- **CIA**, a modern definition. Confidentiality: prevent unauthorized reading of information. Integrity: detect unauthorized writing of information. Availability: data is available in a timely manner when needed.

- **Network Security**. Various protocols play a critical role, and cryptography matters a lot in protocol (especially network protocols) design and analysis.

- **Kerckhoof's Principle**. The system is completely known to the attacker; only the key is secret; the crypto algorithms are not secret.

- **Confusion and Diffusion**. Confusion: obscuring the relationship between plaintext and ciphertext. Diffusion: spreading the plaintext statistics through the ciphertext. A little note: hash function can be viewed as *one way cryptography*.

- **Stream Cipher**. Both A5/1 and RC4 are examples of this symmetric cryptosystem. It generalized the idea of a one-time pad, except that we trade provably security with a relatively small (and manageable) key. The key is stretched into a long stream of bits, which is then used just like a one-time pad.

- **Block Cipher**. It's really just an "electronic" version of a codebook, and employs both confusion and diffusion.

Algorithm 1 RC4 Keystream Byte

```
$i = (i + 1) \mod 256$
$j = (j + S[i] \mod 256)$
swap $(S[i], S[j])$
$t = (S[i] + S[j] \mod 256)$
$Keystream\ byte = S[t]$
```

- **Feistel Cipher**. It's a general cipher design principle. $L_i = R_{i-1}$ and $R_i = L_{i-1} \oplus F(R_{i-1}, K_i)$.

- **DES**. The security of this cryptosystem has much to do with *S-box*. Steps: an initial permutation before round 1; halves are swapped after last round; a final permutation applied to R_{16}, L_{16}.

Algorithm 2 TEA Encryption

```
$(K[0], K[1], K[2], K[3]) = 128\ bit\ key$
$(L, R) = plaintext\ (64 - bit$ block$)$
$delta = 0x9e3779b9$
$sum = 0$
for $i = 1$ to 32 do
    $sum = sum + delta$
    $L = L + (((R \ll 4) \oplus K[0]) \oplus (R + sum) \oplus ((R \gg 5) \oplus K[1]))$
    $R = L + (((L \ll 4) \oplus K[2]) \oplus (L + sum) \oplus ((L \gg 5) \oplus K[3]))$
    next $i$
end for
$ciphertext = (L, R)$
```

- **Block Cipher Modes**. ECB: encrypt each block independently. CBC: chain the blocks together. For this mode, a random initialization vector is required. CTR: block cipher acts like stream one.

- **Data Integrity**. The encryption process does provide confidentiality, but no guarantee of integrity.

Algorithm 3 Key generation for RSA public key encryption

Ensure: Each entity creates an RSA public key and a corresponding private key. Each entity A should do the following:

1. Generate two large random and distinct primes p and q, each roughly the same size.
2. Compute $n = pq$ and $\phi = (p - 1)(q - 1)$.
3. Select a random integer e, $1 \leq e \leq \phi$, such that $\gcd(e, \phi) = 1$.
4. Use the extended Euclidean algorithm to compute the unique integer d, such that $ed \equiv 1 \mod \phi$.
5. A's public key is (n, e), private key is d.

- **RSA Validity Proof**.
  - Since $ed \equiv 1 \mod \phi$, there exists an integer k such that $ed = 1 + k\phi$.
  - Now if $\gcd(m, p) = 1$, then by Fermat's theorem, $m^{p-1} \equiv 1 \mod p$.
  - Raising both sides of this congruence to the power $k(q - 1)$ and then multiplying both sides by m yields $m^{1+k(p-1)(q-1)} \equiv m \mod p$.
  - On the other hand if $\gcd(m, p) = p$, then this last congruence is valid since each side is congruence to 0 mod p.
  - Hence, in all cases, $m^{ed} \equiv m \mod p$. By the same argument, $m^{ed} \equiv m \mod q$.
  - Finally, since p and q are distinct primes, it follows that $m^{ed} \equiv m \mod n$. And hence, $c^d \equiv (m^e)^d \equiv m \mod n$.

- **Cube Root attack** on RSA. A simple but practical way to prevent is to pad message with random bits.

- **Cryptographic Hash Function**. This function must provide the following:
  - Compression. For any size input x, the output length, i.e. $h(x)$ is small. Usually a fixed length is predefined.
  - Efficiency. It must be easy to compute $h(x)$ for any input x.
  - One way. Given any value y, it's computationally infeasible to find a value x such that $h(y) = x$.
  - Weak Collision Resistance. Given x and $h(x)$, it's infeasible to find any y, with $y \neq x$, such that $h(y) = h(x)$.
  - Strong Collision resistance. It's (and should be so) infeasible to find any $x \neq y$ such that $h(x) = h(y)$.

- **Birthday Problem**. Strong one. How large much the N be before the probability that someone shares the same birthday with me? Weak one. How many people must be in a room before the probability of at least two share the same birthday is larger than 0.5?

- **Access Control**. Two easy-to-understand comparisons. Authentication: are you who you say you are? Authorization: are you allowed to do that fucking (forgive my rudeness; I am tired.) stuff?

- **Common Attacks on Passwords**. Usually, this applies to many other similar stuff in security as well. Outsider, and then you "act as if" you are a normal user. Some time when "the day" comes, you may have the privilege to "self-promoting" to (one of) the administrators.

- **Iris Scan Attacks**. One thing pointed out in the slides, "scanners could use light to" make sure that it's scanning a live eye.

- **Web Cookies**. According to our official textbook, web cookies have "some interesting security implications". Web cookie is simply a numerical value that is stored and managed by one's browser. The website that one is visiting also stores the cookie, which is used to index a database that retains information about Alice (in textbook flavor). In a slightly stronger "expression", a password is used to initially authenticate Alice, after which the cookie is considered sufficient.

- **Evaluation Assurance Level**. Note that a product with a higher EAL does not necessarily mean it *does* possess a higher security power (forgive me for the lack of words). For example, suppose that product A is certificated EAL4, while product B carries EAL5 rating. All it means is that product A was evaluated for EAL4 (and passed), while product B was actually evaluated for EAL5 (and, at the same time, passed). It is possible that product A could have achieved EAL5 or higher, but the developers simply felt it was not worth the cost and effort to try a higher EAL.

- **Classification and Clearances**. Classification applies to *objects*. Clearance applies to *subjects*.

- **Compartments**. They serve to enforce the *need to know* principle, that is, subjects are only allowed to know the information that they *must* know for their work.

- **Covert Channel**. Three things are required for a covert channel to exist. First, the sender and receiver must have access to a shared resource. Second, the sender must be able to vary some property of the shared resource that the receiver can observe. Finally, the sender and receiver must be able to synchronize their communication.

- **CATPCHA** Completely Automated Public Turing test to tell Computers and Humans Apart. It is a program that can generate and grade tests that it itself cannot pass. (Indeed much like some professors, or a lot of professors, in China.)

- **Firewall**. In computing, a firewall is a software or hardware-based network security system that controls the incoming and outgoing network traffic by analyzing the data packets and determining whether they should be allowed through or not, based on applied rule sets. (Firewall can be defined in many ways according to your level of understanding.) A firewall establishes a barrier between a trusted, secure internal network and another network (well, take our beloved Internet for example) that is *not assumed to be* secure and trusted.

- **Packet Filter**. It works by inspecting the "packets" which transfer between computers on the Internet. If a packet matches the packet filter's set of rules, the packet filter will drop the packet, or reject it. This type of packet filtering pays no attention to *whether a packet is part of an existing stream or traffic.*

- **Stateful Filter**. Its work is achieved by retaining packets until enough information is available to make a judgment about its state. Obviously, from the description above, we can infer that this sort of firewall works on the *transport layer.*

- **Proxy Server**. A proxy server is a gateway from one network to another for a specific network application, in the sense that it functions as a proxy on behalf of (most probably, inner) network users.

- **Authenticate over Network**. For attackers, they can: passively observe messages; they can also *replay* messages; they can also actively attack you, like insert, delete or change your messages.

- **Keys or Passwords**. Yet another time according to our beloved textbook. After all, passwords are a little more than a crutch used by *humble humans* because most of the case we are incapable of remembering keys. That is, passwords are about the closest thing to a key that, *a humble human being* can remember. So if Alice and Bob are "actually machines" (that's the case) in real world communications, they should use keys instead of password, as *the* authentication tool.

- **Authentication Using Symmetric Keys**. Final version as suggested by textbook. Encrypt the user's identity together with the nonce. In this manner, Trudy cannot use a response from Bob for the third message—or it "seems" that it's encrypted by Bob himself.

- **Something more about Public Key**. As the concept in (modern) cryptography goes, "the public key is public". Or, should it be public? Should it be so public? Could it possibly be, say, private?

- **Authentication Using Session Keys**. Instead of signing *or* encrypting the messages, we sign *and* encrypt the message.

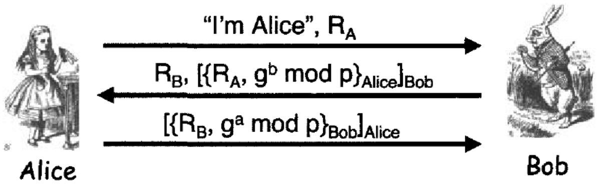
"I'm Alice", R_A
R_B, $[\{R_A, g^b \bmod p\}_{Alice}]_{Bob}$
$[\{R_B, g^a \bmod p\}_{Bob}]_{Alice}$
Alice
Bob

Figure 9.22: Mutual Authentication, Session Key and PFS

- **Public Key Authentication Drawback**. There may be one approach that is dangerous. If you encrypt your message with the other one's public key, and your time stamp is also included in this encryption. Later on you sign this specific message with your private key (aka, as the "digital signature"). It may suffer attack!

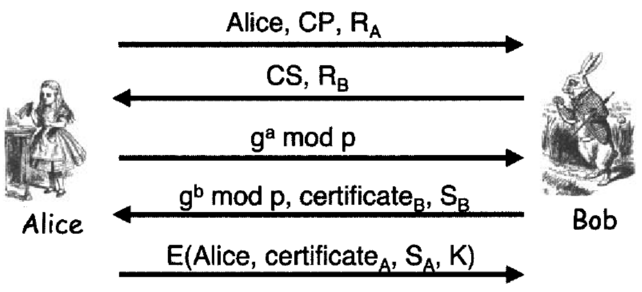
Alice, CP, R_A
CS, R_B
g^a mod p
g^b mod p, certificate$_B$, S_B
E(Alice, certificate$_A$, S_A, K)
Alice
Bob

Figure 10.1: Simplified SSH

- **God Bless SSH**. Such a login typically requires a password and tools like *rlogin* simply sends the password in the clear (aka, plaintext, or rather, message that can be eavesdropped easily), which might be observed by a snooping Trudy. By first establishing an SSH session, any inherently insecure command such as *rlogin* will be secure. That is, an SSH session provides confidentiality and integrity protection.

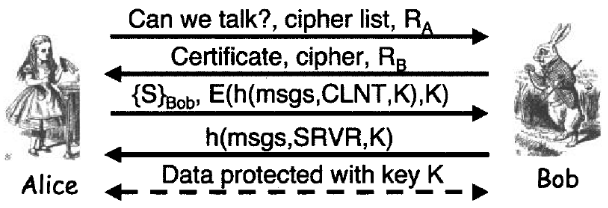
Can we talk?, cipher list, R_A
Certificate, cipher, R_B
$\{S\}_{Bob}$, E(h(msgs,CLNT,K),K)
h(msgs,SRVR,K)
Data protected with key K
Alice
Bob

Figure 10.4: Simplified SSL

Algorithm 4 Simple SSL in A Nutshell

```
1: Alice passes a list of ciphers that she supports, along with a nonce $R_A$.
2: Bob selects one of the ciphers from the cipher list that Alice sent in the message one, and he sends a nonce $R_B$.
3: Alice sends a pre-master secret $S$, along with a hash that is encrypted with the key $K$.
4: Bob responds with a similar hash.
5: At this point, Alice has authenticated Bob. And Alice and Bob has established a shared session key $K$.
```

- **SSL versus IPSec**.
  - SSL is very simple whereas IPSec is, well, indeed it is extremely complex.
  - SSL lives at socket layer, thus residing is user space. IPSec lives at network layer and is therefore not directly accessible from user space—it's in the domain of *the operating system.*
  - Both SSL and IPSec provide encryption, integrity protection, and authentication.

- **Internet Key Exchange**. Mutual authentication is supported. Session key should (and would) be established. Phase 1, IKE security association. Phase 2, AH/ESP security association. Something more: the main mode *must* be implemented whereas the aggressive mode *should* be implemented.

- **Transport and Tunnel Modes**. Independent of whether ESP or AH is used, IPSec employs either *transport mode* or *tunnel mode.*
  - In transport mode, the new ESP/AH header is sandwiched between the IP header and the data.
  - Transport mode is more efficient since it adds a minimal amount of additional header information.
  - The downside of transport mode is that a passive attacker can see the headers. This mode is mainly designed for host-to-host communication.

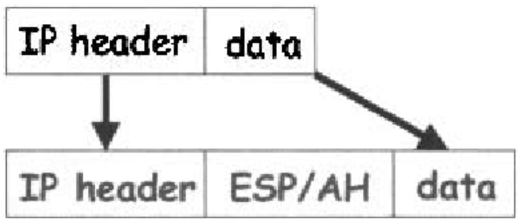

Figure 10.15: IPSec Transport Mode

  - In tunnel mode, the entire IP packet is encapsulated in a new IP packet.

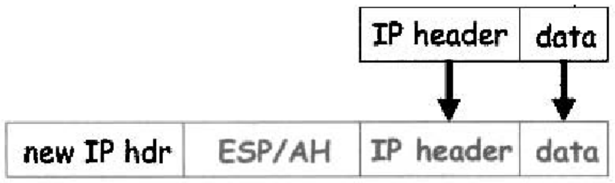

Figure 10.17: IPSec Tunnel Mode

- **AH versus ESP**. Authentication Header: provides only integrity. Encapsulating Security Payload: integrity and confidentiality are provided at the same time. For ESP, everything is protected *except* IP header.

- **Buffer Overflow**. A buffer overflow must exist in the code. But not all buffer overflows are exploitable. However, if exploitable, the attacker can *inject code.*

- **Race Conditions**. To prevent race conditions, one option is to make security-critical processes *atomic.*