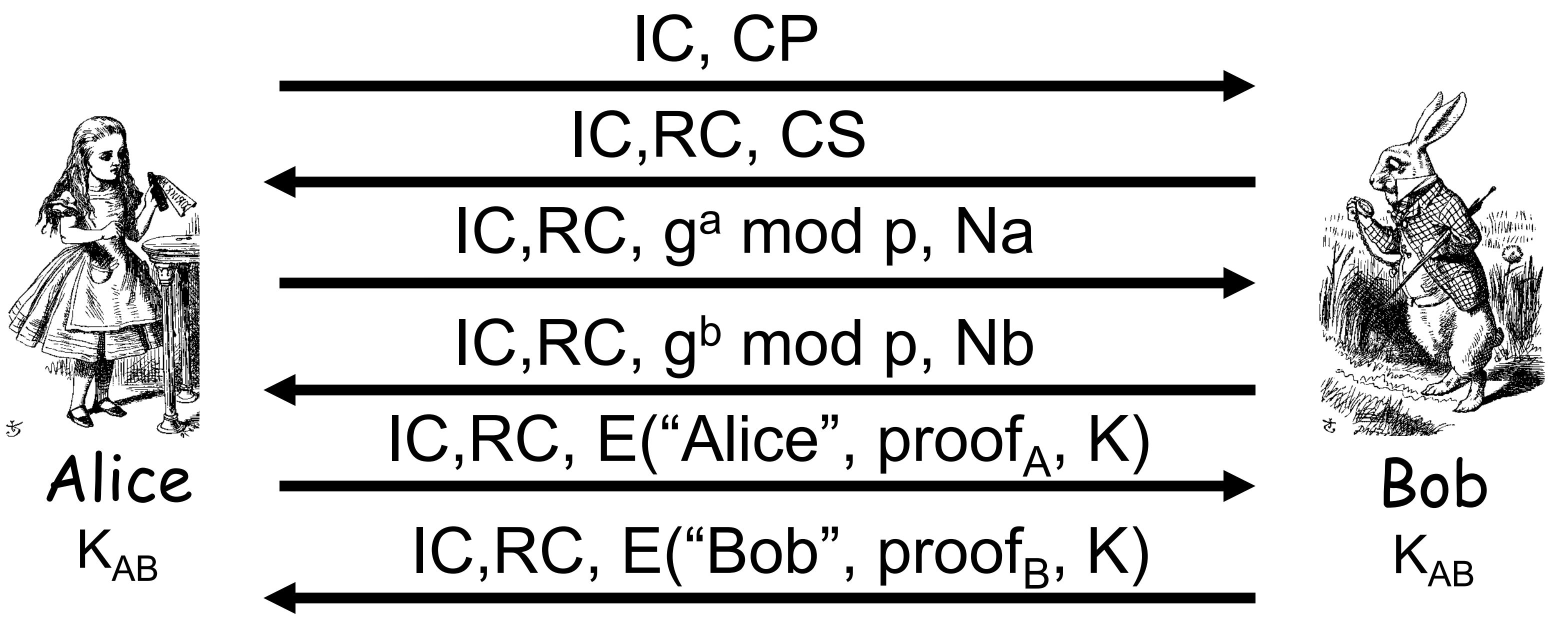


- Main differences from main mode
  - Not trying to hide identities
  - Cannot negotiate g or p

Main vs Aggressive Modes

- Main mode **MUST** be implemented
- Aggressive mode **SHOULD** be implemented
  - So, if aggressive mode is not implemented, "you should feel guilty about it"
- Might create interoperability issues
- For public key signature authentication
  - **Passive attacker** knows identities of Alice and Bob in aggressive mode, but not in main mode
  - **Active attacker** can determine Alice's and Bob's identity in main mode

IKE Phase 1: Symmetric Key (Main Mode)



- Same as signature mode except:
  - K_{AB} = symmetric key shared in advance
  - K = h(IC,RC,g^{ab} mod p,Na,Nb,K_{AB})
  - SKEYID = h(K, g^{ab} mod p)
  - proof_A = h(SKEYID,g^a mod p,g^b mod p,IC,RC,CP,"Alice")

Problems with Symmetric Key (Main Mode)

- Catch-22
  - Alice sends her ID in message 5
  - Alice's ID encrypted with K
  - To find K Bob must know K_{AB}
  - To get K_{AB} Bob must know he's talking to Alice!
- Result: **Alice's IP address used as ID!**
- Useless mode for the "road warrior"
- Why go to all of the trouble of trying to hide identities in 6 message protocol?

IKE Phase 1: Symmetric Key (Aggressive Mode)

Alice

IC, “Alice”, g^a mod p, Na, CP $\rightarrow$

$\leftarrow$ IC,RC, “Bob”, Nb, g^b mod p, CS, proof_B

IC,RC, proof_A $\rightarrow$

Bob

- Same format as digital signature aggressive mode
- Not trying to hide identities…
- As a result, does **not** have problems of main mode
- But does not (pretend to) hide identities

IKE Phase 1 “Cookies”

- IC and RC — cookies (or “anti-clogging tokens”) supposed to prevent DoS attacks
  - No relation to Web cookies
- To reduce DoS threats, Bob wants to remain **stateless** as long as possible
- But Bob must remember CP from message 1 (required for proof of identity in message 6)
- Bob must keep state from 1st message on
  - So, these “cookies” offer little DoS protection

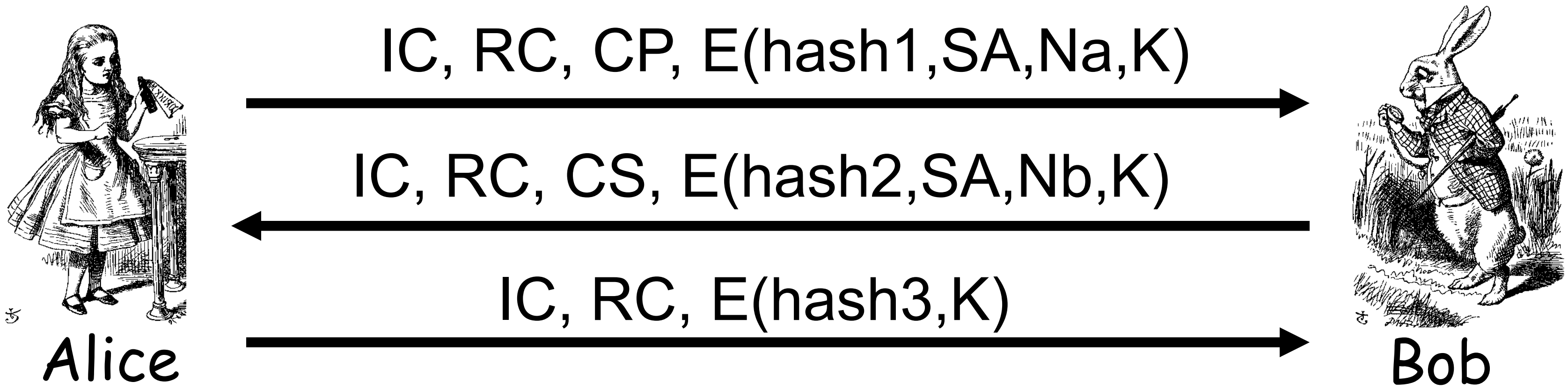
IKE Phase 1 Summary

- Result of IKE phase 1 is
  - Mutual authentication
  - Shared symmetric key
  - IKE **Security Association (SA)**
- But phase 1 is expensive
  - Especially in public key and/or main mode
- Developers of IKE thought it would be used for lots of things — not just IPSec
  - Partly explains the over-engineering…

IKE Phase 2

- Phase 1 establishes IKE SA
- Phase 2 establishes IPSec SA
- Comparison to SSL
  - SSL session is comparable to IKE Phase 1
  - SSL connections are like IKE Phase 2
- IKE **could** be used for lots of things…
- …but in practice, it's **not!**

IKE Phase 2



- Key K, IC, RC and SA known from Phase 1
- Proposal CP includes ESP and/or AH
- Hashes 1,2,3 depend on SKEYID, SA, Na and Nb
- Keys derived from KEYMAT = h(SKEYID,Na,Nb,junk)
- Recall SKEYID depends on phase 1 key method
- Optional PFS (ephemeral Diffie-Hellman exchange)

IPSec

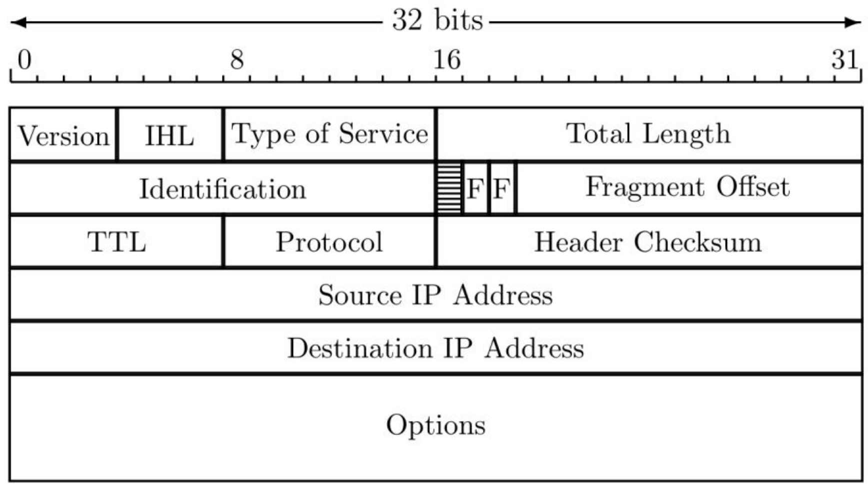
- After IKE Phase 1, we have an IKE SA
- After IKE Phase 2, we have an IPSec SA
- Authentication completed and have a shared symmetric key (session key)
- Now what?
  - We want to protect **IP datagrams**
  - But what is an IP datagram?
  - From the perspective of IPSec…

IP Review

- IP datagram is of the form

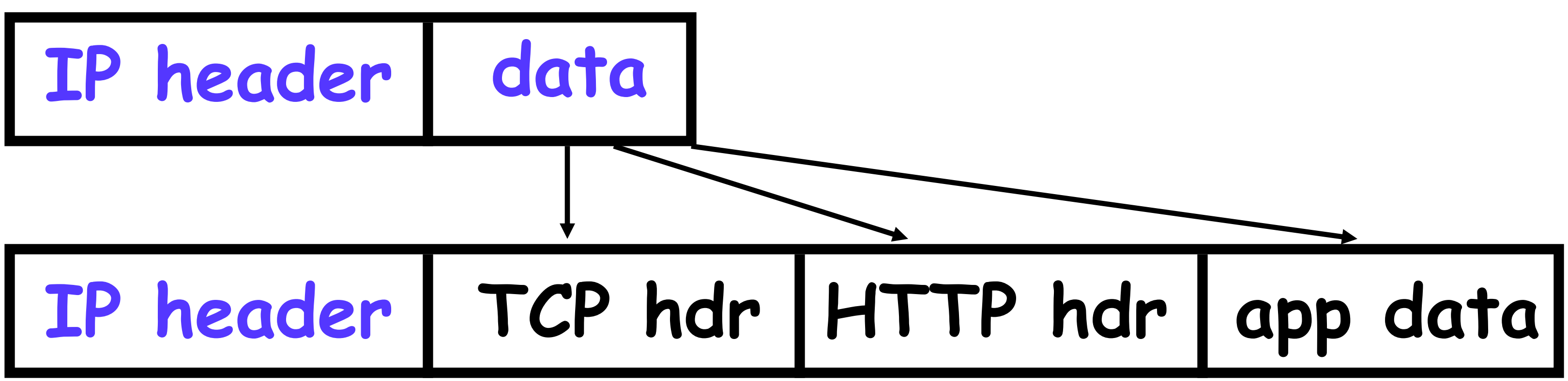
IP header	data

- Where IP header is



IP and TCP

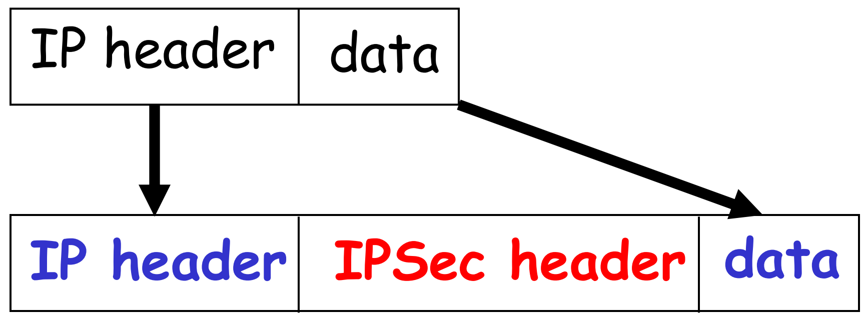
- Consider Web traffic, for example
  - IP encapsulates TCP and…
  - …TCP encapsulates HTTP



- IP **data** includes TCP header, etc.

IPSec Transport Mode

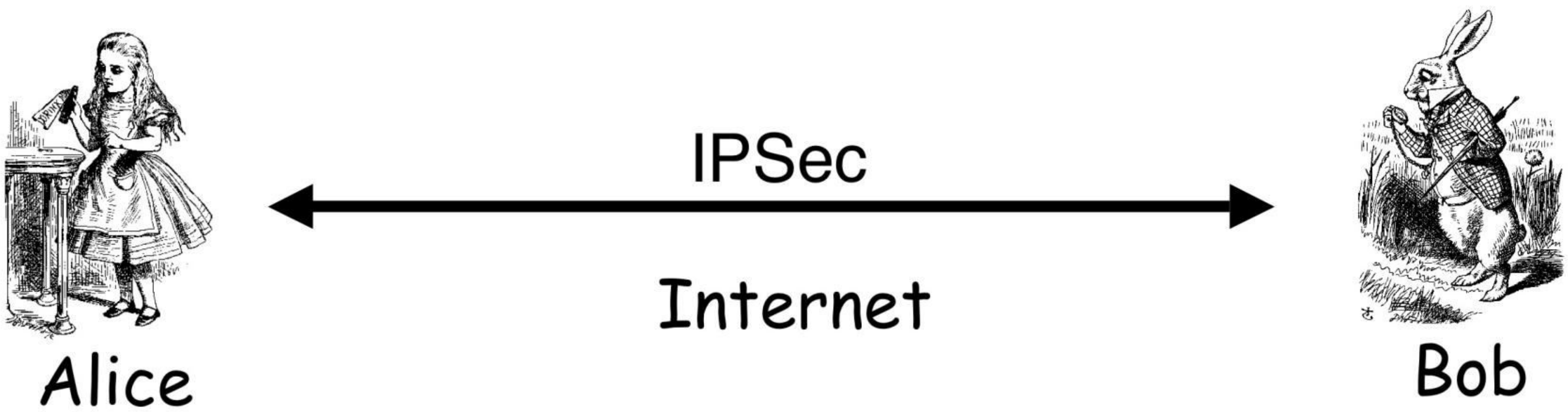
- IPSec **Transport Mode**



- Transport mode designed for ***host-to-host***
- Transport mode is efficient
  - Adds minimal amount of extra header
- The original header remains
  - Passive attacker can see who is talking

IPSec: Host-to-Host

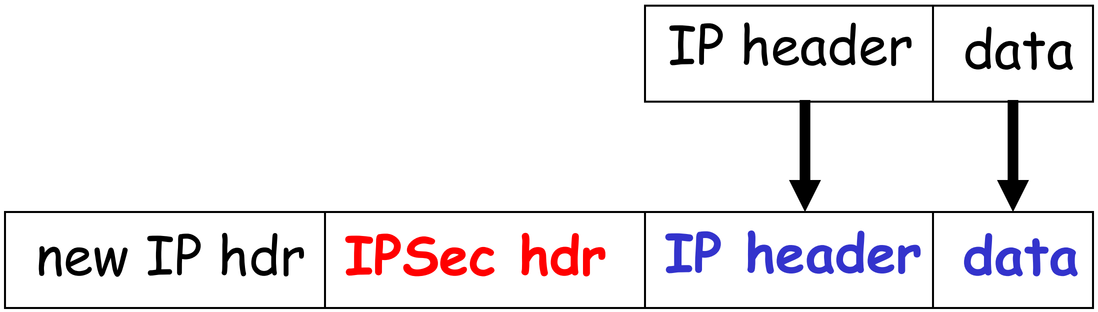
- IPSec transport mode used here



- There may be firewalls in between
  - If so, is that a problem?

IPSec Tunnel Mode

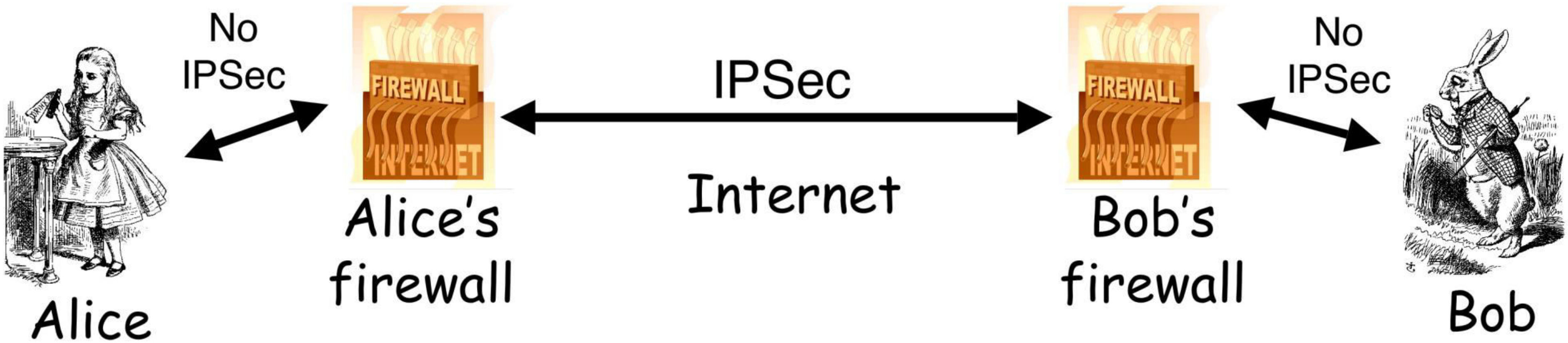
- IPSec **Tunnel Mode**



- Tunnel mode for ***firewall-to-firewall*** traffic
- Original IP packet encapsulated in IPSec
- Original IP header not visible to attacker
  - New IP header from firewall to firewall
  - Attacker does not know which hosts are talking

IPSec: Firewall-to-Firewall

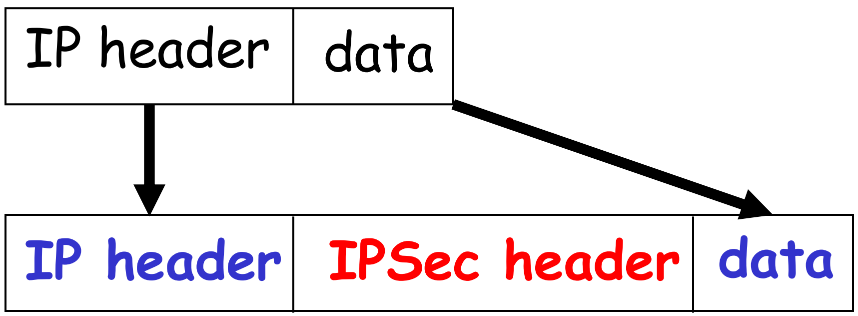
- IPSec tunnel mode used here



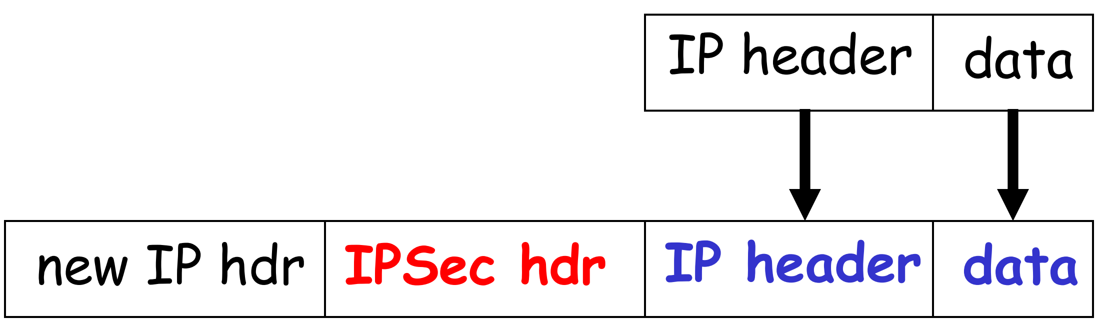
- Note: Local networks not protected
- Is there any advantage here?

Comparison of IPSec Modes

- Transport Mode



- Tunnel Mode



- Transport Mode
  - Host-to-host
- Tunnel Mode
  - Firewall-to-firewall
- Transport Mode not necessary…
- …but it's more efficient

IPSec Security

- What kind of protection?
  - Confidentiality?
  - Integrity?
  - Both?
- What to protect?
  - Data?
  - Header?
  - Both?
- ESP/AH do some combinations of these

AH vs ESP

- AH — Authentication Header
  - **Integrity only** (no confidentiality)
  - Integrity-protect everything beyond IP header and some fields of header (why not all fields?)
- ESP — Encapsulating Security Payload
  - Integrity and **confidentiality** both **required**
  - Protects everything beyond IP header
  - Integrity-only by using NULL encryption

ESP NULL Encryption

- According to RFC 2410
  - NULL encryption "is a block cipher the origins of which appear to be lost in antiquity"
  - "Despite rumors", there is no evidence that NSA "suppressed publication of this algorithm"
  - Evidence suggests it was developed in Roman times as exportable version of Caesar's cipher
  - Can make use of keys of varying length
  - No IV is required
  - Null(P,K) = P for any P and any key K
- Bottom line: Strange option for ESP

Why Does AH Exist? (1)

- Cannot encrypt IP header
  - Routers must look at the IP header
  - IP addresses, TTL, etc.
  - IP header exists to route packets!
- AH protects **immutable fields** in IP header
  - Cannot integrity protect all header fields
  - TTL, for example, will change
- ESP does not protect IP header at all

Why Does AH Exist? (2)

- ESP encrypts everything beyond the IP header (if non-null encryption)
- If ESP-encrypted, firewall cannot look at TCP header (e.g., port numbers)
- Why not use ESP with NULL encryption?
  - Firewall sees ESP header, but does not know whether null encryption is used
  - End systems know, but **not** the firewalls

Why Does AH Exist? (3)

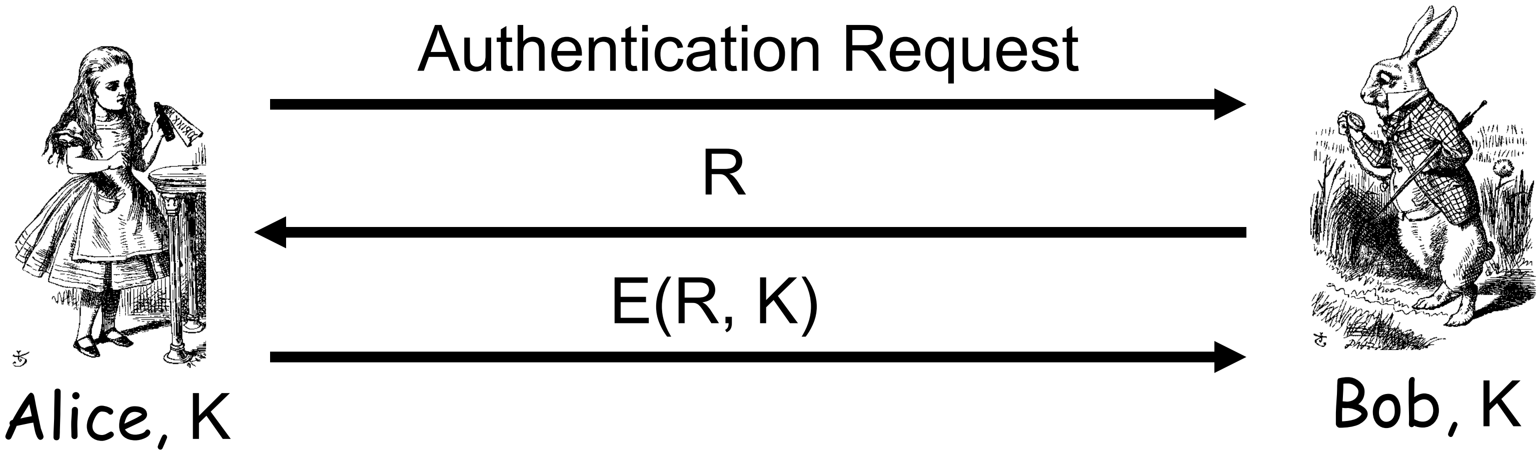
- The real reason why AH exists:
  - At one IETF meeting "someone from Microsoft gave an impassioned speech about how AH was useless…"
  - "…everyone in the room looked around and said `Hmm. He's right, and we hate AH also, but if it annoys Microsoft let's leave it in since we hate Microsoft more than we hate AH.' "

WEP

WEP

- WEP — Wired Equivalent Privacy
- The stated goal of WEP is to **make wireless LAN as secure as a wired LAN**
- According to Tanenbaum:
  - *"The 802.11 standard prescribes a data link-level security protocol called WEP (Wired Equivalent Privacy), which is designed to make the security of a wireless LAN as good as that of a wired LAN. Since the default for a wired LAN is no security at all, this goal is easy to achieve, and WEP achieves it as we shall see."*

WEP Authentication



- Bob is ***wireless access point***
- Key K shared by access point and **all users**
  - Key K seldom (if ever) changes
- WEP has many, many, many security flaws

WEP Issues

- WEP uses RC4 cipher for confidentiality
  - RC4 is considered a strong cipher
  - But WEP introduces a subtle flaw…
  - …making cryptanalytic attacks feasible
- WEP uses CRC for "integrity"
  - Should have used a MAC, HMAC, or similar
  - CRC is for error detection, not crypto integrity
  - *Everyone* should know ***NOT*** to use CRC here…

WEP Integrity Problems

- WEP "integrity" gives no crypto integrity
  - CRC is linear, so is stream cipher (XOR)
  - Trudy can change **ciphertext and CRC** so that checksum on ***plaintext*** remains valid
  - Then Trudy's introduced changes go undetected
  - Requires no knowledge of the plaintext!
- CRC does ***not*** provide a cryptographic integrity check
  - CRC designed to detect random errors
  - Not to detect intelligent changes

More WEP Integrity Issues

- Suppose Trudy knows destination IP
- Then Trudy also knows keystream used to encrypt IP address, since

  C = destination IP address $\oplus$ **keystream**

- Then Trudy can replace **C** with

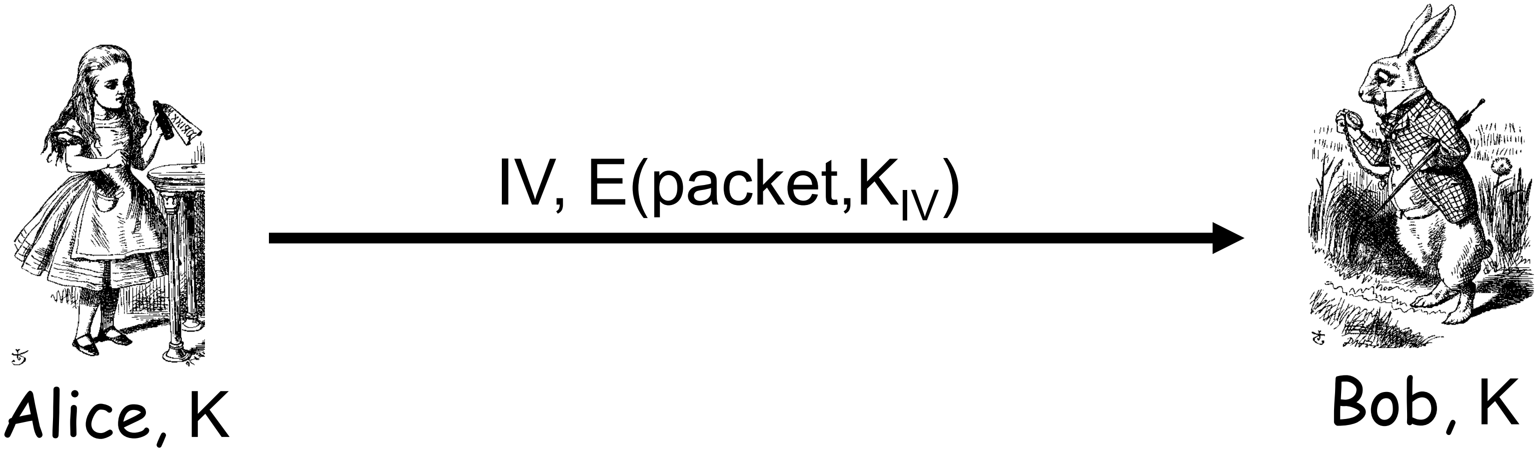
  C′ = Trudy's IP address $\oplus$ **keystream**

- And change the CRC so no error detected
  - Then what happens??
- Moral: Big problems when integrity fails

WEP Key

- Recall WEP uses a long-term key K
- RC4 is a stream cipher, so each packet must be encrypted using a different key
  - Initialization Vector (IV) sent with packet
  - Sent in the clear, that is, IV is **not** secret
  - Note: IV similar to “MI” in WWII ciphers
- Actual RC4 key for packet is (IV,K)
  - That is, IV is **pre-pended** to long-term key K

WEP Encryption



- K_{IV} = (IV,K)
  - That is, RC4 key is K with 3-byte IV pre-pended
- Note that the IV is known to Trudy

WEP IV Issues

- WEP uses 24-bit (3 byte) IV
  - Each packet gets its own IV
  - Key: IV pre-pended to long-term key, K
- Long term key K seldom changes
- If long-term key and IV are same, then same keystream is used
  - This is bad, bad, really really bad!
  - Why?

WEP IV Issues

- Assume 1500 byte packets, 11 Mbps link
- Suppose IVs generated in sequence
  - Since $1500 \cdot 8/(11 \cdot 10^6) \cdot 2^{24} = 18{,}000$ seconds…
  - …an IV repeat in about 5 hours of traffic
- Suppose IVs generated at random
  - By birthday problem, some IV repeats in seconds
- Again, repeated IV (with same K) is ***bad***

Another Active Attack

- Suppose Trudy can insert traffic and observe corresponding ciphertext
  - Then she knows the keystream for some IV
  - She can decrypt any packet that uses that IV
- If Trudy does this many times, she can then decrypt data for lots of IVs
  - Remember, IV is sent in the clear
- Is such an attack feasible?

Cryptanalytic Attack

- WEP data encrypted using RC4
  - Packet key is IV and long-term key K
  - 3-byte IV is pre-pended to K
  - Packet key is (IV,K)
- Recall IV is sent in the clear (not secret)
  - New IV sent with every packet
  - Long-term key K seldom changes (maybe never)
- So Trudy always knows IV and ciphertext
  - Trudy wants to find the key K

Cryptanalytic Attack

- 3-byte IV pre-pended to key
- Denote the RC4 key **bytes**…
  - …as $K_0, K_1, K_2, K_3, K_4, K_5, \ldots$
  - Where IV = (K_0, K_1, K_2) , which Trudy knows
  - Trudy wants to find $K = (K_3, K_4, K_5, \ldots)$
- Given enough IVs, Trudy can easily find key K
  - Regardless of the length of the key
  - Provided Trudy knows first keystream byte
  - **Known plaintext** attack (1st byte of each packet)
  - Prevent by discarding first 256 keystream bytes

WEP Conclusions

- Many attacks are practical
- Attacks have been used to recover keys and break real WEP traffic
- How to prevent WEP attacks?
  - Don't use WEP
  - Good alternatives: WPA, WPA2, etc.
- How to make WEP a little better?
  - Restrict MAC addresses, don't broadcast ID, …