넷째 주 넷째 날 아카이브 & tar 압축, 해제, 암호학

노트북:	필기노트		
만든 날짜:	2019-06-20 오후 8:02	**업데이트:**	2019-06-22 오후 10:19
작성자:	이종민		
URL:	https://github.com/torlgit/cccr/blob/master/4th/15day		

15DAY

lto tape : 테이프, 기업에서 컴퓨터나 Rack에 연결해서 사용한다. / 클라우드에서도 쓰인다.
cd rom, hard disk처럼 연결해서 사용한다. / 테이프 하나의 30TB의 용량 저장이 가능하다.
재작년, 작년 기준으로 판매한 용량을 계산해봤는데, 10PB가 나올 정도로 생각보다 많은 곳에서 쓰인다.
분 당 1테라 정도의 전송이 가능하다고 한다.

요즘 추세는 lto tape 보다 디스크를 더 많이 사용한다.

Archive : clod data를 저장하기 위한 방법
Backup : 말 그대로 백업

요즘은 아카이브나, 백업을 디스크에 하는 경우가 많다.
옛날에는 테이프도 표준이 많았지만, 현재닌 lto 표준만이 남아있다.

기본적으로 디스크가 더 빠르다. 하지만 lto 방식도 7200RPM 속도로 일반 하드 디스크의 속도와 거의 유사하다.
중요한건 디스크가 테이프보다 속도도 더 빠르고, 테이프의 가장 큰 단점 Random access 이다.

Read
Random Access
Sequence Access

데이터가 연속적으로 배치 되어 있을수도 있고, 아닐수도 있다.
테이프를 물리적으로 감아야한다. 디스크는 데스크탑 (HD) 기준으로 분 당 7200번 돈다.
서버 같은 경우는 분 당 12000번 돈다고 한다.
하드디스크는 테이프 보다 랜덤액세스가 빠르지만, SSD보다 느리다 (lto tape의 큰 단점은 랜덤액세스 시간이다)
ssd는 반도체이므로 데이터가 흩어져있든 뭉쳐있든 상관이 없다.

그럼에도 lto tape이 쓰이는 이유
테이프는 보관에 용이하고, 디스크보다 데이터의 수명이 길다. 그리고 가격이 디스크에 비해 저렴하다. ssd < hard < tape
하드디스크나, SSD는 물리적인 충격에 약하고 수명이 짧다 (보통 디스크는 수명을 5년에서 10년 미만으로 보는데, 테이프는 20~30년 보관이 가능하다)

online Hot data
offline cold data

우리 기준으로 hot data는 하드 디스크, cold data는 외장하드로 비유 할 수 있다.
스토리지에서 계속 쓰고 변경하고 해야하기 때문에 hot data로 볼 수 있다.

cold data 는 보통 로그 같은 경우는 법적으로 얼마정도 보관해야하는 법이 있다.
이런 데이터 들은 문제가 생겼을 때만 봐야하는 데이터로 그 외에는 보관용 등에 사용되는 cold data로 볼 수 있다.

아카이브
아카이브와 압축을 동시에 지원을 한다. 여러가지 포맷들이 아카이브와 압축을 동시에 지원하는 경우도 있고, 아닌 경우도 있다.
zip도 실제로 그렇게 만들어서 압축을 한다.

unix에서는 아카이브를 tar로 한다.
데이터를 묶기 위한 목적으로 사용 (아카이브)

tar :
아카이브 파일을 생성하거나 해제할 수 있으며, 저장 장치에 저장할 수 있다.

jar :
자바 아카이브, 자바 프로그램을 압축할 때 사용한다. 기본적으로 tar의 아카이브에 더해서 zip으로 자동 압축한다.

tar, jar은 비슷하면서도 다르다. 옵션은 100% 똑같지만 목적이 다르다.

생성 cf a.tar
확인 tf a.tar
풀기 xf a.tar

f 뒤에는 파일명이 와야한다.

tar 기능
c : 아카이브 생성
t : 아카이브 내용확인
x : 아카이브 풀기
f : 아카이브 파일이나 테이프 장치를 지정한다. / 장치가 있디면 dev 밑에 있을 것이다.
v : tar 명령어 수행과정을 출력한다. / 유닉스에서 v옵션은 크게 두 가지로 version, verbose (자세하게) 라는 의미로 쓰인다.
h : 아카이브 하려는 파일이 심볼릭 링크 파인 경우 이 옵션을 사용하면 원본 파일을 아카이브 해준다.

특징 : 아카이브 명령어는 옵션에 대시가 안 붙는 경우가 많고, 없어도 사용이 가능하다.

lh - human readable : 휴먼리더블 / 단위를 사람이 읽을 수 있는 단위로 바꿔준다.

압축 & 해제

압축 포맷이 여러가지가 있는데, 기업에서 보통 데이터를 압축해서 이력서를 보내라고 하는데,
알집 압축 방식인 alz, egg 로 압축해서 보내는 경우가 있는데, 기업에선 알집은 돈주고 사야한다.
= 표준화 된 걸 사용하자 !

그리고 항상 문서는 pdf를 사용하고 압축은 zip으로 하자 ! / PDF 없는 회사 없다.

압축 명령어
gzip / gunzip
bzip2 / bunzip2

특징 : 원본 tar은 없어진다. / 옵션으로 설정 가능하다. (기본값)

아카이브 따로 압축 따로 하는게 정상적인 방법이지만,
tar 옵션으로 한번에 압축과 아카이브가 가능하다.
xz : 최근에 많이 사용하는 압축 명령어

압축이 오래 걸리는 이유 : 압축을 오래할수록 압축률이 높다. (압축률이 높아지면 용량이 작아진다)

tar 압축 명령어
tar [옵션][파일명/확장자][f] [압축 할 파일]
v옵션을 넣으면 tar 명령어 수행과정을 자세하게 볼 수 있다.

z : gzip j : bzip2 J : xz

ex) tar cvJf abc.bz2 / tar tvJf abc.gz / tar xvJf abc.xz | 압축 타입에 맞게 풀어주면된다.

file format (unix는 다 이름)
gzip 형식이면 확장자를 xz로 해도 gzip으로만 압축이 풀린다.

윈도우에서 확장자 개념도 unix와 같지만, 기본 프로그램 때문에 확장자 개념이 있는거다.
hwc를 doc로 바꾸고 doc로 열면 열리지 않는다.
확장자는 큰 의미가 없다. / 포맷형식과 관련이 있다.

압축 파일을 받았을 때 오류가 나면 파일형식을 보고 적절하게 압축을 풀면된다.

암호학

데이터 프로토콜들은 암호화를 한다.
우리가 사용하는 인터넷이라는 공간은 다 같이 사용을 한다. = 즉 누구든지 엿 볼 수 있다는거다.
ID, PW, Email / 인터넷에서 왔다갔다 하는 데이터들은 도감청이 가능하다.

암호화를 왜 하느냐? -> 프리즘 프로젝트
몇 년전 에드워드 스로든 이라는 사람이 NSA 기밀을 폭로하는 사건이 있었는데, NSA에서 전 세계 네트워크를 도감청을 했다는 걸 폭로한다.

우리나라도 합법적으로 다 도감청한다. 어떤 것을 검색하는지, 뭘 하는지.. 다 알 수 있다. -> 그래서 암호화를 한다.
ETE (END TO END) : 망을 경유한 단말간 종단간 통신을 말한다.

학문, 암호학이라고 한다. / IT의 계열이 아니라 수학의 영역이다.
기본적으로 일반적인 메시지를 얼마나 알 수 없게 복잡하게 만드는거에 대한 알고리즘 = 암호학방식 / 수학이다.
암호화를 제대로 얘기하려면 수학을 얘기해야한다.

암호화 : 암호화가 정해주는 기능은 4가지이다.
기밀성
(confidential) : 적법하지 않은 사용자가 데이터를 접근 할 수 없게, 데이터 접근을 하더라도 알 수 없게 하는 것

무결성(Integrity) : 허가되지 않은 사용자가 허가 받은 사용자만 데이터를 조작 할 수 있는 것

인증(Authertiation)

부인방지(Non-repudiation) :
부인을 하는 것을 방지
계약서에 보통 서명, 날인을 하는 이유가 뭐냐?? -> 나중에 딴 소리 못하게 하려고
IT 에서는 전자서명이 있다.

필요할 때 언제든지 사용 할 수 있는 것 : 가용성

공인인증서 이슈 :
인증서라는게 KEY인데, 이걸 사용자한테 관리하라고 하고, 거기에 대한 책임을 안진다. / 공인인증서는 4가지에 해당된다.

기밀성, 무결성, 인증, 부인방지 4가지 정의를 알고 있어야한다.

보안의 3 요소
C
I
A(vailability)

우리가 봐야하는건 크게 3가지

symmetric key :
대칭이라는 의미, 즉 암호화와 복호화에 같은 암호키를 쓰는 알고리즘 (기밀성만 제공)
메세지를 암호화 하는 것 (인크립션) 메시지를 원래대로 되돌리는걸 복호화

asymmetric key :
비대칭이라는 의미, 퍼블릭키 암호화 시스템 (기밀성, 인증, 부인방지 제공)

hash function :
해시 알고리즘, 줄여서 해시라고도 한다. (무결성만 제공)
암호화는 아니지만 암호화 학문에서 다룬다. -> 왜 암호화가 아니냐? -> 키가 없다. 암호화는 대칭이든 아니든 키가 있어야한다.
엄밀하게 말하면 암호화라가 할 수 없지만 암호화와 같이 설명된다.

적절하게 섞어서 4가지를 적용해서 사용해야한다.

무결성 :
Ethernet frame 이 어떻게 생겼나? [L2 | Payload | L2F] Trailer / footer. <- cheak sum = 오류 검출

오류 검출이 뭐냐 ? -> 무결성

A -> B 데이터 전송 Header, Payload 를 보내고 xor 연산으로 L2F을 A가 만든다.
B가 받아서 xor 연산으로 L2F를 푸는데 값이 다르면 문제가 있다. (무결성이 깨진거다)

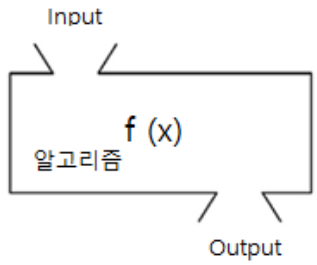
A -> B
데이터 전송 할 때 Header, Payload를 보내고

IP Geader, Tcp Header
cheack sum fiId가 있다. 같으면 상관없지만 다르면 문제가 있다.

ICMP type 12 : IP header cheak sum error / 이 오류가 발생했다면, IP Header에 문제가 있는거다.

hsh function
algorithm
message digst
one way function

Data file



= Hash vlaue
fixed legth 128bit

경우의 수는 정해져있다. 2진수가 1 자리수면 0,1
입력 값은 길이를 정하지 않아 무한하다.

값은 값을 넣으면 값은 값이 나온다.
문제는 다른 값을 넣어도 같은 값이 나올 수 있다.

birthday paradox :
Input과 같은 길이면 출력 값이 같아져서 문제가 생기는데 이걸 birthday attack 라고 한다.
유추해선 안된다. 엮으로 만들 수 없으면 좋은 함수다. / 역방향이 되지 않아야한다.

collision attacks :
암호학에서 암호 해시에 대한 해시 충돌을 생성하는 입력 값을 찾는다. -> 엮으로 계산 할 수 있다. 입력값 < 출력값 일 때

block size :
데이터 블록 하나를 압축한 뒤의 내부적인 해시값

MD5 : 가장 기본 알고리즘

SHA :
-SHA-0
-SHA-1

안전한 암호학적 국가표준 해시 함수 였었다. 취약점이 발견이 되어서 현재는 사용하지 않는다.

현재 사용하는 형태
SHA-2 - 256
- 512

최근 표준은 SHA3이다.
일반 파일은 SHA1도 사용을 한다.
암호화는 컴퓨터에서 제일 많은 CPU를 사용한다고함

encrypt
decrypt
palintest 평문 (encrypt = 암호화)
cyphertext 암호문 (decrypt + sy 복호화)
ciphertext

cryptology

취업을 하면 암호학 관련 책 한 권 이상은 읽어보는게 좋다.

Rot13 (rotate) 시저 암호, 13자리를 미는 것 [13이라는게 key다] / shift 시킨다 <알고리즘>

Confusion 혼돈 / 잘 알아보지 못하게 혼돈 시킨다.
diffusion 확산 / 알아보지 못하게 얼마나 확산 시키나.

substitution 치환 / 문자를 바꾸는 것
transposition 전치 / 위치를 바꾸는 것

수학적으로 얼마나 치환, 전치가 잘 일어나는가 ? -> 잘 일어날수록 암호화가 잘된것

Rounds : 알고리즘을 몇 번 돌리나? 혼돈과 확산을 한번의 Rounds라고 한다. / 혼돈과 확산을 계속 만들어낸다.

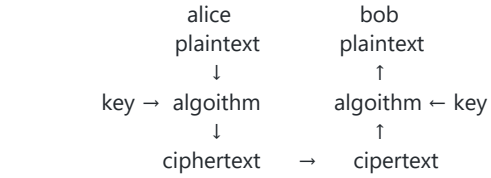
부호화 / 인코딩
복호화 / 디코딩

암호화가 아니다. 데이터를 잘 저장하고 잘 전달하기 위해서 코드로 나타낸다.
인코딩 방식 중에 base64가 있다.

protocol에 어떤 특정 값을 부호화하는 경우가 있는데, 보통 http 프로토콜에서 많이 사용한다.
왜 하는지? -> 알아보기 어렵게 하려고 // 간혹 사용하는데 알아보는 사람한테는 효과가 없다.

인코딩 , 디코딩 방식 이므로 쉽게 풀어낼 수 있다.

대칭 키



대칭키의 문제는 키를 안전하게 분배할 방법이 없다.
암호화키와 복호화키가 동일하다.
네트워크에서 쓰지 않는다. / 쓰는 이유 - 빠르다. 비대칭 방식은 너무 느려서 못쓴다.
key = secret key / private key

DES 암호화 알고리즘 :
IBM이 70년도에 만들었다. 90년도에 키가 유출이 되서 다른 방식이 나오는데,
3DES - DES를 3번 돌리는건데, 어차피 취약점이 유출 되었기에 3번을 돌리든 만 번을 돌리든 같다.
AES (표준키) 128~256 bit / 키를 512 ~ 1024까지 늘릴 수는 있는데, 지금은 256을 많이 사용해서 256까지 사용한다.

1024까지 늘릴 수 있지만 256을 쓰는 이유는 256bit 알고리즘 방식으로도 충분하기도 하고,
요즘 스마트폰을 사용하는데 512, 1024 bit 까지 늘려놓으면 계산하는데 시간이 오래걸려서 느리다.

윈도우 95, 98
옛날에는 영문윈도우와 비영문윈도우로 따로 판매를 하였는데
영문윈도우는 암호화 방식을 64bit를 사용했고,
비영문은 48~56bit 의 암호화방식으로 만들어서 판매했다.
-> 패권을 가지고 싶어서 암호화방식을 다르게해서 판매했지만 법이 바뀌어서 같아지게 되었다.

우리나라는 128bit 키를 전 세계에서 3번째로 만들었다. / 하지만 지금은 안쓴다 AES라는 더 좋은 표준이 있기 때문에
SEED(공인인증서), ARIA, HAS-160 등이 있지만 사용을 안한다.

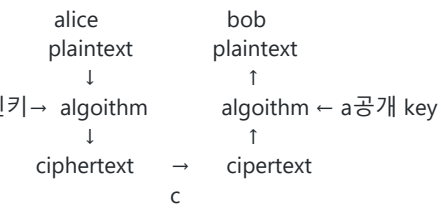
DH
Diffe - Hellman
대칭키의 키를 교환하는 형태로만 썼다. / 키 교환만 한다. -> 속도가 빠르기 때문에
key pair : 비대칭키 방식은 키에 쌍이 존재해야한다.
RSA : 공개키의 표준화 (1024-2048bit) / 산업표준
ECC : 타원곡선 알고리즘을 통해서 구현을 한 공개키 (256-512bit)

public key :
누구에게나 공개해도 상관 없는 키

개인키 :
반드시 나만 갖고 있어야 되는 키

alice - 공개키 bob - 공개키
- 개인키 - 개인키

대칭키 방식의 문제점
-> 사용자가 많아지면 키 개수가 많아진다. $\frac{n(n-1)}{2}$



기밀성 : 공개키로 암호화 하면 개인키로만 복호화 할 수 있다.
개인키로 암호화 하면 공개키로만 복호화 할 수 있다.

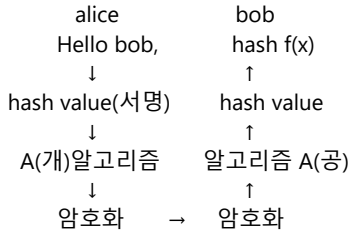
키를 안전하게 보낼 수 없던 걸 안전하게 보낼 수 있게 되었다.
개인키가 외부에 노출 되지 않는다면, 절대로 풀 수 있는 방법이 없다. / 속도가 엄청나게 느리다.

A의 개인키로 cihpertext를 했을 때 C가 전달된 정보를 도청 및 조작한 후 다른 쪽으로 전달한다.

전자서명 알고리즘 : 공개키 방식과 해시를 섞어서 같이 사용한다.
RSA SHA1 - 알고리즘

A : Hello, bob -> message -> hash f(x) -> hash value (서명) -> 알고리즘 A의 개인키로 암호화 -> 전송
B : 암호 -> A의 공개키 -> hash value (서명) -> hash value -> hash f(x)

plaintext를 암호화해도 법으로 문제가 없다. B의 공개키로 암호화해서 보내면된다.



MITM 중간자 공격
모든 암호화는 중간자 공격에 취약하다. -> public key infrastructure (PKI) - 인증서X.509
인증서 안에는 공개키, 서명, 지문(fingerpoint) 제 3자의 성명 - CA(인증기관)
공개키를 hash를 뜨면 - hash value가 지문이다.

인증기관은 여러 개 있을 수 있는데 그 중 최상위 인증기관을 Root CA라고 한다.
Root CA는 보통 정부기관으로, 우리는 한국인터넷진흥원이 한국의 Root CA이다.

RA 등록기관 -> user (사용자) / 등록을 하면 특정기관으로부터 인증서와 개인키를 준다.
국내에는 6개의 CA가 있는데, 은행이 CA로부터 인증을 받아서 공인인증서를 인증 받아주는거다.

인증서 관련된걸 관리하는 서버가 있는데 LDAP이라고 한다.
인증서에는 개인키가 없다. 공개키를 가지고 암호화를 한다.

http = http + tls
ssl = netscape 사용하지 않는다. 개인회사 이므로 조작을 할 수 있다고 해서 tls를 만들었다. (상징성으로 남아있다)
tls = pki

인터넷 뱅킹은 양방향 인증이다. 서버와 사용자가 서로 인증

보통 인터넷을 할 때 https 하면서 자물쇠가 있는데 눌러보면 서버에 대한 인증이 나온다.
왜 인터넷을 할 때는 서로 인증을 안하나? 라고 생각을 하면 인터넷을 할 때 마다 인증을 받아서 해야하는 점이 있다.
서버에 대한 인증은 사용자가 신뢰 할 수 있는 서버인지를 판별을 하고 접속을 해야 하기 때문에 서버에 대한 인증을 받는다. = 단방향 인증

fqdn 기본적으로 인증서는 서버 한 대 당 한개의 인증을 받아야 했었는데, 최근에는 1개의 인증서로 서브 도메인 무제한 적용이 가능하다.

EV-SSL 확장, 검증 / 별도의 기능은 아니다. (보통 인증이 되면 초록색, 인증이 안되었으면 빨간색으로 주소창이 표시된다.

PGP : Email 암호화표준
국내에서는 인증을 안한다. 외국에서는 자주 사용하는데, 외국계회사와 거래할 때 PGP 암호화 방식으로
상대방의 공개키로 암호화하고 상대방의 개인키로 푸는 형식을 사용한다.
전 세계 pgp는 동기화가 되어있다.

서명파일 확장자 :
보통 서명 파일은 .sig .sign 등이 있다.

리눅스 꿀팁!
드래그 + 휠클릭 : 복사 붙여넣기가 된다.