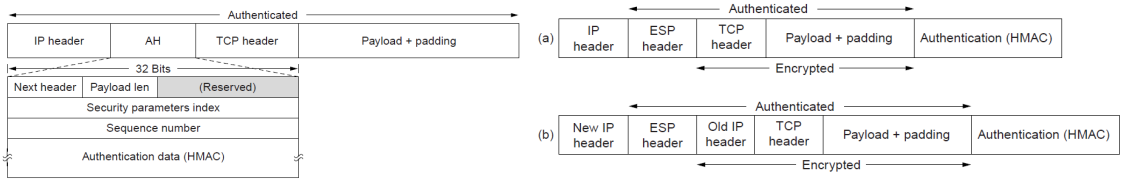
Hash Functions(message digest)

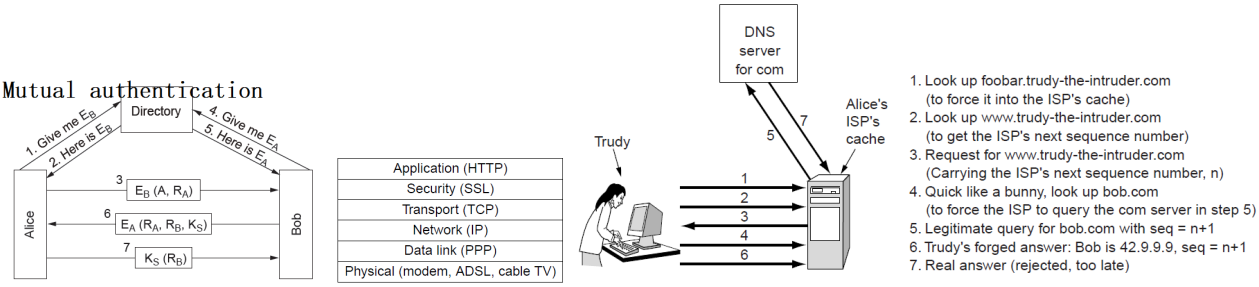
takes an arbitrary string of bits and transform them into a uniform (fixed size) result. h(m1) = h(m2) then, m1 = m2(Collision resistance property) **ideal hash** function: random mapping, **attack on a hash function** is a **non-generic method**(依赖于特定哈希函数的结构、弱点或特性)
Iterative hash functions : padding(last block:inputs length), divide, Compression.H0:fixed value, $H_i = h'(H_{i-1}, m_i)$ Hk: last block hash function outcome
Message Diagest 5 (MD5) 128 bit hash function. 1.512bits block, 2. 4 (32 bits)words, 512→16个32bit.初始化四个 3. compression function h' mixes (message block+state), 4 rounds(XOR, AND, OR etc, efficient on 32-bit CPUs). 4. input stat+result → output of h'
SHA-1 NIST. 160 bit result size. **SHA-224, SHA-256, SHA-384, SHA-512** (支持128, 192, 256key size)of AES(Advanced Encryption Standard分组加密) and the 112 bit 3DES(Data). **SHA-3** Permutation-based Hash, Extendable-Output. on KECCAK algorithm (wining algorithm).
Hash Functions weaknesses Length Extensions : $m \to m_1 \dots m_k$, $m' \to m_1 \dots m_{k+1}$, $h(m') = h'(h(m), m_{k+1})$ --no special processing at the end. $h(m) = h(X||m)$, where : X : **secret** know to Bob and Alice, h is a not ideal function.→ Eve can append text, updated the authentication code to match the new message. Partial-Message Collision Inheriting the iterative structure, 攻击者希望找到$h(m) = h(m||X)$, X : the **authentication key**. birthday attack. succeeds, an iterative hash function; fails, it is the ideal hash function. Weakness Fix SHA-3 addresses this, they are designed to include resistance against collision, preimage, second preimage attacks.h(iterative hash function) Short term fix or workaround $h_{DBL} := h(h(m)||m)$ →processing slow, pre-stage in buffer. Efficient short term fix h_d是中间哈希函数b: len(block underlying compression function. $h_d(m) := h(h(O^b(全0)||m))$ security level of **min(k, n/2)** ,k(security level of h), n(hash result size).
Message Authentication Code (MAC) Chaining Block Cipher (CBC)-MAC create MAC, m is encrypted, keep last block of cipher text. message $P1, \dots, Pk$, $H0 := IV$, $Hi := Ek(Pi \oplus Hi - 1)$, $MAC := Hk$. 初始IV=0. en,au用不同key. collision attacks--1/2len(block size)安全. **A collision attack**: M(CBC-MAC function)→ $M(a) = M(b) -> M(a||c) = M(b||c)$. c:a single block, $M(a||c) = E_k(c \oplus M(a))$, $M(b||c) = E_k(c \oplus M(b))$, birthday paradox. find M(a)=M(b). attack: get the sender to authenticate a||c, replace with b||c and not changing MAC。
Implementing CBC-MAC 1.Construct a string s from the concatenation of l and m, l=len(encoded m) 2. Pad s直到block size整数倍. 3. Apply CBC-MAC to the padded s. 4. Output the last(part) ciphertext block,不要输出中间值. Instead of using CBC-MAC directly use CMAC CMAC NIST, based on CBC-MAC. CMAC treats the last block不同. CMAC XORs one of two special values (derived from CMAC key) into the last block prior the last block cipher encryption. CMAC key dependency in the length of messages and the cipher's block length. Keyed-Hash based MAC a,b(specified constants) $h(K \oplus a||h(K \oplus b||m))$ Works with any iterative function. HMAC with SHA-1 less insecure than SHA-1. HMAC avoids key recovery attacks that would reveal K to the attacker. HMAC is limited by n/2 bit security. Galois MAC efficient, 128 bit block ciphers. Authentication function input: key, message and a nonce(初始随机数). uses a universal hash function, encrypts the output with a block cipher in CTR mode(nonce和counter一起输入,块密码转换成流密码) to obtain MAC. The IV(initial vector) is created using a function of its nonce. 64 bits of security. Not for short MAC values. --**recommend**(Block size) SHA3-224 144, SHA3-256 136, SHA3-384 104, and SHA3-512 72 MAC usage : Preventing the **reply attack**重复发送先前消息来欺骗系统,组合d||m发送, **Horton principle** "Authenticate what is meant, not what is said", Authentication包括 protocol identifier, version number message identifier, sizes of various fields. Layered OSI protocol, authentication isolation认证隔离在各层中

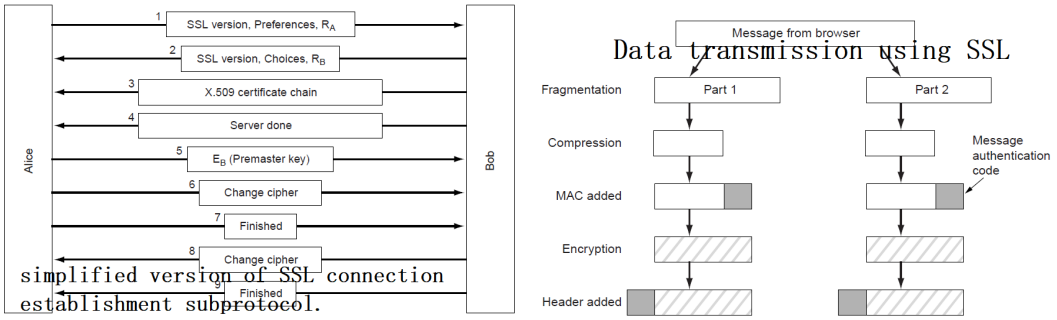
Communication Security

IPsec(Internet Protocol Security): Firewalls, Virtual private networks, Wireless security.节点建立安全通信时会建立一个或多个**SA** (security association)有**security identifier**. IPsec authentication header in transport mode for IPv4. 用于建立 VPN，加密IP 数据包，同时验证数据包的来源
Virtual Private Networks

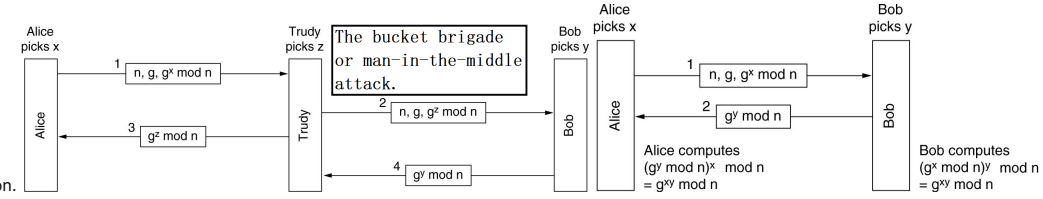


transport(avoid traffic analysis)/tunnel(end-to-end faster) Hypertext Transfer Protocol directory储存管理认证凭据的数据库





Web Security: Threats, Secure naming(Domain Name System, 攻击DNS 发回去的地址是错误), SSL The Secure Sockets Layer, Mobile code security
Service Provider: DNSsec fundamental services: Proof of data originated. Public key distribution. Transaction and request authentication. SIG is encoded from A& Key
SSL server and websitesecurity connection: Parameter negotiation between client and server. Authentication of the server by client. Secret communication. Data integrity protection.



Difflie-Hellman Key Exchange

The Secure Channel

Shared key, known to Bob, Alice only. Session Key K, one single communication. security level = 1/2 length
Distributing data among entities: only consider Streams of data that can be distributed in a discrete. A data stream is then separated into discrete messages, assembled at the receiver side. Transport system (layer) cryptographically not reliable. TCP/IP不可靠,可靠通信协议不存在(for traffic analysis)
Security Properties: secrecy(Difficult to achieve Eve can monitor secured data traffic (size and timing) among 交流双方. Bob 只有 a subset(dropped or intercepted) of such sequence from Alice. No resend again. not implemented(not incentive)
Authentication and Encryption 1.ea: in parallel, 2.ae: see only(a), modify only(e) 3.ea, concatenate results, ae组合
A secure channel design initialize message numbering(Must increase monotonically, unique 32→2^32 -1), authentication and encryption.
Authentication HMAC-SHA-256, 256- bit. MAC input (mi , xi), xi(contextual information ex. protocol, files negotiated and protocol version) same xi for Alice, Bob. The Mac value a: $a_i = MAC(i||l(x_i)||x_i||m_i)$,连接 (32-bit)i and len(xi)=l(x) .
Encryption AES `E()` , CTR, nonce handled by secure channel. message size 16 . 2^32 bytes, in turns the block counter is limited by 32 bits. Key stream密钥流k0, k1, ... for a message with nonce i. $k_0, \ldots, k_{2^{32}-1} = E(K, 0||i||0)||E(K, 1||i||0)||\ldots||E(K, 2^{32}-1||i||0)$ Plaintext block: 32-bit block number, 32 bit message number and 64 bits of zeros. Only used the first l(mi)+32 bytes. (Concatenation of mi and ai) are XOR with $k_0, \ldots, k_{l(mi)+31}$
Frame Format encoded i as a 32 bit integer. First the least significant byte(最低有效字节), then the encrypted mi and ai. Initialization Two functions: Setting the keys, Setting the message numbers Sending a message Message reception

function InitializeSecureChannel
input: K Key of the channel, 256 bits.
R Role. Specifies if this party is Alice or Bob.
output: $\mathcal{S}$ State for the secure channel.

First compute the four keys that are needed. The four strings are ASCII strings without any length or zero-termination.
KeySendEnc ← SHA_d-256(K ‖ "Enc Alice to Bob")
KeyRecEnc ← SHA_d-256(K ‖ "Enc Bob to Alice")
KeySendAuth ← SHA_d-256(K ‖ "Auth Alice to Bob")
KeyRecAuth ← SHA_d-256(K ‖ "Auth Bob to Alice")
Swap the encryption and decryption keys if this party is Bob.
if R = "Bob" **then**
swap(KeySendEnc, KeyRecEnc)
swap(KeySendAuth, KeyRecAuth)
fi
Set the send and receive counters to zero. The send counter is the number of the last sent message. The receive counter is the number of the last received message.
(MsgCntSend, MsgCntRec) ← (0, 0)
Package the state.
$\mathcal{S}$ ← (KeySendEnc, KeyRecEnc, KeySendAuth, KeyRecAuth, MsgCntSend, MsgCntRec)
return $\mathcal{S}$

function ReceiveMessage
input: $\mathcal{S}$ Secure session state.
t Text received from the transmitter.
x Additional data to be authenticated.
output: m Message that was sent.

The received message must contain at least a 4-byte message number and a 32-byte MAC field. This check ensures that all the future splitting operations will work.
assert $\ell(t) \geq 36$
Split t into i and the encrypted message plus authenticator. The split is well-defined because i is always 4 bytes long.
$i \| t \leftarrow t$
Generate the key stream, just as the sender did.
$K \leftarrow$ KeyRecEnc
$k \leftarrow E_K(0 \| i \| 0) \| E_K(1 \| i \| 0) \| \cdots$
Decrypt the message and MAC field, and split. The split is well-defined because a is always 32 bytes long.
$m \| a \leftarrow t \oplus$ First-$\ell(t)$-bytes(k)
Recompute the authentication. The values $\ell(x)$ and i are encoded in four bytes, least significant byte first.
$a' \leftarrow$ HMAC-SHA-256(KeyRecAuth, $i \| \ell(x) \| x \| m$)
if $a' \neq a$ **then**
destroy k, m
return AuthenticationFailure
else if $i \leq$ MsgCntRec **then**
destroy k, m
return MessageOrderError
fi
MsgCntRec ← i
return m

function SendMessage
input: $\mathcal{S}$ Secure session state.
m Message to be sent.
x Additional data to be authenticated.
output: t Data to be transmitted to the receiver.

First check the message number and update it.
assert MsgCntSend < $2^{32} - 1$
MsgCntSend ← MsgCntSend + 1
i ← MsgCntSend
Compute the authentication. The values $\ell(x)$ and i are encoded in four bytes, least significant byte first.
$a \leftarrow$ HMAC-SHA-256(KeySendAuth, $i \| \ell(x) \| x \| m$)
$t \leftarrow m \| a$
Generate the key stream. Each plaintext block of the block cipher consists of a four-byte counter, four bytes of i, and eight zero bytes. Integers are LSByte first, E is AES encryption with a 256-bit key.
$K \leftarrow$ KeySendEnc
$k \leftarrow E_K(0 \| i \| 0) \| E_K(1 \| i \| 0) \| \cdots$
Form the final text. Again, i is encoded as four bytes, LSByte first.
$t \leftarrow i \| (t \oplus$ First-$\ell(t)$-bytes(k))
return t

Message order: checks, expected increasing. IPsec maintains a **reply protection window** instead of keeping track of the message number. IPsec uses a bit map if (last received)d: [d-31, d-30,..., d-1, d]. **Alternatives to the secure channel** Dedicated use of block cipher modes: CCM mode (Counter with CBC-MAC), OCB mode (Offset Codebook Mode), CWC Mode (Carter–Wegman + CTR mode), GCM Mode (Galois Counter mode) CCM and GCM are recommended, not provide full secure channel. Adaptability of the secure channel will allow the use of this modes.

Implementation Issues

The weakest link properly木桶效应. Individual components. attak更难对比implement. successful brakes可能使得系统构建问题不被发现. Implementation design: Specifies how the program works internally. Modularized approach, program→sub-programs (modules)
testing can only show the presence of bugs, never the absence of bugs. Per bug finding, implement a test that detects the bug. Fix the bug, check that the test did not find the bug. Review what is causing the bug whenever you find it. Keep track of every bug event. Statistical analysis of the bugs found. **Lack of functionality** is the difference between correct and secure software. Secure software should be able to implement it, Eve无论如何不能做X. Functionality can be tested但lack of functionality no.
Wiping state (Transient secrets are kept/store in memory)Wipe any type of information as soon as it not used or needed. Involve wiping memory locations. &C++: destructor function for each object, heap-allocated object are refurbish翻新. &Java: all objects live in Heap memory is garbage collected, finalization routing are run at program exit. &Not possible to wipe the state of the CPU registers manually. Swap file Virtual memory system(Windows Linux) increase parallel. 避免敏感数据存储在交换文件中 Caches a copy of the most recently used data from main memory. Data retention by Memory Overwriting data in memory does not delete the data. **Static RAM (SRAM)**. Power cycling the memory could return previous states (old data). **Dynamic RAM (DRAM)**. store a small charge on a small capacitor. An attacker with physical access could recover the data. DRAM Slow decay of charge in capacitors 室温. **Partial solution.** 不存m,存R random string m → R, h(R) ⊕ m. read both, hash the first and XOR them to get m.
Multi-access 同一台机器设置访问等级Super user and regular user. **Data Integrity** Assumption the hardware use is reliable. Memory could use an Error-correcting code. **Quality of Code** Simplicity: Complexity is worst any for security. Modularization. Assertions: Professional paranoia. 断言错误程序终止. Buffer overflow. (From Algol 60 array bound checking防止程序员无意中访问数组之外的内存)
Testing 1.Generic functional tests developed from the module's functional specifications, 2. Developed by the programmer of the module itself. **Side channel attacks** Timing时序信息 information. ex. Magnetic field, RF emissions, power consumption, timing ...

RSA Ronal Riverst, Adi Shamir, Leonard Alleman (1978)

Based on a trapdoor one-way function(在一个方向上很容易计算，但在反向上很难计算), such the one used in Diffie-Hellman key generation algorithm(基于数论离散对数问题，依赖陷门单向函数的属性,计算给定基数（通常是一个大素数）的幂所得到的结果的离散对数是困难的). 1.Choose two large primes, p and q 2.Compute `n = p × q` and `z = ( p − 1) × (q − 1)` . 3.Choose number relatively prime to z call it d. 4.Find e such that `e × d = 1` mod z. P (plaintext), `0 ≤ P < n` . Plaintext is group into blocks of k bits, k is the largest integer for which $2^k < n$ is true. To encrypt a message, P, compute $C = P^e$ (mod n). To decrypt C, compute $P = C^d$ (mod n) The public key consists of the pair (e, n), and the private key consists of (d, n).

Plaintext (P)			Ciphertext (C)		After decryption	
Symbolic	Numeric	P^3	P^3 (mod 33)	C^7	C^7 (mod 33)	Symbolic
S	19	6859	28	13492928512	19	S
U	21	9261	21	1801088541	21	U
Z	26	17576	20	1280000000	26	Z
A	01	1	1	1	01	A
N	14	2744	5	78125	14	N
N	14	2744	5	78125	14	N
E	05	125	26	8031810176	05	E
		Sender's computation		Receiver's computation		

Using the RSA public key crypto-system with a=1, b=2 .... Y=25, z=26.

(a) If p=5 and q=13. List five legal values for d.

How about 5, 7, 11, 13, and 17.

(b) If p=5, q=31, and d=37, find *e*.

If z=120=(5-1)(31-1)

Then,
37. *e=1mod(120)*
121, 241, 361, 481....

For 481/37 = 13=*e*

Implementation of Cryptographic protocols: Cryptographic protocols: exchange between participants. Trust: Ethics, Reputation, Law, Physical, Threat, Mutually Assured Destruction (MAD). Incentive, Real motivation/intention behind creation of the cryptographic protocol. **Trust in Cryptographic Protocols**: --minimize the amount of trust required. Paranoia model as a tool. **Messages and Steps** Modularization: Functionality can be split into several protocol layer. The transport layer: For cryptographers, the transport layer is the underlying communication system that permits parties to communicate **Protocol and Message Identity** Provides protocol and message identifiers: Which protocol belongs to and which message within that protocol is. Message encoding and parsing解析: Data elements of the message transformed into a sequence of bytes, Encoding is use for variable size data. Protocol Execution States: states contains all info to complete the protocol. Errors: Checks to verify the: protocol type, message type, protocol execution state, Avoid error retransmission, local handling.