Ch7. Security

Created	@Jun 18, 2020 6:21 PM
Tags	컴망

네트워크 보안

- **Confidentiality(기밀성)** : 송신자와 의도된 수신자만이 메시지를 이해할 수 있어야 한다.

- **Authentication(진정성)** : 서로 누군지 알 수 있어야 한다.
- **Integrity(무결성)** : 중간에 메시지가 변경되지 않아야 한다.
- **Availability(가용성)** : 서비스는 사용자가 사용 가능해야 한다.

Symmetric key cryptography : 대칭 키 암호화

Alice도, Bob도 **똑같은 Key를 사용**한다. → KS라 하자.

m → Alice가 암호화해서 KS(m) → 전송 → Bob이 복호화해서 KA(KA(m)) = m → m

단순한 Encryption scheme

Substitution cipher : 글자 치환

→ **Encryption key에는 글자 간의 mapping**이 들어 있다.

좀 더 복잡한 Encryption

Substitution cipher가 하나가 아니라 **N개**가 있고, **Cycling pattern**이 있다.

Cycling pattern에 따라, 1번째 글자에서는 M1번째 Substitution cipher를 쓰고, 2번째 글자에서는 M2번째...

→ **Encryption key에는 Substitution cipher N개랑, Cycling pattern**이 있다.

DES : Data Encryption Standard

대칭 키 암호화 쓰는 대표적인 표준.

56비트 대칭키, 64비트 입력.

AES : Advanced Encryption Standard

DES 강화판. 128비트 입력.

Public key cryptography : 공개 키 암호화 (비대칭 키 암호화)

Public key, Private key를 둘 다 만든다.

- Public key는 Encryption 알고리즘까지 다 알려져 있다.
- Private key를 가지고 있는 사람만 Decryption할 수 있다.

💡 Notation:
Public key K+
Private key K-

조건 : **Public key가 주어질 때, Private key를 구하는 게 불가능해야 한다.**

RSA 알고리즘

중요한 특징 :

- **Private key로 암호화**하고 **Public key로 복호화**해도 원문 나오고,
- **Public key로 암호화**하고 **Private key로 복호화**해도 원문 나온다.
- 어느 순서로 해도 원문이 나온다.

Session key

RSA는 exponential 함수를 쓰기 때문에 상대적으로 느리다. 긴 메시지를 대상으로 RSA를 쓰기에는 너무 느리다.

→ 긴 거에는 **대칭키 암호화를 먼저** 쓰고, **그 키를 전달해줘야 하는데, 이 과정에 RSA 암호화** 먹이는 식으로 쓸 수도 있다.

Authentication

"내가 누구다."

Protocol ap1.0

"I'm Alice" 메시지만 보내면?

→ 누구든 위조할 수 있다.

2.0

메시지와 함께 자신의 IP주소를 같이 넘겨주면?

→ IP주소도 위조할 수 있다.

3.0

Alice, Bob 둘만 알고 있는 비밀번호를 같이 넘겨주면?

→ Trudy가 도청해서 나중에 그대로 써먹으면 그만이다.

= **Playback attack!**

3.1

Encrypted된 비밀번호를 전송해도?

→ Playback attack은 여전히 가능. 도청해서 그냥 써먹으면 그만.

4.0

Issue : Playback attack를 해결해야 한다.

→ **Nonce : 단 한번만 유효한 번호**

1. A → B : "I'm Alice"
2. A ← B : nonce R (단 한번만 유효한 번호)
3. A → B : 공유된 대칭키로 암호화한 K(R) 송신
4. B가 이를 복호화해보고, A인지 아닌지 판단 가능.

5.0

4.0처럼 대칭키가 아니라, 공개키를 사용.

3. A → B : KA-로 먼저 R을 encryption 후 보내줌

4. A ← B : A에게 KA+ 요청

5. A → B : KA+ 보내줌

그래도 허점이 있다. Man in the middle attack이라는 게 있다.

Digital signatures

서명 : 그 사람이 맞는지 확인하기 위해.

공개키 암호화를 사용한다.

B의 메시지 m을 **B의 Private key로 암호화**한다. → KB-(m)

수신 측에서는 **원본 m**과 **복호화한 m을 비교**해서 동일하면 B가 맞다.

Message digest

메시지 자체를 암호화하는 건 너무 일이 크다.

큰 메시지 m에 대해, 이 자체를 encryption하는 게 아니고, **메시지에 Hash function을 돌려서 짧은 Digest H(m)를 뽑아내서 얘를 암호화하는 것.**

수신측에서는 비슷하게 **원본 H(m)**과 **복호화한 H(m)**을 비교한다.

Hash function은 뭐 쓰나?

- **MD5** : 128bit 출력
- **SHA-1** : 160bit 출력

Certification Authorities (CA)

공개된 Public key가 정말 그 사람의 key인지 어떻게 알지?

→ Public key에 대한 인증서를 발급해주는 기관.

인증서에는 이 Public key가 이 사람 게 맞다는 정보가 들어 있다. Bob의 Public key를 CA의 Private key를 사용해 암호화한다.

복호화할 때 인증 기관의 Public key도 필요한데, 이는 컴퓨터에 들어 있다.

인증 기관은 믿는 것으로 한다.

Secure E-mail

원칙은 똑같다. **메시지는 대칭 키로 암호화, 키 자체는 공개 키로 암호화.**

그런데, 이렇게만 하면 Authentication 기능이 없다.

- Auth도 똑같이 Hash 먹인 Message digest를 암호화한 것과 원본 digest 둘 다 보내서, 수신 측에서 확인하게 한다.

SSL

1. **Handshake** : 그들의 certificate, private key를 이용해 서로 authenticate
2. **Key derivation** : Encryption을 위한 key, Auth를 위한 key 두 개를 만든다.
3. **Data transfer**
4. **Connection closure**

IPsec

- **AH 프로토콜** : 무결성, 소스 Auth
- **ESP 프로토콜** : 암호화까지 지원

방화벽

망을 보호하자. 외부로부터의 침입을 막고, 외부로 나가는 것도 막자.

- **DoS 공격 방지**
  - **SYN flooding** : TCP 연결을 계속 만드는 것. (Remind: SYN이 그 TCP에서 연결 맺자는 거였다)
    → 서버의 리소스 버퍼가 꽉 차서 뻗어버리게 된다.
- **데이터의 Illegal한 수정 방지**
- **허가된 액세스만 네트워크에 들어올 수 있도록**

세 가지 타입이 있다.

- **Stateless packet filters** : 패킷 단위로 모두 뜯어보는 것
- **Stateful packet filters** : 상태를 보겠다
- **Application gateways** : 항상 거쳤다 간다

Stateless packet filtering

헤더만 보는 게 아니라, 그 안에 있는 메시지까지 모든 패킷을 다 본다.

Policy에 따라 Forward할지, Drop할지 결정.

ACL : Access Control Lists
Rule(Policy)의 리스트. Source 주소, Dest 주소, ... 등에 대해 Allow할지, Deny할지 정의해둔 것

Stateful packet filtering

모든 TCP 연결에 대해 상태를 추적하자.

SYN, FIN 등 연결이 되어 있는지를 먼저 체크.

Application gateways

Gateway를 거치지 않고 들어오는 패킷은 모두 Drop한다.

IDS : Intrusion Detection Systems

침입을 막자.

- **Signature-based** : 공격 패턴이 알려져있어서, 패킷을 뜯어봐서 공격 유형이면 블락. 알려져 있는 공격만 막을 수 있다.
- **Anomaly-based**