Information and Network Security

컴퓨터개론

(Introduction to Computer Systems)

GEN1030

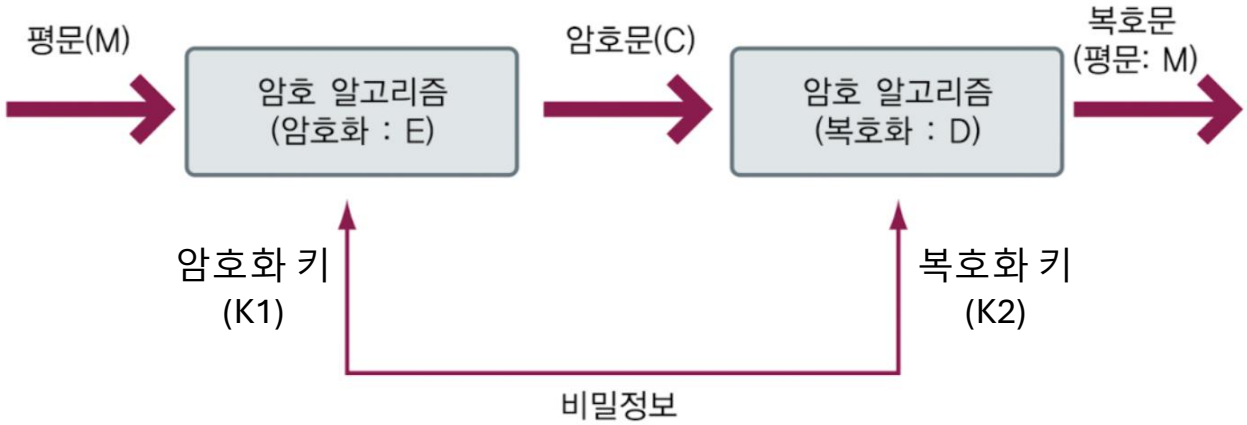
암호화

암호화

- 정보보호를 위한 기본적인 방법
- 암호(Cryptography)
  - 평문을 해독 불가능한 형태로 변형하거나 암호화된 데이터를 해독 가능한 형태로 복원하기 위한 원리, 수단, 방법 등의 기술
- 암호화 및 복호화
  - 평문(Plaintext): 암호화 되지 않은 원문 메시지
  - 암호문(Ciphertext): 평문이 암호화 된 결과
  - 암호화(Encryption): 평문이 암호문으로 변환하는 과정
  - 복호화(Decryption): 암호문을 평문으로 변환하는 과정

암호화

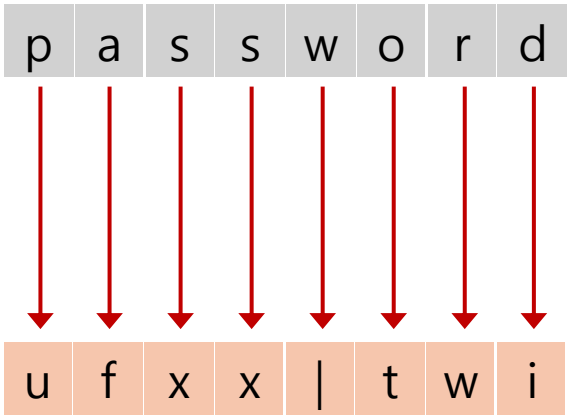
- 암호화 및 복호화
  - 암호 알고리즘 및 키(Key)로 구성
  - 같은 알고리즘이라도 키가 다르면 다른 암호문이 생성됨



- 암호화: C=E(M, K1) 복호화: M=D(C, K2)

암호화

- Example
  - 원문의 각 문자에 숫자 5을 더함
  - 암호 알고리즘: 각 문자의 ASCII Decimal 값에 Key 값을 더함
  - Key: 5



Decimal	Char		Decimal	Char
97	a		112	p
98	b		113	q
99	c		114	r
100	d		115	s
101	e		116	t
102	f		117	u
103	g		118	v
104	h		119	w
105	i		120	x
106	j		121	y
107	k		122	z
108	l		123	{
109	m		124	|
110	n		125	}
111	o		126	~

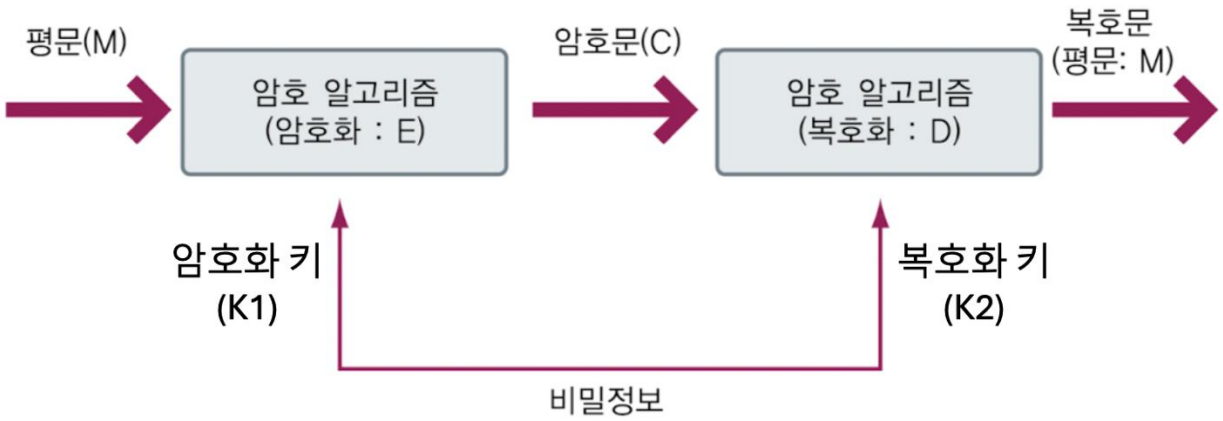
암호화 기법

- 여러 기준으로 분류 가능

- 기본 원리에 따라
  - 평문의 각 원소를 다른 원소에 대응시키는 대체(substitution)
    - A → G, B → H
  - 평문의 원소들을 재배열하는 치환(transposition)
    - abcd → dcab
- 처리 단위에 따라
  - 스트림 암호(Stream cipher): 데이터를 연속적으로 처리(ex. bit 혹은 byte 단위)
  - 블록 암호(Block cipher): 데이터를 일정한 크기의 블록 단위로 처리 (ex. 64-bit block, 128-bit block)
- 키 사용 방식에 따라
  - 대칭키/비밀키 암호화
  - 비대칭키/공개키 암호화

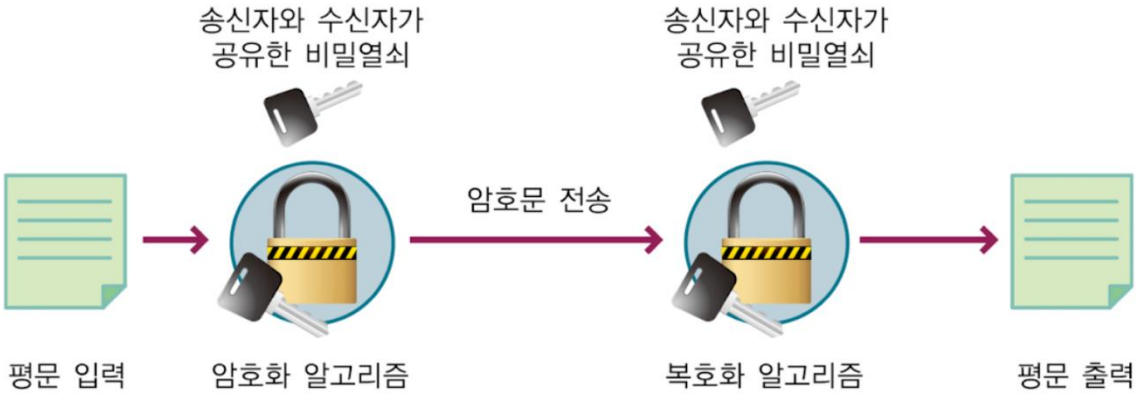
키 기반 암호화 기법

- 대칭키/비밀키(Symmetric key/secret key) 암호화 방식
  - K1 = K2
- 비대칭키/공개키(Asymmetric key/public key) 암호화 방식
  - K1 =/= K2



비밀키 암호화

- 비밀키(Secret key) 혹은 대칭키(Symmetric key) 암호화
  - 보내는 사람과 받는 사람이 같은 키를 가지고 있음
  - 송신자 및 수신자 모두 동일한 키로 암호화 및 복호화 수행



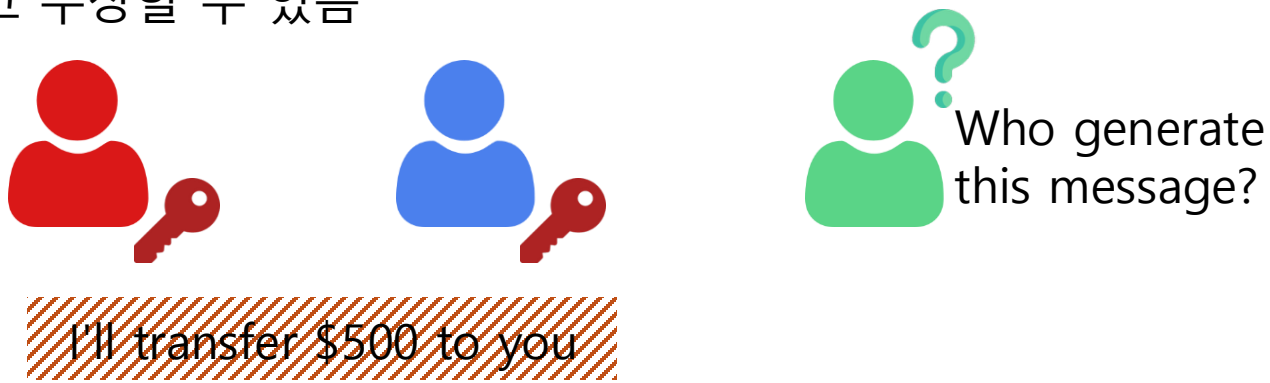
- 시저 암호(Caesar cipher): 비밀키 원리 보여주는 간단한 예
  - 각 문자를 일정한 수만큼 뒤에 있는 문자로 변화
    - Ex) 각 문자를 5문자 뒤의 문자로 → key = 5

비밀키 암호화 – 단점

- 통신 상대가 많아질수록 관리해야 할 비밀키가 많아짐
  - 여러 상대가 동일한 키를 사용하면 서로의 메시지를 읽을 수 있음
  - 한 사용자에게 N명의 통신 상대가 있는 경우, 보통 N개의 비밀키가 필요

비밀키 암호화 – 단점

- 통신 상대가 많아질수록 관리해야 할 비밀키가 많아짐
  - 여러 상대가 동일한 키를 사용하면 서로의 메시지를 읽을 수 있음
  - 한 사용자에게 N명의 통신 상대가 있는 경우, 보통 N개의 비밀키가 필요
- 부인방지를 제공하기 어려움
  - 송신자와 수신자가 같은 키를 공유하므로, 누가 메시지를 만들었는지 제3자에게 증명하기 어려움
  - A와 B가 같은 키를 공유하면, 서로 다른 사람이 그 메시지를 생성했다고 주장할 수 있음



비밀키 암호화 – 단점

- 통신 상대가 많아질수록 관리해야 할 비밀키가 많아짐
  - 여러 상대가 동일한 키를 사용하면 서로의 메시지를 읽을 수 있음
  - 한 사용자에게 N명의 통신 상대가 있는 경우, 보통 N개의 비밀키가 필요
- 부인방지를 제공하기 어려움
  - 송신자와 수신자가 같은 키를 공유하므로, 누가 메시지를 만들었는지 제3자에게 증명하기 어려움
  - A와 B가 같은 키를 공유하면, 서로 다른 사람이 그 메시지를 생성했다고 주장할 수 있음
- 키 분배 문제
  - A와 B가 비밀키를 처음 공유하려면 어떻게 해야 될까?

공개키 암호화

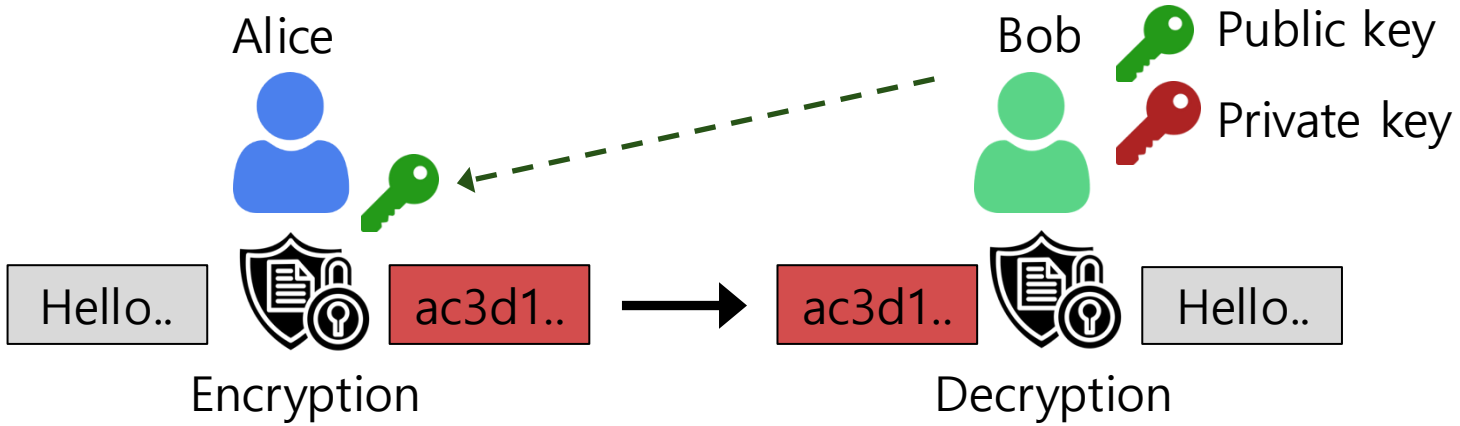
- 암호의 최대 난제?
  - 암호화 과정에서 사용되는 키를 안전하게 분배하는 일

- 1976년 Diffie와 Hellman이 공개키 암호기법 개념 제안

- 공개키(Public key) 혹은 비대칭키(Asymmetric key) 암호화
  - 두 개의 분리된 키를 사용하는 비대칭적 암호화
  - Public key: 누구에게나 공개 가능
  - Private key: 당사자(소유자)만 비밀로 보관
  - 하나의 키로 암호화한 메시지는 pair가 되는 다른 키로만 복호화 가능

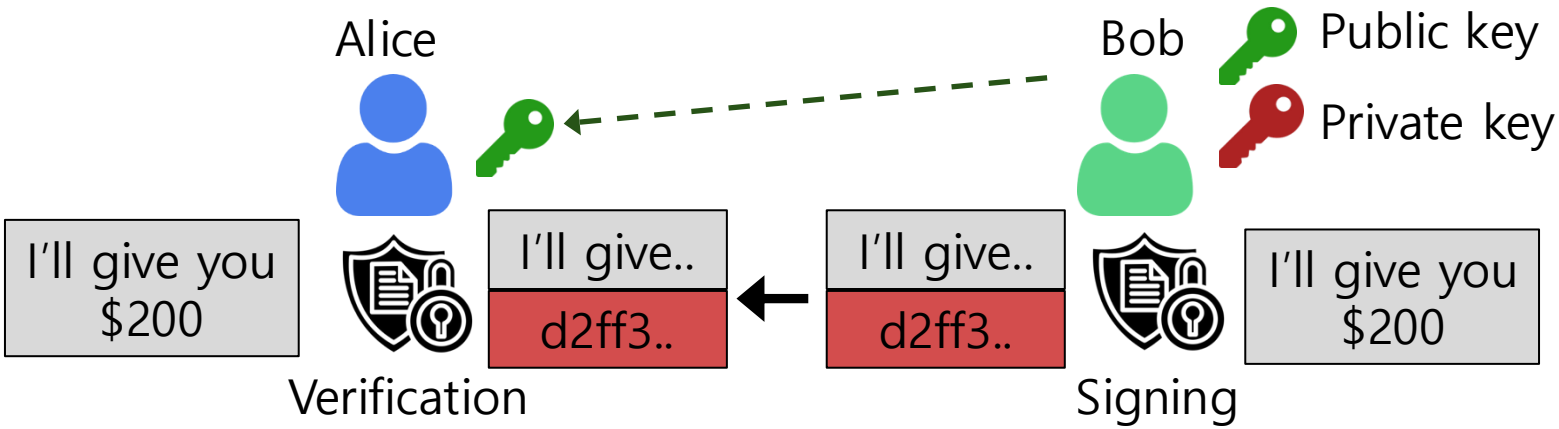
공개키 암호화

- 암호화(Encryption) 및 복호화(Decryption) 과정
  - 메시지 전달을 위해 송신자는 수신자의 public key를 얻음
  - 송신자는 수신자의 public key를 사용하여 메시지를 암호화하고 수신자에게 전송
  - 암호화된 메시지는 수신자의 private key로만 복호화 가능



디지털 서명(Digital Signature)

- 메시지 및 전자문서의 송신자와 무결성을 확인하기 위한 기술
  - 송신자는 자신의 private key로 메시지에 서명
  - 수신자는 송신자의 public key로 서명을 검증



- 제공하는 보안성
  - 인증: 누가 서명한 메시지인지 확인
  - 부인 방지: 송신자가 전송 또는 서명 사실을 부인하기 어렵게 함
  - 무결성: 메시지가 변경되지 않음

공개키 암호화 특징

- 장점
  - 키 분배 문제가 개선
    - 비밀키 암호화처럼 모든 사용자 쌍마다 비밀키를 미리 공유할 필요가 없음
  - 디지털 서명(Digital Signature)을 통해 송신자 인증, 무결성, 부인 방지를 제공할 수 있음

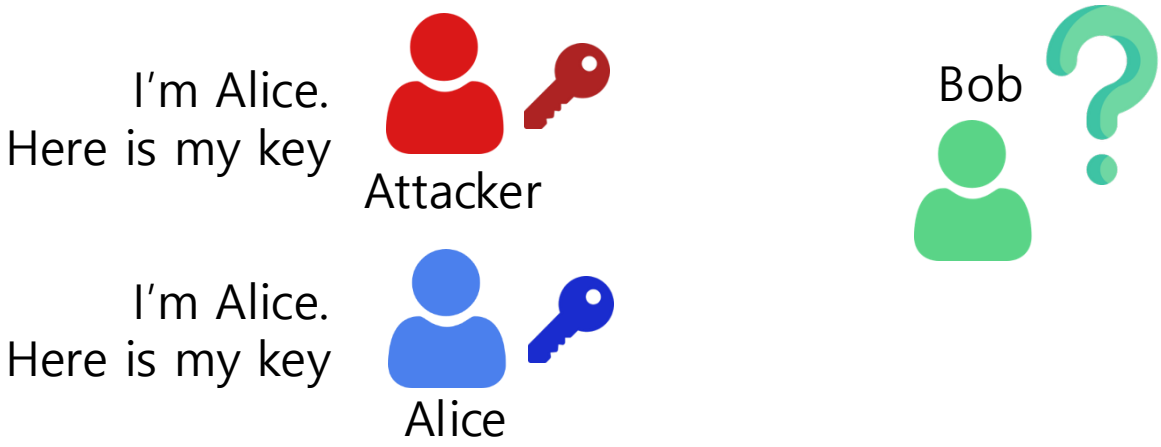
- 단점
  - 암호화 처리속도가 비밀키 암호화에 비해 느림
    - 공개키 암호화가 수학적으로 복잡한 연산을 주로 사용
  - public key가 정말 해당 사용자의 key인지 확인해야 함
    - 공격자가 가짜 public key를 제공하면 위장(impersonation) 공격 가능

    → 인증서(Certificate)와 Public Key Infrastructure (PKI)

Public Key

- Public key는 임의의 bit string
  - Private key도 마찬가지
  - Public key 만으로는 그 key의 실제 소유자를 알 수 없음

- 누구라도 public key/private key pair 생성할 수 있음



Public key와 identity를 어떻게 신뢰할 수 있게 연결할 수 있을까?

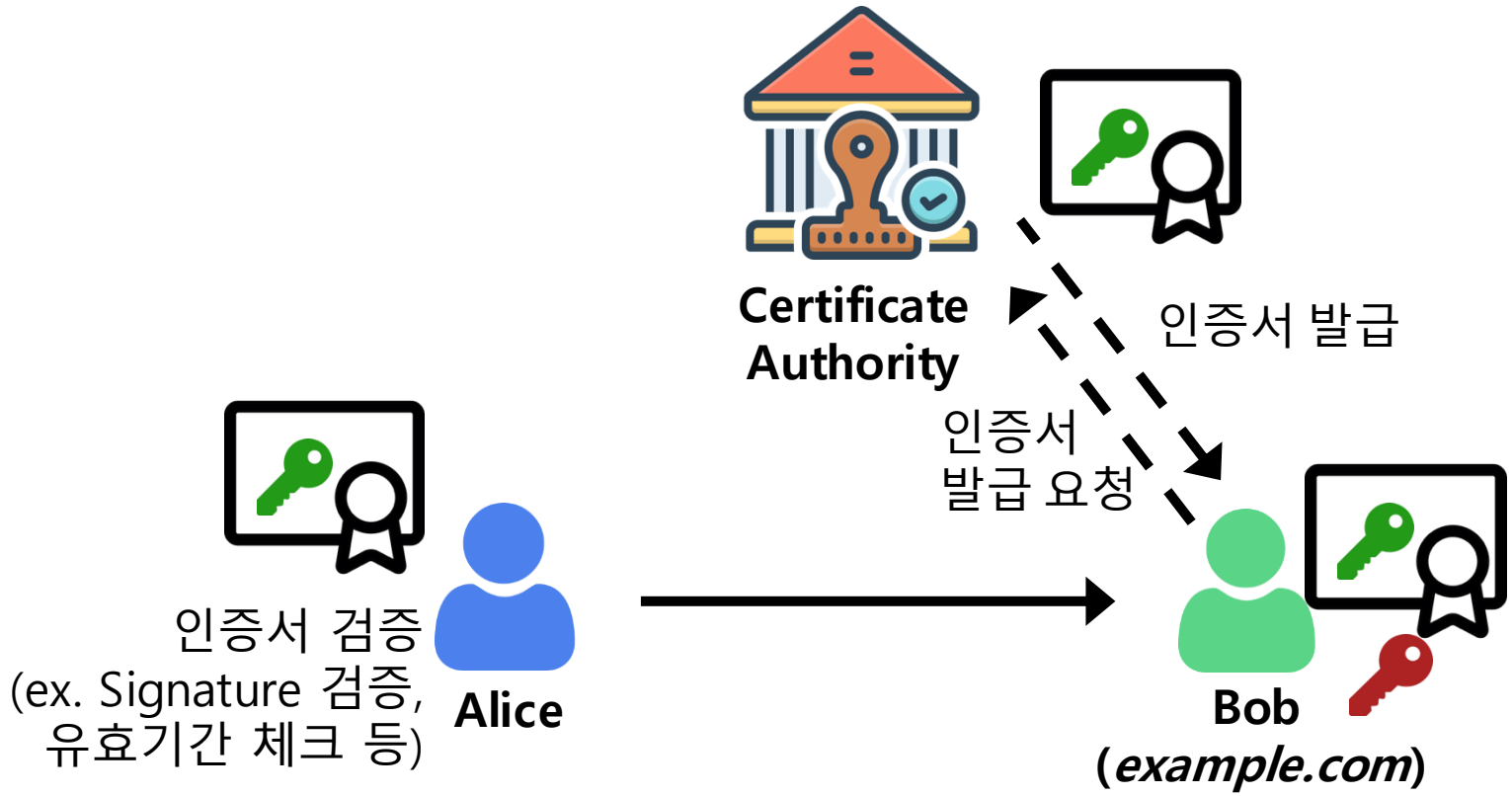
Public Key Infrastructure (PKI)

- 공개키 기반 보안 서비스를 운영하기 위한 신뢰 관리 인프라
  - Identity와 public key의 관계를 검증하고, 관리하는 체계
  - 공개키 기반 암호화, digital signature, TLS/HTTPS 등의 보안 서비스를 가능하게 함

- 주요 구성 요소
  - **인증서(Certificate)**
    - Identity와 public key를 묶은 전자 문서
    - 포함 정보: 소유자 이름(ex. Domain name), public key, 발급자(인증기관), 유효기간, 발급자의 digital signature
    - "이 public key는 이 사용자(identity)의 것이다" 증명
  - **인증기관(Certificate Authority, CA)**
    - 인증서를 발급하고 서명하는 신뢰 기관
    - 인증서에 서명하여 public key와 identity의 binding을 보증

Public Key Infrastructure (PKI)

- 인증서 발급 및 사용 과정



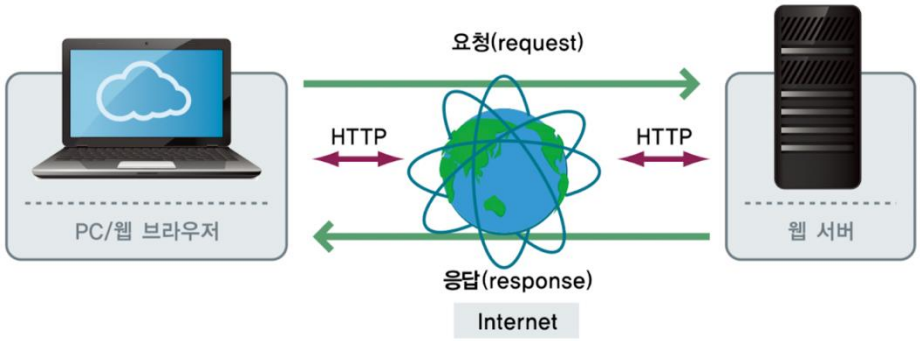
인터넷 보안

개요

- 인터넷 서비스는 네트워크 위에서 동작
- 네트워크는 기본적으로 보안을 자동으로 제공하지 않음
  - 데이터가 평문으로 전송되면 공격자가 중간에서 트래픽을 훔쳐보거나 변조할 수 있음
- 인터넷 기반 서비스마다 보호해야 하는 대상이 다름 → 서비스의 목적에 맞는 보안 기술 사용
  - Web: 브라우저(browser)와 웹 서버 사이의 통신
  - DNS: domain name을 올바른 IP address로 변환
  - Email: 송신자, 수신자, 메시지 내용을 보호

Web Security

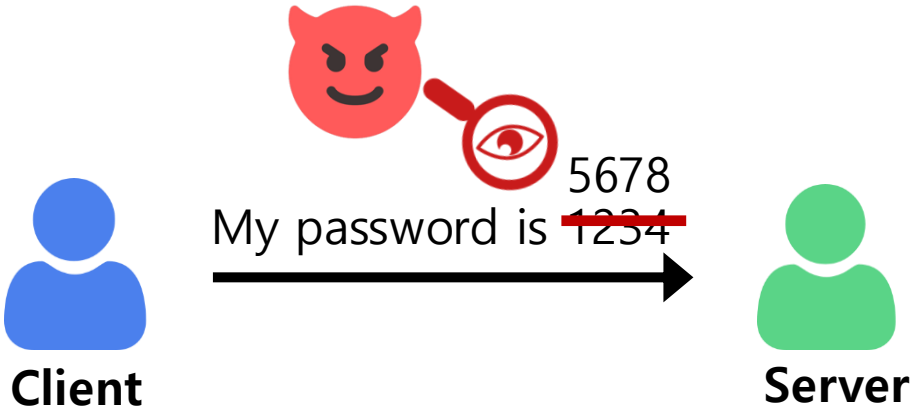
- WWW은 HTTP(HyperText Transfer Protocol) 사용
  - 클라이언트(브라우저)와 서버 사이에서 데이터를 주고받는 프로토콜



- HTTP는 기본적으로 데이터를 평문으로 전송

Web Security

- WWW은 HTTP(HyperText Transfer Protocol) 사용
  - 클라이언트(브라우저)와 서버 사이에서 데이터를 주고받는 프로토콜



- HTTP는 기본적으로 데이터를 평문으로 전송
  - 공격자가 중간에서 트래픽 볼 수 있음
  - 공격자가 중간에서 데이터 변조 가능
  - 사용자는 자신이 진짜 서버와 통신하는지 확인하기 어려움

Transport Layer Security (TLS)

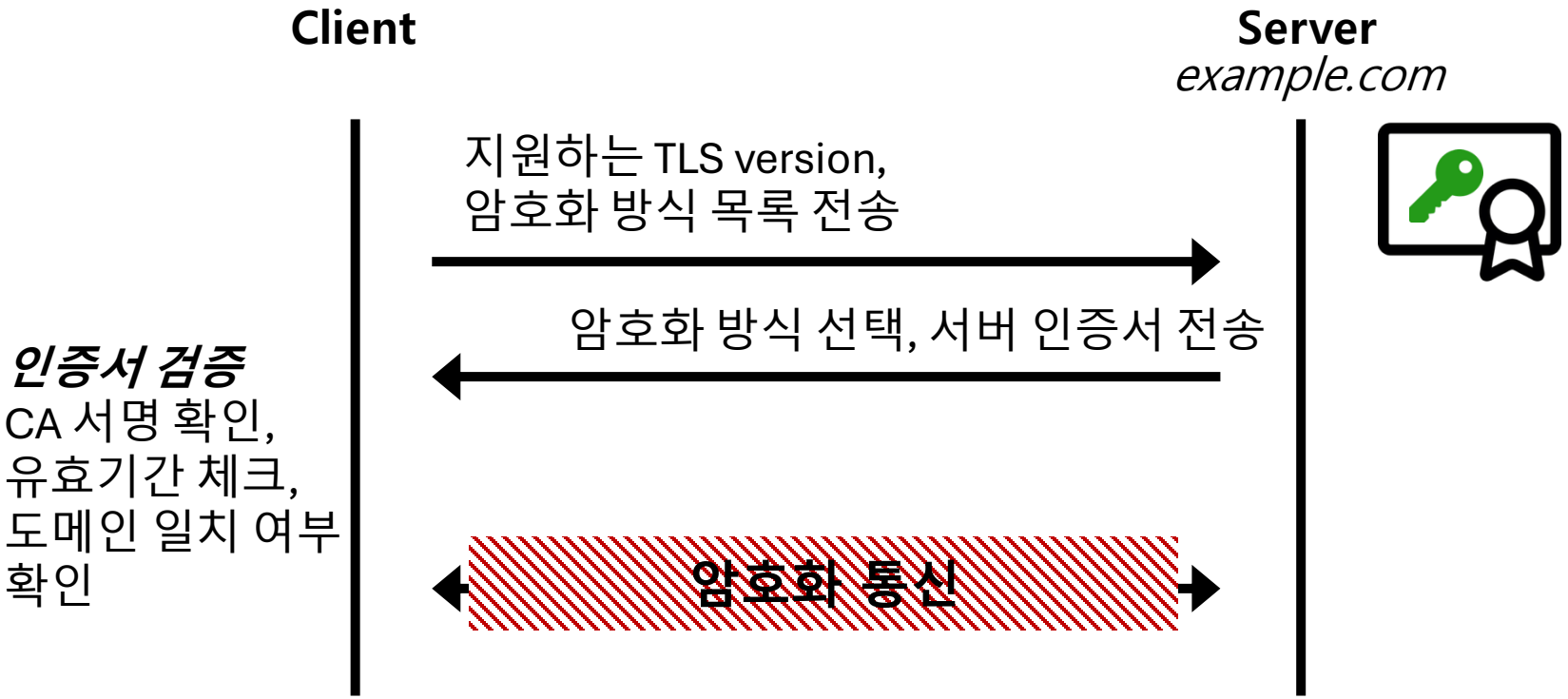
- 클라이언트와 서버 사이에 안전한 통신 경로(채널)을 만드는 보안 프로토콜
  - 채널(Channel): 두 컴퓨터가 데이터를 주고받는 경로
  - HTTP, 이메일 등 다양한 응용 프로토콜에 보안 기능 적용

- 제공하는 보안 기능
  - Confidentiality: 통신 내용 암호화
  - Integrity: 통신 중 데이터 변조 방지
  - Authentication: 서버가 진짜인지 인증서(Certificate)를 통해 확인

- Versions
  - SSL, TLS 1.2, TLS 1.3

Transport Layer Security (TLS)

- TLS Handshake
  - 실제 데이터를 주고받기 전, 클라이언트와 서버가 보안 채널을 설정하는 협상 과정
  - 어떤 암호화 방식을 사용할지 정하고, 서버가 진짜 인지 확인, 암호화에 사용할 키 생성



HTTPS (HTTP Secure)

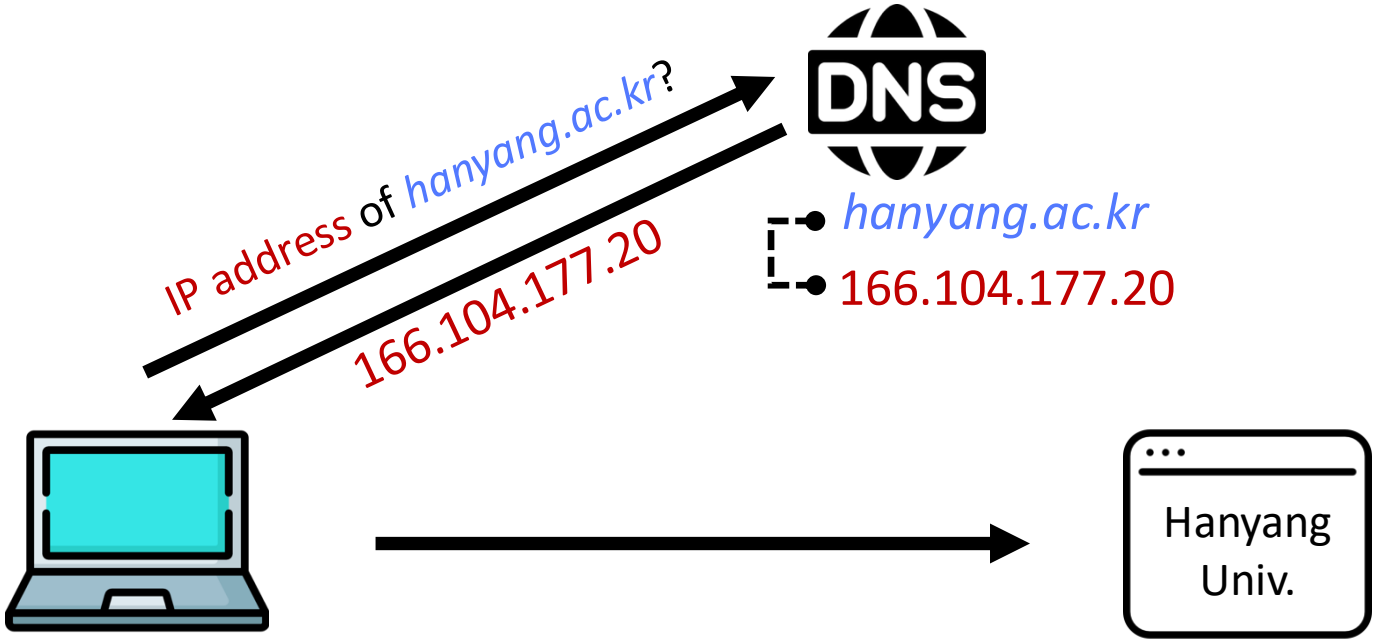
- HTTP + TLS
  - 기존 HTTP 통신에 TLS 보안 레이어를 추가
- 클라이언트(browser)와 web server 사이의 통신 보호
  - HTTP 메시지를 TLS가 보호하는 안전한 채널 위에서 전송
  - 포트 번호: HTTP는 80번, HTTPS는 443번 사용
- 현대 Web에서 HTTPS
  - 로그인, 결제, 개인정보 등 민감한 정보 다루는 사이트에서 필수
  - 주요 사이트는 HTTP 접속시 HTTPS로 리다이렉트(redirect)
  - 주요 브라우저는 HTTPS를 강제하는 기능 내장하기도 함

HTTPS (HTTP Secure)

- 브라우저에서 표시
  - http:// 시작 → 평문 전송
  - https:// 시작 → 암호화 전송

DNS Security

- Domain Name System (DNS)
  - 도메인 네임(Domain name)과 관련된 정보 사이의 매핑을 저장하고 제공하는 계층적이고 분산된 시스템
  - ex) 사람이 읽기 쉬운 domain name을 컴퓨터가 사용하는 IP 주소로 바꿔 주는 시스템



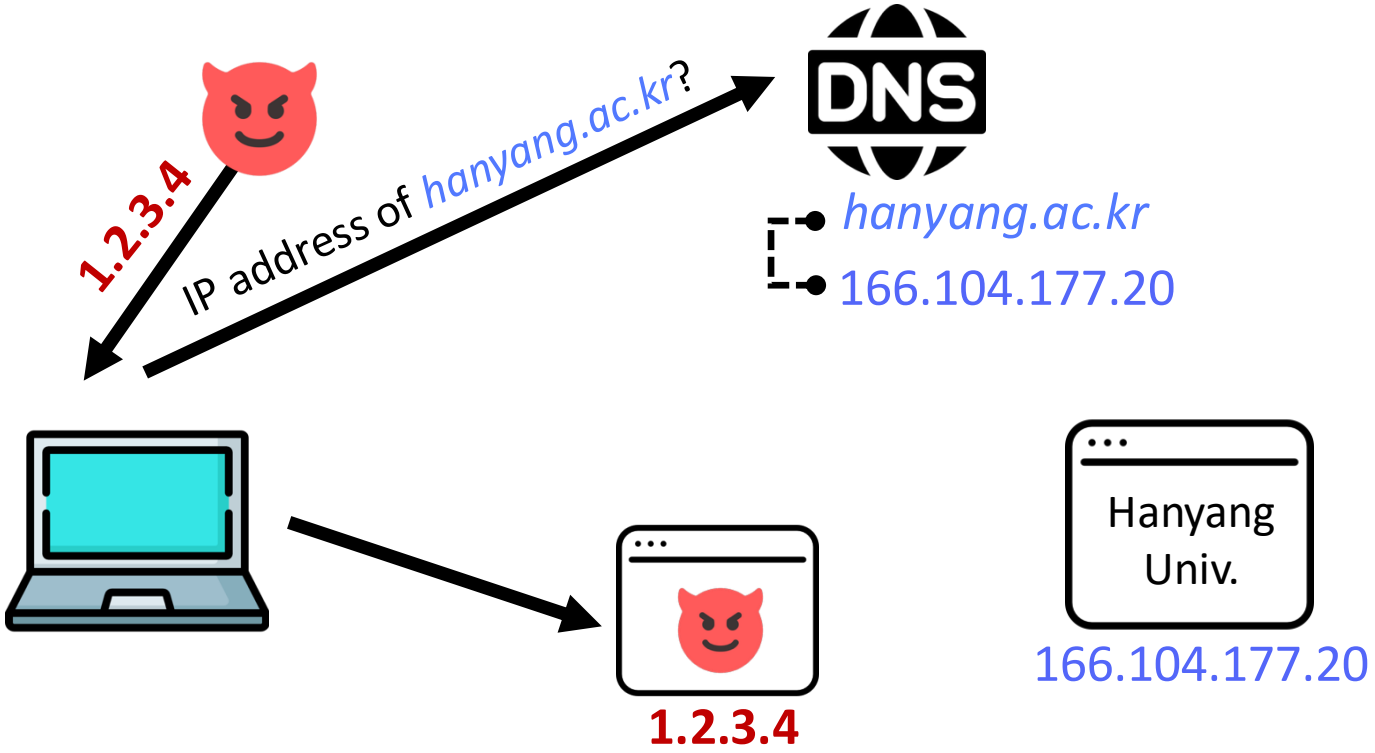
DNS Security

- Domain Name System (DNS)
  - 도메인 네임(Domain name)과 관련된 정보 사이의 매핑을 저장하고 제공하는 계층적이고 분산된 시스템
  - ex) 사람이 읽기 쉬운 domain name을 컴퓨터가 사용하는 IP 주소로 바꿔 주는 시스템

- DNS는 설계 당시 보안을 고려하지 않음
  - 응답 내용이 전송 중 변조되었는지 확인할 수 없음
  - 응답이 진짜 DNS 서버에서 온 것인지 확인 할 수 없음

DNS 공격 유형

- DNS spoofing
  - 공격자가 DNS query에 대해 가짜 응답을 보냄
  - 피해자는 공격자가 만든 가짜 사이트로 연결 될 수 있음
    - 만약 A record (IP address)를 가짜로 보내는 경우



DNS Security

- DNS security mechanisms
  - DNS Security Extensions (DNSSEC)
  - DNS over Encryption

DNS Security Extensions (DNSSEC)

- DNS 보안 취약점을 보완하기 위해 제안된 DNS 확장 표준
- DNS 응답에 디지털 서명(Digital signature)를 추가하여 응답의 무결성과 출처를 검증
  - 무결성(Integrity): 응답이 전송 중 변조되지 않았음
  - 인증(Authentication): 응답이 정당한 DNS 서버(authoritative nameserver)에서 온 것임
- DNSSEC 위한 DNS 레코드 필요
  - **DNSKEY** record: 도메인이 사용하는 public key를 저장
  - **RRSIG** record: 각 DNS 레코드에 대한 디지털 서명을 저장
    - Public key의 pair인 private key를 이용해서 생성
  - **DS** record: 하위 계층 DNS 서버의 public key를 상위 계층이 저장
    - 상위 계층 DNS 서버가 하위 계층 DNS 서버의 DNSKEY를 신뢰한다고 보증 → 신뢰 체인(Chain of trust) 형성

DNS Security Extensions (DNSSEC)

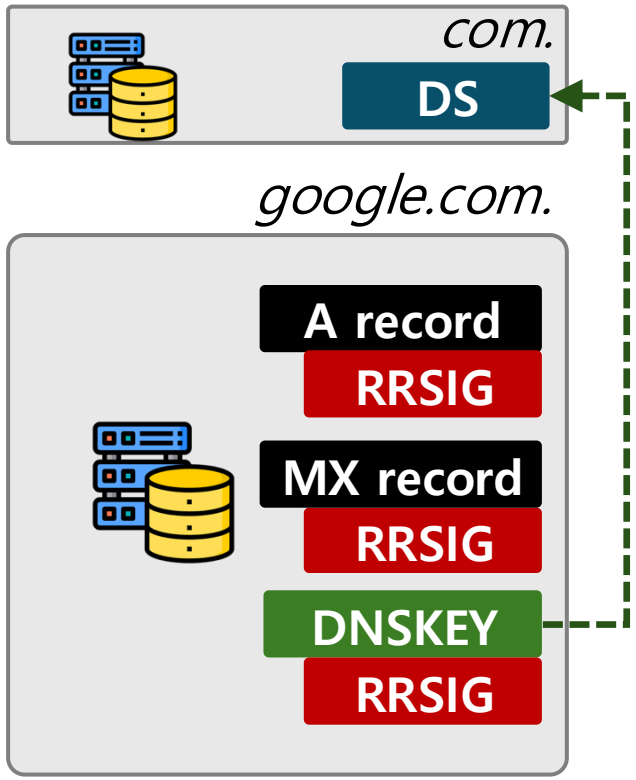
- 동작 방식
  1. DNS 서버(authoritative nameserver)는 자신의 DNS record에 대해 디지털 서명(RRSIG record)을 private key 이용하여 생성
  2. 서버는 응답을 보낼 때 요청받은 DNS record와 서명(RRSIG)을 함께 전송
  3. 수신자는 DNS 서버의 public key (DNSKEY record)로 서명 검증
  4. 서명이 유효하면 응답이 진짜이고 변조되지 않았음을 확인 가능
     - 유효하지 않으면 DNS 응답을 사용하지 않고 오류 처리

DNSSEC 동작

Authoritative Nameserver
- 도메인 관련 정보를 저장하고 있음 **+ DNS records for DNSSEC**
- DNS 요청이 오면 그 응답을 반환함 (ex. IP address)

DNS Resolver
- 클라이언트 대신 Authoritative nameserver에 DNS 요청을 보내서 응답을 찾음

Client
- DNS 요청을 보냄 (ex. 웹 브라우저)

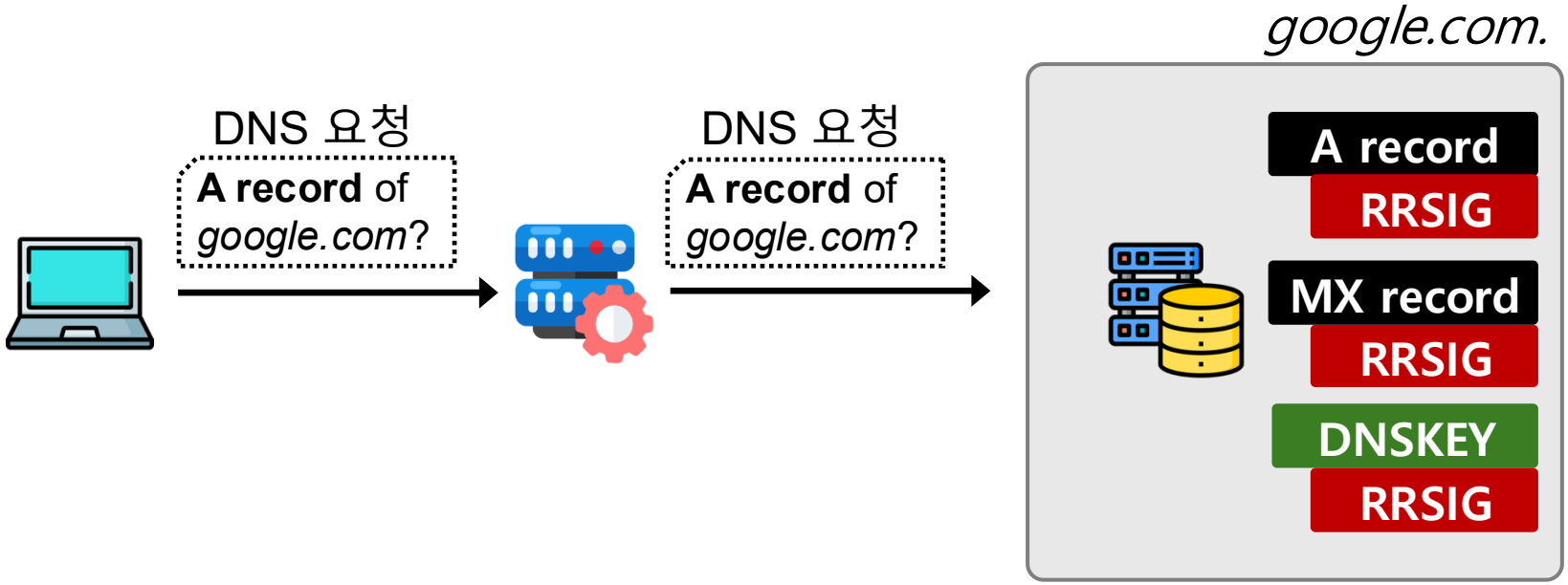


DNSSEC 동작

Authoritative Nameserver
- 도메인 관련 정보를 저장하고 있음 **+ DNS records for DNSSEC**
- DNS 요청이 오면 그 응답을 반환함 (ex. IP address)

DNS Resolver
- 클라이언트 대신 Authoritative nameserver에 DNS 요청을 보내서 응답을 찾음

Client
- DNS 요청을 보냄 (ex. 웹 브라우저)

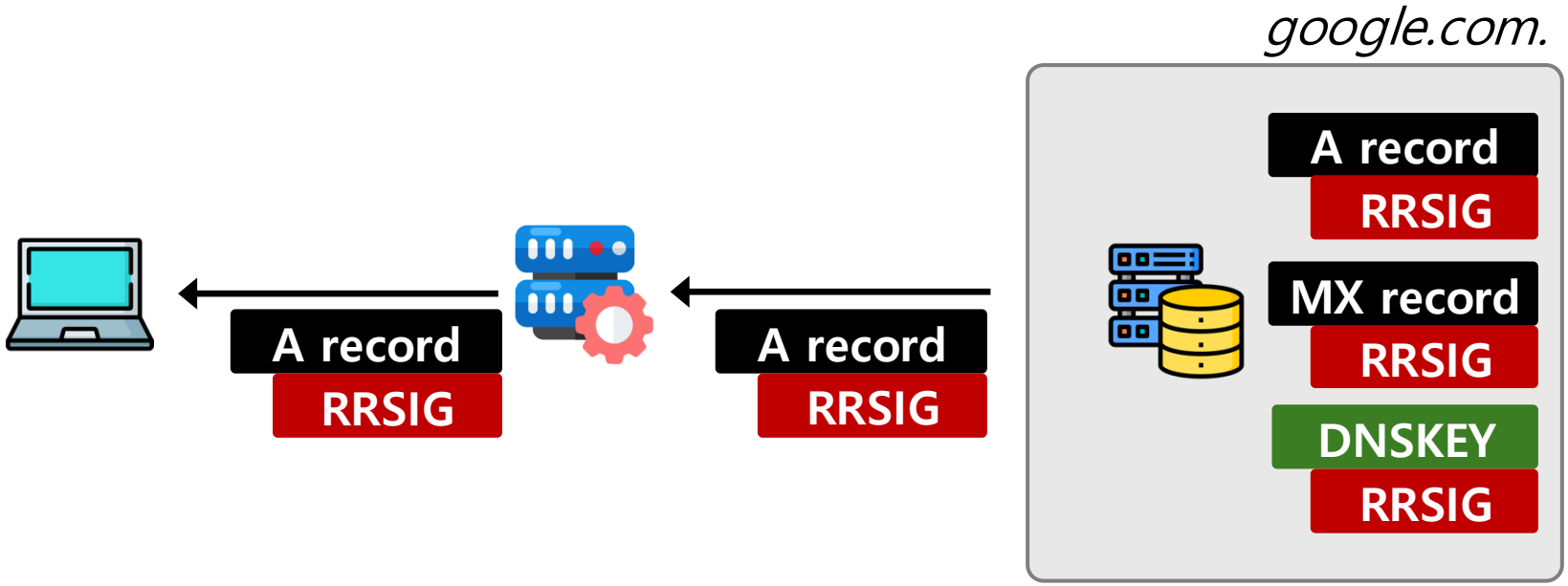


DNSSEC 동작

Authoritative Nameserver
- 도메인 관련 정보를 저장하고 있음 **+ DNS records for DNSSEC**
- DNS 요청이 오면 그 응답을 반환함 (ex. IP address)

DNS Resolver
- 클라이언트 대신 Authoritative nameserver에 DNS 요청을 보내서 응답을 찾음

Client
- DNS 요청을 보냄 (ex. 웹 브라우저)

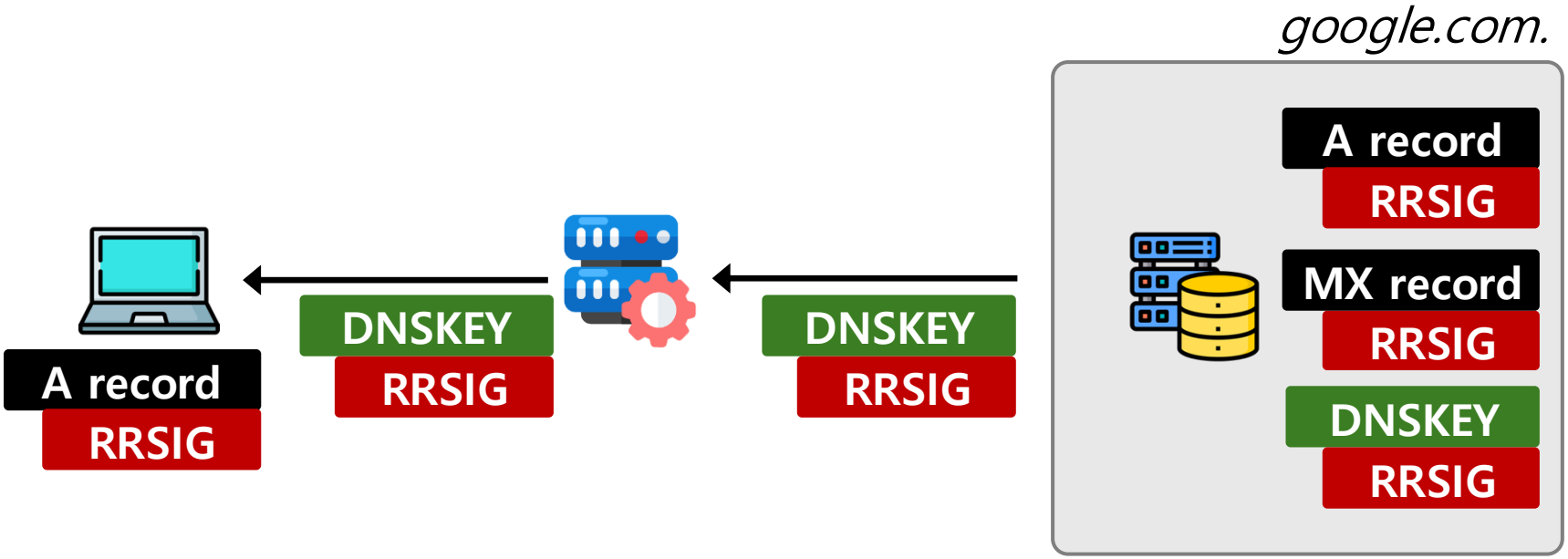


DNSSEC 동작

Authoritative Nameserver
- 도메인 관련 정보를 저장하고 있음 **+ DNS records for DNSSEC**
- DNS 요청이 오면 그 응답을 반환함 (ex. IP address)

DNS Resolver
- 클라이언트 대신 Authoritative nameserver에 DNS 요청을 보내서 응답을 찾음

Client
- DNS 요청을 보냄 (ex. 웹 브라우저)

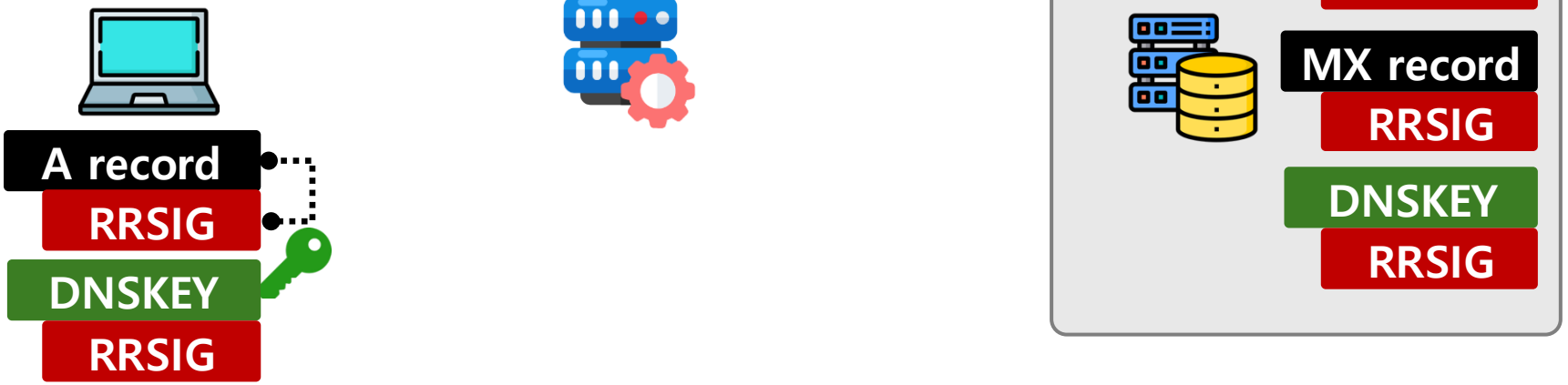


DNSSEC 동작

Authoritative Nameserver
- 도메인 관련 정보를 저장하고 있음 **+ DNS records for DNSSEC**
- DNS 요청이 오면 그 응답을 반환함 (ex. IP address)

DNS Resolver
- 클라이언트 대신 Authoritative nameserver에 DNS 요청을 보내서 응답을 찾음

Client
- DNS 요청을 보냄 (ex. 웹 브라우저)

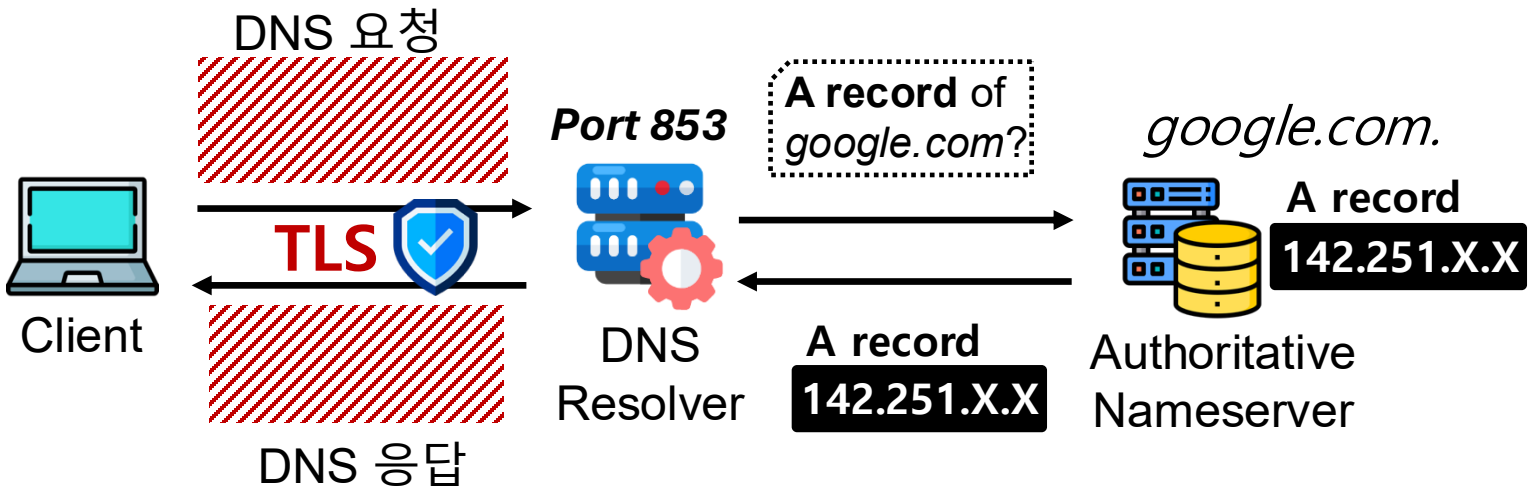


DNS over Encryption

- DNSSEC은 integrity 및 authenticity만 보장 → confidentiality는 보장하지 않음
  - 누가 어떤 도메인을 조회했는지 네트워크 상에서 그대로 노출
  - 공격자 등이 사용자의 DNS 질의를 모니터링할 수 있음
- 이를 보완하기 위해 DNS 질의/응답 자체를 암호화하는 기술 등장
  - Client와 DNS resolver 사이의 메시지를 암호화
- 종류
  - DNS over TLS (DoT)
  - DNS over HTTPS (DoH)

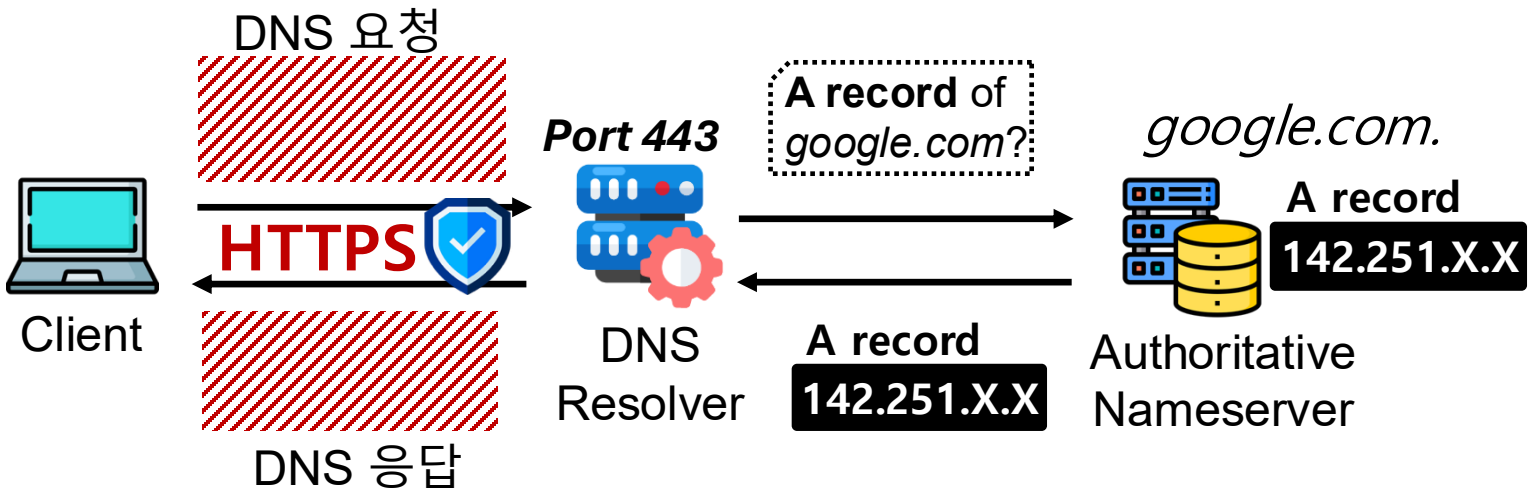
DNS over Encryption – DoT

- DNS over TLS (DoT)
  - DNS 질의/응답를 TLS로 암호화하여 전송
    - DNS 메시지 포맷은 그대로 유지, TLS 레이어만 추가
  - 별도 포트(853) 사용
    - DNS 트래픽임을 식별 가능
  - Android OS 등에서 지원



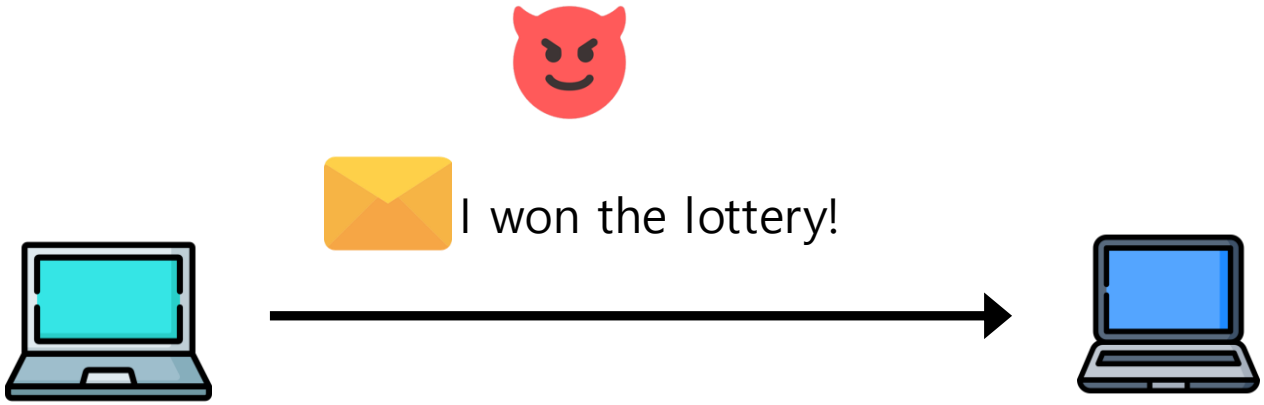
DNS over Encryption – DoH

- DNS over HTTPS (DoH)
  - DNS 질의/응답을 HTTPS 통해 전송
    - DNS 요청/응답을 HTTP 포맷으로 전송
  - HTTPS 포트(443번) 그대로 사용
    - 일반 HTTPS 웹 트래픽과 구분이 어려움
    - 프라이버시 보장
  - Chrome, Firefox 등 주요 브라우저에서 지원



Email Security

- SMTP 역시 설계 당시 보안을 고려하지 않음
  - 기본적으로 평문으로 전송
  - 전송 중 내용이 노출 될 수 있음
  - 전송 중 내용이 변조될 수 있음



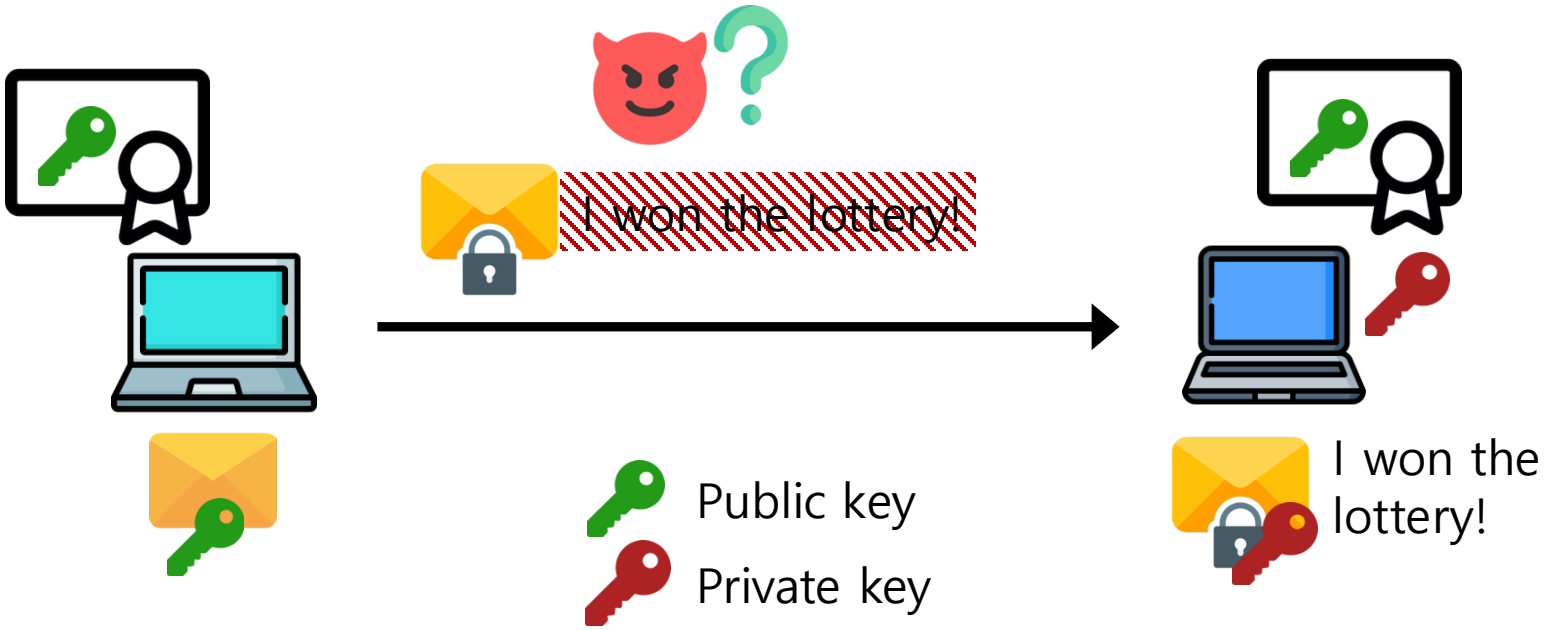
Email Security

- S/MIME (Secure/Multipurpose Internet Mail Extensions)
  - 이메일 내용(본문, 첨부파일)을 암호화
  - PKI 인프라 활용
    - CA가 발급한 인증서(Certificate) 필요
  - 송신자에서 수신자까지 보호
  - 디지털 서명도 가능 → 발신자 인증, 무결성, 부인방지 제공

  - 과정
    1. 송신자가 수신자의 인증서(및 public key)를 가져옴
    2. 수신자의 public key로 이메일 내용 암호화
    3. 송신자는 암호화 된 이메일을 전송
    4. 수신자는 자신의 private key로 암호화된 이메일 복호화

Email Security

- S/MIME (Secure/Multipurpose Internet Mail Extensions)
  - 이메일 내용(본문, 첨부파일)을 암호화
  - PKI 인프라 활용
    - 인증서(Certificate) 필요
  - 송신자에서 수신자까지 보호
  - 디지털 서명도 가능 → 발신자 인증, 무결성, 부인방지 제공



Summary

- 암호화
  - 비밀키 암호화
  - 공개키 암호화
    - Public Key infrastructure
- 인터넷 보안
  - Web security
  - DNS security
  - Email security