



2013 CodeEngn Conference 08

2013.07.13 SAT 13:00

www.CodeEngn.com | www.RCEHub.com



2013 CodeEngn Conference 08

2013.07.13 SAT 13:00

www.CodeEngn.com | www.RCEHub.com

CodeEngn [코드엔진]

Since 2007

1. 코드엔진 컨퍼런스 소개

코드엔진 컨퍼런스는 현업에 있는 실무자가 직접 주최 및 주관을 하고 있으며 국내 리버스엔지니어링에 대한 정보공유를 위해 2007년 부터 시작되었고 해당 분야에 관심이 많은 학생 및 실무자를 대상으로 개최하고 있습니다. 또한, 리버스엔지니어링이라는 하나의 큰 주제로 소프트웨어 보안에 대한 다양한 시각으로 불법이 아닌 발전적인 접근 및 연구를 주제로 개최하고 있습니다. 코드엔진 컨퍼런스는 상업적 이익이 없는 비영리로 운영되고 있으며 도움을 주시는 분들과 몇몇 회사의 협찬으로 운영되고 있습니다.

2. 다루는 주제

- Anti ReverseEngineering Technic
- Usermode / Kernelmode Debugging Technic
- OS / Application Vulnerabilities Research
- OS / Application Protection Technic
- Windows / Unix / Mac OS Secure Programming
- Packing / Manual Unpacking
- Malware Analysis
- SmartPhone [Android, Bada, iOS]
- Virtualization Technic Research
- Code Obfuscation & Deobfuscation
- Advanced Hooking Techniques
- More Things...

3. 발표자 분들의 혜택

- CodeEngn Conference 평생 무료입장
- 해당 Conference 발표 동영상 HD화질 DVD 제공
- 소정의 발표비
- 기념품
- 받고 싶은 서적 5권
- CodeEngn Groups 가입

실력있는 발표자를 모십니다!!

http://codeengn.com/call_for_paper

4. 2007 ~ 2013년 발표주제

	발표자	발표제목
2007.07.21 1회	김기오	NASM 어셈블러 사용법과 Calling Convention
	이강석	Malware Analysis Start
	송창현	Manual Unpacking
	구사무엘	Windows 커널단의 후킹
	윤재근	Linux Virus Analysis
	박영호	Art of Hooking
2008.11.08 2회	송민호	임베디드 시스템에서의 펌웨어 보호
	윤재근	Immunity Debugger 활용과 플러그인 제작
	태인규	정적 링크된 Stripped ELF 바이너리 상에서의 함수 탐지 기법
	강봉구	스타크래프트 맵핵 제작을 통해 알아보는 리버싱
2009.07.04 3회	최상명, 김태형	(파일바이러스 치료로직 개발자 입장에서 본) 파일 바이러스 분석
	고홍환	윈도우 커널 악성코드에 대한 분석 및 방법
	박찬암	DEFCON CTF 2009 Binary Leetness 100-500 Solutions
	안기찬	Reversing Undocumented File Formats using a Hex Editor and your Brain

	발표자	발표제목
2010.07.03 4회	심준보	Taint analysis for vulnerability discovery
	김은수	Defcon 18 CTF 문제풀이
	강병탁	Art of Keylogging - 키보드보안과 관계없는 키로거들
	Max	Fighting against Botnet
2011.07.02 5회	박상호	DBI(Dynamic Binary Instrumentation)를 이용한 프로그램 취약점 분석
	박천성	안드로이드 리눅스에서의 시스템 해킹
	손충호	x64 아키텍쳐 분석과 x64와 x86 비교 분석
	차민석	virse program messge DOS to Win
	한정화	파일바이러스 분석 및 치료로직 개발
2012.07.07 6회	박병진	Defcon 20th : The way to go to Las Vegas
	권혁	Secuinside 2012 CTF 예선 문제풀이
	유동훈	모바일 스마트 플랫폼 원격, 로컬 취약점 공격 분석
	이승진	Everyone has his or her own fuzzer

	발표자	발표제목
2012.12.01 7회	서만기	Exploit Writing Technique의 발전과 최신 트랜드
	신정훈	NFC, Play on real world
	정재훈	Defcon 20th : 본선 CTF 문제풀이
	고홍환	Manual UnPack by Debugger
	차민석	iThreat
2013.07.13 8회	권혁	Pwning multiplayer game - case Starcraft Broodwar
	김슬기	각종 취약점과 대응방안 & 해킹, 보안 문제풀이
	김창수	Android 악성앱 필터링 시스템
	신정훈	Android Platform 3G, 4G 통신 분석
	서만기	Windows 8 Exploit

2013 CodeEngn Conference 08 Sponsor



숙명여대보안동아리, SISS

(주)NSHC

한국정보기술연구원

(주)사이버원

(주)보안뉴스

www.CodeEngn.com | www.RCEHub.com



We need Heroes by NSHC



전세계 6명 중 1명은 스마트폰을 사용하고 있습니다
이제 아마존 원주민도 스마트폰으로 사냥법을
검색할 날도 멀지 않았습니다



스마트폰은 우리의 삶에 많은 변화를 가져왔습니다
언제 어디서나 음악을 듣고, 카톡도 하죠
참 편리해졌습니다



편리하다고 좋은 것만 있는 것은 아닙니다
나쁜 해커들은 사이버 테러를 통해 나쁜 짓을 하고
이제, 우리의 스마트폰을 노리고 있습니다



악성코드를 심어 개인정보를 빼가고
피싱을 통해 신용카드 정보를 가로채기도 합니다



심지어 개인사진, 주소록, 의료정보, 위치정보까지
빼내가고 있습니다
생각만 해도 너~~~무 무섭습니다



그래서, 우리에겐 영웅이 필요합니다



진정한 영웅들이 한자리에 모였습니다



우리가 바로 NSHC입니다

함께 영웅이 되어보지 않겠습니까?



031-458-6456



recruit@nshc.net



[Facebook.com/NSHC.Story](https://www.facebook.com/NSHC.Story)



경기도 의왕시 광진말길 55 N스퀘어 (주)NSHC



KITRI 소개.

최고를 추구하는 IT 연구기관 한국정보기술연구원

한국정보기술연구원(Korea Information Technology Research Institute)은
산업통상자원부' 산하의 전문생산기술연구소로써 대한민국 IT 발전의 미래를 만들어가고 있습니다.

KITRI는 최초 컴퓨터교육훈련센터(1985년)로 설립됐으며, 1990년 한국컴퓨터기술원으로 개편했습니다.
그리고 1992년에 한국정보기술연구원'으로 확대 · 개편했습니다.

주요사업으로 정보화관련 전문기술인력양성사업과 국제협력사업, 연구개발사업을 진행하고 있으며
현재까지 약 15,000여명의 교육생들을 배출한 IT전문연구교육기관입니다.

KITRI는 2011년부터 정보보안사업을 전략사업으로 추진 중에 있습니다.
특히 차세대 보안리더 양성프로그램(Best of the Best)은 국 · 내외 해킹방어대회 입상자 등 대한민국 1%의
우수인재를 대상으로 서바이벌방식의 도제식 교육을 실시하여 최종 명품인재를 선발하는 프로그램입니다.
미국 CNN, 프랑스 AFP통신, 영국 sky news 등 수많은 국내외 언론으로부터 많은 관심과 조명을 받고 있습니다.

KITRI 사업영역.





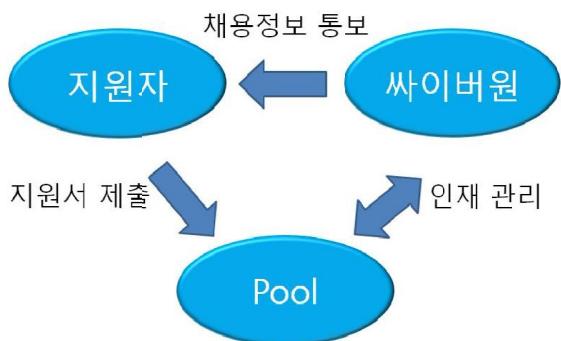
코드엔진의 성공을 기원합니다.



싸이버원에서 유능한 인재를 모십니다.

인재 Pool System에 등록해 주세요.

미래창조과학부지정 지식정보보안컨설팅전문업체, 보안관제전문업체 / ISO27001인증 (주)싸이버원



- 지원서식(입사지원서 및 경력소개서)을 잡코리아에서 다운받거나 recruit@cyberone.kr로 문의하세요.
- 지원서식 작성 후 recruit@cyberone.kr로 보내주세요.
- 싸이버원의 인재 Pool System에 등록된 귀하는 수시채용 및 공개채용의 자동 심사대상이 됩니다.
- 인재 Pool System에 등록되어 있더라도 채용의 기회가 발생하지 않을 수 있습니다.

보안관제 서비스

- MSS(Managed Security Service)
- 통합관제 시스템 구축
- 실시간 모니터링 서비스
- 성능/장애/이벤트 분석 서비스
- 침해대응 서비스
- 취약성분석 서비스
- 보안전문가 파견 서비스

보안컨설팅 서비스

- 기본시설 컨설팅
- 정보보호관리체계 인증 컨설팅 (KISA ISMS, ISO27001 등)
- 개인정보보호 컨설팅
- 어플리케이션 취약성 점검 및 모의해킹
- 정보보호 마스터플랜 및 보안전략 수립
- U-Security 보안 컨설팅

통합보안관리 서비스

- 웹보안&DLP : Websense
- SSI(Security System Integration)
- ESM(통합보안관리 시스템) 구축
- SecuPlat ESM(UNIX, WIN)
- CAS(종합분석 시스템) 구축
- ISAC(정보공유분석센터) 구축
- RMS(위험관리 시스템) 구축
- Web Eyes(웹위변조 모니터링 시스템) 구축
- Anti-Dos : Radware, 스나이퍼DDX

융합보안 서비스

- 스마트카드 시스템 구축 컨설팅
- 시스템 구축 및 Integration
- Embedded 단말기 개발 및 공급
- RFID 기반 응용솔루션 개발 및 공급
- 카드발급 대행
- 보안솔루션 공급

인터넷데이터 서비스

- Co-Location
- 서버호스팅
- VPN
- 웹호스팅
- 보안/백업 서비스
- 메일ASP
- 웹로그분석
- 네트워크장비임대

보안뉴스
www.boannews.com



보안이 무너지면
기업이 무너집니다

성공 비즈니스의 핵심 '보안'
그 해답을 제시하는
보안뉴스

보안전문 기자들의 생생한 현장소식과 함께 심층 분석된 최신 뉴스와 콘텐츠로 가득 차 있습니다!

뉴스

국내외 보안관련
최신 뉴스
실시간 공급

콘텐츠

사회 각 분야별
베스트프랙티스
제공

서비스

웹취약점
점검과 무료 백신 등
보안서비스 제공

뉴스레터

최신 보안이슈
데일리
업데이트

시장트랜드

국내외
보안시장
트랜드 분석

www.CodeEngn.com | www.RCEHub.com



Pwning multiplayer games

- case Starcraft-Broodwar

Kwon Hyuk (pwn3r) | B10S | 2013/07/13

www.CodeEngn.com
2013 CodeEngn Conference 08

Code**⚡**Engn

Outline

1. Introduce

1. Who am I?
2. Summary
3. Why multiplayer game?
4. Choosing target

2. Starcraft

1. What is starcraft?
2. Attack vectors

3. Fuzzing starcraft

1. Structure of map file
2. Attack scenario
3. Fuzzing scenario
4. Exploitable crashes

4. Exploitation

1. Heap spray
2. Process continuation

5. Conclusion

1. Timeline
2. I have talked about ..
3. Q & A

Chapter 1: Introduce

- 1) Who am I?**
- 2) I will talk about ..**
- 3) Why multiplayer game?**
- 4) Choosing target**

1. Introduce

\$ whoami

Name : Kwon Hyuk
Age : 19
FB : <http://fb.com/kwonpwn3r>
Blog : <http://pwn3r.tistory.com>
Job : 고삼완전체



1. Introduce

I will talk about ..

- BoB 에서 진행했던 '멀티플레이어 게임 취약점 점검' 프로젝트
- Multiplayer game 중 하나인 Starcraft-broodwar에서 발견한 RCE(remote code execution) 취약점과 공략하기까지의 과정
- 간단한 Exploitation 기술들



1. Introduce

Why multiplayer game?

1. 수 많은 게임 이용자

- 통계자료를 대지 않아도 다들 알고 있는 엄청난 이용자 수

2. 게임 보안의 불균형한 발전

- Multiplayer 게임의 시장규모와 이용자 수가 증가함에 따라 게임보안도 발전함.
- But, 게임의 룰에서 탈출하는 Abusing bug를 막는 데에 주력하여 취약성 측면에서의 보안을 놓치고 있음.

3. 다양한 공격벡터

- 게임에 따라 Map, Chatting, Image, Sound 등 다양한 공격 벡터가 존재함.

4. Just for fun

- 취약점 찾다가 잘 안되면 게임 한판 ㅋㅋ

1. Introduce

Choosing target



- 현재 가장 유명한 게임



- 그림으로 대화하는 기능



- 음성 채팅 기능



- 그림 영상 정보를 주고받는 기능

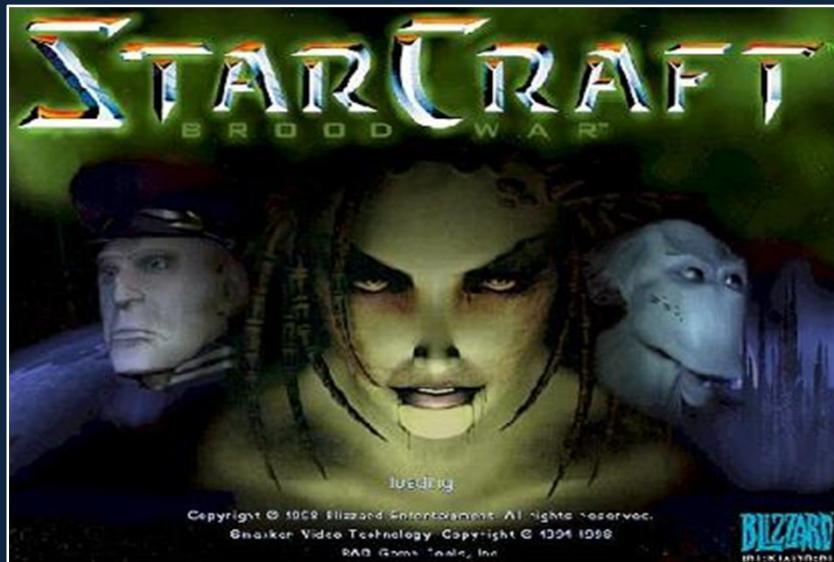
Chapter 2: Starcraft

- 1) What is starcraft?**

- 2) Attack vectors**

2. Starcraft

What is starcraft?



2. Starcraft

게임 진행 방식

- 기본적으로 하나의 맵으로 방을 개설하고, 타 플레이어들이 방에 입장하여 같이 게임을 진행하는 방식.
- 방에 입장한 플레이어들에게 방장이 맵을 보내줌.
- 즉, 플레이어들은 같은 맵 파일을 공유하여 게임을 진행하게 됨.



2. Starcraft

게임 진행 방식

- 기본적으로 하나의 맵으로 방을 개설하고, 타 플레이어들이 방에 입장하여 같이 게임을 진행하는 방식.
- 방에 입장한 플레이어들에게 방장이 맵을 보내줌.
- 즉, 플레이어들은 같은 맵 파일을 공유하여 게임을 진행하게 됨.



Chapter 3: Fuzzing Starcraft

- 1) Structure of map file**
- 2) Fuzzing scenario**
- 3) Analyzing crashes**

3. Fuzzing Starcraft

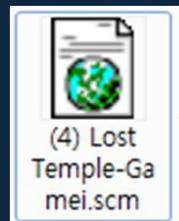
Map file structure

1) 확장자

- .scm / .scx

2) File format

- MPQ compressed file format
(MPQ = Mike O'Brien Pack)
- 맵 파일은 하나의 MPQ 압축파일.
- 맵을 구성하는 파일들이 하나의 MPQ 압축 파일에 압축되어 있는 구조



Offset(h)	00	01	02	03	04	05	06	07	08	09	0A	0B	0C	0D	0E	0F
00000000	4D	50	51	1A	20	00	00	00	3D	7C	04	00	00	00	03	00
00000010	ED	3B	04	00	ED	7B	04	00	00	04	00	10	05	00	00	00
00000020	EE	E4	B5	86	7A	D9	7B	AE	F5	4B	A1	7D	1B	AC	C6	BA
00000030	AD	DA	3C	14	54	AD	FB	8A	B5	E2	FB	B6	16	18	84	6B

The memory dump shows the raw byte data of the file. A red arrow points from the '(4) Lost Temple-Ga mei.scm' file icon to the first row of the table, indicating the connection between the file and its binary representation.

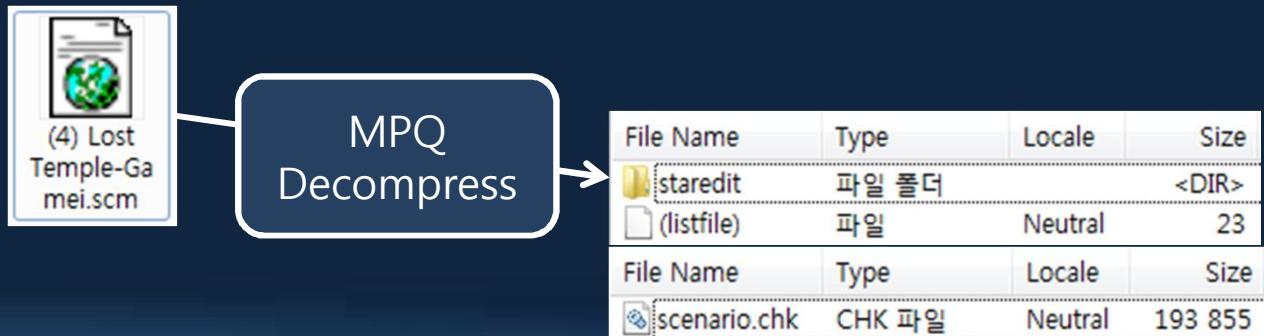
3. Fuzzing Starcraft

Map file structure

3) MPQ decompressed

: MPQ 압축파일에 압축된 파일들

- (listfile)
=> 압축된 파일명 list
- staredit\scenario.chk
=> 맵의 설정 정보를 저장하는 핵심 파일
- etc ..
=> 그 외 게임설정에 따라 음악파일, 그래픽 파일 등이 추가될 수 있다.



3. Fuzzing Starcraft

Map file structure

4) Scenario.chk

- 1) 맵의 실질적인 설정 정보가 저장된 파일
- 2) 섹션 단위로 구성됨.
- 3) TYPE, VER, OWNR, TRIG, ... 등 39개 종류의 Section이 존재.
- 4) [Section name] [Section size] [Section data] 가 반복된 단순한 구조
(4 byte) (4 byte) (n bytes)

Offset(h)	00	01	02	03	04	05	06	07	08	09	0A	0B	0C	0D	0E	0F
00000000	54	59	50	45	04	00	00	00	52	41	57	53	56	45	52	20
00000010	02	00	00	00	3F	00	49	56	45	32	02	00	00	00	0B	00
00000020	56	43	4F	44	10	04	00	00	34	19	CA	77	99	DC	68	71
00000030	0A	60	BF	C3	A7	E7	75	A7	1F	29	7D	A6	D7	B0	3A	BB

TYPE ... RAWVER
...IVE2...
VCOD....4.ÈwÜhç
.`ÃŞçu\$.)}!xº::»

Section name
Section size
Section data

3. Fuzzing Starcraft

Attack scenario

취약점 발생시점에 따른 공격 시나리오



1) Victim이 악성 맵으로 방을 Open하다가 취약점 유발

- Victim에게 악성 맵을 전송해주고, Victim이 해당 맵으로 방을 개설하는 과정에서 취약점이 유발됨.
- 악성 맵을 인터넷에 유포해도 해당 맵을 받아서 방을 개설할 사람들은 많지 않기 때문에, 공격대상이 적어 파급력이 떨어짐.



2) 공격자가 악성 맵으로 방을 Open하고, Victim이 접속시 취약점 유발

- 공격자가 만든 방에 접속하는 Victim은 모두 취약점이 유발됨.
- 감염된 Victim은 퇴장시켜버리면 또 다른 Victim들의 접속을 기다릴 수 있기 때문에 파급력이 높음.



3) 공격자가 악성 맵으로 방을 Open하고, 게임 시작 시 취약점 유발

- 공격자가 만든 방에 접속하고, 게임을 시작하면 Victim은 같은 방에 있는 Victim은 모두 취약점이 유발됨.
- 게임을 시작해야 감염이 된다는 점 때문에, 한번에 방에 입장할 수 있는 최대인원(8명)까지만 취약점을 유발시킬 수 있음.
(2번에 비해 파급력이 떨어지긴 하지만, 자동화 시킨다면 낮지 않은 파급력)

3. Fuzzing Starcraft

Attack scenario



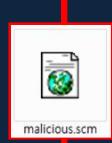
1) Victim이 악성 맵으로 방을 Open하다가 취약점 유발

- Victim에게 악성 맵을 전송해주고, Victim이 해당 맵으로 방을 개설하는 과정에서 취약점이 유발됨.
- 악성 맵을 인터넷에 유포해도 해당 맵을 받아서 방을 개설할 사람들은 많지 않기 때문에, 공격대상이 적어 파급력이 떨어짐.

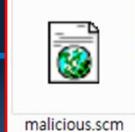


Attacker

Send
Map file



Victim



Choosing map



3. Fuzzing Starcraft

Attack scenario

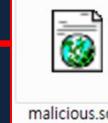


2) 공격자가 악성 맵으로 방을 Open하고, Victim이 접속시 취약점 유발

- 공격자가 만든 방에 접속하는 Victim은 모두 취약점이 유발됨.
- 감염된 Victim은 퇴장시켜버리면 또 다른 Victim들의 접속을 기다릴 수 있기 때문에 파급력이 높음.



Attacker



Choosing map
& Create game

방제목:
2:2 로템 ㅋㅋㅋ

Send
Map file



Victim

Join the game



3. Fuzzing Starcraft

Attack scenario



Attacker

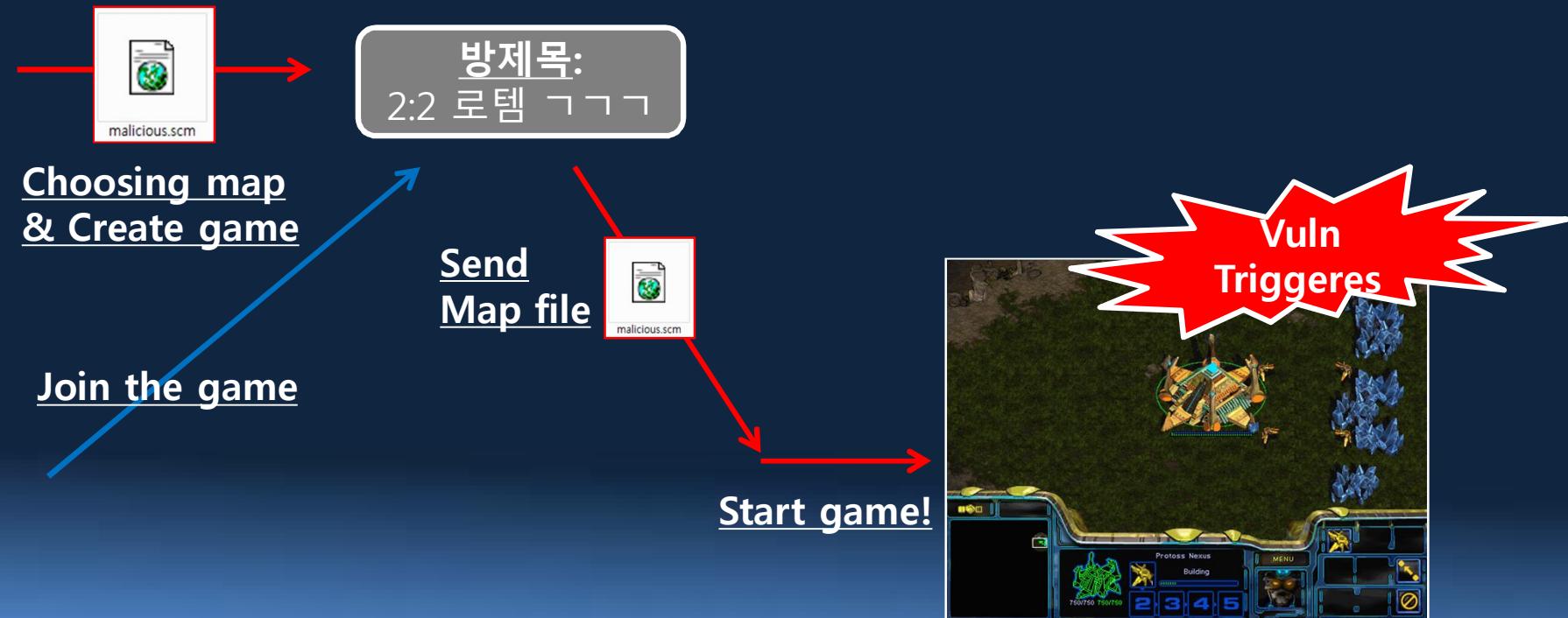


Victim



3) 공격자가 악성 맵으로 방을 Open하고, 게임 시작 시 취약점 유발

- 공격자가 만든 방에 접속하고, 게임을 시작하면 Victim은 같은 방에 있는 Victim은 모두 취약점이 유발됨.
- 게임을 시작해야 감염이 된다는 점 때문에, 한번에 방에 입장할 수 있는 최대인원(8명)까지만 취약점을 유발시킬 수 있음.



3. Fuzzing Starcraft

Fuzzing scenario

Scenario 1

- 1개의 Client 필요

Client 1 : 스크립트가 맵 파일을 조작한 후 Starcraft를 실행하여 방을 개설함.

Scenario 2

- 2개의 Client 필요

Client 1 : Starcraft를 실행하여 UDP 모드로 방을 개설하고, 스크립트가 Starcraft 프로세스의 메모리에 있는 맵 파일을 조작함을 반복

Client 2 : 스크립트가 Starcraft를 실행하여 UDP 모드에서 Client (1)이 개설한 방에 참가함을 반복 -> Exception 발생여부 확인

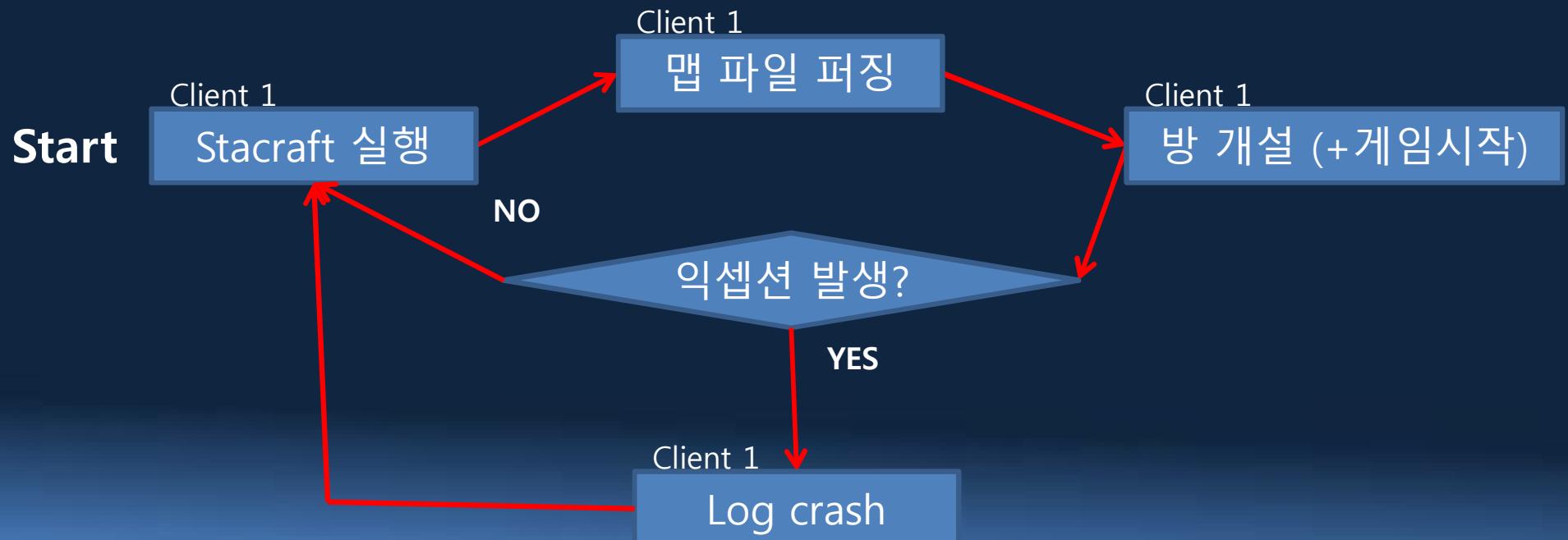
3. Fuzzing Starcraft

Fuzzing scenario

Scenario 1.

- 1개의 Client 필요

Client 1 : 스크립트가 맵 파일을 조작한 후 Starcraft를 실행하여 방을 개설함.



3. Fuzzing Starcraft

Fuzzing scenario

Scenario 1.

장점

- 스크립트 작성이 간단함.
- 1대의 클라이언트만 필요

단점

- 맵 파일을 압축 해제하지 않고 조작할 경우, 방을 개설할 때에 크래시가 발생하기 때문에 압축 해제를 할 때에만 적용 가능한 시나리오

3. Fuzzing Starcraft

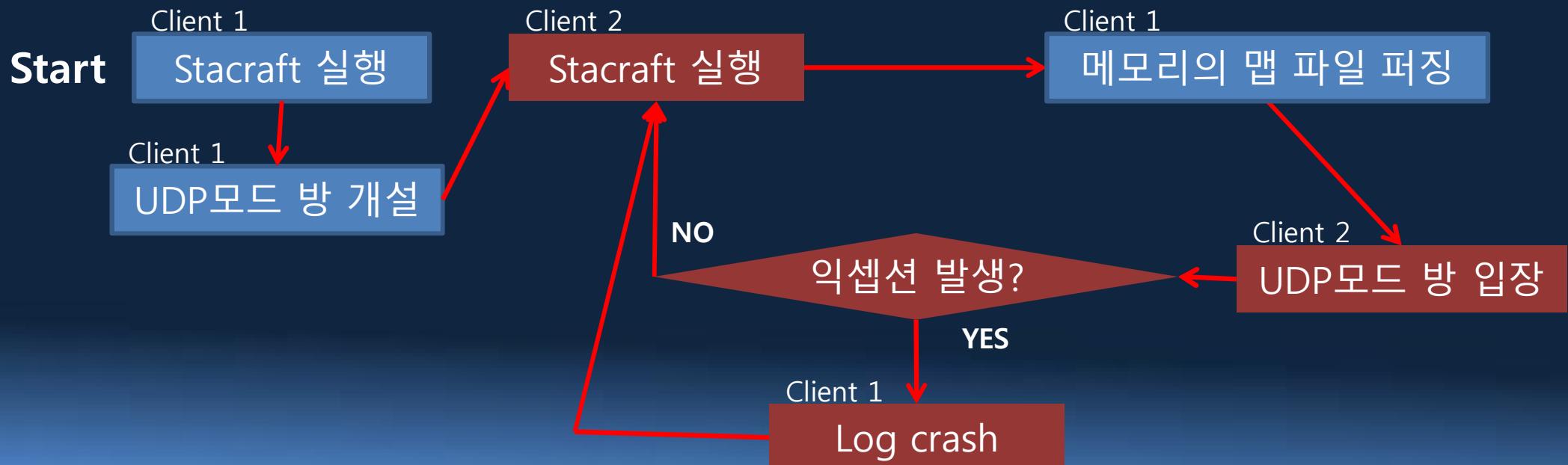
Fuzzing scenario

Scenario 2.

- 2개의 Client 필요

Client 1 : Starcraft를 실행하여 UDP 모드로 방을 개설하고, 스크립트가 Starcraft 프로세스의 메모리에 있는 맵 파일을 조작함을 반복

Client 2 : 스크립트가 Starcraft를 실행하여 UDP 모드에서 Client (1)이 개설한 방에 참가함을 반복 -> Exception 발생여부 확인



3. Fuzzing Starcraft

Fuzzing scenario

Scenario 2.

장점

- 타 플레이어가 방에 접속하는 상황에서 퍼징할 수 있음.
- 방을 만드는 Client는 메모리에 있는 맵 파일을 조작하기 때문에, 방을 한번만 열어두면 됨.

단점

- 구현이 어려움.
- 구현이 힘듬.
- 구현이 짜증남.

3. Fuzzing Starcraft

Fuzzing scenario

취약점 발생 시점	방 접속 시 취약점 유발	방 접속 후 게임 시작 시 취약점 유발
압축해제 전	①	②
압축 해제 후	③	④

3. Fuzzing Starcraft

①, ③ 압축 해제 전, 후 + 게임 시작 전



3. Fuzzing Starcraft

①, ③ 압축 해제 전, 후 + 게임 시작 전

Scenario 3 적용

동작

- 1) Client 1 : UDP 모드로 방 개설
- 2) Client 1 : 메모리에 올라와 있는 맵 파일 조작 (압축 해제 전/후)
- 3) Client 2 : UDP 모드에서 Client 1이 개설한 방 입장
- 4) Client 1 : Client 2에게 맵 파일 전송
- 5) Client 2 : 익셉션 발생여부 확인

3. Fuzzing Starcraft

①, ③ 압축 해제 전, 후 + 게임 시작 전

Scenario 3 적용

구현

Client 1 : Immunity debugger의 pycommand 이용 (Memory read/write)

```
def mutate(imm, loop, ptr, size):
    mutate_db = {}

    for i in range((size//100)*5):
        rand_idx = random.randrange(0x20, size)
        rand_val = random.randrange(0x0, 0x100)
        mutate_db[rand_idx] = chr(rand_val)
        imm.writeMemory(ptr+rand_idx, chr(rand_val))

    return mutate_db
```

Client 2 : 키보드 단축키를 이용해 매크로 형식으로 구현

```
sendmsg(c_void_p(self.hWnd), 0x0102, ord("m"), 0)
time.sleep(1)
sendmsg(c_void_p(self.hWnd), 0x0102, ord("e"), 0)
time.sleep(3)
for i in range(0,5):
    sendmsg(c_void_p(self.hWnd), 0x0100, 0x28, 0
time.sleep(0.5)
sendmsg(c_void_p(self.hWnd), 0x0102, ord("o"), 0)
time.sleep(1.5)
sendmsg(c_void_p(self.hWnd), 0x0102, ord("o"), 0)
time.sleep(1.5)
sendmsg(c_void_p(self.hWnd), 0x0102, ord("g"), 0)
```

3. Fuzzing Starcraft

Sub problems

Problem 1) MPQ 압축 포맷을 자동으로 decompress 시켜야 함.

=> 외국에서 MPQ 포맷 파싱을 위해 만들어둔 stormlib라는 오픈소스 라이브러리 이용.

```
9    TCHAR buf[256] = {0,};
10   HANDLE hMpq=0, hFile=0;
11   LPSTR lpFileData;
12   DWORD dwFileSize, dwByteRead, dwTmp;
13   const TCHAR* name = _T("C:\\mylogs\\rwqqwqrqwr.scm");
14
15   SFileOpenArchive(name, 0, 0, &hMpq);
16   MultiByteToWideChar(CP_ACP, MB_PRECOMPOSED, argv[1], 256, buf, 256);
17   SFileAddFileEx(hMpq, (const TCHAR*)buf, argv[1], MPQ_FILE_IMPLODE, NULL, NULL);
18
19   SFileCompactArchive(hMpq, NULL, NULL);
20   SFileCloseArchive(hMpq);
21   return 0;
```

3. Fuzzing Starcraft

Sub problems

Problem 2) UDP 모드로 게임 목록 확인 후 10초가 지나면 더 이상 같은 게임이 검색되지 않음.

=> 직접 udp로 broadcast 패킷을 전송하여 해결

```
1  from socket import *
2  import time
3
4  f = open("data", "rb")
5  data = f.read()
6  f.close()
7
8  s = socket(AF_INET, SOCK_DGRAM)
9  s.setsockopt(SOL_SOCKET, SO_REUSEADDR, 1)
10 s.bind(("0.0.0.0", 6333))
11 while 1:
12     s.sendto(data, ("192.168.123.104", 6111))
13     time.sleep(2)
14 s.close()
```

3. Fuzzing Starcraft

Sub problems

Problem 3) 방장이 클라이언트에게 맵을 전송할 때, 체크섬과 같이 전송함.
=> Starcraft를 분석하여 checksum생성루틴 찾아내 구현.

```
xor_table = [0, 1996959894, 3993919788L,
def get_checksum(data):
    global xor_table
    a = 0xffffffff
    for i in range(0, len(data)):
        tmp = ord(data[i])
        tmp = tmp ^ a
        a = a >> 8
        tmp = tmp & 0xff
        tmp = xor_table[tmp]
        tmp = tmp ^ a
        tmp = tmp % 0x100000000
        a = tmp
    return a
```

3. Fuzzing Starcraft

② 압축 해제 전 + 게임 시작 후

압축을 해제 하기 전에 맵을 조작 시

- 1) 방을 개설하다가 익셉션이 발생
- 2) 방을 개설/입장하자마자 익셉션이 발생
⇒ 게임 시작 자체가 불가능

따라서 Scenario 2 제외.



3. Fuzzing Starcraft

④ 압축 해제 후 + 게임 시작 후



3. Fuzzing Starcraft

④ 압축 해제 후 + 게임 시작 후

Scenario 1 적용

동작

- 1) 파일 조작
- 2) Starcraft 실행 -> 방 개설
- 3) 게임 시작
- 4) 익셉션 발생여부 확인

구현

- 키보드 단축키를 이용해 매크로 형식으로 구현 (1, 3번 케이스와 같은 방식)

```
sendmsg(c_void_p(self.hWnd), 0x0102, ord("m"), 0)
time.sleep(1)
sendmsg(c_void_p(self.hWnd), 0x0102, ord("e"), 0)
time.sleep(3)
for i in range(0,5):
    sendmsg(c_void_p(self.hWnd), 0x0100, 0x28, 0
time.sleep(0.5)
sendmsg(c_void_p(self.hWnd), 0x0102, ord("o"), 0)
time.sleep(1.5)
sendmsg(c_void_p(self.hWnd), 0x0102, ord("o"), 0)
time.sleep(1.5)
sendmsg(c_void_p(self.hWnd), 0x0102, ord("g"), 0)
```

3. Fuzzing Starcraft

Result

- Fuzzing scenario (4)에서 27종류의 unique crash획득
(다른 시나리오에서도 crash는 발생했지만 종류가 다양하지 않으며
exploitable하지 않음)
- 이 중 2종류는 Exploitable하다고 판단 및 Exploit 작성
(나머지는 exploitable하지 않거나 분석 중)

받은편지함	Fuzzer notify ..	ected ! The file saved as C:\mylogs\ci
받은편지함	Fuzzer notify ..	n detected ! The file saved as C:\myc
받은편지함	Fuzzer notify ..	tion detected ! The file saved as C:\rr
받은편지함	Fuzzer notify ..	- Exception detected ! The file saved a
받은편지함	Fuzzer notify ..	tion detected ! The file saved as C:\my
받은편지함	Fuzzer notify ..	:option detected ! The file saved as C:\v
받은편지함	Fuzzer notify ..	the file saved as C:\mylogs\cr
받은편지함	Fuzzer notify ..	! ! ! The file saved as C:\mylogs\cra
받은편지함	Fuzzer notify ..	- Exception detected ! The file saved a
받은편지함	Fuzzer notify ..	ceted ! The file saved as C:\mylogs\cra
받은편지함	Fuzzer notify ..	Exception detected ! The file saved a:
받은편지함	Fuzzer notify ..	on detected ! The file saved as C:\my



3. Fuzzing Starcraft

Exploitable crashes

Case (1) ADD [edi*4 + DATA_TABLE], ebx

Range of EDI : 0x00000000 ~ 0x0000ffff (Controllable)

Range of EBX : 1 or -1 (Fixed)

- 임의의 주소에 ebx 값(1 or -1)을 더할 수 있는 취약점.
- > DATA_TABLE 메모리 뒤에 있는 함수포인터 값을 반복적으로 ADD하여 원하는 코드의 주소로 바꿔준 후, 함수포인터가 호출되도록 하여 공략.

Case (2) CALL [edx* 4 + FUNC_TABLE]

Range of EDX : 0x00 ~ 0xff (Controllable)

- FUNC_TABLE은 0x3c * 4의 크기를 가지며, 그 뒤로는 쓰레기 값이 들어가 있음.
- EDX를 0x3d~ 0xff 사이의 값으로 넣어주면, 쓰레기 값을 코드 주소로서 호출할 수 있음.
- > 쓰레기 값이 나타내는 주소에 쓸만한 가젯이 있거나, 맵 파일의 데이터가 있다면 best case.
- > 아니라면 Heap spray로 해당 메모리를 할당시켜 공략.

Chapter 4: Exploitation

- 1) Heap spray**
- 2) Process continuation**

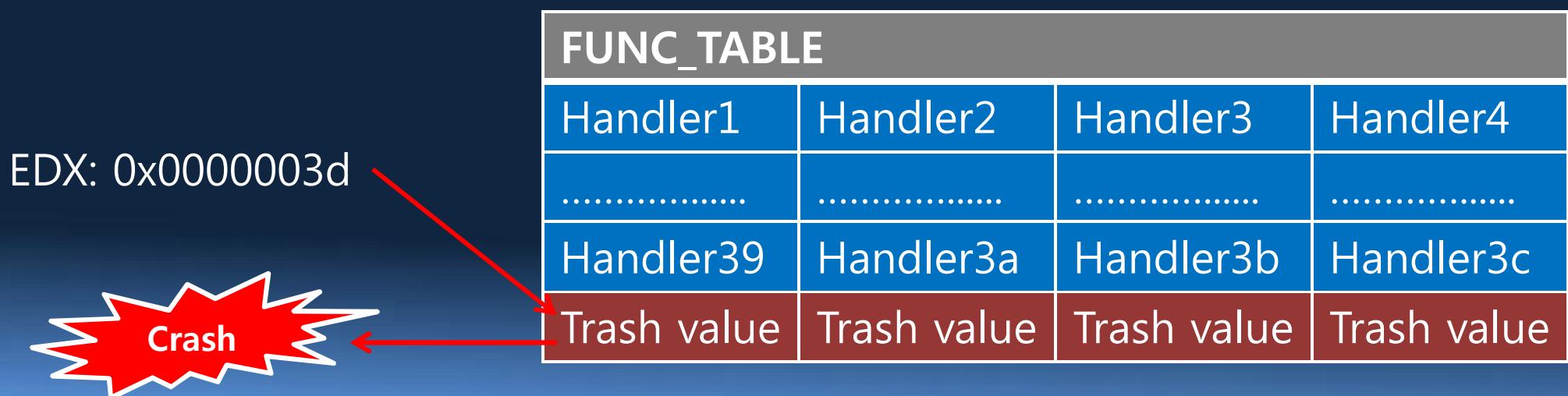
4. Exploitation

How to exploit?

Crash case (2)

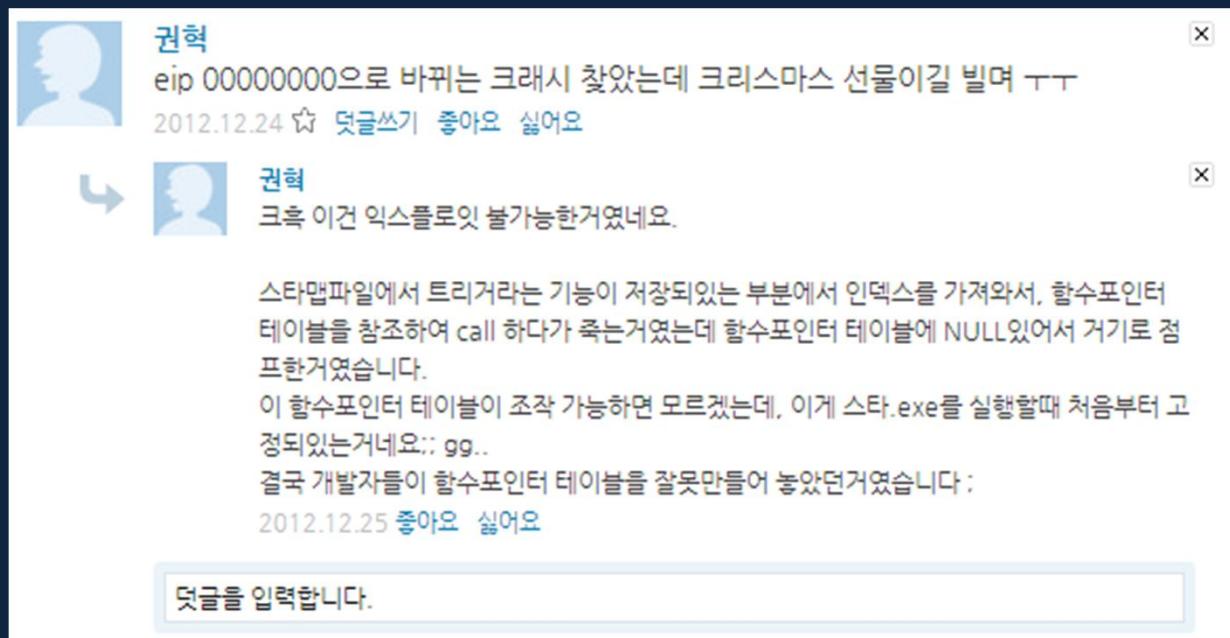
CALL [edx* 4 + FUNC_TABLE]

- 기존 FUNC_TABLE의 크기는 $0x3c * 4$ 이지만, edx의 크기를 제한하지 않음.
- edx 값을 $0x3c < idx \leq 0xff$ 의 범위로 만들어주면 FUNC_TABLE에 있는 정상 함수포인터가 아닌, 쓰레기 값을 호출하다가 crash 발생.



4. Exploitation

It can not be exploited.. 12/12/24



4. Exploitation

It can be exploited!! 13/02/02

권혁
eip 00000000으로 바꾸는 크래시 찾았는데 크리스마스 선물이길 빌며 ㅜㅜ
2012.12.24 ☆ 덧글쓰기 좋아요 싫어요

↳ 권혁
크큭 이건 익스플로잇 불가능한거였네요.

스타맵파일에서 트리거라는 기능이 저장되있는 부분에서 인덱스를 가져와서, 할수포인터
테이블을 참조하여 call 하다가 죽는거였는데 할수포인터 테이블에 NULL있어서 거기로 점
프한거였습니다.
이 할수포인터 테이블이 조작 가능하면 모르겠는데, 이게 스타.exe를 실행할때 처음부터 고
정되있는거네요;; gg..
결국 개발자들이 할수포인터 테이블을 잘못만들어 놓았던거였습니다;
2012.12.25 좋아요 싫어요

덧글을 입력합니다.

권혁
스타 계산기 떳네요 ㅋㅋㅋㅋ 아 이제서야 뜨다니 -_- 암울하네요 ㅜㅜ ㅋㅋㅋ
원래 exploit불가능하다고 생각해서 버려뒀던 취약점 있었는데 준비형수업듣고 힙스
프레이 하는 방법 더 찾아봤는데 결국 힙스프레이로 공략 성공했습니다 ㅋㅋㅋ 감사
합니다 있다가 영상을릴게여 ㅋㅋ
2013.02.02 ☆ 덧글쓰기 좋아요 싫어요

4. Exploitation

Heap spray

1) What is heap spray?

- One of exploitation technique
- Heap에 nop와 쉘코드를 spray하듯이 엄청나게 뿌려서 원하는 메모리에 쉘코드를 할당시키는 기술.



4. Exploitation

Heap spray

2) How to heap spray at Starcraft Broodwar?

- Browser나 Flash처럼 script를 사용할 수 없음.
- 맵 파일 데이터를 무작정 넣는다고 해서 그대로 메모리에 올라가지 않음.

→ 'STR' section in scenario.chk

- STR (String) section : 맵에서 필요로 하는 문자열을 저장해두는 section
- Starcraft는 scenario.chk파일의 STR section을 파싱할 때 STR section의 size를 제한하거나 검사하지 않음.
- STR section에 있는 값은 heap메모리에 그대로 복사됨.

Offset(h)	00	01	02	03	04	05	06	07	08	09	0A	0B	0C	0D	0E	0F
000313F0	FF	FF	FF	FF	FF	53	54	52	20	E6	19	00	00	00	04	ÿÿÿÿÿÿSTR æ....
00031400	03	08	15	08	2A	08	52	08	92	08	B2	08	BA	08	1F	09
00031410	86	09	8B	09	9A	09	ED	09	02	0A	1D	0A	2F	0A	54	0A
00031420	85	0A	99	0A	C9	0B	ED	0B	2E	0C	56	0C	6B	0C	81	0C
00031430	8D	0C	97	0C	EE	0D	1C	0F	5C	0F	8C	10	BD	10	DE	10
00031440	0E	12	2D	12	6D	12	AE	12	F2	12	3E	13	62	13	73	13
00031450	9D	13	CB	14	EB	14	F6	14	37	15	BC	15	19	16	87	16
00031460	93	16	D5	16	04	18	12	18	51	18	59	18	5D	18	61	18
00031470	65	18	69	18	6D	18	71	18	75	18	79	18	7D	18	81	18
00031480	85	18	89	18	8D	18	91	18	95	18	99	18	9D	18	A1	18
00031490	A5	18	A9	18	AD	18	1B	18	B5	18	B9	18	BD	18	C1	18
000314A0	C5	18	C9	18	CD	18	D1	18	D5	18	D9	18	DD	18	E1	18
000314B0	E5	18	E9	18	ED	18	F1	18	F5	18	F9	18	FD	18	01	19
000314C0	05	19	09	19	OD	19	11	19	15	19	19	19	1D	19	21	19
000314D0	25	19	29	19	2D	19	31	19	35	19	39	19	3D	19	41	19
000314E0	45	19	49	19	4D	19	51	19	55	19	5E	19	62	19	66	19
000314F0	6A	19	6E	19	72	19	76	19	7A	19	7E	19	82	19	86	19
00031500	8A	19	8E	19	92	19	96	19	9A	19	9E	19	A2	19	A6	19
00031510	AA	19	AE	19	B2	19	B6	19	BA	19	BE	19	C2	19	C6	19
00031520	CA	19	CE	19	D2	19	D6	19	DA	19	DE	19	E2	19	02	08
00031530	00	46	72	65	64	6F	6D	20	28	53	2E	45	2E	45	2E	.Freedom (S.E.E.
00031540	44	29	00	07	20	C7	C1	B7	CE	BA	EA	20	32	BO	B3	20
00031550	C7	CA	BF	E4	C7	D4	2E	00	20	20	04	44	72	6F	70	73
00031560	68	69	70	20	3A	20	22	43	61	6E	20	49	20	74	61	6B
00031570	65	20	79	6F	75	72	20	6F	72	64	65	72	3F	22	20	00

çéjàçô.. .Drops
hip : "Can I tak
e your order?" .

4. Exploitation

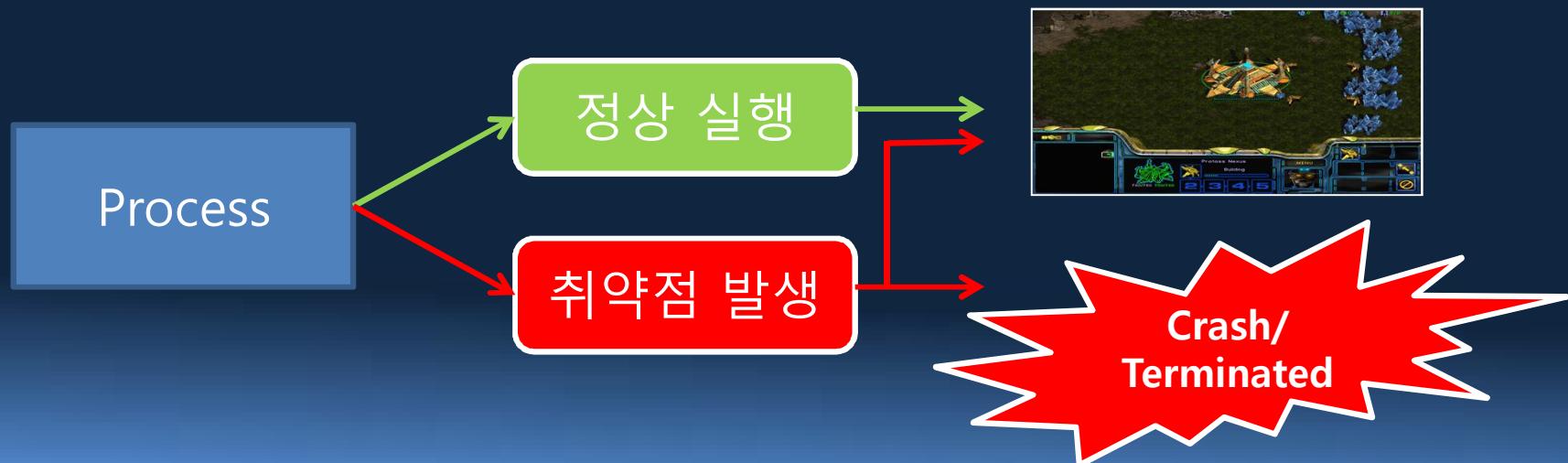
DEMO



4. Exploitation

Advanced Exploitation – Process continuation

- 일반적으로 임의의 코드를 실행하는 Exploit을 작성했을 때, 코드의 실행이 끝나면 해당 process는 종료되거나, 크래시로 이어짐.
- 하지만 취약점이 발생하지 않았을 때의 환경(Stack, Register, ...)을 복구해 주어 Process가 종료되지 않고 정상적으로 Continue 되도록 해주는 것을 Process continuation이라 부름.
- 취약점의 유형이나 상황마다 Process continuation을 구현하는 방법은 다양함.



4. Exploitation

Advanced Exploitation – Process continuation

`CALL FUNC_TABLE [idx]`

- 이 경우에선 취약점의 특성상 **process continuation**을 구현하기가 굉장히 간단함.
 - 취약점은 함수 포인터 테이블에서 잘못된 인덱스에 있는 함수를 호출함으로써 발생함.
- ⇒ 즉, 원래 정상적인 인덱스를 참조했을 때에 반환되는 레지스터/메모리 상태를 만들어주고 `return`하면 정상 프로세스 흐름을 이어갈 수 있음.
(하지만 운이 좋게도 레지스터 상태를 함수 호출 전과 똑같이 유지한 채 `return`해도 정상 프로세스 흐름을 이어감.)

4. Exploitation

Advanced Exploitation – Process continuation

- 1) PUSHAD와 같은 명령으로 레지스터 상태를 보존
- 2) 목표하는 헬코드 실행
 - 단, 스택 아래에 있는 값을 변경하거나 스택 외에 메모리에 있는 값을 변경하면 안됨.
- 3) POPAD로 레지스터 상태를 함수 호출 직후로 복구

60 31C9	PUSHAD XOR ECX, ECX	
Shellcode		
(ESP 밑에 있는 값을 변경하지 않아야함)		
2E:61 C3	POPAD RETN	Superfluous prefix

4. Exploitation

DEMO



Chapter 5: Conclusion

- 1) Timeline**
- 2) I have talked about**
- 3) Q & A**

5. Conclusion

Timeline

- **2013/12/24**
 - Exploitable crash 발견 & 취약점 원인 분석 완료
(but, Heap spray할 방법 모름)
- **2013/02/02**
 - Heap spray할 방법 찾음 -> Exploit 작성 완료
- **2013/02/15**
 - Process continuation 성공
- **2013/02/27**
 - 취약점 분석 보고서 작성 & Blizzard 취약점 신고 담당 메일로 패치 권고.
 - ~ 2013/05/04 응답 없음
- **2013/05/05**
 - 취약점 패치 재권고
 - ~2013/07/13 응답 없음
- ~ **현재**
 - 응답 없음

5. Conclusion

I have talked about ..

- Multiplayer 게임들의 공격 벡터
- 맵 파일을 공격 벡터로 하여 취약점 점검하기
- Starcraft 공략 story
- Heap spray
- 매우 간단한 process continuation

5. Conclusion

Q & A



Thank you!

www.CodeEngn.com | www.RCEHub.com

Basic Reversing

[+] website hacking & security



Kim seul-gi · CherishCat
<http://hack-me.org/>

목

차

- 0x01 소개

- » CherishCat 소개
- » Hack-Me.org 사이트 소개

- 0x02 Basic Reversing

- » 기초적인 이야기

- 0x03 재미있게 공부하려면?

- » How ?

목

차

- 0x04 Reversing 문제풀이
 - » Easy Reverse Engineering
- 0x05 [+] web site 취약점
 - » URL변조 & XSS (+software)
- 0x06 END
 - » Quiz Time
 - » 느낀점? Q&A!
 - » Bye Bye .

0x01

소개

About me

CherishCat

컴퓨터
공학

보안

슈르깅

LOL

About me



The screenshot shows a dark-themed web application interface. At the top, there's a navigation bar with links: Home, About, Challenges, Ranking, Board, Chat, Crack, Research, Links, and Logout. Below the navigation is a user profile section featuring a blue cat logo and the text "11.09.08". A horizontal menu bar below the profile lists categories: Crypto(10), Stegano(6), Web(9), Network(2), Binary(6), Reverse(6), and Forensic(1). The main content area displays a table of challenges. The table has columns for ID, Title, Author, Score, Solvers, Age, Questions, and Answers. There are 23 challenges listed:

#	Title	Author	Score	Solvers	Age	Questions	Answers
1	View Sourcecode	by Cheatsheat	100	2232	10/21/7	?	1
2	Easy Steganography	by Cheatsheat	200	742	10/21/7	?	1
3	MP3 file tags Test	by Cheatsheat	100	233	10/21/0	?	1
4	Extract file from pswp	by Cheatsheat	100	371	10/21/0	?	1
5	ZIP Password Recovery	by Cheatsheat	100	243	10/21/0	?	1
6	Decode strange file	by Cheatsheat	200	295	10/21/0	?	1
7	All Roads lead to Rome	by Cheatsheat	100	239	10/20/9	?	1
8	Breaks like computer	by Cheatsheat	300	116	10/20/9	?	1
9	Pass MySQL root	by Cheatsheat	100	171	10/20/8	?	1
10	Bentzen star collision	by Cheatsheat	200	45	10/20/8	?	1
11	Save the U boat	by Cheatsheat	200	179	10/09/7	?	1
12	I Still Know Where You Been	by Blackbird	100	231	10/09/4	?	1
13	Server Web Shell Client	by Cheatsheat	300	36	10/08/5	?	1
14	Assassins: Fishery	by Blackbird	200	29	10/07/1	?	1
15	Easy Cryptography	by Cheatsheat	100	468	10/06/5	?	1
16	Polyalphabetic substitution	by Cheatsheat	200	124	10/06/5	?	1
17	Apache 1.1.2 Mac OS X	by Blackbird	200	67	10/06/3	?	1
18	I want to play a game	by Cheatsheat	200	100	10/06/0	?	1
19	Read by Apel on road	by Blackbird	800	0	10/01/6	?	1
20	View Sourcecode 2	by Cheatsheat	100	361	10/01/3	?	1
21	Let's play... for game	by Blackbird	600	38	99/96	?	1
22	The meaning of life game	by Blackbird	200	5	9/08/9	?	1
23	The meaning of life game	by Blackbird	200	5	9/08/9	?	1

<http://hack-me.org/>



Register now! :)

About me



Hack-me 는 자신의 해킹실력을 테스트 해 볼 수 있는 사이트입니다 :)

The challenges cover a wide range of subjects: **network, cryptography, steganography, forensic, and reverse engineering.**

About me

<http://www.wechall.net/>

The screenshot shows a user profile for 'Inferno'. The bio text is as follows:

For the people not familiar with challenge sites, a challenge site is mainly a site focussed on offering computer-related problems. Users can register at such a site and start solving challenges. There exist lots of different challenge types. The most common ones are the following: **Cryptographic**, **Crackit**, **Steganography**, **Programming**, **Logic** and **Math/Science**. The difficulty of these challenges vary as well.

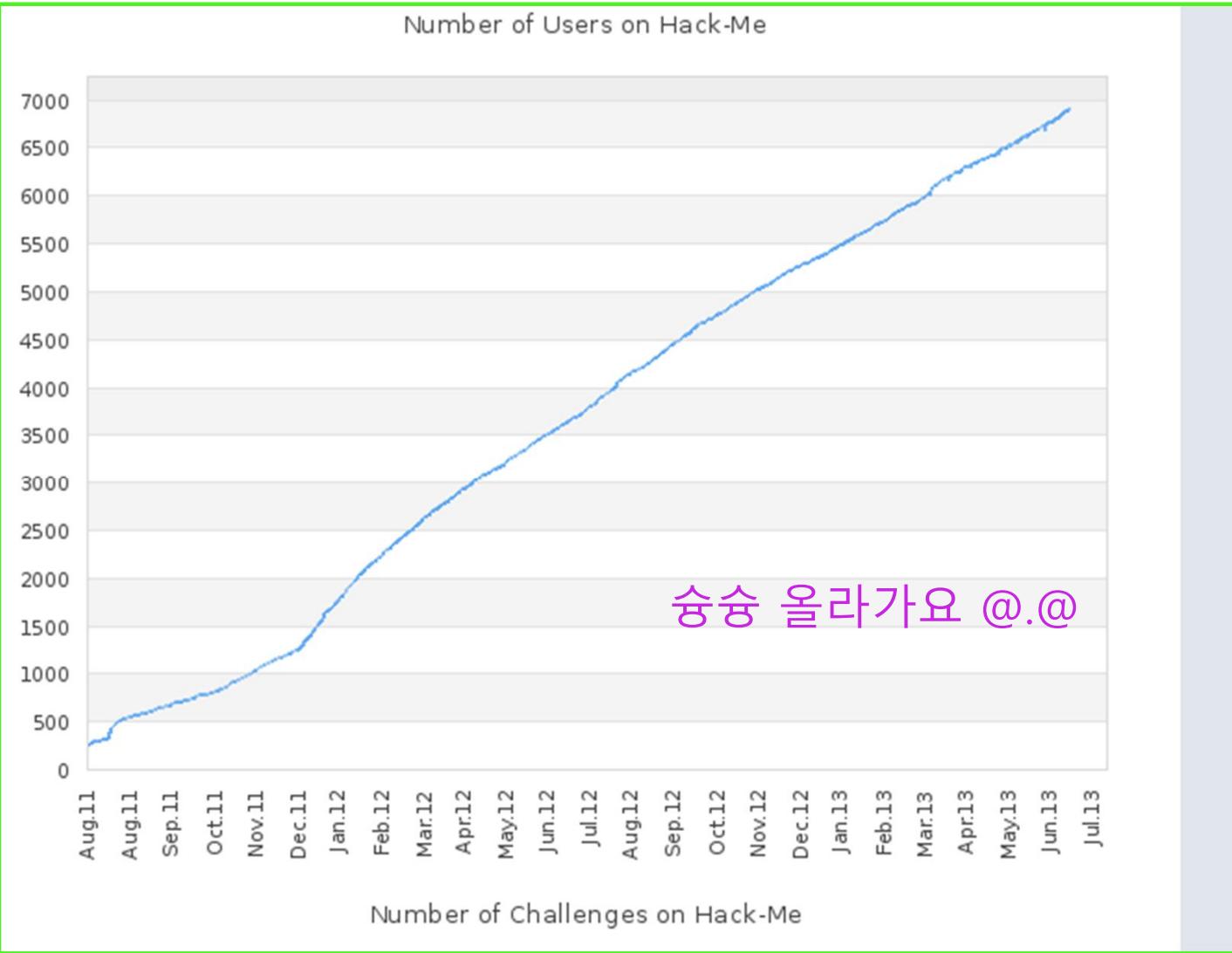
When I first started out I was familiar with working with a computer, but that was about it. In about 5 years I roamed the internet searching for different kind of challenge sites. I have played at many different sites, did a lot of challenges and I am still learning new (and exciting) stuff.

I also found an IRC network where most of the bigger challenge sites have their own IRC channel. These channels often have bots allowing users to check how other people are ranked at a certain site, or sites. From time to time the screens are filled with these simple requests. That is the reason I started coding this site. It allows the users to get a complete overview of a users challenge history on every site he signed up for. Now how does this work exactly?

A user on this site can indicate which challenge sites he plays at with a specific username. Challenge sites can be selected from a simple drop-down list. His profile page will then contain an entry for this specific challenge site with the given username, along with some text describing how high he is ranked

	Hack-Me
Sites origin country	
Language	Korean
Category Tags	Crypto , Cracking , Exploit
Site Admins	cherishcat
Auto update	no
Has OnSiteRank	yes
WarBoxes	0
Score	20472
Base-Score	10000
Users	6900
Challs	50
Linked users	142
Average	18.03%
Difficulty	75.00%
Enjoyment	75.00%
IRC Contact	not exist 

About me



0x02

Basic Reversing

Basic Reversing

Reverse

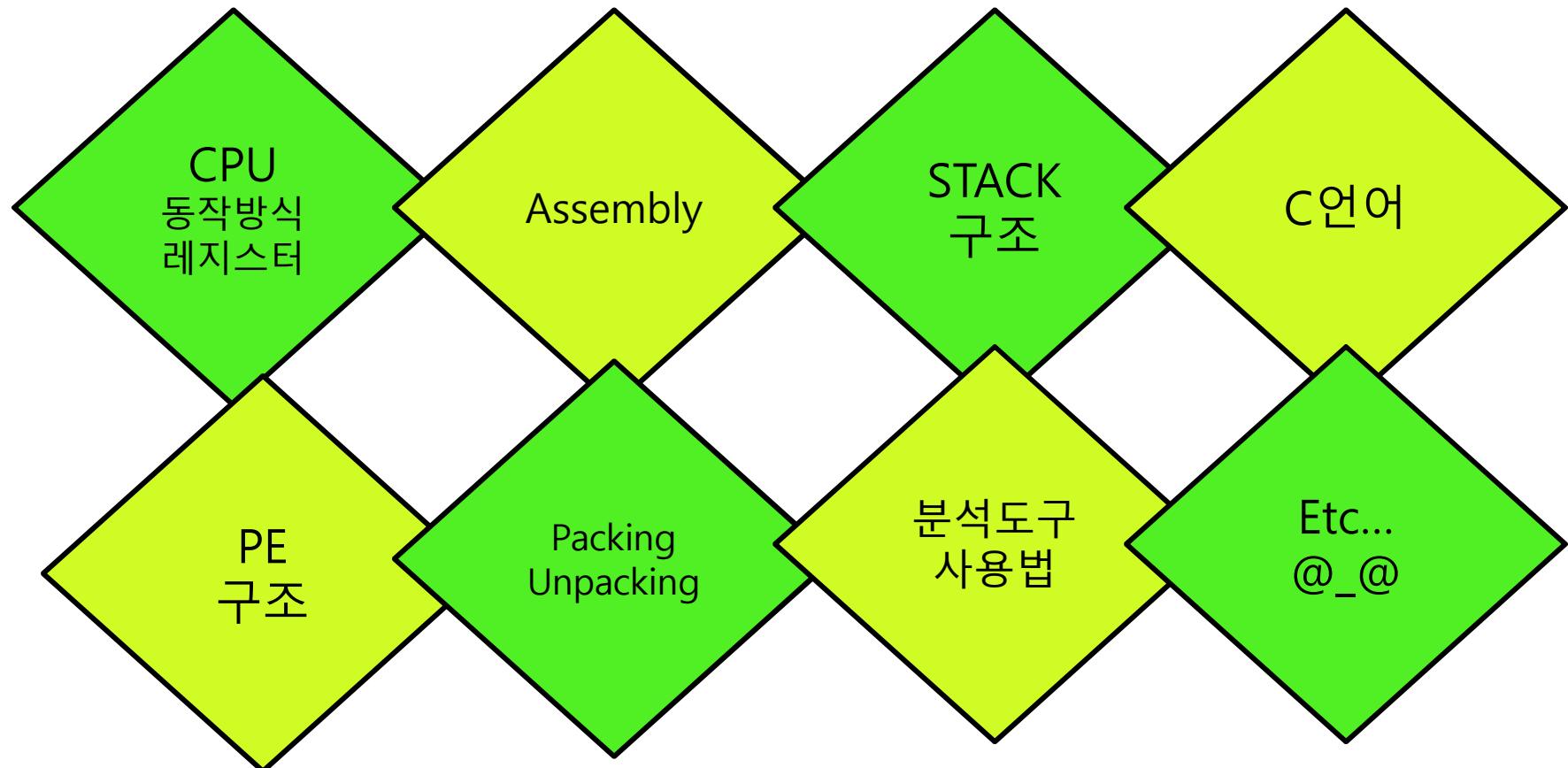
+

Engineering

= Reverse Engineering

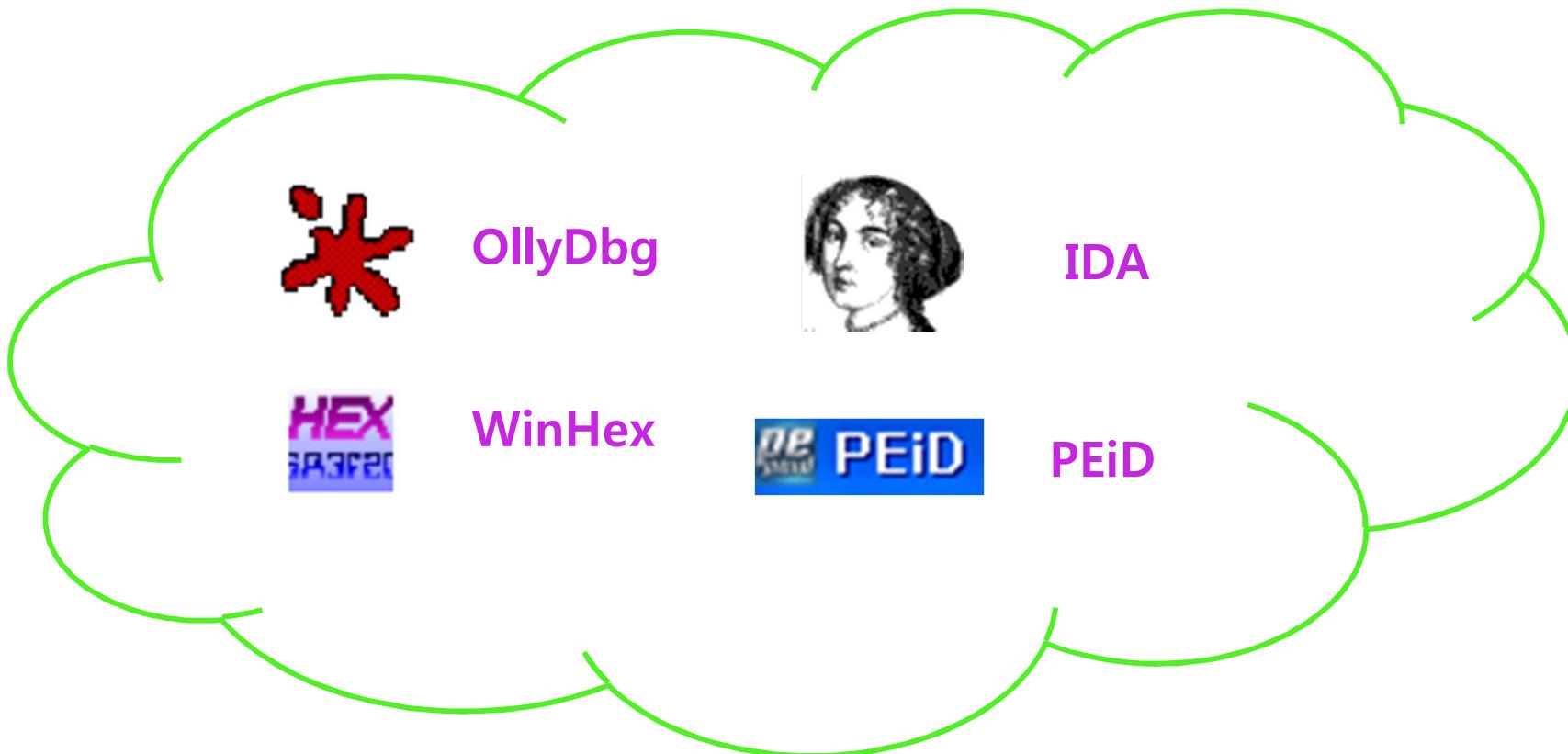
- 시스템의 구조나 기능 분석
- 기술적인 원리 분석
- 어떻게 작동되는가?
- 악성코드 분석
- 안전성 테스트
- 개발 효율성 증가

Basic Reversing



리버싱을 공부하기 위한 과정

Basic Reversing



Debugger / Disassembler / File Analyzer / HexEditor

0x03

재미있게 공부하려면?

How to study excitingly



How to study excitingly

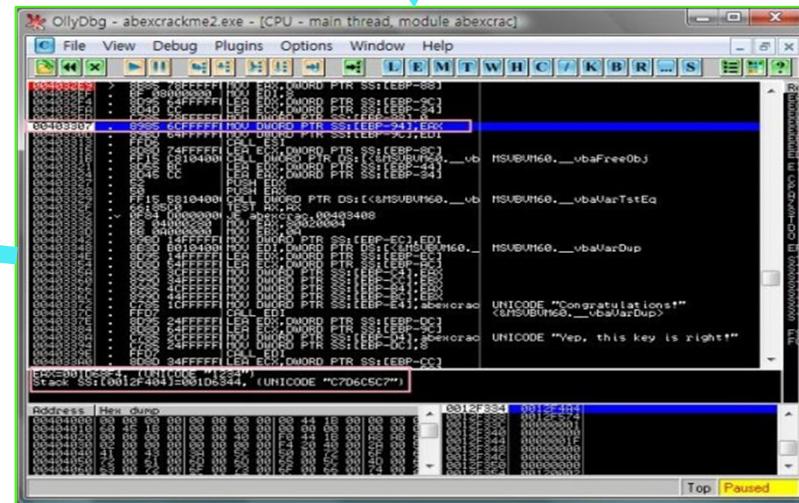
Crack Me = CM

The screenshot shows the homepage of Crackmes.de. At the top, there is a navigation bar with links for Home, Latest, Search archive, Register, and FAQ. Below the navigation bar, there is a login form titled "Reversers login here" with fields for username and password, and a "login" button. There are also links for "Forgot your password?" and "New user? Register here.". To the right of the login form, there is a section titled "Profile for abex". It displays a message stating "This account is locked." followed by the registration date "Registered: 05. Nov, 2001" and the last seen date "Last seen: 01. Jan, 1970". Below this, there is a section titled "User submissions" which lists several entries with dates, user names, and solution details. For example, on 19.09.2002, user abex4 solved Exhuman Scooby Doo Cipher thomas.idpz.net. Other submissions include abex2 solving Exhuman Cipher Tsongkie AttilhaZ thomas.idpz.net, abex3 solving Exhuman Cipher thomas.idpz.net, abex5 solving Hague Cipher Tsongkie thomas.idpz.net, and abex1 solving Hague Cipher barcode_ ShadowMaster Squajbob thomas.idpz.net.

<http://crackmes.de/users/abex>

CrackMe 는 Reverse Engineering 실력 테스트를 위해 만들어진 프로그램 :D

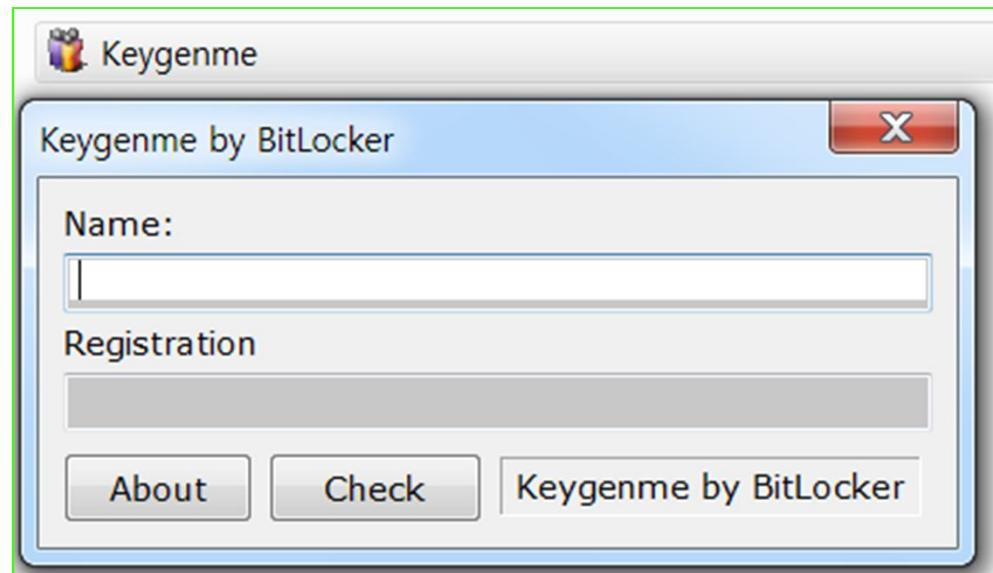
How to study excitingly



<http://llove94.blog.me/50110920747>

How to study excitingly

Keygen Me



```
[x][x] Babylon KeygenMe [x][x] coded by haiklr  
[x] Name : _
```

Keygenme 는 CrackMe와 비슷하게 Reverse Engineering 테스트를 위해 만들어진 프로그램 :D 다른점은 조금 더 난이도가 있고 [+]로 프로그래밍을 하여 시리얼 도출하는 문제가 많습니다 :D

How to study excitingly

```
void main(void)
{
unsigned short enc[] = { 0x0D8, 0x1C1, 0x189, 0x231, 0x1C9, 0xF9, 0x181, 0xF8, 0x189, 0x241 };
const char *name = "CherishCat";
int i;

for (i = 0; i < 10; i++)
{
unsigned char a = ((enc[i] + 900) ^ ((name[i] >> 5) | (name[i] << 3)));

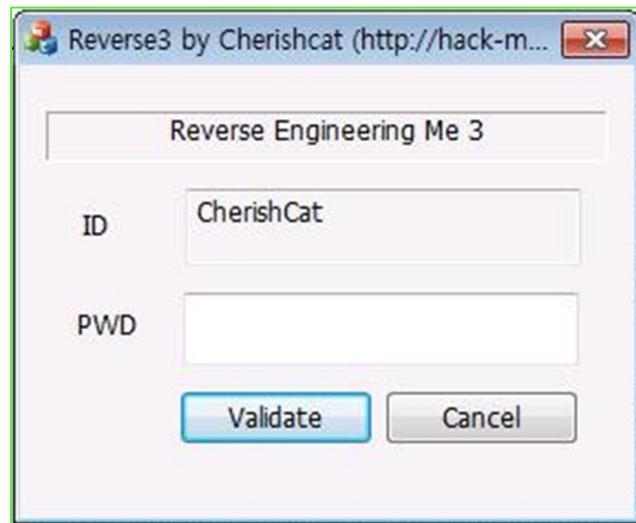
a = (a >> 5) | (a << 3);

printf("%c", a);
}
}
```

Keygen source 예시 :)

How to study excitingly

Make EXE by myself



직접 CreckMe를 만들어보자 :D Test 해보자 ! 그리고, 지인들에게 추천추천 :D

0x04

hack-me Reversing 문제풀이

Easy Reverse Engineering

Easy Rerverse Engineering

Description :

Do you know about Reverse Engineering?

Are you good at IDA?

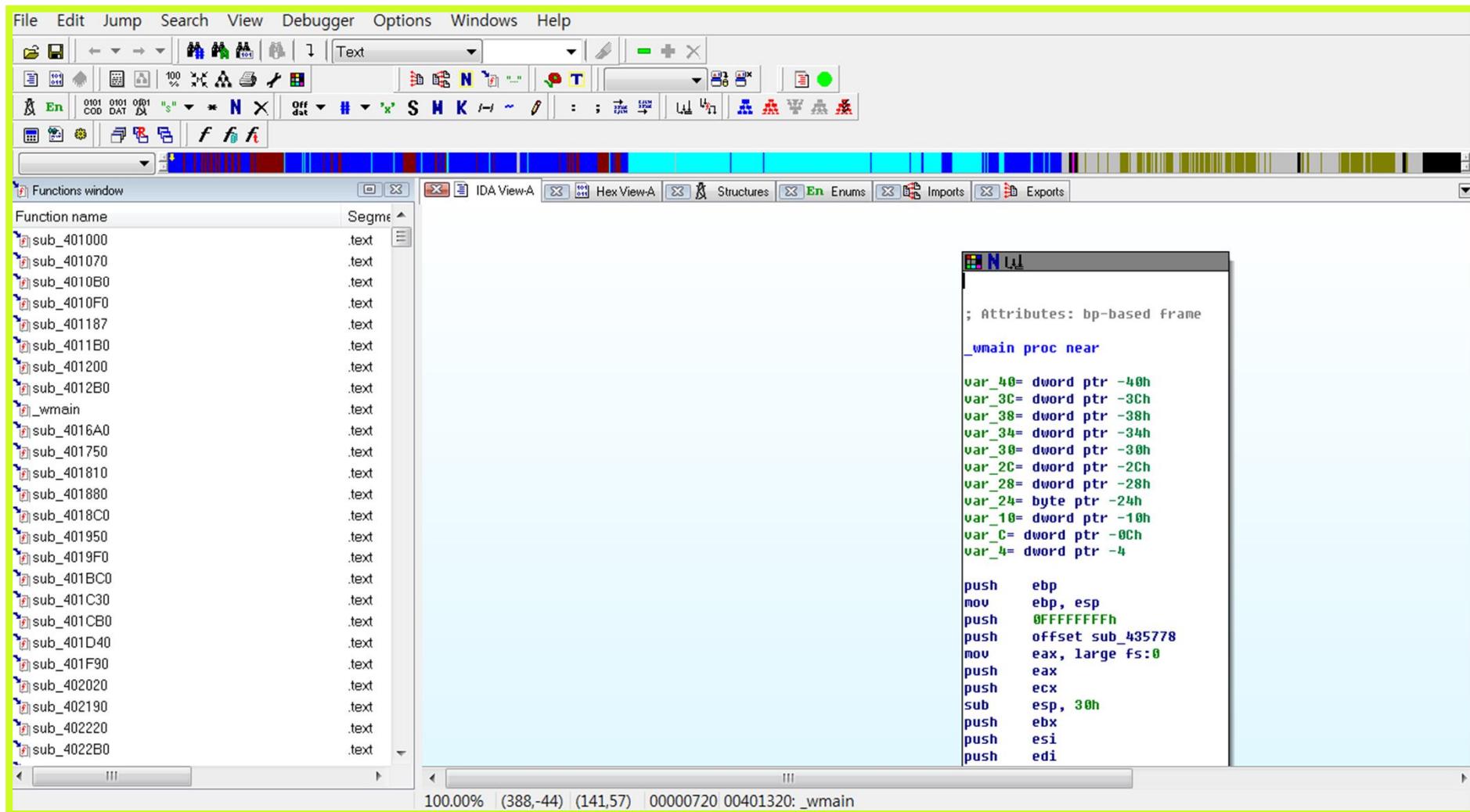
So, You can take this 350 points easily XD

Hint :

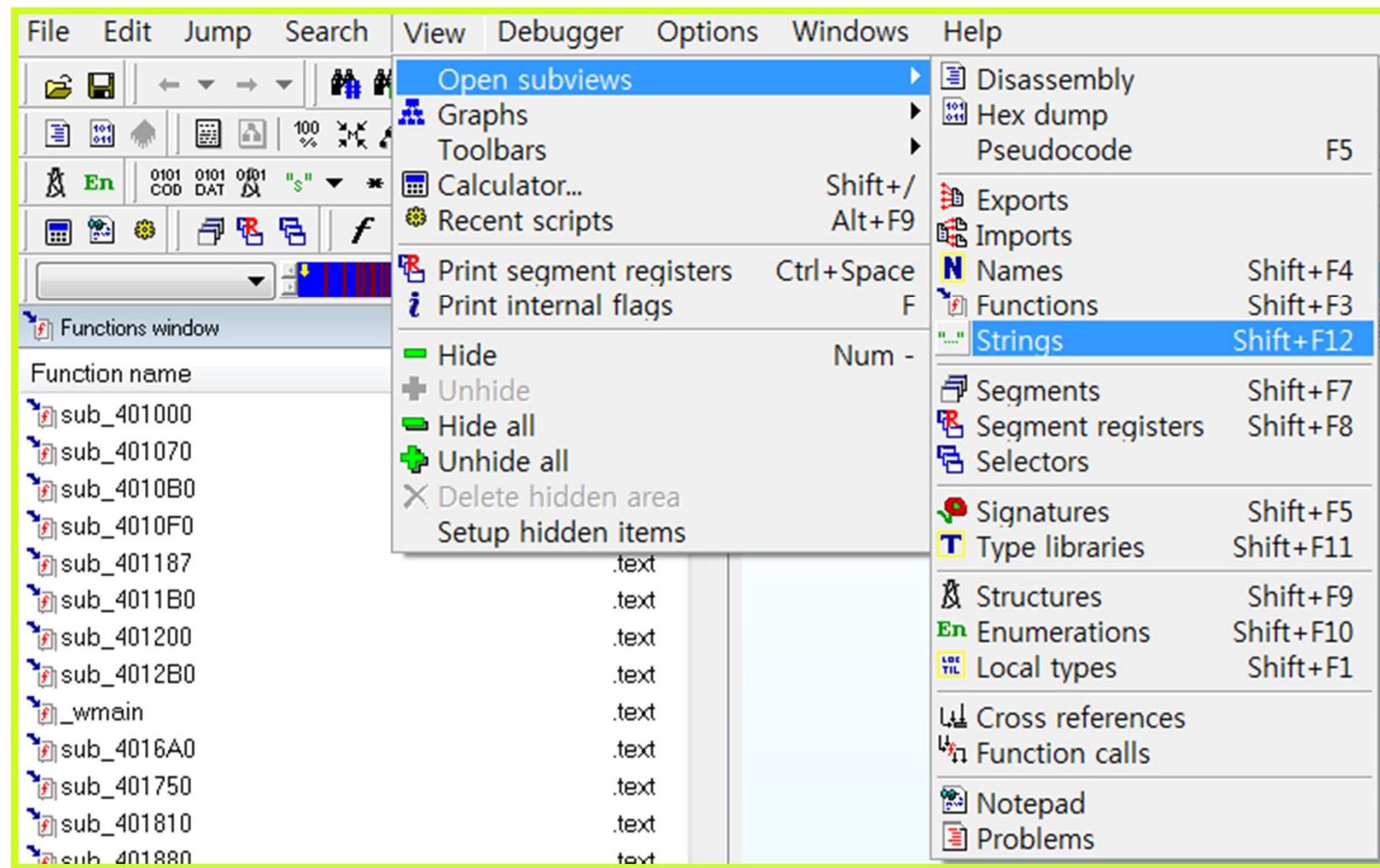
- 1) Not Packed
 - 2) Windows x86 binary
 - 3) Classical cipher(not modern)

H a c k m e
<http://hack-me.org>
Reverse Engineering
Basic training
by Bluebird

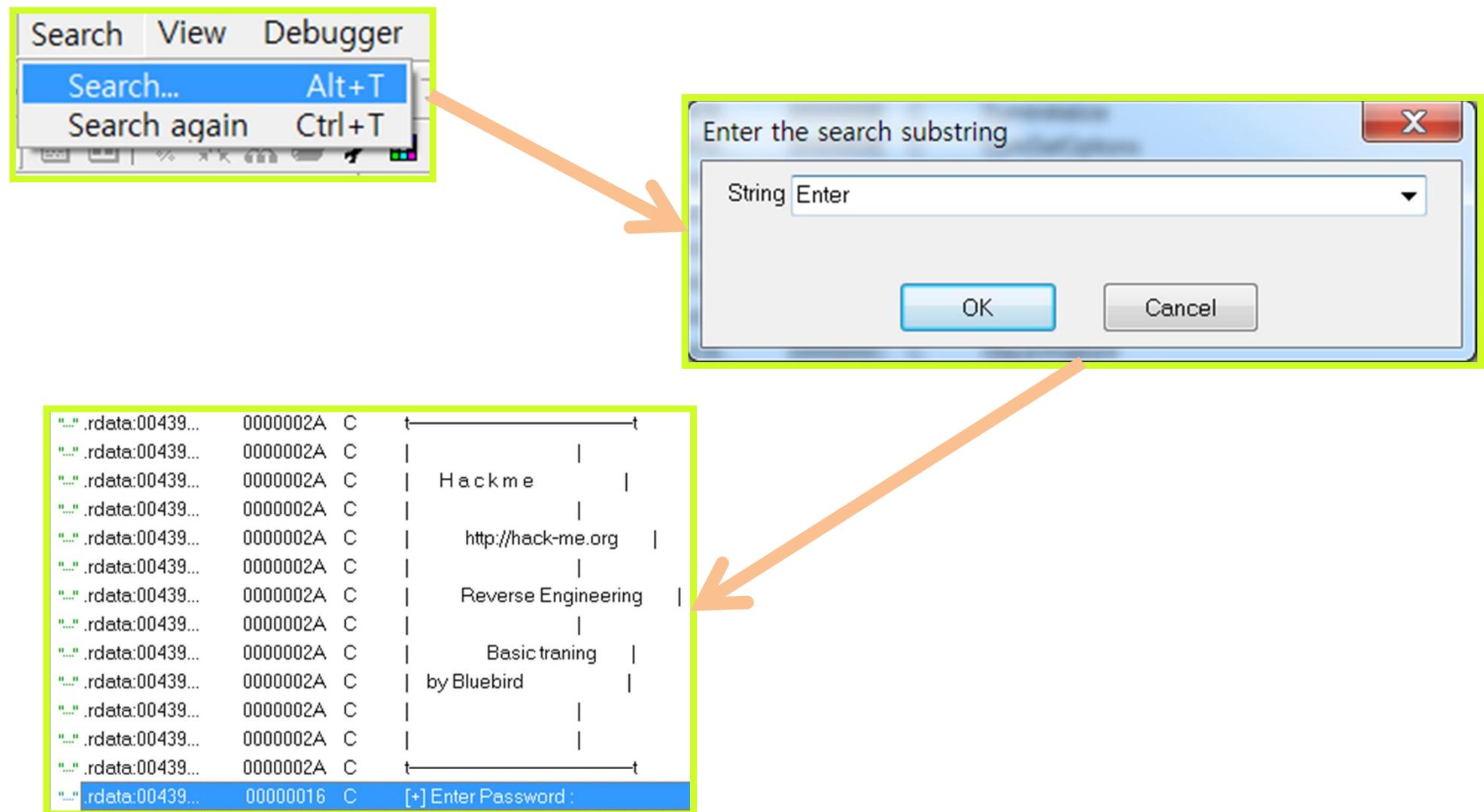
Easy Reverse Engineering



Easy Reverse Engineering



Easy Reverse Engineering



Easy Reverse Engineering

"...".rdata:00439...	0000002A	C		
"...".rdata:00439...	0000002A	C	Hack me	
"...".rdata:00439...	0000002A	C		
"...".rdata:00439...	0000002A	C	http://hack-me.org	
"...".rdata:00439...	0000002A	C		
"...".rdata:00439...	0000002A	C	Reverse Engineering	
"...".rdata:00439...	0000002A	C		
"...".rdata:00439...	0000002A	C	Basic training	
"...".rdata:00439...	0000002A	C	by Bluebird	
"...".rdata:00439...	0000002A	C		
"...".rdata:00439...	0000002A	C		
"...".rdata:00439...	0000002A	C	t—————t	
"...".rdata:00439...	00000016	C	[+] Enter Password :	
"...".rdata:00439...	0000001A	C	NtQueryInformationProcess	
"...".rdata:00439...	00000013	C	MapFileAndCheckSum	
"...".rdata:00439...	0000000F	C	Password is :	
"...".rdata:00439...	00000005	C	RSDSM	

Easy Reverse Engineering

The screenshot shows a debugger interface with two main windows. The top window displays assembly code:

```
    .rdata:004396A4          unicode 0, <1ImageNip.DLL>,0
    .rdata:004396BE          align 10h
    .rdata:004396C0 ; char aMapfileandchec[]
    .rdata:004396C0 aMapfileandchec db 'MapFileAndCheckSum',0 ; DATA XREF: sub_401810+25To
    .rdata:004396D3          align 4
    .rdata:004396D4 aPasswordIs      db 'Password is : ',0 ; DATA XREF: sub_401880+5To
    .rdata:004396E3          align 8
    .rdata:004396E8 __load_config_used dd 48h           ; Size
    .rdata:004396EC          dd 0                   ; Time stamp
    .rdata:004396F0          dw 2 dup(0)           ; Version: 0.0
```

The bottom window is a 'xrefs to aPasswordIs' dialog box:

Dire...	T.	Address	Text
Up	o	sub_401880+5	push offset aPasswordIs; "Password is : "

An orange arrow points to the 'OK' button at the bottom of the dialog box. The status bar at the bottom of the dialog box says 'Line 1 of 1'.

Easy Reverse Engineering

```
sub_10E1880 proc near
    mov     eax, ds:@cout@std@@3V?$basic_ostream@DU?$char_traits@D@std@@@1@A ; std::basic_ostream<char, std::
    push    offset aPasswordIs ; "Password is : "
    push    eax
    call    sub_10E19F0
    mov     ecx, ds:@cout@std@@3V?$basic_ostream@DU?$char_traits@D@std@@@1@A ; std::basic_ostream<char, std::
    push    offset Decrypted
    push    ecx
    call    sub_10E19F0
    mov     edx, ds:@endl@std@@YAAV?$basic_ostream@DU?$char_traits@D@std@@@1@AAV21@0Z ; std::endl(std::ba
    add    esp, 10h
    push    edx
    mov     ecx, eax
    call    ds:@?__6?$basic_ostream@DU?$char_traits@D@std@@@std@@QAEAAV01@P6AAAV01@AAV01@0Z@Z ; std::basic_ostream<
    retn
sub_10E1880 endp
```

Easy Reverse Engineering

The screenshot shows the IDA View-A window with assembly code. A yellow box highlights the assembly code area. A callout arrow points from the bottom of the highlighted area to the 'OK' button in a modal dialog titled 'xrefs to Decrypted'.

IDA View-A window:

```
65CB db 0
65CC dd 1 ; DATA XREF: __set_fpsr_sse2+1C↑r
65CC ; __set_fpsr_sse2+4C↑w
65D0 dd offset aBadAllocation_1 ; "bad allocation"
65D4 align 8
65D8 ; char Decrypted[]
65D8 Decrypted db 0E0h ; DATA XREF: sub_10E1750+4↑r
; sub_10E1750+D↑o ...
65D9 db 0F7h ; ÷
65DA db 0EAh ; è
65DB db 0C7h ; ç
65DC db 0F1h ; ñ
65DD db 0EBh ; è
65DE db 0C7h ; ç
```

xrefs to Decrypted dialog:

Direction	Type	Address	Text
Up	r	sub_10E1750+4	cmp Decrypted, 0
Up	o	sub_10E1750+D	mov eax, offset Decrypted
Up	w	sub_10E1750+A7	xor Decrypted[edi], al
Up	o	sub_10E1880+16	push offset Decrypted

Buttons at the bottom of the dialog: OK, Cancel, Search, Help.

Easy Reverse Engineering

```
char __cdecl sub_10E1750()
{
    signed int szLen; // ebp@1
    int p; // eax@2
    char result; // al@5
    signed int i; // edi@11
    char *v4; // eax@12
    unsigned int v5; // edx@12
    int szLen2; // ebx@12
    char v7; // cl@13

    szLen = 0;
    if ( Decrypted[0] )
    {
        p = (int)Decrypted;
        do
        {
            ++p;
            ++szLen;
        }
        while ( *(_BYTE *)p );
    }

    if ( v4 != &szUserInput[1] )
    {
        do
        {
            if ( !szUserInput[v5] )
                break;
            ++szLen2;
            ++v5;
        }
        while ( v5 < strlen(szUserInput) );
        Decrypted[i] ^= szLen2 + byte_1127669; | 
    }
    result = 1;
}
else
{
    result = 0;
}
return result;
```

XOR(exclusive or)

Easy Reverse Engineering

The screenshot shows the IDA Pro interface with the 'Hex View-A' tab selected. A context menu is open over a block of memory starting at address 01126648. The menu items are:

- Edit... F2
- Data format ▶
- Columns ▶
- Text ▶
- Save to file... (highlighted)
- Synchronize with ▶

The hex dump shows several strings in various encodings, including ASCII, UTF-8, and possibly OEM or custom encodings. An orange arrow points from the top right towards the menu.

Address	Hex Value	Dec Value	ASCII
01126598	50 83 22 18 4E 4B 65 62 FD 83 8F AF 06 94 7D 11	80 83 22 18 4E 4B 65 62 FD 83 8F AF 06 94 7D 11	P, ".NKeþý,,-...).
011265A8	E4 2D DE 9F CE D2 C8 04 DD A6 D8 0A 00 00 00 00	80 83 22 18 4E 4B 65 62 FD 83 8F AF 06 94 7D 11	à-þ .íðÈ.Ý;Ø.....
011265B8	00 00 00 80 10 44 00 00 01 00 00 00 00 00 00 80	80 83 22 18 4E 4B 65 62 FD 83 8F AF 06 94 7D 11D.....
011265C8	00 30 00 00 01 00 00 00 7C 93 11 01 00 00 00 00	80 83 22 18 4E 4B 65 62 FD 83 8F AF 06 94 7D 11	,0,.....
011265D8	E0 F7 EA C7 F1 EB C7 FD F9 EB F1 EB EC C7 FD F6	80 83 22 18 4E 4B 65 62 FD 83 8F AF 06 94 7D 11	à÷ èçñëçýüëñëïçýö
011265E8	FB EA E1 E8 EC F1 F7 F6 C7 F5 FD EC F0 F7 FC F3	80 83 22 18 4E 4B 65 62 FD 83 8F AF 06 94 7D 11	ûëáèïñ÷ öçöýí ã÷ üö
011265F8	E4 F9 D4 E2 F8 D4 EE EA F8 E2 F8 FF D4 EE E5 E8	80 83 22 18 4E 4B 65 62 FD 83 8F AF 06 94 7D 11	äùûâø ðîëø âø yûîæ
01126608	F9 F2 FB FF E2 E4 E5 D4 E6 EE FF E3 E4 EF A7 B0	80 83 22 18 4E 4B 65 62 FD 83 8F AF 06 94 7D 11	ùòûýâääûæ îýääí § °
01126618	AD 80 B6 AC 80 BA BE AC B6 AC AB 80 BA B1 BC AD	80 83 22 18 4E 4B 65 62 FD 83 8F AF 06 94 7D 11	.¶, % ¶, %, ± ¼
01126628	A6 AF AB B6 B0 B1 80 B2 BA AB B7 B0 BB B4 A3 BE	80 83 22 18 4E 4B 65 62 FD 83 8F AF 06 94 7D 11	±«¶, °, ±, °, «, °, » , ±%
01126638	93 A5 BF 93 A9 AD BF A5 BF B8 93 A9 A2 AF BE B5	80 83 22 18 4E 4B 65 62 FD 83 8F AF 06 94 7D 11	.¥, .®, ¥, .®, .®, -¾ u
01126648	BC B8 A5 A3 A2 93 A1 A9 B8 A4 A3 A8 00 00 00 00	80 83 22 18 4E 4B 65 62 FD 83 8F AF 06 94 7D 11	
01126658	40 75 11 01 00 00 00 00 2E 50 41 56 4C 45 78 63	80 83 22 18 4E 4B 65 62 FD 83 8F AF 06 94 7D 11	
01126668	65 70 74 69 6F 6E 40 53 79 73 74 65 6D 40 4C 69	80 83 22 18 4E 4B 65 62 FD 83 8F AF 06 94 7D 11	
01126678	67 68 74 40 40 00 00 00 00 00 00 00 00 00 00 00	80 83 22 18 4E 4B 65 62 FD 83 8F AF 06 94 7D 11	
01126688	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00	80 83 22 18 4E 4B 65 62 FD 83 8F AF 06 94 7D 11	
01126698	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00	80 83 22 18 4E 4B 65 62 FD 83 8F AF 06 94 7D 11	
011266A8	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00	80 83 22 18 4E 4B 65 62 FD 83 8F AF 06 94 7D 11	
011266B8	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00	80 83 22 18 4E 4B 65 62 FD 83 8F AF 06 94 7D 11	
011266C8	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00	80 83 22 18 4E 4B 65 62 FD 83 8F AF 06 94 7D 11	
011266D8	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00	80 83 22 18 4E 4B 65 62 FD 83 8F AF 06 94 7D 11	

00044848 | 01126648: .data:01126648

Easy Reverse Engineering

```
#include "stdafx.h"
#include <windows.h>
#include <conio.h>

int _tmain(int argc, _TCHAR* argv[])
{
    unsigned char buff[] = {0xE0, 0xF7, 0xEA, 0xC7, 0xFD, 0xF9};
    unsigned char buff_copy[7] = {0,};

    for(int i = 0; i < 255; i++)
    {
        memcpy(buff_copy, buff, 6);

        for(int j = 0; j < 6; j++)
        {
            buff_copy[j] ^= i;
        }

        printf("%d = %s\n", i, buff_copy);
    }
}
```

```
_getch();

FILE *fp = fopen("input.txt", "r");
unsigned char buff2[1024] = {0,};
fread(buff2, 1024, sizeof(unsigned char) ,fp);

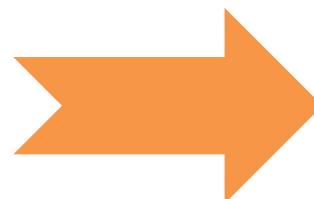
for(int i = 0; i < strlen((const char *)buff2); i++)
{
    buff2[i] ^= 152;
}

printf("new : %s\n",buff2);

fclose(fp);
_getch();
return 0;
```

Easy Reverse Engineering

140	= 1KfRqu
141	= mzgJpt
142	= nydIsW
143	= oxeHru
144	= pgzWmi
145	= qf<U1h
146	= rexUok
147	= sdyTnj
148	= tc^\$im
149	= ub@Rh1
150	= ua!Qko
151	= w^>Pjn
152	= xor_ea
153	= yns^d^
154	= zmp1gc
155	= <1qWfb
156	= Iku[ae
157	= >jwZ^d
158	= ~itYcg
159	= &huXbf
160	= @WJg]Y
161	= AVKfwX
162	= BUHe_E
163	= CTId^Z
164	= DSNeYJ



140	= 1KfRqu
141	= mzgJpt
142	= nydIsW
143	= oxeHru
144	= pgzWmi
145	= qf<U1h
146	= rexUok
147	= sdyTnj
148	= tc^\$im
149	= ub@Rh1
150	= ua!Qko
151	= w^>Pjn
152	= xor_ea
153	= yns^d^
154	= zmp1gc
155	= <1qWfb
156	= Iku[ae
157	= >jwZ^d
158	= ~itYcg
159	= &huXbf
160	= @WJg]Y
161	= AVKfwX
162	= BUHe_E
163	= CTId^Z
164	= DSNeYJ

Easy Reverse Engineering

```
+L = →  
251 = ←♀◀◀◀]  
252 = ←♂↑; Γ]  
253 = ↔  
|:  
254 = ▲ ↗ ↘ L  
new : PASSWORD IS XXXXXXXX_XXXX_XXXX laLz`Lvr`z`g  
3↑")$5>?3.(.)†*?"3/(#,;86='615'=' 61:78-$ =;;691 <;0
```

Easy Reverse Engineering

Category :								
Category :	Crypto(10)	Stegano(6)	Web(9)	Network(2)	Dummy(6)	Reverse(9)	Forensic(1)	
#	Title	Author	Score	Solvers	Age	Questions	Answers	
24	Easy Reverse Engineering	by Bluebird	350	40	17285	?	!	
25	Make EXE by myself!	by Cherishcat	300	27	17284	?	!	
27	Hello? My name is M_IX64	by Cherishcat	600	30	17252	?	!	
31	Extract exe from memory dump	by Cherishcat	380	25	17216	?	!	
37	The Hardest Game in the World	by Bluebird	500	119	17167	?	!	
42	Rape me, If you can	by Bluebird	300	45	17118	?	!	
46	Hello World!	by Cherishcat	150	23	17084	?	!	
47	Ghost In The Shell	by Cherishcat	350	6	17034	?	!	
49	Linux Strip Binary	by Cherishcat	500	10	16998	?	!	

이 외에도 . . . ☺ 리버싱 문제가 다양하게 있습니다.

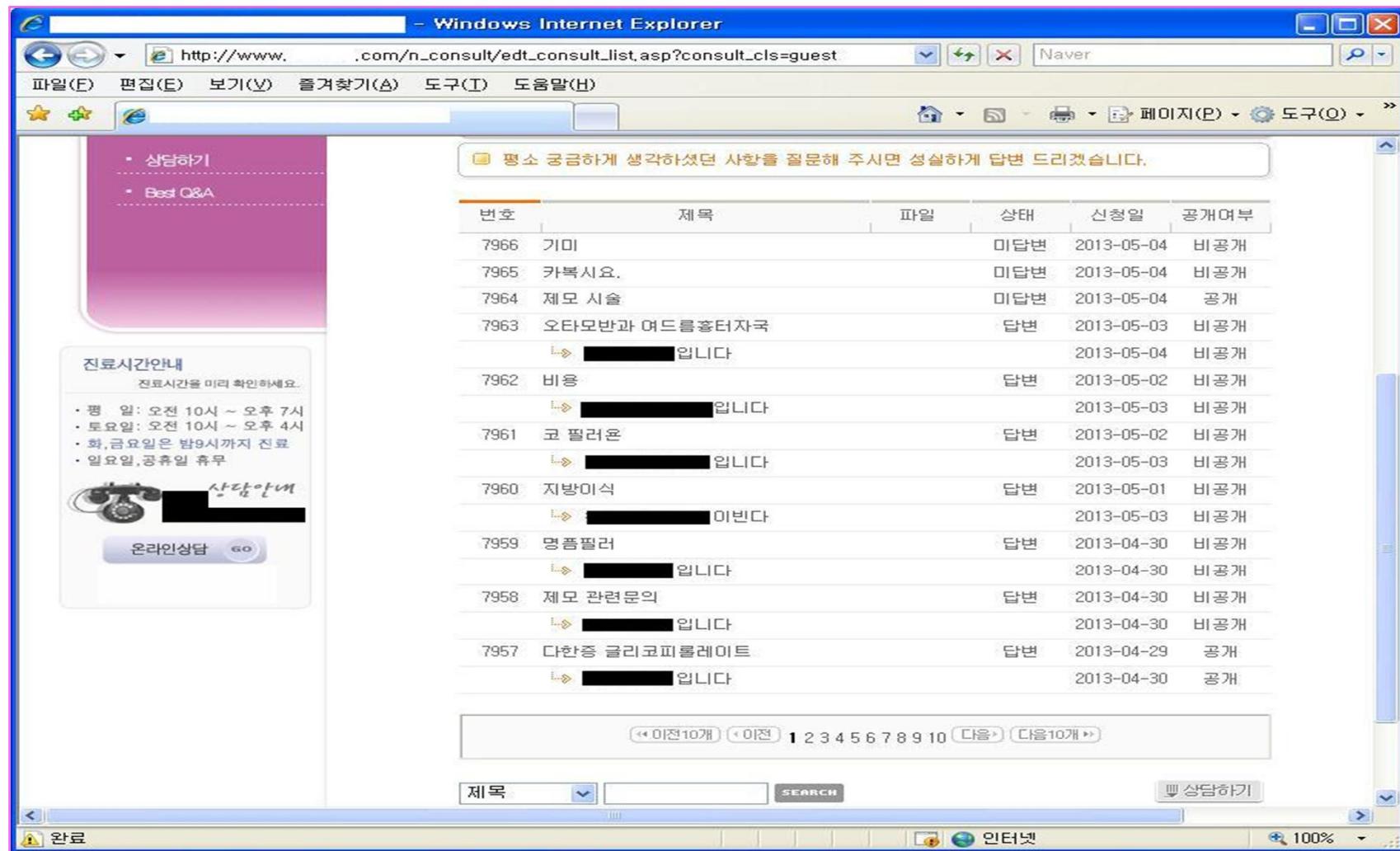
0x05

web site 취약점 (+software)

Web site URL Hacking



Web site URL Hacking



Web site URL Hacking

▣ 평소 궁금하게 생각하셨던 사항을 질문해 주시면 빠르고 성실하게 답변 드리겠습니다.

▣ 상담의(과) []	▣ 프로필보기
▣ 신청자 CherishCat	▣ 이메일 C [] @ C.C
▣ 비밀번호 [*****]	▣ 전화번호 666 - 6666 - 6666
▣ 성별 남 [여]	▣ 나이 6
▣ 공개여부 공개 [비공개]	
▣ 첨부파일 []	찾아보기...

▣ 제목 질문

▣ 상담내용 모공이 너무 넓어서 고민이에요 T_T

▣ 자동등록방지 8 2 3 4 8234 * 옆에 보이는 숫자를 입력해주세요.

정확한 상담을 위하여 아래의 항목에 답해주세요.
진한 글씨는 상담을 위한 필수입력항목입니다.

▣ 상담하기 ▣ 다시작성 ▣ 취소

Web site URL Hacking

평소 궁금하게 생각하셨던 사항을 질문해 주시면 성실하게 답변 드리겠습니다.

번호	제목	파일	상태	신청일	공개여부
7967	질문 new		미답변	2013-05-05	비공개
7966	기미		미답변	2013-05-04	비공개
7965	카복시요.		미답변	2013-05-04	비공개
7964	제모 시술		미답변	2013-05-04	공개
7963	오타모반과 며드를 흘터자국 ↳ [REDACTED]입니다		답변	2013-05-03	비공개
7962	비용 ↳ [REDACTED]입니다		답변	2013-05-02	비공개
7961	코 필러운 ↳ [REDACTED]입니다		답변	2013-05-02	비공개
7960	지방미식 ↳ [REDACTED]이빈다		답변	2013-05-01	비공개
7959	명품필러 ↳ [REDACTED]입니다		답변	2013-04-30	비공개
7958	제모 관련문의 ↳ [REDACTED]입니다		답변	2013-04-30	비공개

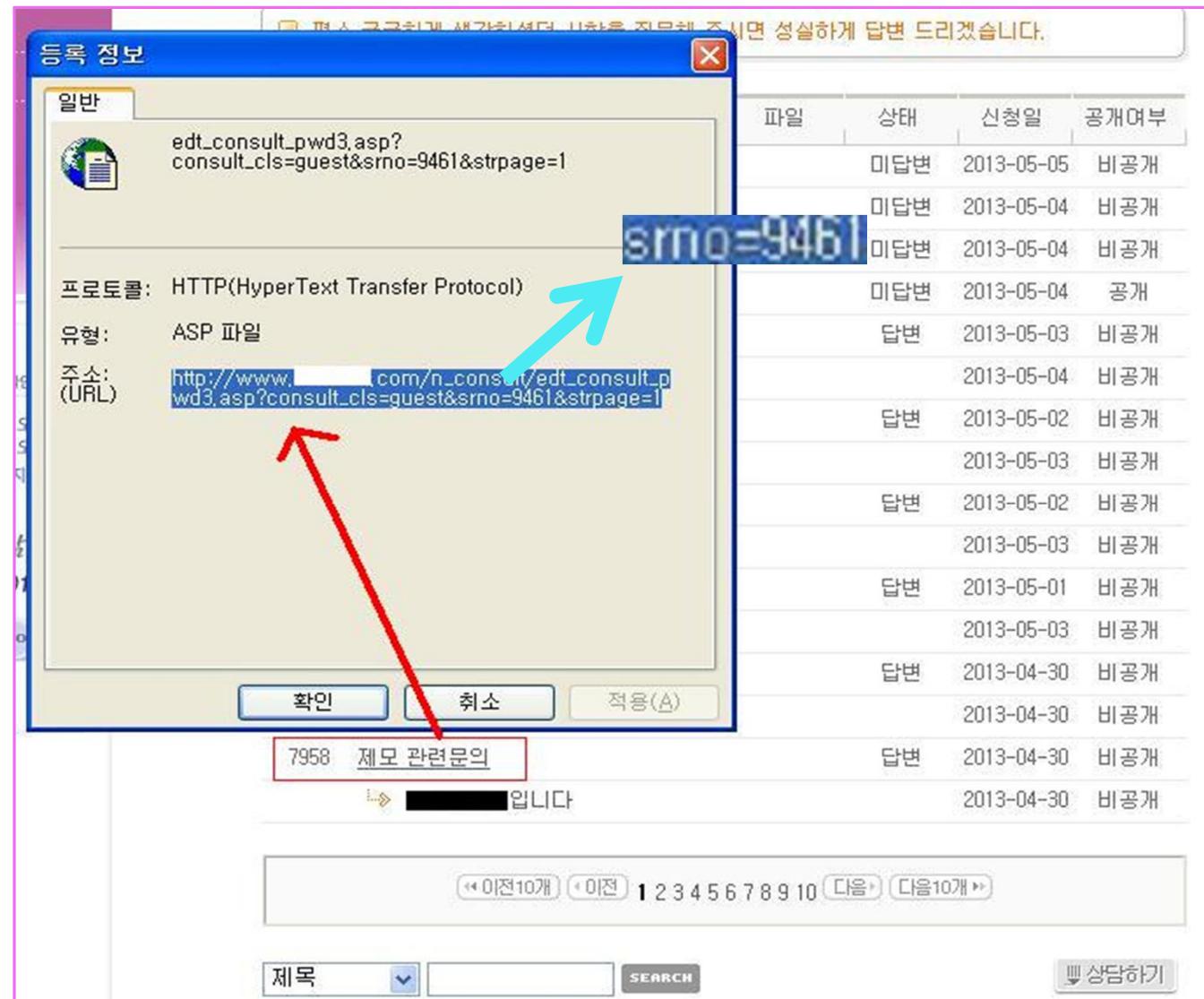
« 이전 10개 이전 1 2 3 4 5 6 7 8 9 10 다음 다음 10개 »

제목SEARCH상담하기

Web site URL Hacking



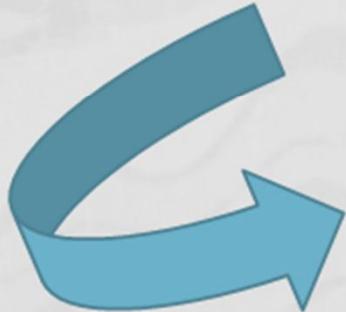
Web site URL Hacking



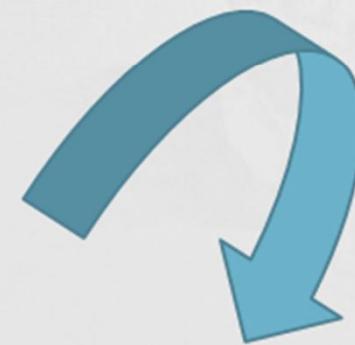
Web site URL Hacking

http://www.XXXXXXXX.com/n_consult/
edt_consult_edit.asp?consult_cls=guest&

srno=9471



srno=9461



http://www.XXXXXXXX.com/n_consult/
edt_consult_edit.asp?consult_cls=guest&

srno=9461

Web site URL Hacking

▣ 평소 궁금하게 생각하셨던 사항을 질문해 주시면 빠르고 성실하게 답변 드리겠습니다.

▣ 상담의 (과)	[REDACTED]
▣ 신청자	김 [REDACTED]
▣ 이메일	[REDACTED]@naver.com
▣ 비밀번호	[REDACTED]
▣ 전화번호	010 - [REDACTED] - 6007
▣ 성별	<input checked="" type="radio"/> 남 <input type="radio"/> 여
▣ 나이	[REDACTED]
▣ 공개여부	<input type="radio"/> 공개 <input checked="" type="radio"/> 비공개
▣ 기존파일	[REDACTED]
▣ 새파일	[REDACTED]
찾아보기...	

내 비밀이
들키다니

제목 제모 관련문의

기본글꼴 보통 줄간격 B I U A A

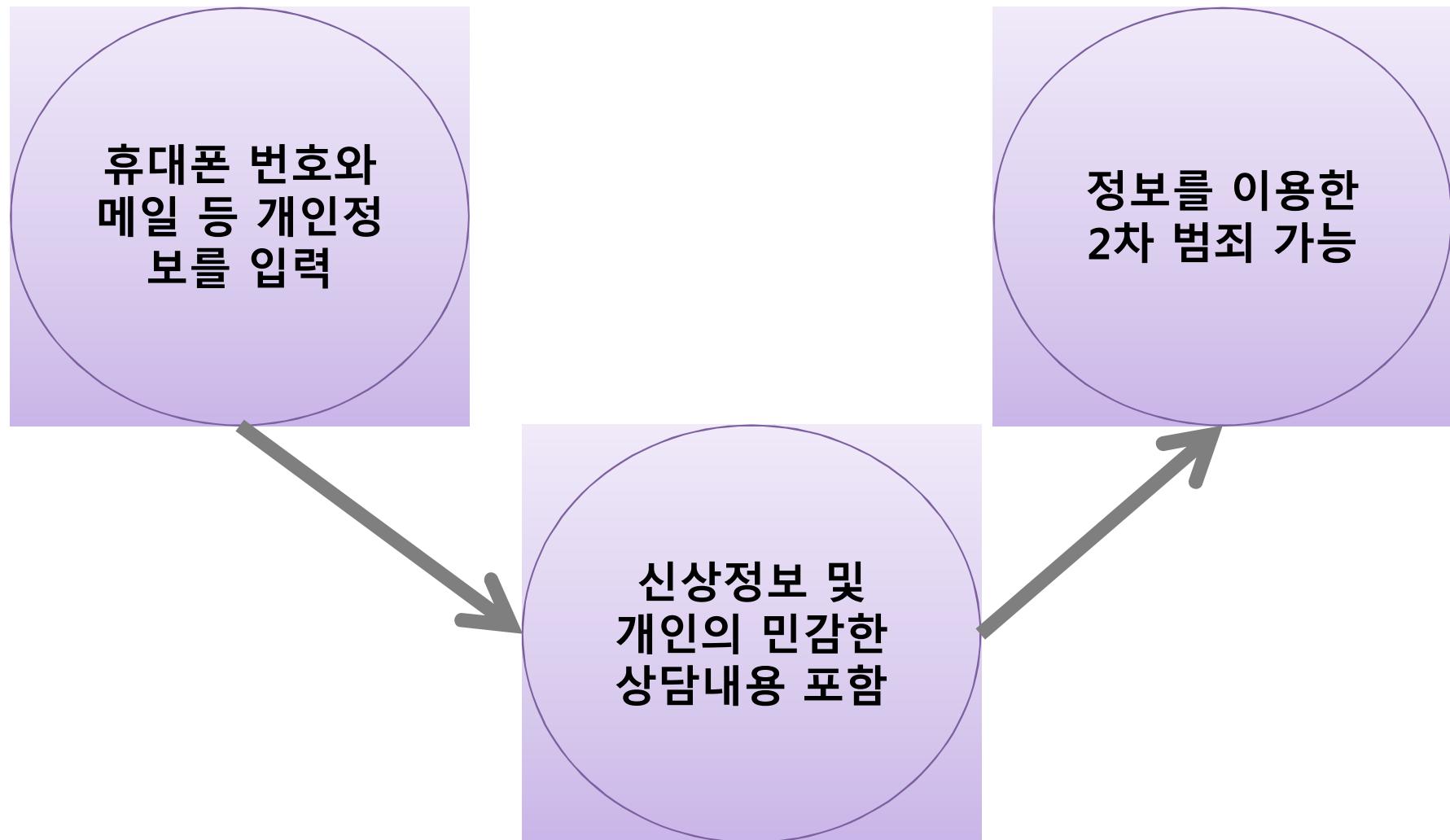
남자영구제모도 되나요??
된다면
팔 하완상완 할때 가격이랑
팔 하완상완 다리 종마리 무릎할때 가격좀 알려주세요
또 손이나 발제모는 추가요금이 더 붙는건지알려주세요
많이하면 가격할인될려나요

편집 HTML 미리보기

정확한 상담을 위하여 아래의 항목에 답해주세요.
진한 글씨는 상담을 위한 필수입력항목입니다.

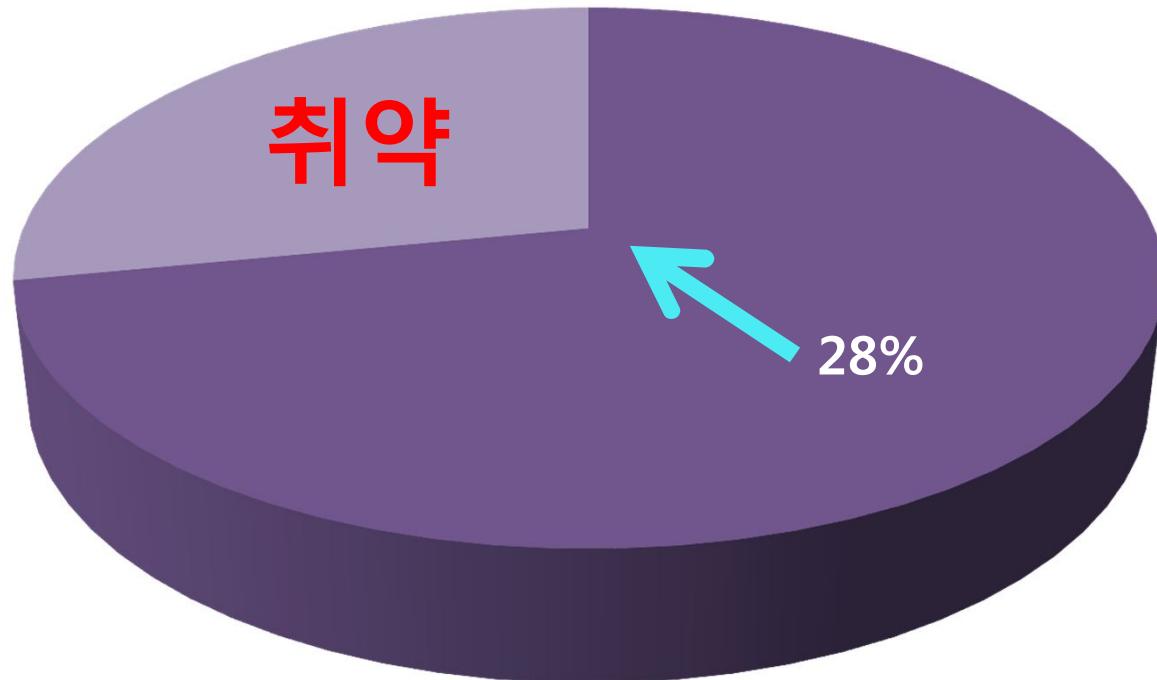


Web site URL Hacking



Web site URL Hacking

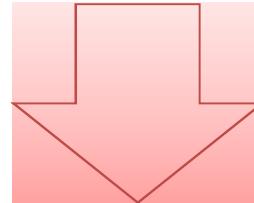
전국 40개 사이트 무작위 실험



7개 사이트 취약

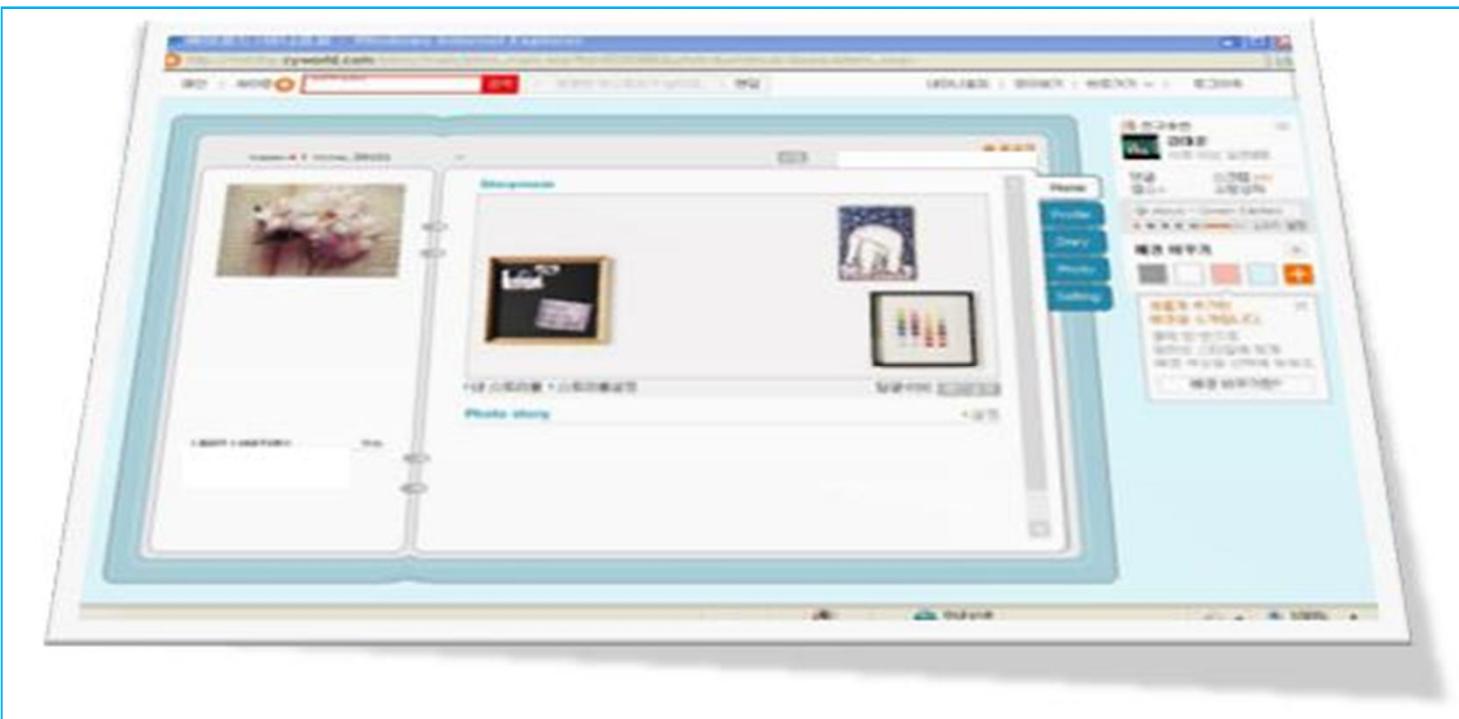
Web site URL Hacking

보안에 취약한 구식 웹 게시판을 이용

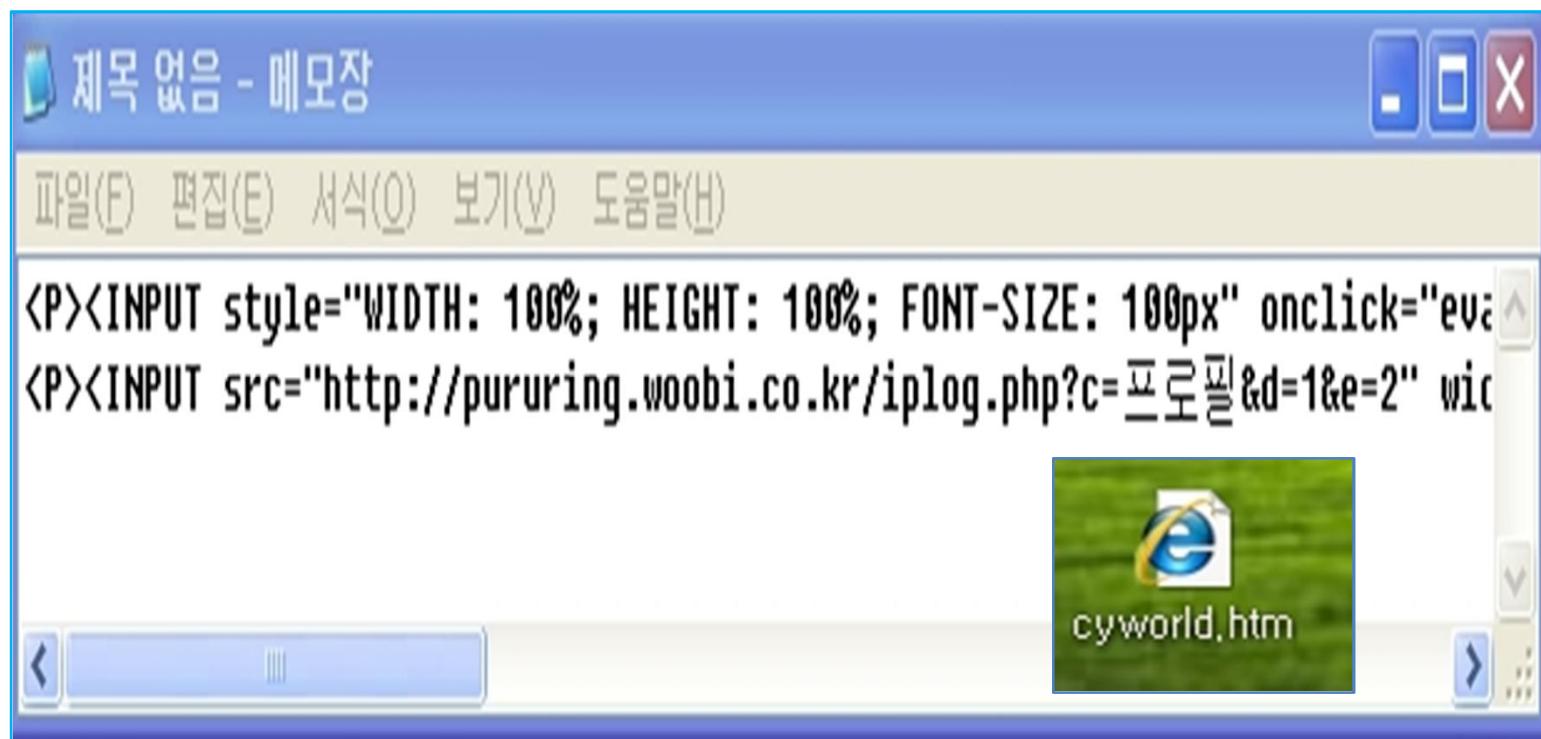


주소창 암호화 X
알기 쉬운 형식 NO
새로운 버전 웹 게시판 사용

Web site XSS Hacking



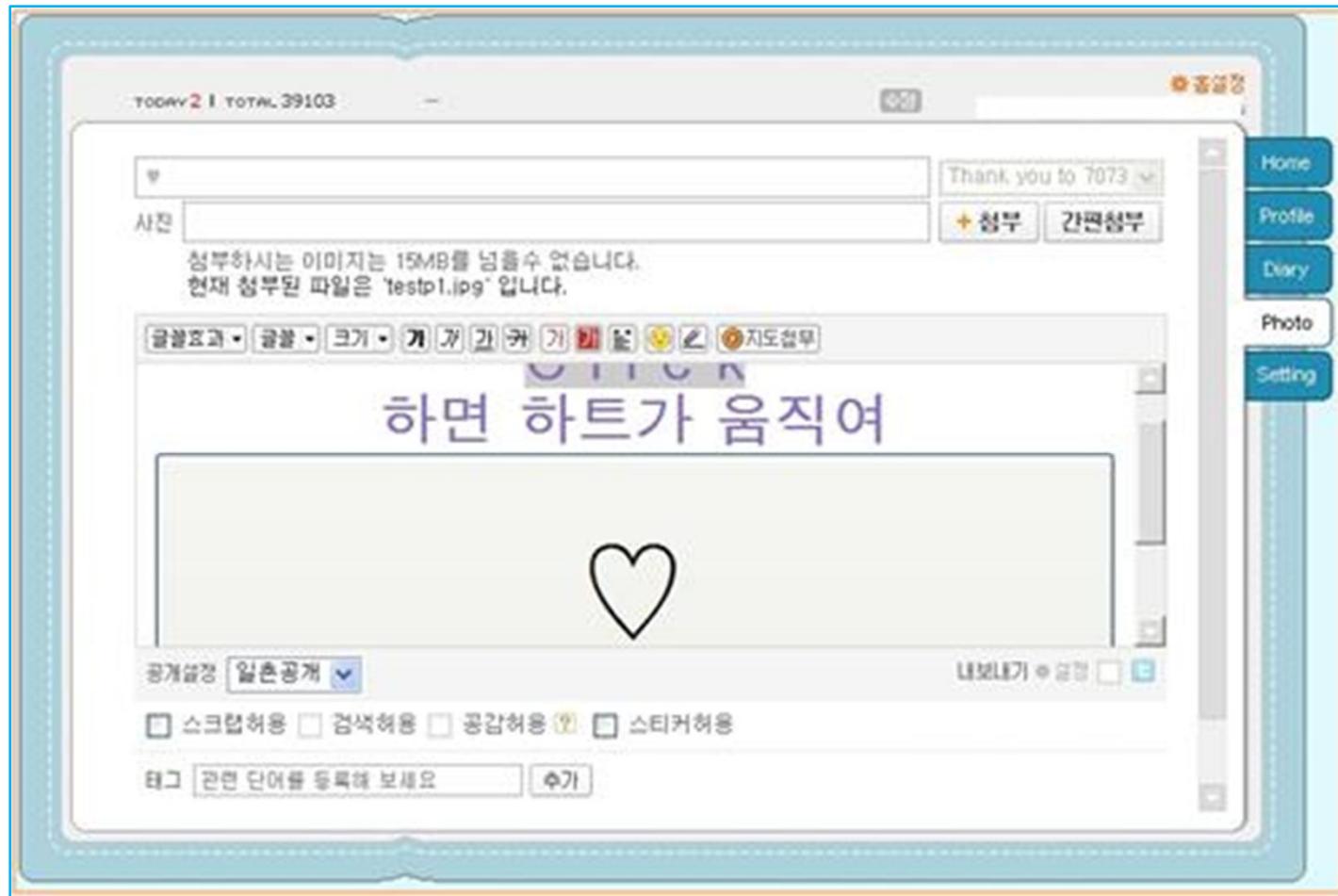
Web site XSS Hacking



Web site XSS Hacking



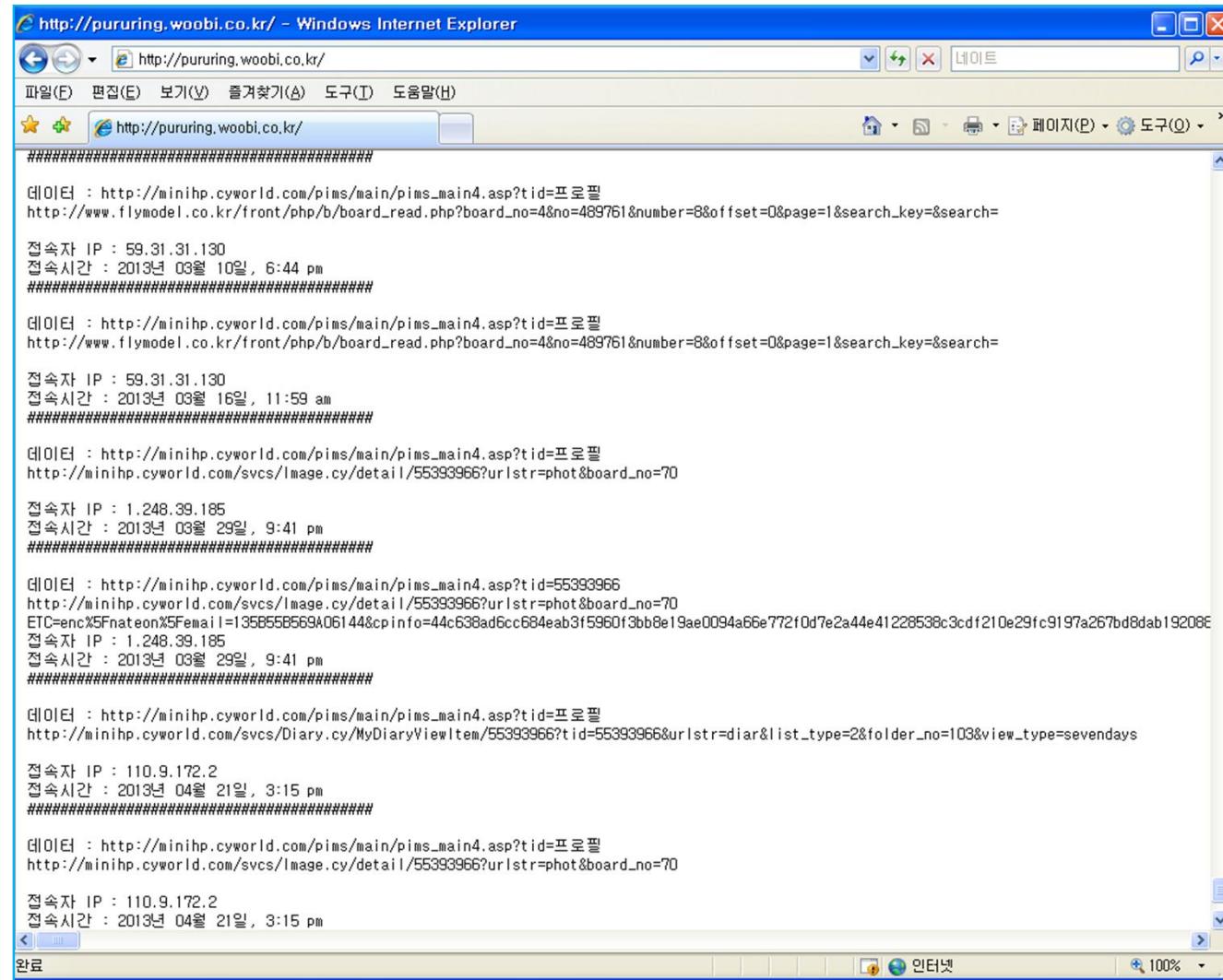
Web site XSS Hacking



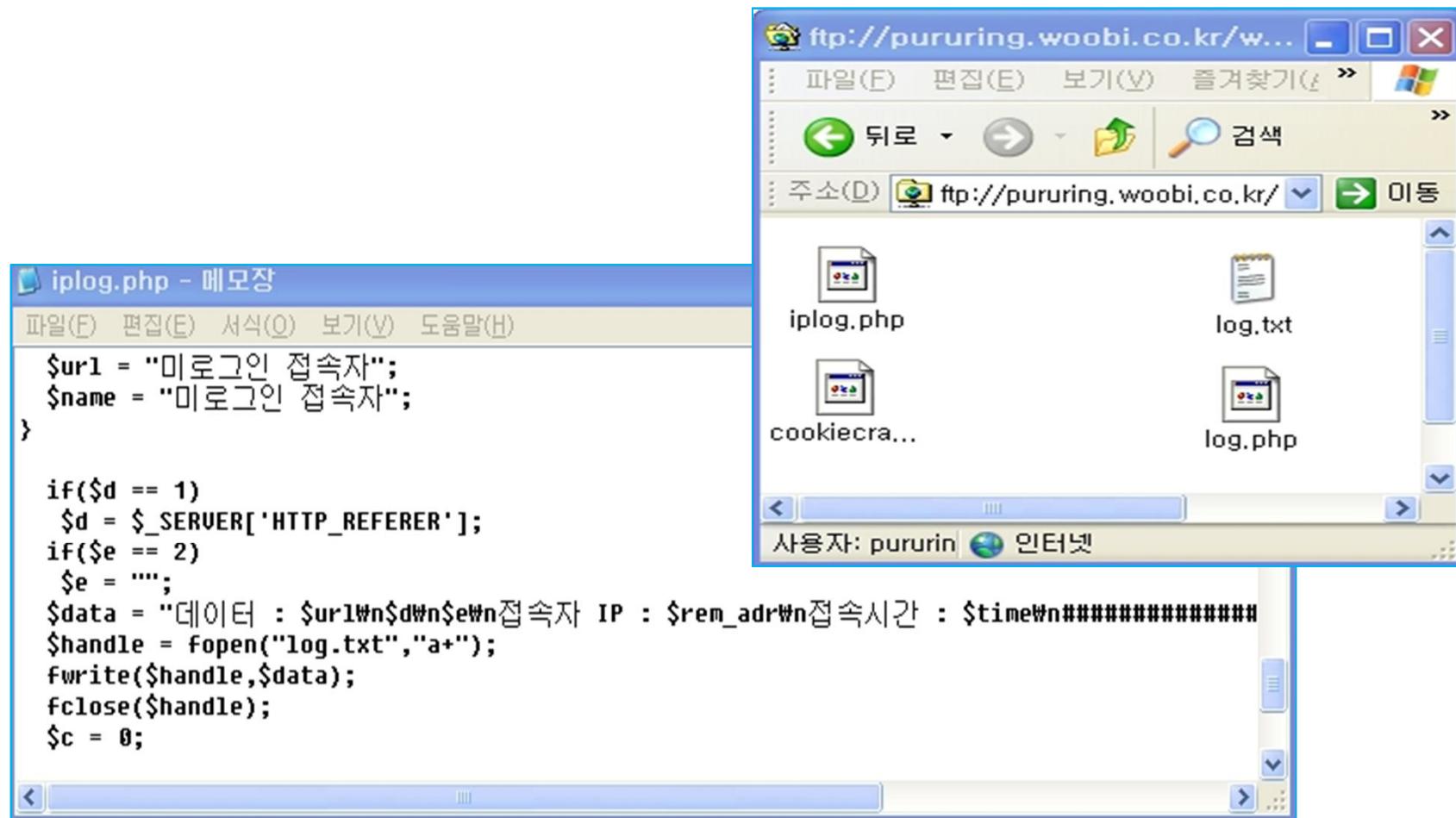
Web site XSS Hacking



Web site XSS Hacking



Web site XSS Hacking



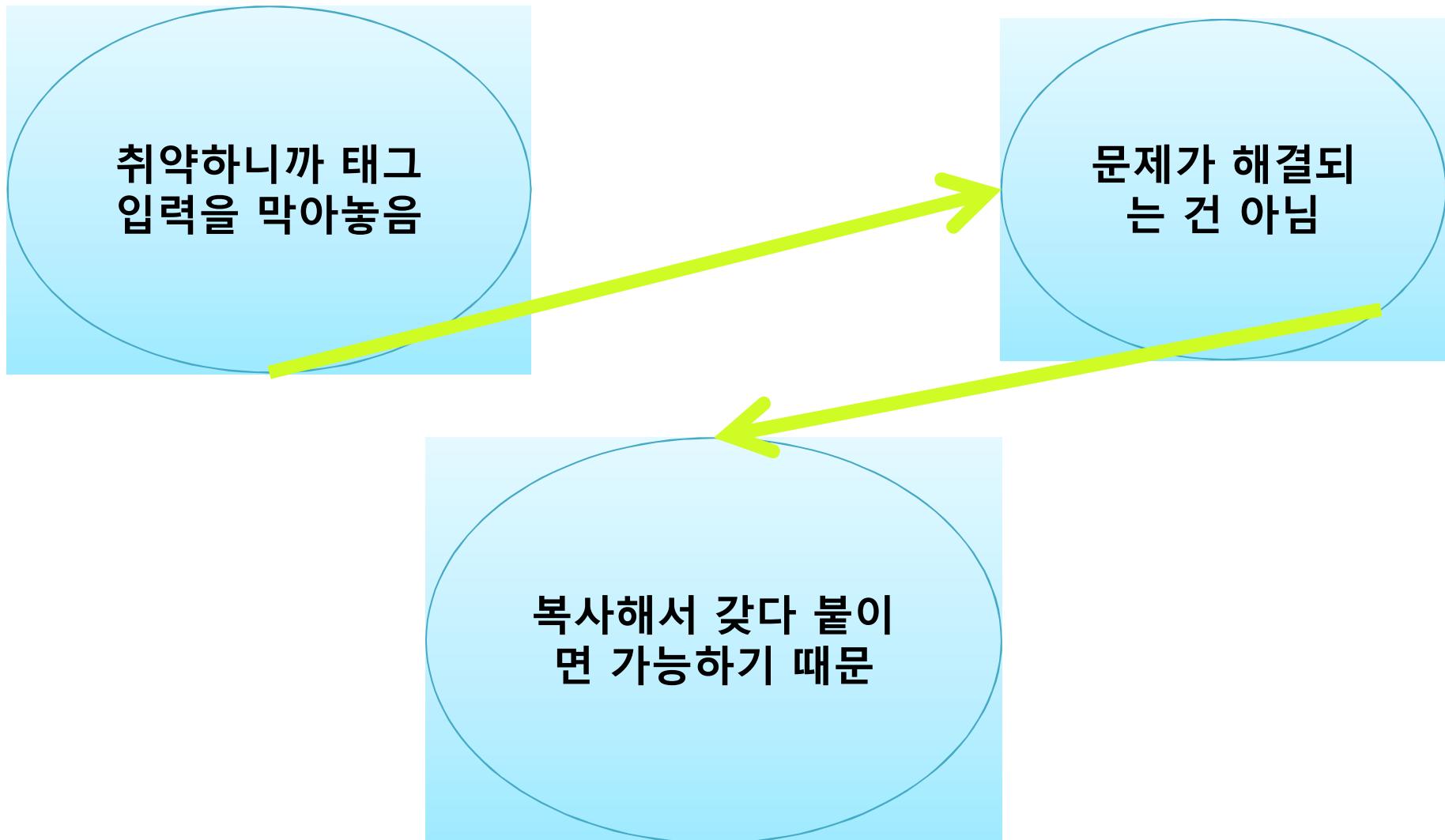
The image shows a screenshot of a Windows desktop environment. On the left, there is a Notepad window titled "ilog.php - 메모장" containing the following PHP code:

```
$url = "미로그인 접속자";
$name = "미로그인 접속자";
}

if($d == 1)
$d = $_SERVER['HTTP_REFERER'];
if($e == 2)
$e = "";
$data = "데이터 : $url\n$d\n$e\n접속자 IP : $rem_addr\n접속시간 : $time\n#####";
$handle = fopen("log.txt","a+");
fwrite($handle,$data);
fclose($handle);
$c = 0;
```

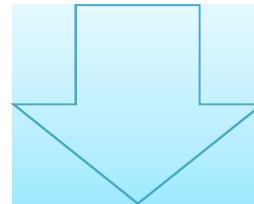
On the right, there is an "Internet Explorer" browser window showing an FTP interface. The address bar shows "ftp://pururing.woobi.co.kr/w...". The file list includes "ilog.php", "log.txt", "cookiecra...", and "log.php". The status bar at the bottom of the browser window displays "사용자: pururin 인터넷".

Web site XSS Hacking



Web site XSS Hacking

많은 웹사이트에서 노출될 것으로 예상



필터링 & 붙여넣기 NO
100%는 아니더라도 99.9%

Software : Anti-Virus



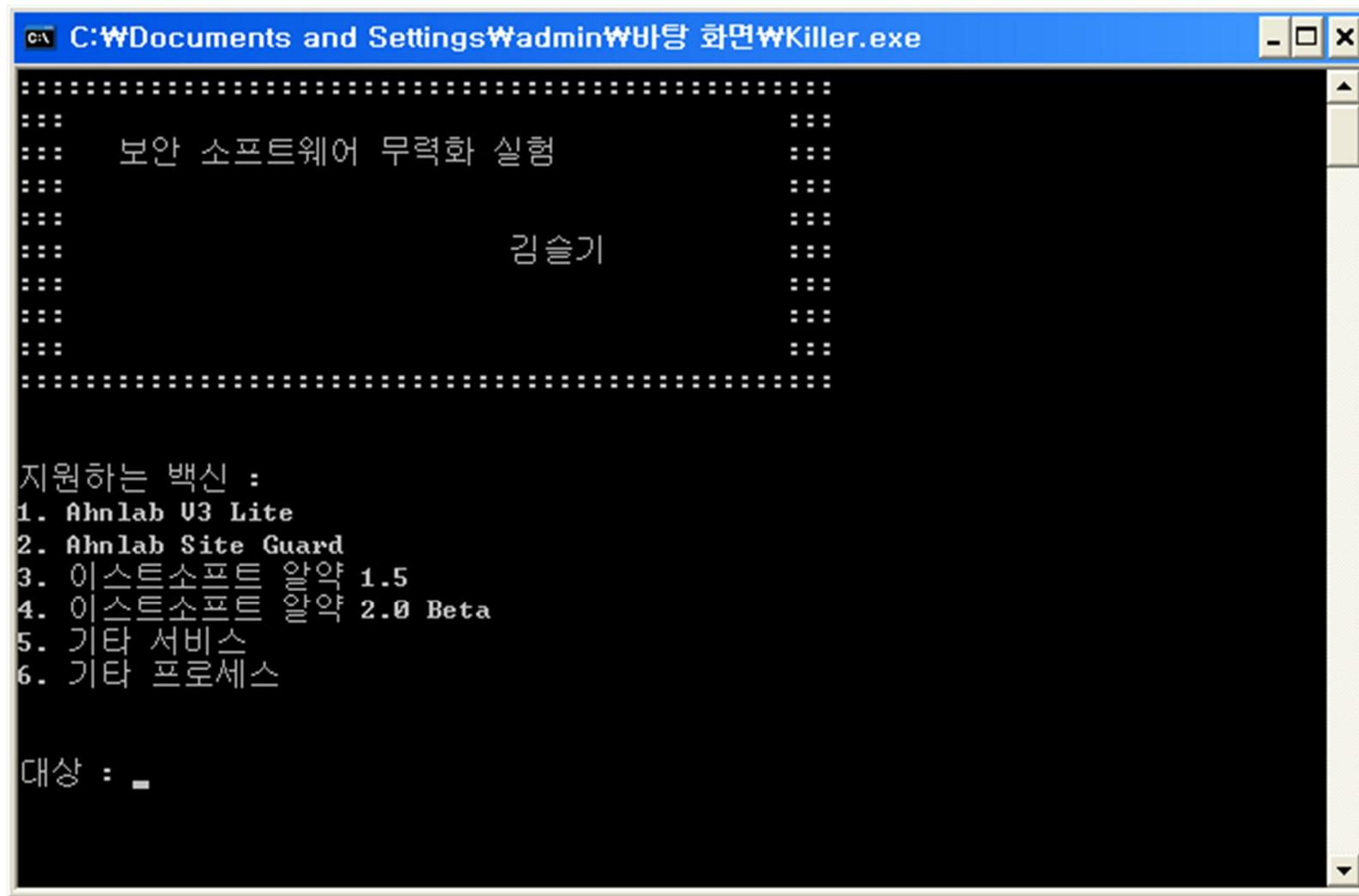
자가보호란?

무결성이 유지되어야 하는
안티바이러스가 자신이 사
용하는 파일, 레지스트리, 프
로세스, 서비스, 커널 객체(이벤
트, 뮤텍스, 네임드 파이프, 메일 슬롯, 공유
메모리) 등을 외부에서 접근
하는 것을 관리 또는 차단
하는 것을 의미합니다 😊

Software : Anti-Virus

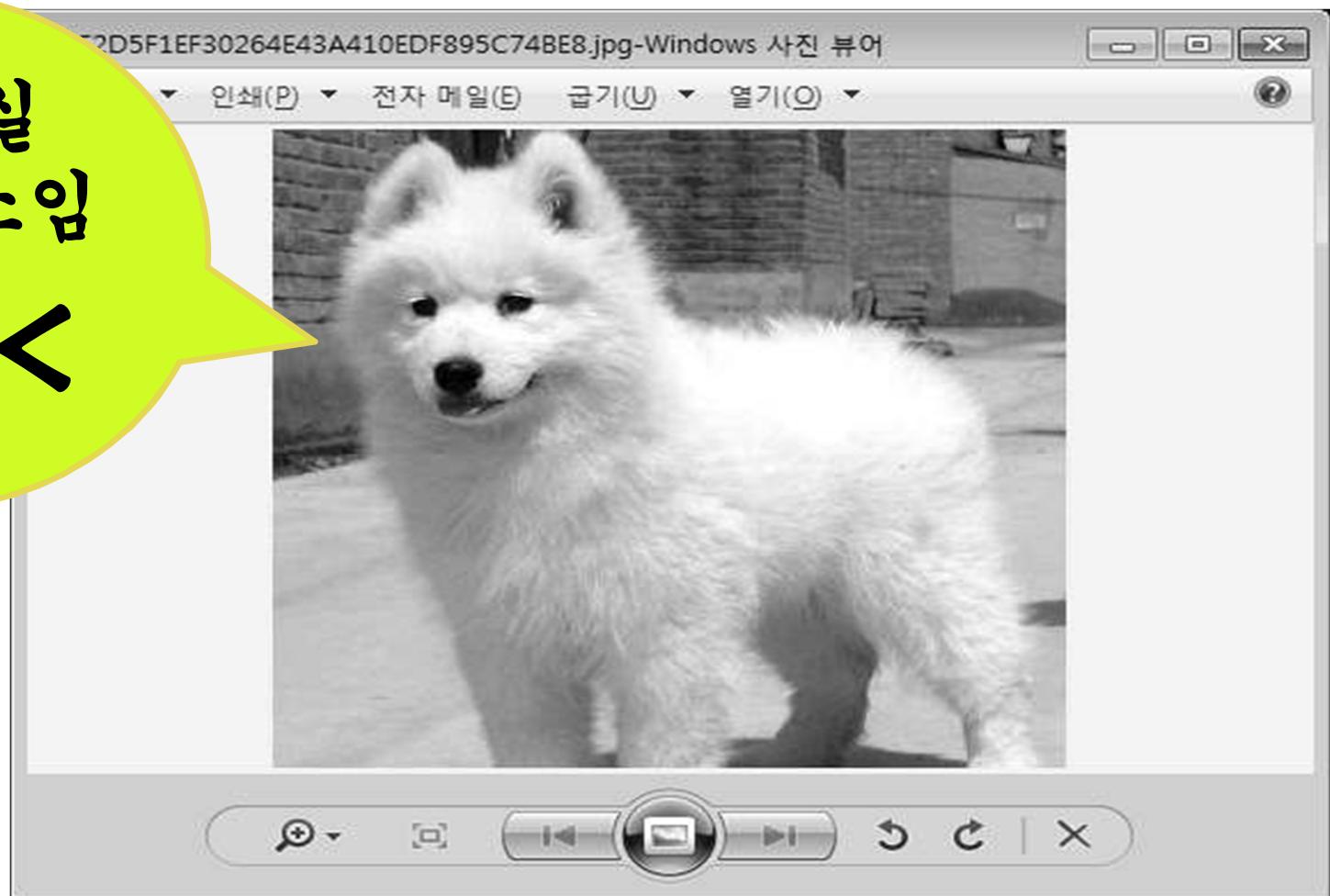


Software : Anti-Virus



Software : Anti-Virus

나 사실
바이러스임
>人<



Software : Anti-Virus



0x06
E.N.D

• • •

발표 잘 들으셨나요?

\(^ω^)/





Q&A



감사합니다.

www.CodeEngn.com | www.RCEHub.com

Android **Malicious** Application Filtering System



세인트시큐리티 김창수 | kcs5287@stsc.com

www.CodeEngn.com
2013 CodeEngn Conference 08

Code**Engn**

Contents

- Intro
- Android Malicious Application
- Malicious Application Analysis
- Automation System

Intro



ANDROID

01

Smartphone



02

Android VS iOS

자유

폐쇄

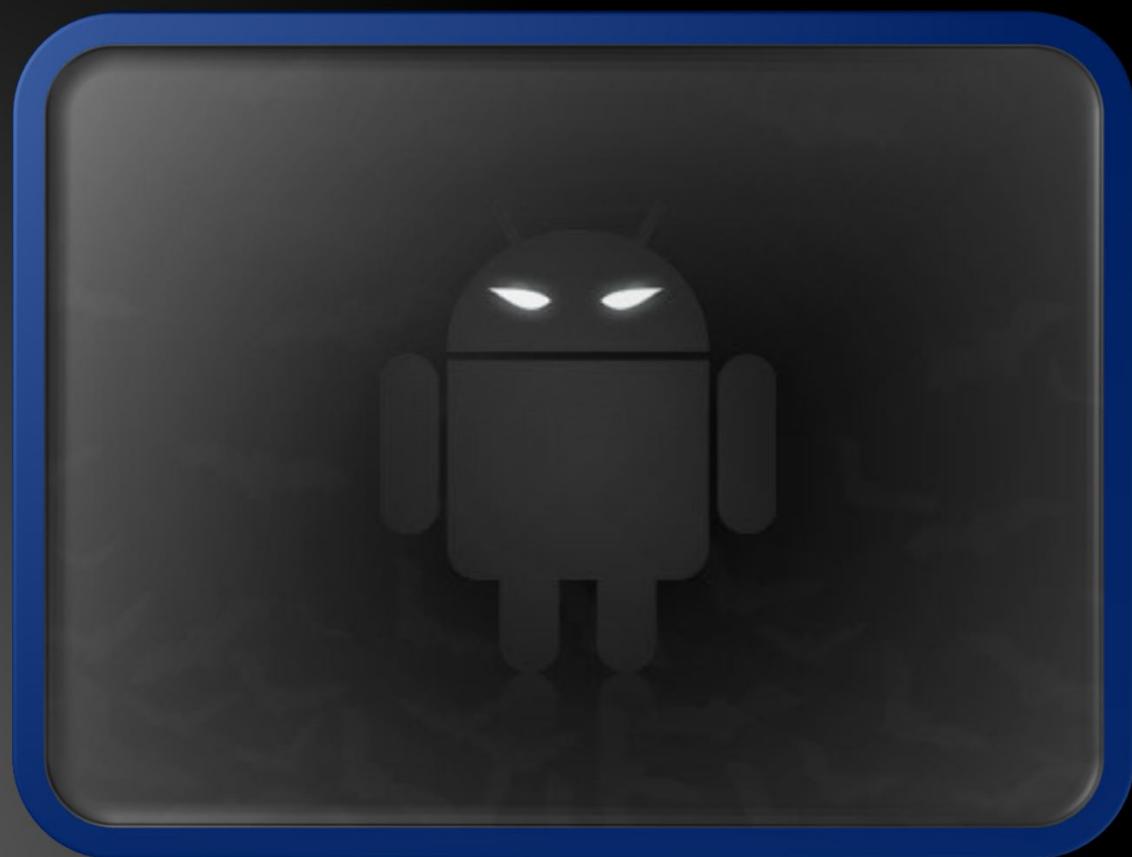
03

Android Architecture



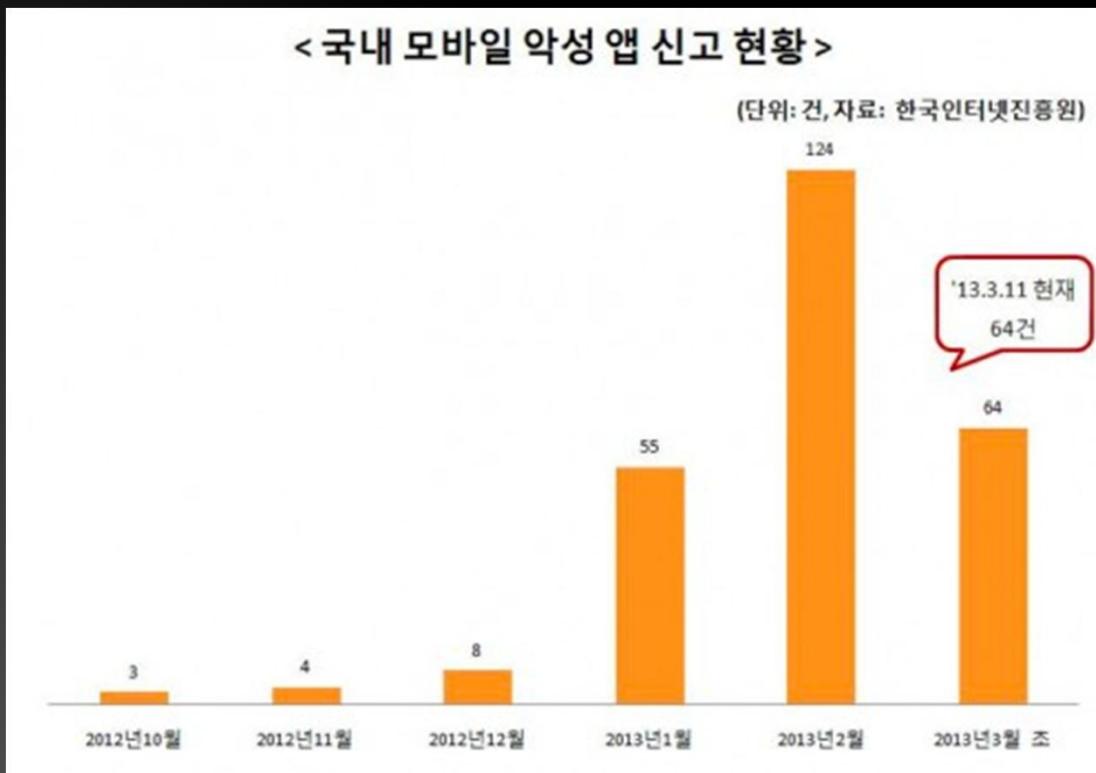
Chapter 1

Android Malicious Application



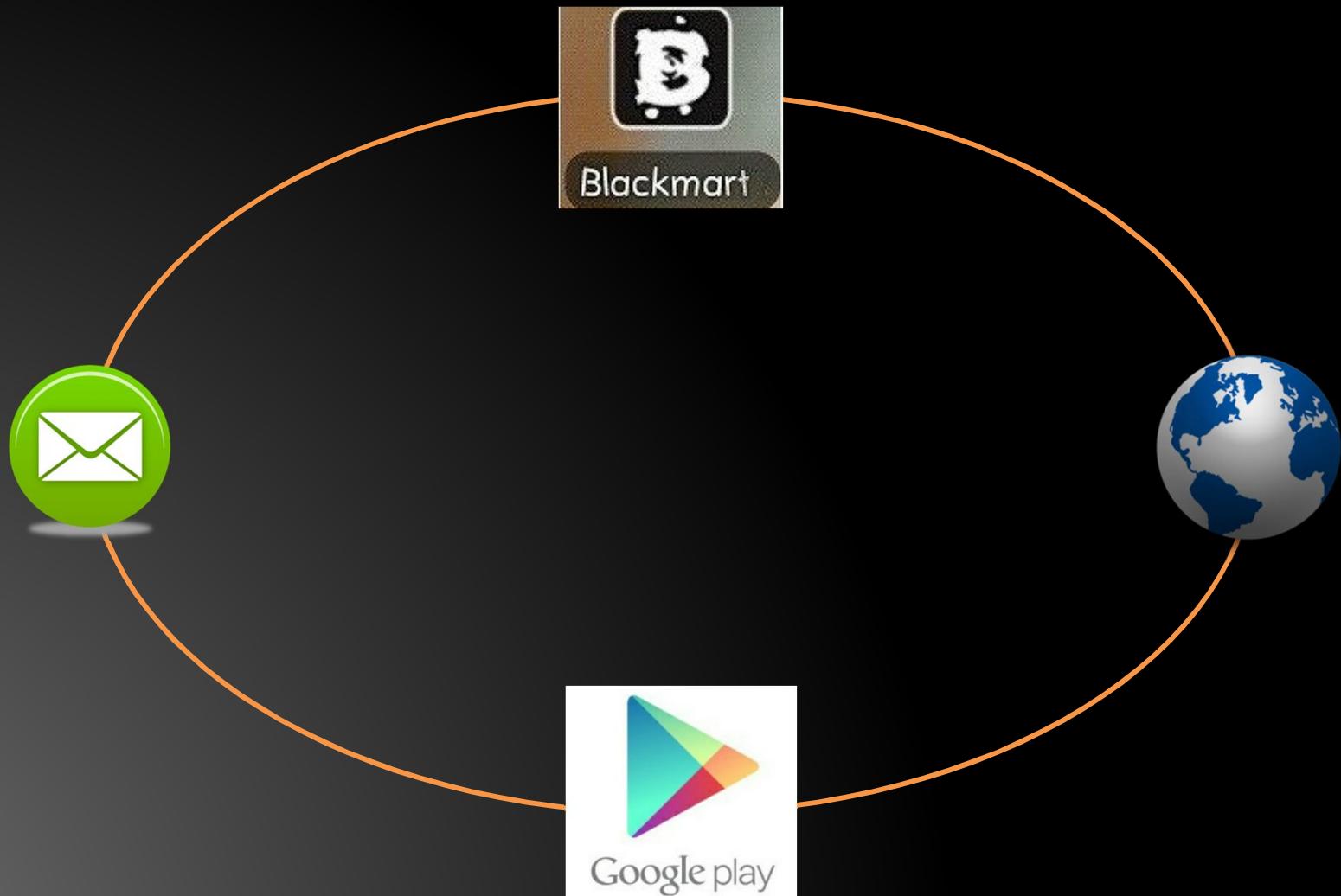
01

Malware Trend



02

Infection Route



- 블랙마켓
 - 크랙 된 어플리케이션으로 사용자를 유인
 - 해커는 자신의 코드를 어플리케이션에 숨기고 배포
- 인터넷 카페 및 블로그
 - 블랙마켓과 비슷한 수법
- SMS
 - 쿠폰 및 할인 등의 문자메시지로 특정 앱을 다운받도록 함
- 구글 마켓
 - 비교적 허술한 구글 마켓의 어플리케이션 등록 절차를 이용
 - 뒤늦게 삭제 조치되는 경우가 있으나, 그 전에는 피해가 발생

03

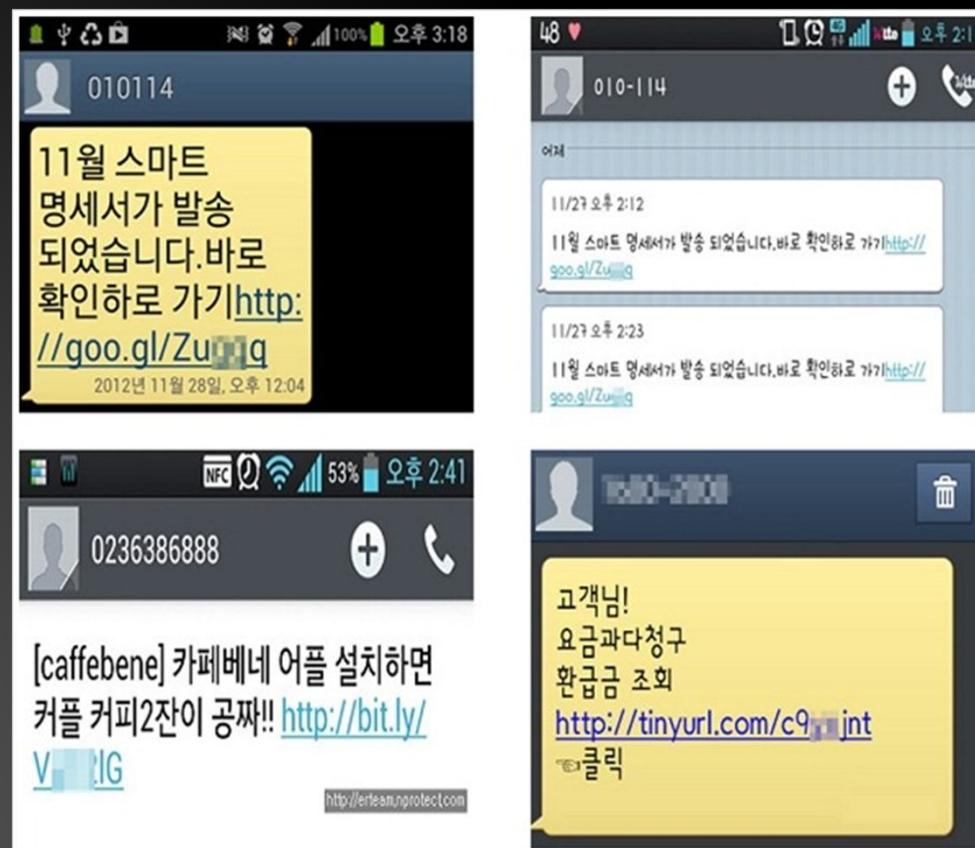
Purpose

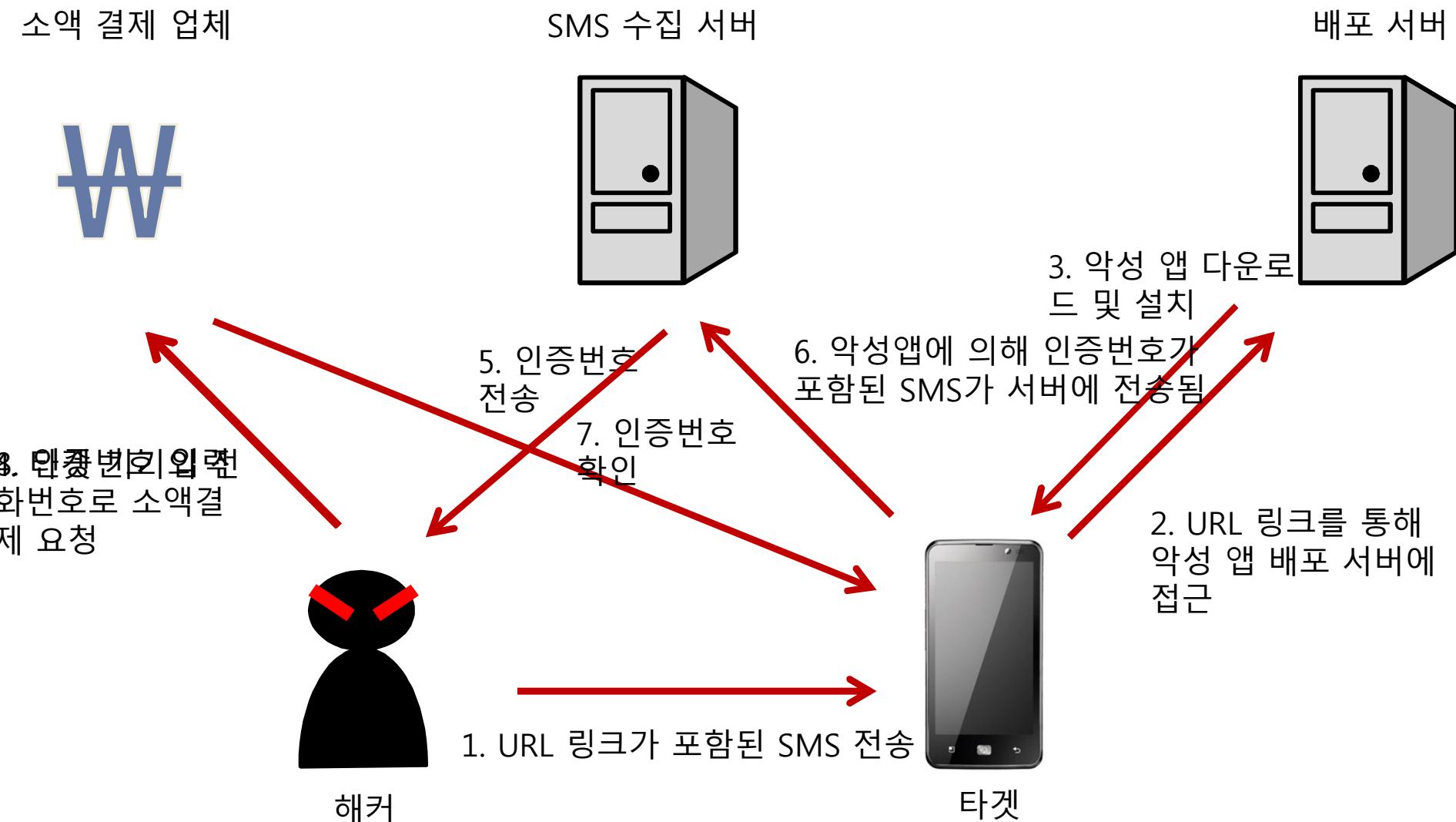
- 개인정보 수집
- 금융정보 탈취
- 소액결제를 통한 이득
- 봇넷
- Just For Fun

04

Smishing

- SMS를 통해 문자메시지를 가로채는 악성 어플리케이션을 다운로드 받게 한 후, 소액결제를 통한 이득을 획득하는 방법





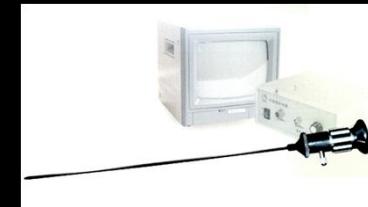
Chapter 2

Malicious Application Analysis



01

Overview



02

AndroidManifest.xml

- 어플리케이션의 큰 틀을 알 수 있음
- 패키지의 이름, 권한 정보, 액티비티 및 서비스의 종류 등을 알아낼 수 있음
- Apktool을 이용하여 추출 가능

02

AndroidManifest.xml

이름	수정한 날짜	유형	크기
tools	2013-06-04 오전...	파일 폴더	
sample.apk	2013-03-13 오후...	APK 파일	21KB

```
C:\Users\김창수\Desktop\발표>tools\apktool d sample.apk
I: Baksmaling...
I: Loading resource table...
I: Loaded.
I: Loading resource table from file: C:\Users\김창수\apktool\framework\1.apk
I: Loaded.
I: Decoding file-resources...
I: Decoding values*//* XMLs...
I: Done.
I: Copying assets and libs...
```

이름	수정한 날짜	유형	크기
res	2013-06-04 오전...	파일 폴더	
smali	2013-06-04 오전...	파일 폴더	
AndroidManifest.xml	2013-06-04 오전...	XML 문서	2KB
apktool.yml	2013-06-04 오전...	YML 파일	1KB

02

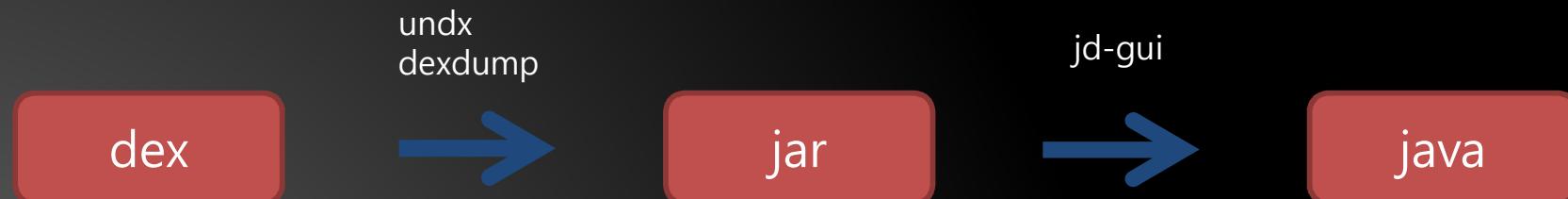
AndroidManifest.xml

```
<?xml version="1.0" encoding="UTF-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android" package="d...5" android:versionName="1.0" android:versionCode="1">
    <uses-permission android:name="android.permission.SEND_SMS"/>
    <uses-permission android:name="android.permission.INTERNET"/>
    <uses-permission android:name="android.permission.READ_SMS"/>
    <uses-permission android:name="android.permission.READ_CONTACTS"/>
    <uses-permission android:name="android.permission.CALL_PHONE"/>
    <uses-permission android:name="android.permission.READ_PHONE_STATE"/>
    <uses-permission android:name="android.permission.RECEIVE_SMS"/>
    <uses-permission android:name="com.android.browser.permission.READ_HISTORY_BOOKMARKS"/>
    <uses-permission android:name="android.permission.READ_CALENDAR"/>
    <uses-permission android:name="android.permission.WRITE_CALENDAR"/>
    <permission android:name="com.me.app.myapp.permission.DEADLY_ACTIVITY" android:protectionLevel="dangerous"/>
    - <application android:debuggable="true" android:icon="@drawable/icon" android:label="@string/app_name">
        - <activity android:name=".MainActivity" android:label="@string/app_name">
            - <intent-filter>
                <action android:name="android.intent.action.MAIN"/>
                <category android:name="android.intent.category.LAUNCHER"/>
            </intent-filter>
        </activity>
        - <receiver android:name=".SMSReceiver">
            - <intent-filter>
                <action android:name="android.provider.Telephony.SMS_RECEIVED"/>
            </intent-filter>
        </receiver>
        <service android:name=".SendDataService"/>
    </application>
</manifest>
```

03

Decompile

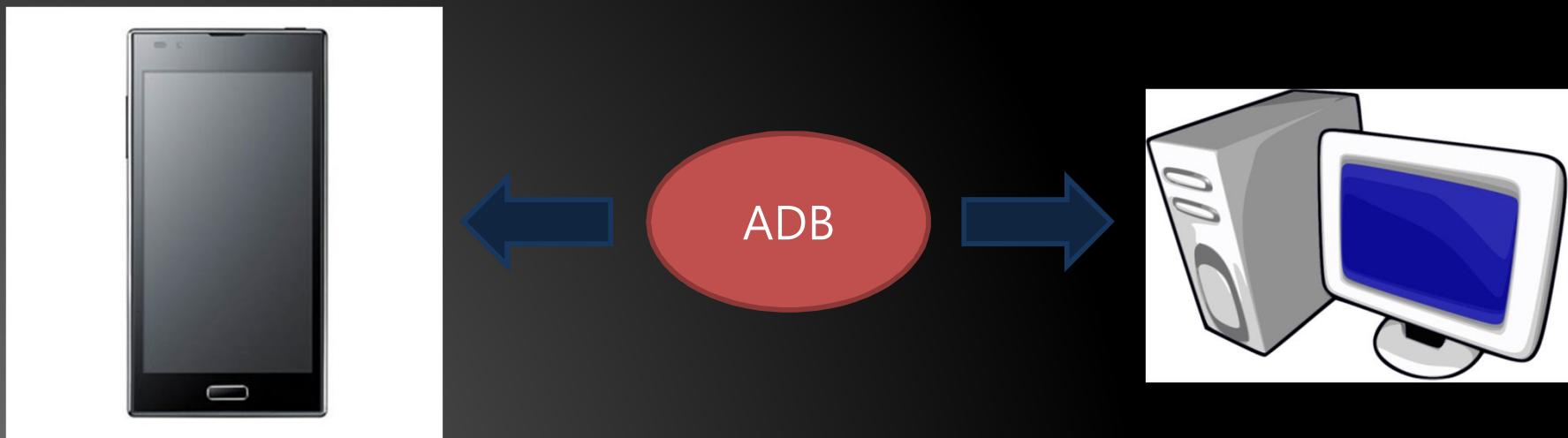
- APK 파일 내의 dex 파일을 java 소스코드로 변환
- undx와 dexdump, jd-gui 를 이용
- 상세한 분석이 필요할 때 효과적



04

ADB

- Android Debug Bridge
- 컴퓨터와 안드로이드 기기 간의 통신을 도움



- shell : 안드로이드 기기의 shell에 연결한다.
- install : 안드로이드 기기에 어플리케이션을 설치한다.
- uninstall : 안드로이드 기기에 설치된 어플리케이션을 삭제한다.
- push : 안드로이드 기기로 파일을 옮긴다.
- pull : 안드로이드 기기에서 파일을 가져온다.

05

Logcat

- 단말에서 발생하는 로그를 보여줌
- 어플리케이션 개발 시에 디버깅 툴로 사용
- 로그의 우선 순위
 - Verbose < Debug < Info < Warning < Error < Fatal < Silent

05

Logcat

```
import android.app.Activity;
import android.os.Bundle;
import android.util.Log;

public class LogCat extends Activity
{
    private static final String TAG = "LogCatTest";

    /** Called when the activity is first created. */
    @Override
    public void onCreate(Bundle savedInstanceState)
    {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.main);

        Log.v(TAG,"Log Priority Level : Verbose");
        Log.d(TAG,"Log Priority Level : Debug");
        Log.i(TAG,"Log Priority Level : Info");
        Log.w(TAG,"Log Priority Level : Warning");
        Log.e(TAG,"Log Priority Level : Error");
    }
}
```



```
V/LogCatTest( 2823): Log Priority Level : Verbose
D/LogCatTest( 2823): Log Priority Level : Debug
I/LogCatTest( 2823): Log Priority Level : Info
W/LogCatTest( 2823): Log Priority Level : Warning
E/LogCatTest( 2823): Log Priority Level : Error
```

05

Logcat

- 어플리케이션 로그 이외에 시스템 이벤트를 확인 가능
- adb logcat -d events

```
I/dalvikvm( 1323): threadid=3: reacting to signal 3
I/dalvikvm( 1323): Wrote stack traces to '/data/anr/traces.txt'
I/ActivityManager(  799): START {act=android.intent.action.PICK dat= typ=vdnd.and
roid.cursor.dir/artistalbum flg=0x4000000 cmp=com.android.music/.ArtistAlbumBrow
serActivity {has extras}} from pid 1323
D/dalvikvm( 1323): GC_CONCURRENT freed 147K, 3% free 9388K/9607K, paused 8ms+5ms

I/Process <  799>: Sending signal. PID: 1323 SIG: 3
I/dalvikvm( 1323): threadid=3: reacting to signal 3
I/dalvikvm( 1323): Wrote stack traces to '/data/anr/traces.txt'
D/dalvikvm(  799): GC_EXPLICIT freed 527K, 8% free 12408K/13383K, paused 7ms+16m
s
I/AudioService<  799>: Remote Control registerMediaButtonIntent<> for Pendin
gIntent{415ae000: PendingIntentRecord{4161faf0 com.android.music broadcastIntent
}}
I/Process <  799>: Sending signal. PID: 1323 SIG: 3
I/dalvikvm( 1323): threadid=3: reacting to signal 3
I/dalvikvm( 1323): Wrote stack traces to '/data/anr/traces.txt'
V/PhoneStatusBar<  852>: setLightsOn<true>
D/gralloc_goldfish( 1323): Emulator without GPU emulation detected.
I/ActivityManager<  799>: Displayed com.android.music/.ArtistAlbumBrowserActivit
y: +1s261ms (total +5s49ms)
W/NetworkManagementSocketTagger<  799>: setKernelCountSet<10021, 0> failed with
errno -2
D/dalvikvm(  881): GC_CONCURRENT freed 352K, 6% free 9551K/10119K, paused 7ms+8ms
```

06

TCP Dump

- ADB를 이용해 shell로 접속하여 TCP Dump 사용 가능

```
C:\APK_DYNAMIC\etc>adb shell
# netcfg
netcfg
lo      UP          127.0.0.1/8  0x00000049  00:00:00:
00:00:00
eth0    UP          10.0.2.15/24 0x00001043  52:54:00:
12:34:56
sit0   DOWN        0.0.0.0/0   0x00000080  00:00:00:
00:00:00
# tcpdump -i eth0 -s 0 -w /data/capture
tcpdump -i eth0 -s 0 -w /data/capture
tcpdump: listening on eth0, link-type EN10MB (Ethernet), capture size 65535 bytes
s
```

No.	Time	Source	Destination	Protocol	Length	Info
22	15.702392	10.0.2.15	10.0.2.3	DNS	73	Standard query 0x7e9f A www.naver.com
23	15.742814	10.0.2.3	10.0.2.15	DNS	125	Standard query response 0x7e9f CNAME www.g.naver.com A 220.95.233.172 A 202.131.30.11
24	15.749284	10.0.2.15	220.95.233.172	TCP	74	48378 > http [SYN] Seq=0 Win=5840 Len=0 MSS=1460 SACK_PERM=1 TSval=289639 TSerr=0 WS=2
25	15.758795	220.95.233.172	10.0.2.15	TCP	64	http > 48378 [SYN, ACK] Seq=0 Ack=1 Win=8192 Len=0 MSS=1460 [ETHERNET FRAME CHECK SEQUENCE INCORRECT]
26	15.759062	10.0.2.15	220.95.233.172	TCP	54	48378 > http [ACK] Seq=1 Ack=1 Win=5840 Len=0
27	15.787240	10.0.2.15	220.95.233.172	HTTP	449	GET / HTTP/1.1
28	15.787334	220.95.233.172	10.0.2.15	TCP	64	http > 48378 [ACK] Seq=1 Ack=396 Win=8760 Len=0 [ETHERNET FRAME CHECK SEQUENCE INCORRECT]
29	15.792807	220.95.233.172	10.0.2.15	HTTP	441	HTTP/1.1 302 Moved Temporarily (text/html)
30	15.793071	10.0.2.15	220.95.233.172	TCP	54	48378 > http [ACK] Seq=396 Ack=388 Win=6432 Len=0
31	15.795615	220.95.233.172	10.0.2.15	TCP	64	http > 48378 [FIN, ACK] Seq=388 Ack=396 Win=8760 Len=0 [ETHERNET FRAME CHECK SEQUENCE INCORRECT]
32	15.832862	10.0.2.15	220.95.233.172	TCP	54	48378 > http [ACK] Seq=396 Ack=389 Win=6432 Len=0
33	15.966314	10.0.2.15	220.95.233.172	TCP	54	48378 > http [FIN, ACK] Seq=396 Ack=389 Win=6432 Len=0
34	15.966399	220.95.233.172	10.0.2.15	TCP	64	http > 48378 [ACK] Seq=389 Ack=397 Win=8760 Len=0 [ETHERNET FRAME CHECK SEQUENCE INCORRECT]
35	15.985935	10.0.2.15	10.0.2.3	DNS	71	Standard query 0x652f A m.naver.com
36	16.050553	10.0.2.3	10.0.2.15	DNS	135	Standard query response 0x652f CNAME m.naver.com.nheos.com A 222.122.195.17 A 202.131.29.100

- 내부의 바이너리를 이용하면 다른 조작없이 스크린샷을 찍을 수 있음
- /system/bin/screencap

```
# /system/bin/screencap -p /data/capture.png
/system/bin/screencap -p /data/capture.png
# ls /data
ls /data
anr
app
app-private
backup
backup.ko
cap.png
capture
capture.png
dalvik-cache
```

- 안드로이드 커널 소스를 수정하거나 커널 함수를 후킹하여 커널 메시지를 출력도록 함
- adb shell cat /proc/kmsg



- 준비물 : Linux Machine, git-core, gnupg, sun-java6-jdk, flex, bison, gperf, libsdl-dev, libsd0-dev, libwxgtk2.6-dev, build-essential, zip, curl, libcurses5-dev, zlib1g-dev, x11proto-core-dev, libreadline6-dev, libgl1-mesa-dev, tofrodos, libxml2-utils, xsltproc
- 기본적인 안드로이드 커널 소스는 <https://android.googlesource.com>에서 받을 수 있음
- 휴대폰 제조사별로 각 휴대폰에 쓰인 커널 소스를 공개
 - 삼성전자 : <http://opensource.samsung.com/>
 - LG 전자 : <http://www.lg.com/global/support/opensource/index>
 - HTC : <http://htcdev.com/devcenter/downloads>
 - 팬택 : <http://opensource.pantech.com/model/list.asp?Category=Mobile>
 - 모토로라 : <http://sourceforge.net/motorola/>

- Linux Machine에 소스코드 다운로드
 - \$ cd YOUR_DEV_DIRECTORY
 - \$ git clone https://android.googlesource.com/kernel/goldfish
 - \$ cd goldfish
 - \$ git branch -a
 - \$ git checkout -t origin/android-goldfish-2.6.29 -b goldfish
- 툴체인 다운로드
 - \$ cd YOUR_DEV_DIRECTORY
 - \$ git clone https://android.googlesource.com/platform/prebuilts/gcc/linux-x86/arm/arm-eabi-4.6

- 컴파일
 - \$ Make ARCH=arm CROSS_COMPILE={TOOL CHAIN DIRECTORY}/arm-eabi- distclean
\$ Make ARCH=arm CROSS_COMPILE={TOOL CHAIN DIRECTORY}/arm-eabi- goldfish-defconfig
\$ Make ARCH=arm CROSS_COMPILE={TOOL CHAIN DIRECTORY}/arm-eabi- zImage
- zImage는 arch/arm/boot/zImage에 생성됨
- emulator –avd <에뮬레이터 이름> -kernel <빌드한 zImage>

- Kprobes : 커널 코드에 원하는 코드를 동적으로 추가할 수 있음
- 커널 설정시 CONFIG_KPROBES 옵션을 선택해야 함
- kprobe, jprobe, kretprobe가 포함됨

- includes/linux/kprobes.h

```
struct kprobe {  
    struct hlist_node hlist;  
    /* list of kprobes for multi-handler support */  
    struct list_head list;  
    /*count the number of times this probe was temporarily disarmed */  
    unsigned long nmissed;  
    /* location of the probe point */  
    kprobe_opcode_t *addr;  
    /* Allow user to indicate symbol name of the probe point */  
    const char *symbol_name;  
    /* Offset into the symbol */  
    unsigned int offset;  
    /* Called before addr is executed. */  
    kprobe_pre_handler_t pre_handler;  
    /* Called after addr is executed, unless... */  
    kprobe_post_handler_t post_handler;  
    /*  
     * ... called if executing addr causes a fault (eg. page fault).  
     * Return 1 if it handled fault, otherwise kernel will see it.  
     */  
    kprobe_fault_handler_t fault_handler;  
    /*  
     * ... called if breakpoint trap occurs in probe handler.  
     * Return 1 if it handled break, otherwise kernel will see it.  
     */  
    kprobe_break_handler_t break_handler;  
    /* Saved opcode (which has been replaced with breakpoint) */  
    kprobe_opcode_t opcode;  
    /* copy of the original instruction */  
    struct arch_specific_insn ainsn;  
    /*  
     * Indicates various status flags.  
     * Protected by kprobe_mutex after this kprobe is registered.  
     */  
    u32 flags;  
};
```

- `kprobe_opcode_t *addr`
 - 커널함수의 주소를 직접 입력하여 후킹 지점을 설정
- `char *symbol_name`
 - 커널함수의 이름을 등록하여 후킹 지점을 설정
- `kprobe_pre_handler_t pre_handler`
 - 후킹한 함수가 실행되기 전에 호출될 함수를 등록
- `kprobe_post_handler_t post_handler`
 - 후킹한 함수가 실행된 후에 호출될 함수를 등록

```
#include <linux/kernel.h>
#include <linux/module.h>
#include <linux/kprobes.h>

static struct kprobe kp = {
    .symbol_name    = "do_fork",
};

static int handler_pre(struct kprobe *p, struct pt_regs *regs)
{
    printk(KERN_INFO "Fork Start!\n");
    return 0;
}

static void handler_post(struct kprobe *p, struct pt_regs *regs,
                        unsigned long flags)
{
    printk(KERN_INFO "Fork Finish!\n");
    return 0;
}
```

```
static int __init kprobe_init(void)
{
    int ret;
    kp.pre_handler = handler_pre;
    kp.post_handler = handler_post;

    ret = register_kprobe(&kp);
    if (ret < 0) {
        printk(KERN_INFO "register_kprobe failed, returned %d\n", ret);
        return ret;
    }
    printk(KERN_INFO "Planted kprobe at %p\n", kp.addr);
    return 0;
}

static void __exit kprobe_exit(void)
{
    unregister_kprobe(&kp);
    printk(KERN_INFO "kprobe at %p unregistered\n", kp.addr);
}

module_init(kprobe_init)
module_exit(kprobe_exit)
MODULE_LICENSE("GPL");
```

- Makefile에 소스 추가 후 컴파일

```
obj-$(CONFIG_REMOTEPROC)      += remoteproc/
obj-$(CONFIG_RPMSG)           += rpmsg/

# Virtualization drivers
obj-$(CONFIG_VIRT_DRIVERS)    += virt/
obj-$(CONFIG_HYPERV)           += hv/

obj-$(CONFIG_PM_DEVFREQ)       += devfreq/
obj-$(CONFIG_EXTCON)           += extcon/
obj-$(CONFIG_MEMORY)           += memory/
obj-$(CONFIG_IIO)               += iio/
obj-$(CONFIG_VME_BUS)          += vme/
obj-$(CONFIG_IPACK_BUS)        += ipack/
obj-m                         += hook_mod.o
```

```
kimchangsoo@kimchangsoo-VirtualBox:~/haha/kernel3/drivers$ ls
accessibility  bcma          clk          devfreq   firmware   hsi          iio          Kconfig
acpi           block         clocksource  dio        gpio       hv          infiniband  leds
amba           bluetooth    connector    dma        gpu        hwmon      input       lguest
ata            built-in.o  cpufreq     edac      hid        hwspinlock iommu      macintosh
atm            bus          cpuidle     eisa      hook_mod.c i2c        ipack      Makefile
auxdisplay    cdrom        crypto      extcon   hook_mod.ko ide        irqchip    md
base           char         dca        firewire  hook_mod.o idle      isdn       media
```

- Adb를 이용해 모듈을 기기로 복사한 후, insmod 명령을 이용해 커널 모듈을 설치

```
C:\APK_DYNAMIC\etc>adb push hook_mod.ko /data  
46 KB/s (10461 bytes in 0.219s)  
  
C:\APK_DYNAMIC\etc>adb shell  
# cd data  
cd data  
# insmod hook_mod.ko  
insmod hook_mod.ko
```

- adb shell cat /proc/kmsg 를 이용하면 printk로 출력되는 내용을 확인 가능

10

Kernel Hooking

- jprobe
 - 함수에 전해지는 파라미터를 확인할 수 있음

```
#include <linux/kernel.h>
#include <linux/module.h>
#include <linux/kprobes.h>

static long jdo_fork(unsigned long clone_flags, unsigned long stack_start,
                     struct pt_regs *regs, unsigned long stack_size,
                     int __user *parent_tidptr, int __user *child_tidptr)
{
    printk(KERN_INFO "jprobe: clone_flags = 0x%lx, stack_size = 0x%lx,"
           " regs = 0x%p\n",
           clone_flags, stack_size, regs);

    jprobe_return();
    return 0;
}

static struct jprobe my_jprobe = {
    .entry            = jdo_fork,
    .kp = {
        .symbol_name = "do_fork",
    },
};
```

```
static int __init jprobe_init(void)
{
    int ret;

    ret = register_jprobe(&my_jprobe);
    if (ret < 0) {
        printk(KERN_INFO "register_jprobe failed, returned %d\n", ret);
        return -1;
    }
    printk(KERN_INFO "Planted jprobe at %p, handler addr %p\n",
           my_jprobe.kp.addr, my_jprobe.entry);
    return 0;
}

static void __exit jprobe_exit(void)
{
    unregister_jprobe(&my_jprobe);
    printk(KERN_INFO "jprobe at %p unregistered\n", my_jprobe.kp.addr);
}

module_init(jprobe_init)
module_exit(jprobe_exit)
MODULE_LICENSE("GPL");
```

- kretprobe
 - 함수의 반환값을 확인할 수 있음

```
#include <linux/kernel.h>
#include <linux/module.h>
#include <linux/kprobes.h>
#include <linux/ktime.h>
#include <linux/limits.h>
#include <linux/sched.h>

static char func_name[NAME_MAX] = "do_fork";
module_param_string(func, func_name, NAME_MAX, S_IRUGO);
MODULE_PARM_DESC(func, "Function to kretprobe; this module will report the"
                 " function's execution time");

struct my_data {
    ktime_t entry_stamp;
};

static int entry_handler(struct kretprobe_instance *ri, struct pt_regs *regs)
{
    struct my_data *data;

    if (!current->mm)
        return 1; /* Skip kernel threads */

    data = (struct my_data *)ri->data;
    data->entry_stamp = ktime_get();
    return 0;
}

static int ret_handler(struct kretprobe_instance *ri, struct pt_regs *regs)
{
    int retval = regs_return_value(regs);
    struct my_data *data = (struct my_data *)ri->data;
    s64 delta;
    ktime_t now;

    now = ktime_get();
    delta = ktime_to_ns(ktime_sub(now, data->entry_stamp));
    printk(KERN_INFO "%s returned %d and took %lld ns to execute\n",
          func_name, retval, (long long)delta);
    return 0;
}
```

```
static struct kretprobe my_kretprobe = {
    .handler           = ret_handler,
    .entry_handler     = entry_handler,
    .data_size         = sizeof(struct my_data),
    /* Probe up to 20 instances concurrently. */
    .maxactive         = 20,
};

static int __init kretprobe_init(void)
{
    int ret;

    my_kretprobe.kp.symbol_name = func_name;
    ret = register_kretprobe(&my_kretprobe);
    if (ret < 0) {
        printk(KERN_INFO "register_kretprobe failed, returned %d\n",
               ret);
        return -1;
    }
    printk(KERN_INFO "Planted return probe at %s: %p\n",
           my_kretprobe.kp.symbol_name, my_kretprobe.kp.addr);
    return 0;
}

static void __exit kretprobe_exit(void)
{
    unregister_kretprobe(&my_kretprobe);
    printk(KERN_INFO "kretprobe at %p unregistered\n",
           my_kretprobe.kp.addr);

    /* nmissed > 0 suggests that maxactive was set too low. */
    printk(KERN_INFO "Missed probing %d instances of %s\n",
           my_kretprobe.nmissed, my_kretprobe.kp.symbol_name);
}

module_init(kretprobe_init)
module_exit(kretprobe_exit)
MODULE_LICENSE("GPL");
```

Chapter 3

Automation System



01

Necessity

- 모든 악성 어플리케이션을 일일이 분석하기에는 많은 시간이 필요함

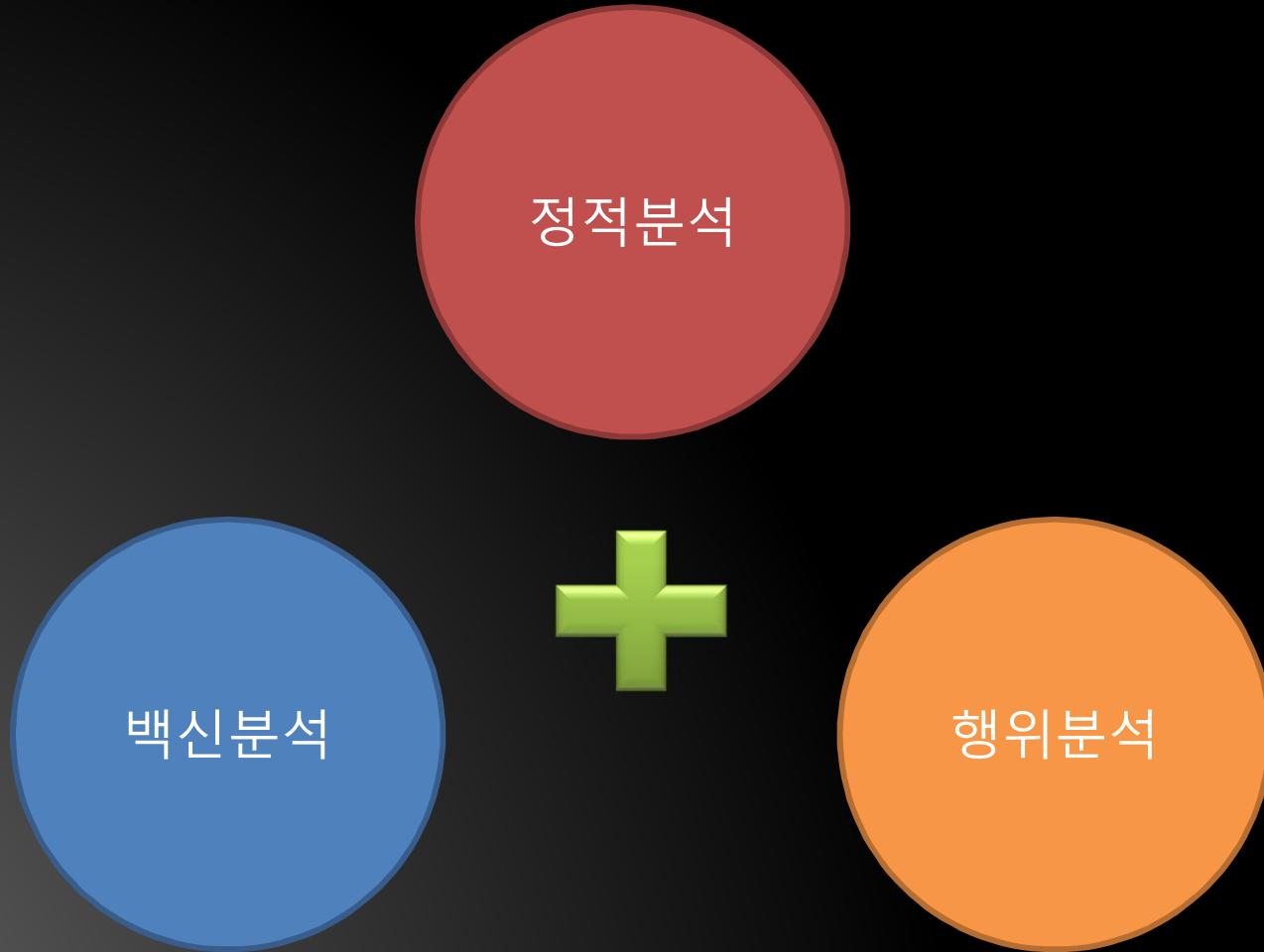


- 24시간 감시가 가능



02

System Component



03

Procedure

1. 수집채널을 통한 샘플 수집

2. 백신 검사

3. 권한 분석

4. API 추출

5. 행위 분석

04

DEMO

Q & A



Thank You !



www.CodeEngn.com | www.RCEHub.com

Hacking Android network

singi
sjh21a@gmail.com
FB : @sjh21a

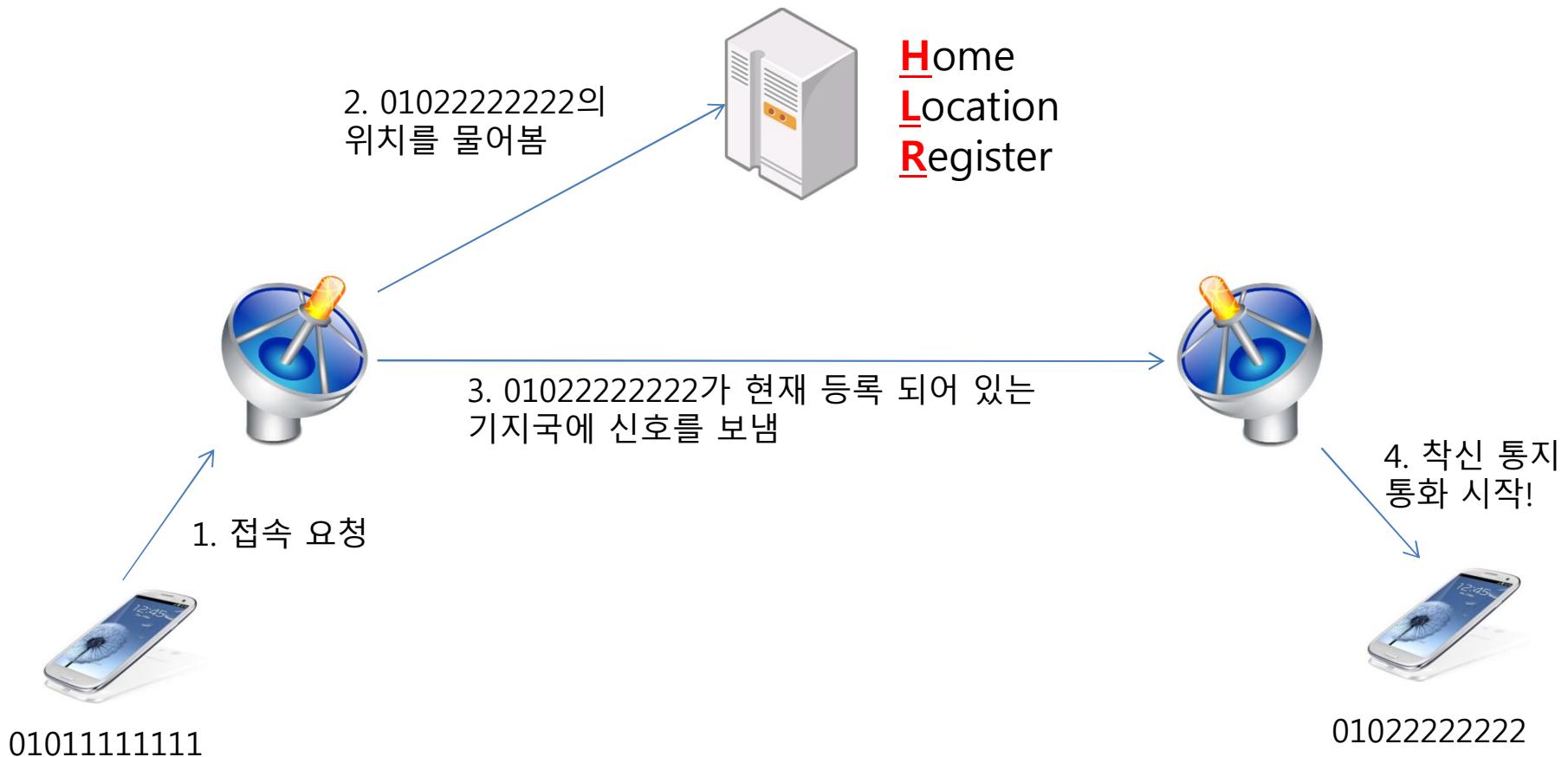
오늘의 목차는?

- 기본 3G/4G 통신망 구조 파악 (**쉬움!**)
- Android 통신구조 (**쉬움!**)
 - Android RIL
 - Vendor-RIL firmware 분석 해 보기
- AT command 종류 및 사용법 (**쉬움!**)
 - AT command로 SMS / 전화 해 보기
- RILD 후킹 및 제어 (**ㅅㅂ**)
 - 전화기에 fuzzing을 해보자!!

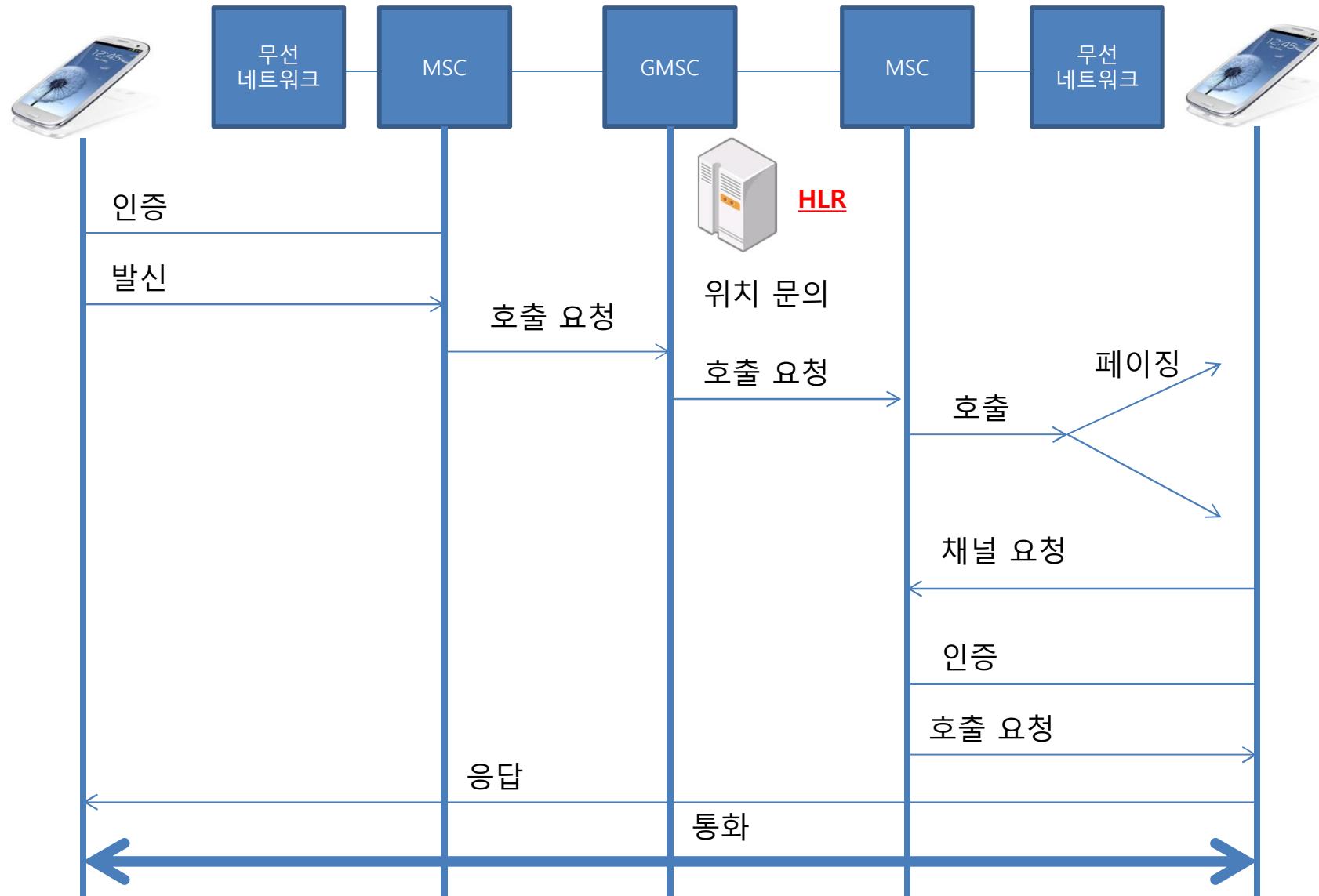
3G? 4G?

- 3G?
 - IMT-2000
 - 주파수 대역 : 2GHz (2000kHz)
 - WCDMA(2~2.4Mbps) -> HSPDA(14.4 Mbps)
- 4G?
 - IMT Beyond 또는 IMT-Advanced
 - All IP Based
 - LTE (는 아님)

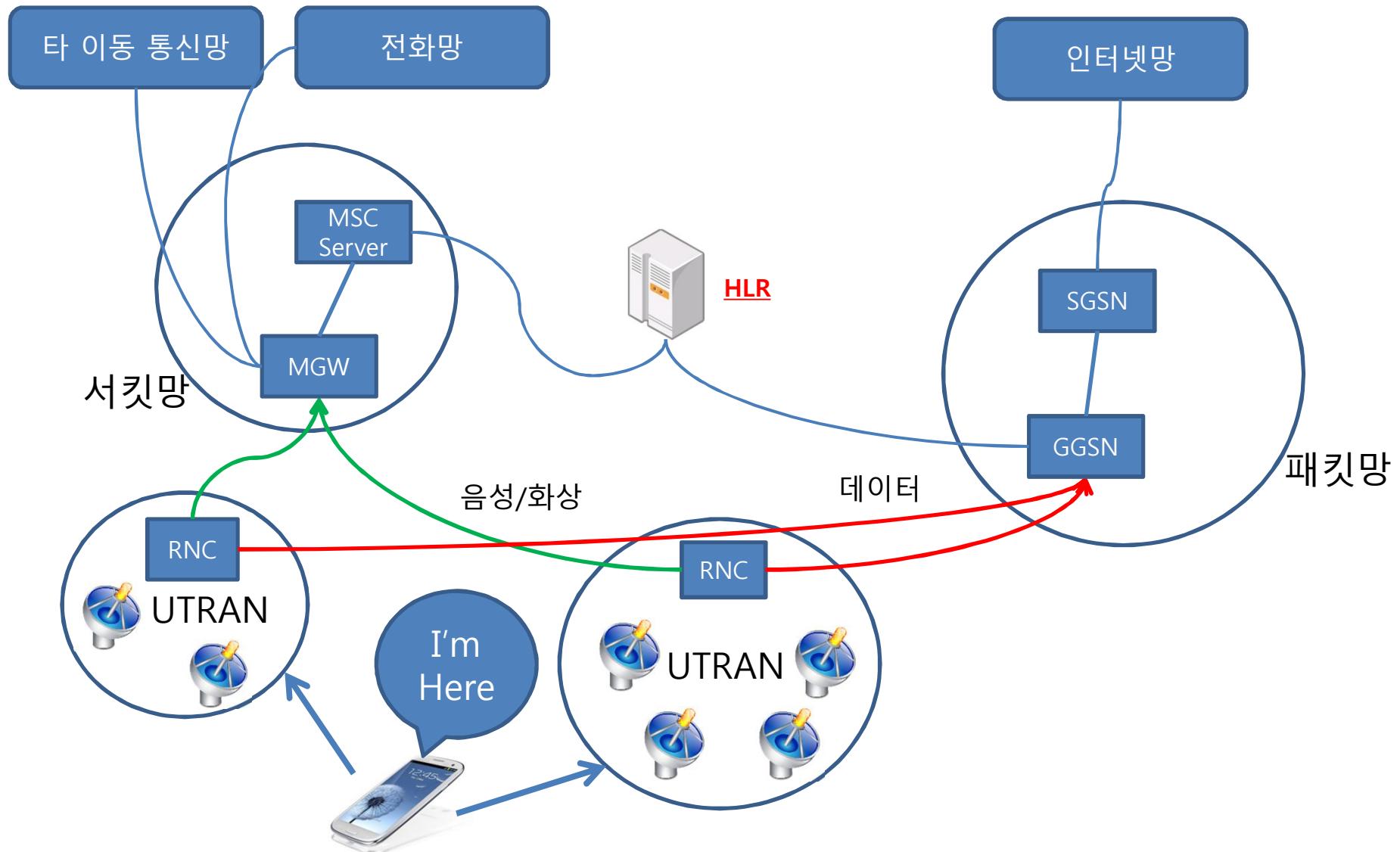
무선 전화는 어떻게 걸리고 받을까?



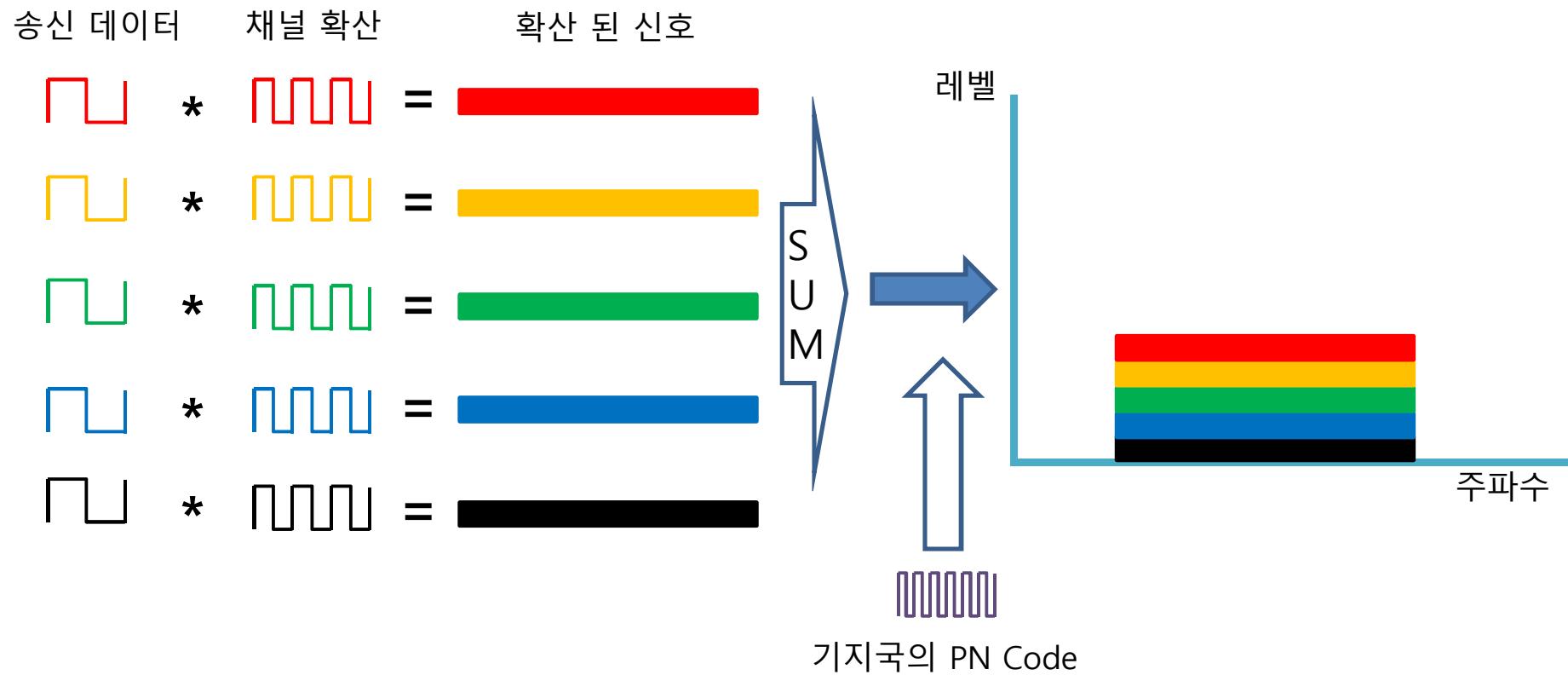
더 자세하게!



3G Network

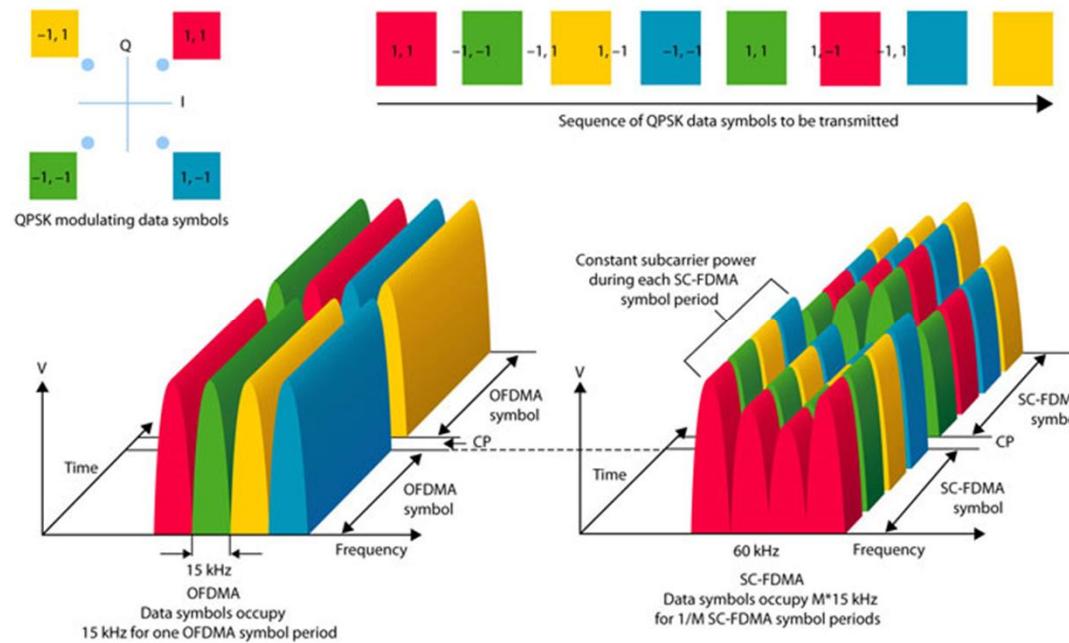


3G 통신 구조(WCDMA)



4G 통신 구조(LTE)

- OFDMA(Orthogonal Frequency Division Multiple Access)

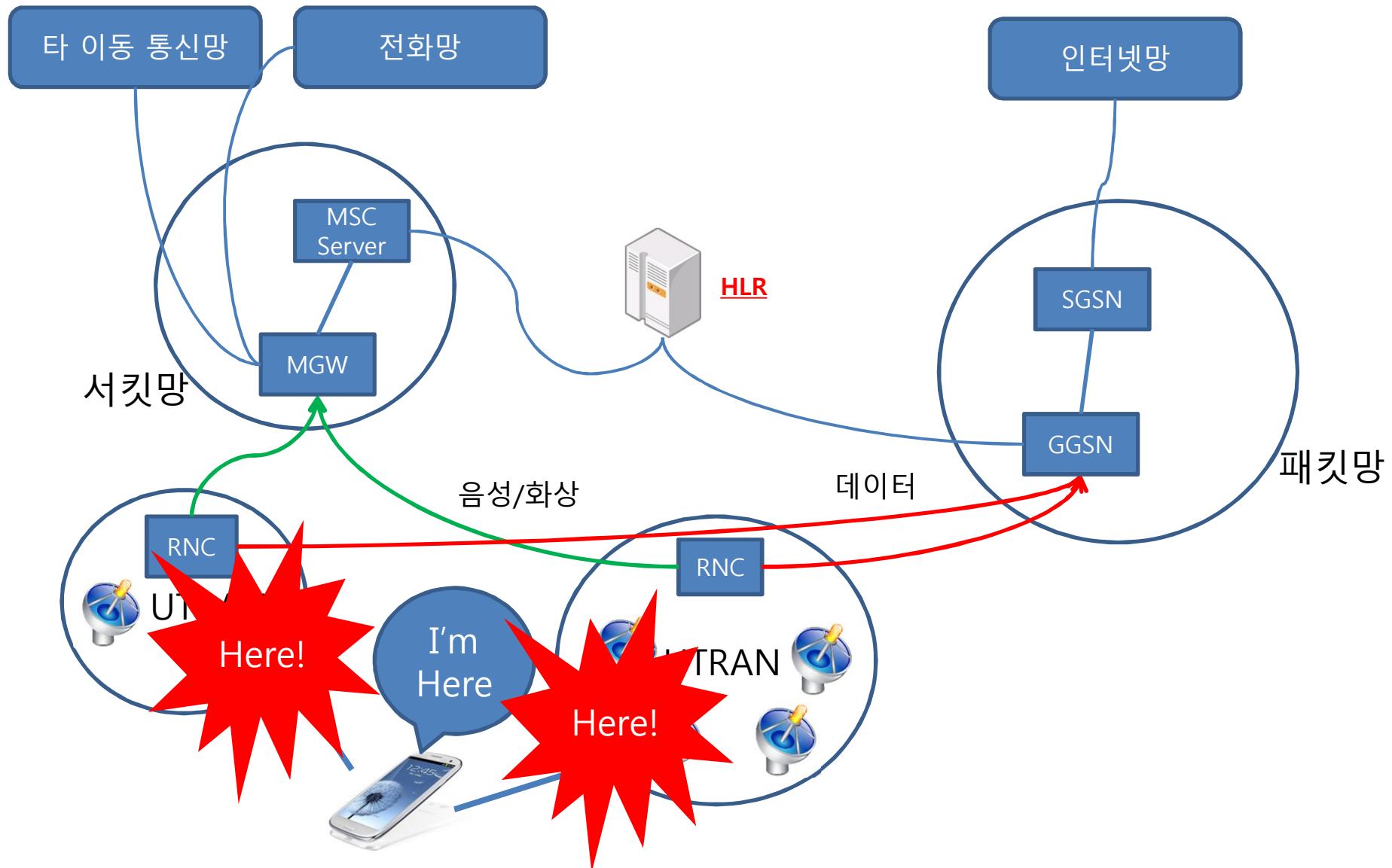


아... 이제 그만... 정확히 뭘 해보려고???

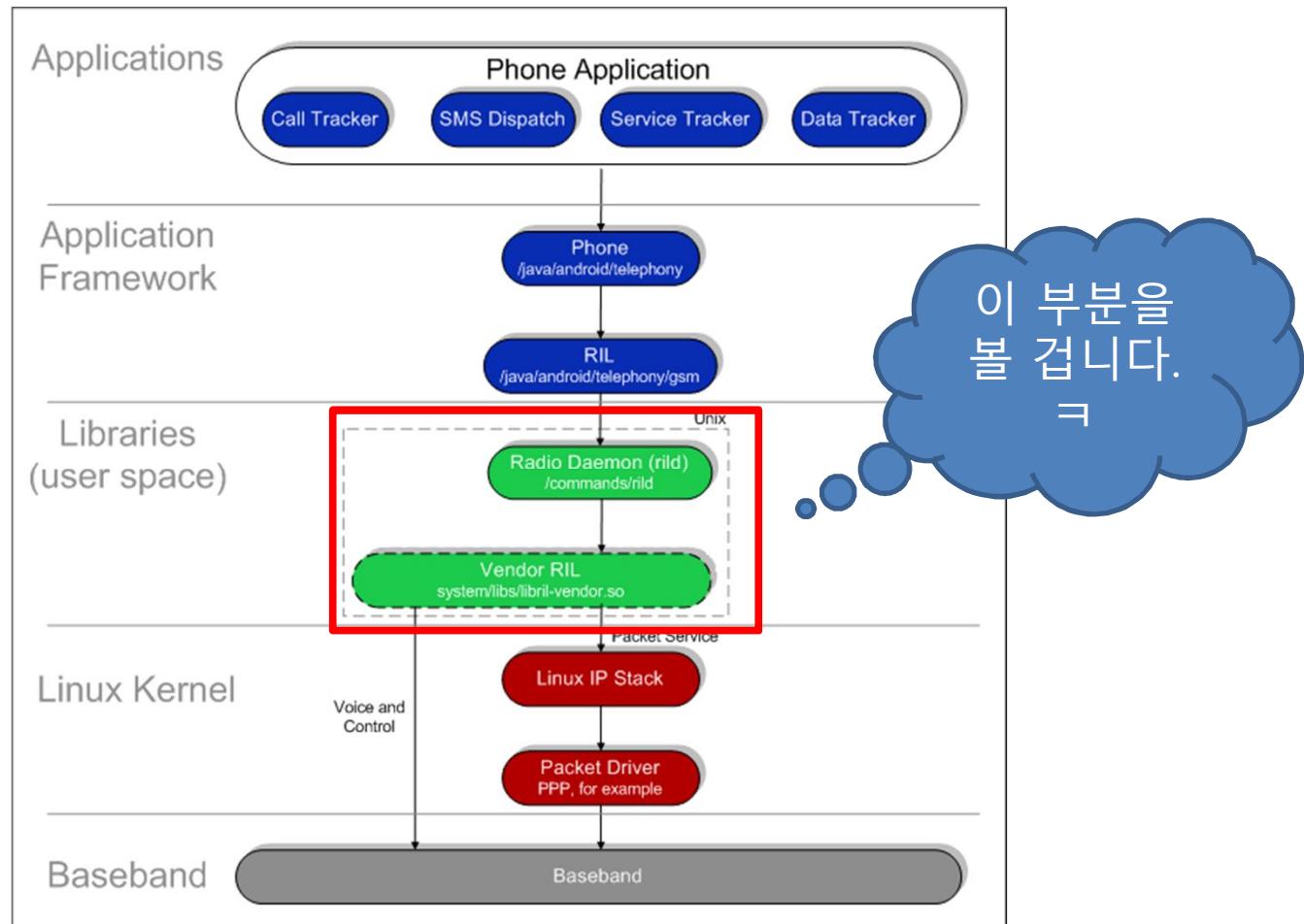
- 대상은 Galaxy S3 / Android
 - Android의 통신 구조
 - RIL, Vendor-RIL
 - Vendor-RIL 추출 / 리버싱
- AT command
 - 단말기에 AT command를 보내기
- RIL 후킹
 - 문자/전화/AT command를 사용 할 때 RIL 구조 파악!



이 부분을 자세히 볼 거예요!



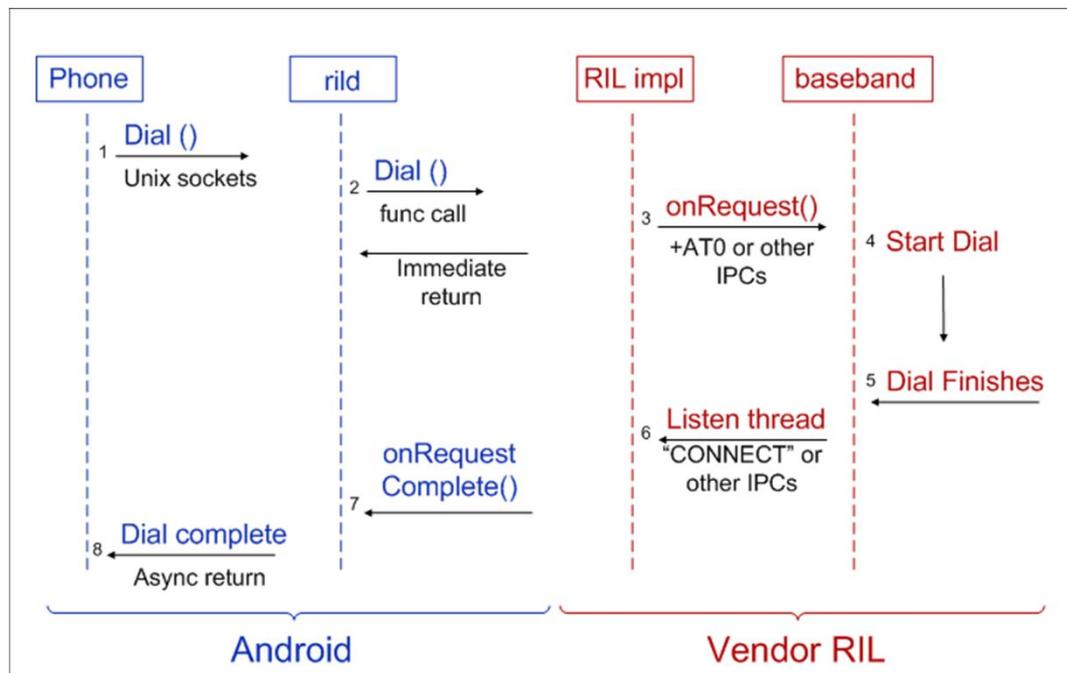
Android 통신 구조를 봅시다.



RIL? Vendor RIL?

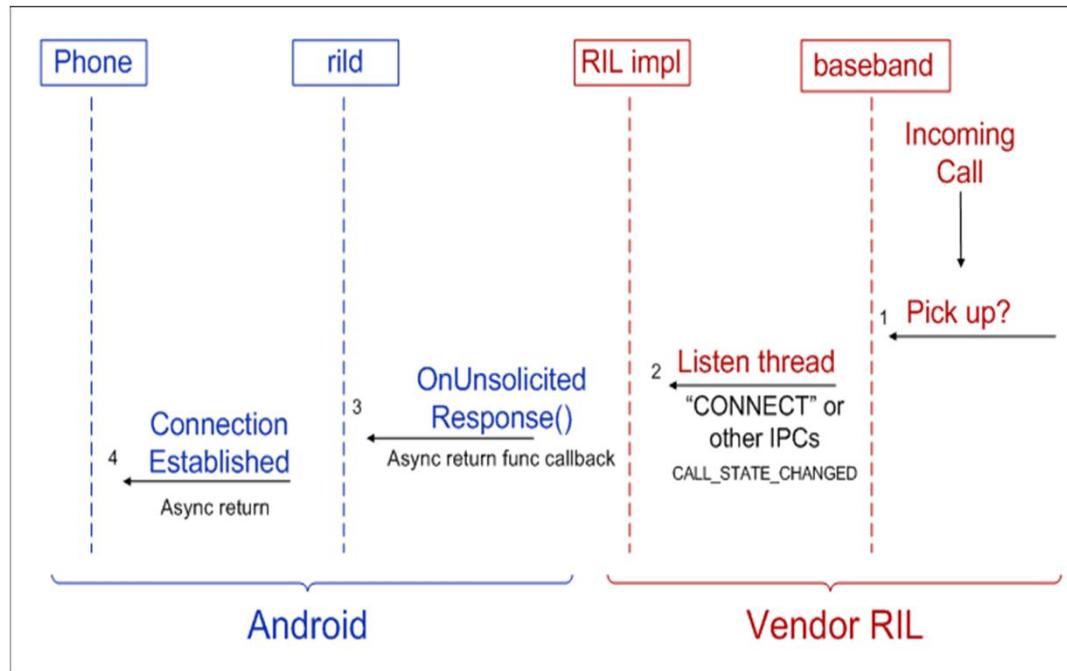
- RIL
 - Android.telephony 와 radio h/w 간의 layer
- RILD(daemon)
 - Android.telephony의 요청을 처리함.
 - Solicited 명령을 통해 Vendor RIL로 넘김.
 - Solicited?
- Vendor RIL
 - Radio h/w의 모든 통신 담당.
 - Unsolicited 명령을 통해 RILD로 넘김.
 - Unsolicited??

Solicited 명령?



- Solicited
 - Android에서 modem쪽으로 통신 할 때 사용됨.
 - SMS 송신, Network, etc...

Unsolicited 명령?



- Unsolicited?
 - Modem쪽에서 Android로 통신 할 때 사용됨.
 - 전화 받기, SMS 수신, etc...

RIL 분석은 어떻게 할까??

- RIL는 OpenSource 다!
 - 그냥 볼 수 있음.
 - Java와 C로 이루어져 있음.
- 하지만, Vendor-RIL은 얘기가 다르다.
 - 각 제조사마다 자신들만의 modem 코드들을 가지고 있음.
 - 분석해 볼 것은 Samsung Galaxy S3의 Modem 코드.
 - 어떻게 분석을 시작해 볼까?

업데이트 시, firmware 추출!

문서 ▶ KKLUCKTTALUCKKKLUJC ▶			
함	공유 대상	새 폴더	
이름	수정한 날짜	유형	크기
📁 KIES_HOME_E210KKKLUJC_E210KTTALJC_371046_REV00_user_low_ship	2012-12-12 오후...	파일 폴더	
tar KIES_HOME_E210KKKLUJC_E210KTTALJC_371046_REV00_user_low_ship.tar	2012-10-26 오전...	ALZip TAR File	1,285,901...
SS_DL.dll	2012-10-26 오전...	응용 프로그램 확장	270KB

이름	수정한 날짜	유형	크기
boot.img	2012-10-25 오후...	디스크 이미지 파일	4,853KB
cache.img	2012-10-25 오후...	디스크 이미지 파일	32,137KB
hidden.img	2012-10-25 오후...	디스크 이미지 파일	14,405KB
modem.bin	2012-10-25 오후...	BIN 파일	18,804KB
modem.idb	2012-12-12 오후...	IDB 파일	75,281KB
recovery.img	2012-10-25 오후...	디스크 이미지 파일	5,541KB
system.img	2012-10-25 오후...	디스크 이미지 파일	1,210,156...

ㅋㅋ
이미 분석
을 했네요

Modem.bin 분석!

Offset	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F	10	11	12	13	14	15	16	17	18	19	1A	1B	1C	1D	1E	1F		
000000000	42	4F	4F	54	00	00	00	00	00	00	00	00	00	02	00	00	00	00	00	50	16	00	00	B4	BD	86	B4	00	00	00	00			
000000020	4D	41	49	4E	00	00	00	00	00	00	00	00	00	60	18	00	00	00	00	40	EC	B5	25	01	24	3C	5A	F5	00	00	00	00		
000000040	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00		
000000060	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	
000000080	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	
0000000A0	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	
0000000C0	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	
0000000E0	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	
000000100	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	
000000120	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	
000000140	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	
000000160	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	
000000180	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	
0000001A0	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	
0000001C0	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	
0000001E0	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	
000000200	16	00	00	EA	11	F1	A0	E3	21	F1	A0	E3	31	F1	A0	E3	41	F1	A0	E3	51	F1	A0	E3	61	F1	A0	E3	71	F1	A0	E3		
000000220	00	52	45	56	40	16	00	00	45	54	41	44	44	16	00	00	45	5A	49	53	50	16	00	00	00	00	41	48	53	00	00	00	00	00
000000240	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	
000000260	D1	F0	21	E3	06	CB	A0	E3	FF	00	AC	E8	0C	00	A0	E1	1F	F0	21	E3	00	9F	A0	E8	00	10	4F	E1	02	60	A0	E8		

IDA로 열어봅시다♥

```
ROM:0005AF6C
ROM:0005AF6C ; ====== S U B R O U T I N E ======
ROM:0005AF6C
ROM:0005AF6C
ROM:0005AF6C
ROM:0005AF6C sub_5AF6C ; CODE XREF: sub_942920:loc_9429A4↓p
ROM:0005AF6C     LDR    R0, =0x4118CA90
ROM:0005AF6E     LDR    R0, [R0,#4]
ROM:0005AF70     BX    LR
ROM:0005AF70 : End of function sub_5AF6C
ROM:0005AF70
ROM:0005AF72
ROM:0005AF72 ; ====== S U B R O U T I N E ======
ROM:0005AF72
ROM:0005AF72
ROM:0005AF72
ROM:0005AF72 sub_5AF72 ; CODE XREF: sub_942920+14↓p
ROM:0005AF72     LDR    R0, =0x4118CA90
ROM:0005AF74     LDRB   R0, [R0]
ROM:0005AF76     BX    LR
ROM:0005AF76 : End of function sub_5AF72
ROM:0005AF76
ROM:0005AF78
ROM:0005AF78 ; ====== S U B R O U T I N E ======
ROM:0005AF78
ROM:0005AF78
ROM:0005AF78
ROM:0005AF78 sub_5AF78 ; CODE XREF: sub_942920+30↓p
ROM:0005AF78     LDR    R1, =0x42EC84E4
ROM:0005AF7A     MOVS   R0, #0
ROM:0005AF7C     LDR    R2, =0x4118CA90
ROM:0005AF7E     MOVS   R3, #0x12
ROM:0005AF80     STR    R0, [R1,#0xC]
ROM:0005AF82     STRB   R0, [R2]
ROM:0005AF84     STR    R3, [R2,#4]
ROM:0005AF86     ADR    R3, unk_5B038
ROM:0005AF88     STR    R3, [R1,#4]
ROM:0005AF8A     ADR    R3, unk_5B0A8
```

이게 뭐지!?ㅋㅋ

```
ROM:0005BDF7 0000001E C A<<< LTE DM Command List >>>\nROM:0005BE18 00000011 C [01] ltedm help\nROM:0005BE2C 0000000F C [02] ltedm on\nROM:0005BE3C 00000010 C [03] ltedm off\nROM:0005BE4C 00000014 C [04] ltedm default\nROM:0005BE60 0000001A C [05] ltedm vu [RATE]\nROM:0005BE7C 0000001B C ex) ltedm vu 1000 \nROM:0005BE98 0000002F C -> LTEDM_VU update rate is 1000ms\nROM:0005BEC8 0000001A C [06] ltedm phy [RATE]\nROM:0005BEE4 0000001A C [07] ltedm l1 [RATE]\nROM:0005BF00 0000001A C [08] ltedm mac [RATE]\nROM:0005BF1C 0000001A C [09] ltedm rlc [RATE]\nROM:0005BF38 0000001A C [10] ltedm pdcp [RATE]\nROM:0005BF54 0000001A C [11] ltedm rrc [RATE]\nROM:0005BF70 0000001A C [12] ltedm nas [RATE]\nROM:0005BF8C 0000001A C [15] ltedm pal [RATE]\nROM:0005BFA8 00000013 C [16] ltedm status\nROM:0005BFBC 00000024 C \nCALPSS\\LteCommon\\Code\\Src\\lte_dm.c\nROM:0005BFE0 0000000D C LTE DM : ON\nROM:0005BFF0 0000000E C LTE DM : OFF\nROM:0005C003 00000018 C @<<< LTE DM Status >>>\nROM:0005C0... 00000010 C %s Rate : %dms\nROM:0005C0... 0000001C C LTE DM set a default value\nROM:0005C048 00000017 C LTE DM VU Rate : %dms
```

LTE Device Manager Command List

통신사 정보 알아내기!

- /data/data/com.android.providers.telephony/databases/telephony.db
 - 각 국 3g/4g 이동통신망 사업자의 접속 정보

21	21	Mobistar	20610	206	10	mworld.be	mobistar		mobistar	212.65.63.143	8080		
22	22	Base	20620	206	20	gprs.base.be	base		base	172.31.198.37	5080		
23	23	BASE MMS	20620	206	20	mms.base.be	base		base		217.72.235.1	8080	http://mmsc.bas
24	24	Orange World	20801	208	01	orange	orange		orange				
25	25	Orange MMS	20801	208	01	orange.acte	orange		orange		192.168.10.200	8080	http://mms.oran
26	26	Orange Entrepris	20801	208	01	orange-mib	orange		orange	172.16.2.8	8000		
27	27	Orange Internet	20801	208	01	orange.fr	orange		orange				
28	28	Orange Internet E	20801	208	01	internet-entrepris	orange		orange				

- /data/data/com.android.providers.telephony/databases/nwk_info.db
 - 현재 기기에 설정 되어 있는 이동통신망 접속 정보

_id	name	numeric	mcc	mnc	apn	user	server	password	proxy	port
1	KT IMS	45008	450	08	ims.ktfwing.com					
2	KT	45008	450	08	lte.ktfwing.com					

AT Command?

- 팩스 모뎀용으로 개발 됨.
- 상업적으로 표준화가 되어, 전 세계에서 사용 중.
- 일반적으로, Terminal을 통해 사용자와 모뎀이 명령어를 주고 받음.
- Android 에서 사용되는 3G Modem의 AT command?

AT Command LIST

- 3GPP의 범용 AT Command

명령어	기능
AT+CAAP	Automatic answer for eMLPP Service
AT+CACM	Accumulated call meter
AT+CAHLD	Leave an ongoing Voice Group or Voice Broadcast Call
AT+CFUN	Set phone functionality
AT+CREG	Network registration
AT+CMOD	Call mode
...이 아래 빙산이 있습니다.	

- 참고 자료 [5], [6]번을 참고 하시면 됨.

AT command의 사용 방법은??

- RIL Daemon에서 사용하는 디바이스 드라이버를 찾아야 함.
 - /system/build.prop
 - /dev/smd0 or /dev/ttyGS0
 - /dev/ttyS0
- 명령어 전송은 어떻게 할까?
 - Exam> echo -ne "ATWr" > [Device]
 - Galaxy S 시리즈에서도 가능.
 - 단, 해외 버전만... 가능 π_π

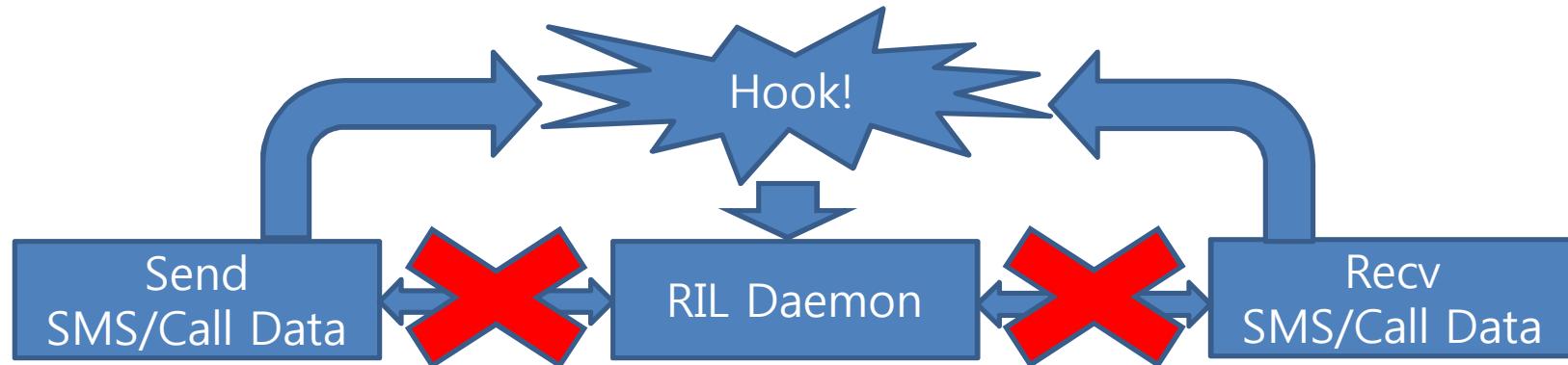
```
#  
# system.prop for sdk4x12  
#  
  
ril.libpath=/system/lib/libsec-ril.so  
ril.libargs=-d /dev/ttyS0  
ro.sf.lcd_density=320  
ro.lcd_min_brightness=20
```



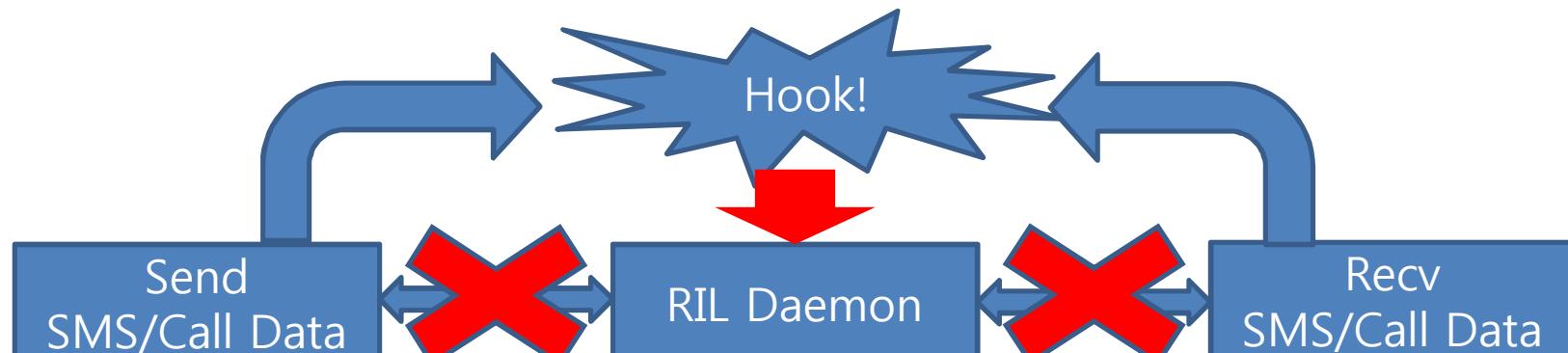
그럼 제 발표는 여기서...?

- AT Command는 전송은 디바이스 문제로 포기!
- 그럼 이제 무엇을? 이대로 진짜 끝?
- rild을 Hooking 해 보기!
- Rild을 통해 오고 가는 SMS/Call 데이터를 제어 하기!
- 잘 되려나? ㅋㅋㅋ

다시 한번 설명 해보면,



그냥 엿보기만!



데이터
변조!
퍼징!

무엇을 어떻게 할까?

- Android용으로 포팅 된 Hijack 툴을 사용.
 - <http://www.mulliner.org/android/>
- 후킹할 대상은?

```
u0_a180@android:/data/data/berserker.android.apps.sshdroid/home # ps | grep rild
28684 radio      0:00 /system/bin/rild
28787 root       0:00 grep rild
u0_a180@android:/data/data/berserker.android.apps.sshdroid/home # cat /proc/28684/maps
```

401a4000-4025a000 r-xp 00000000 b3:09 1366	/system/lib/libsec-ril.so
4025a000-4025b000 r--p 000b5000 b3:09 1366	/system/lib/libsec-ril.so
4025b000-40260000 rw-p 000b6000 b3:09 1366	/system/lib/libsec-ril.so

잘 들어갔는지 확인!ㅋㅋ

```
/data/local/tmp # ./hijack -p 1669 -l libsingi.so -d
mprotect: 0x400e85d8
dlopen: 0xb000585d
pc=400e8c74 lr=400f52fd sp=beed3b68 fp=beed3d38
r0=fffffc r1=beed3b70
r2=0 r3=0
stack: 0xbeeb3000-0xbeed4000 leng = 135168
executing injection code at 0xbeed3b18
library injection completed!
/data/local/tmp # cat /proc/1669/maps
00008000-00009000 r-xp 00000000 5d:18 183          /system/bin/rild
00009000-0000a000 rw-p 00001000 5d:18 183          /system/bin/rild
00832000-00837000 rw-p 00000000 00:00 0           [heap]
10000000-10001000 ---p 00000000 00:00 0
10001000-10100000 rw-p 00000000 00:00 0
40016000-40017000 r--p 00000000 00:00 0
40020000-40035000 r-xp 00000000 5d:18 818          /system/lib/libm.so
40035000-40036000 rw-p 00015000 5d:18 818          /system/lib/libm.so
40059000-4005c000 r-xp 00000000 5d:18 817          /system/lib/liblog.so
4005c000-4005d000 rw-p 00003000 5d:18 817          /system/lib/liblog.so
4005d000-4006c000 r-xp 00000000 5d:18 778          /system/lib/libcutils.so
4006c000-4006d000 rw-p 0000f000 5d:18 778          /system/lib/libcutils.so
4006d000-4007c000 rw-p 00000000 00:00 0
4007c000-4007f000 r-xp 00000000 5d:18 1248         /system/lib/libsingi.so
4007f000-40080000 rw-p 00002000 5d:18 1248         /system/lib/libsingi.so
40091000-40099000 r--s 00000000 00:0c 462          /dev/_properties_ (deleted)
40099000-400b0000 r-xp 00000000 5d:18 883          /system/lib/libz.so
400b0000-400b1000 rw-p 00017000 5d:18 883          /system/lib/libz.so
400b1000-400d4000 r-xp 00000000 5d:18 765          /system/lib/libbinder.so
400d4000-400da000 rw-p 00023000 5d:18 765          /system/lib/libbinder.so
400dc000-4011e000 r-xp 00000000 5d:18 766          /system/lib/libc.so
4011e000-40121000 rw-p 00042000 5d:18 766          /system/lib/libc.so
```

Opcode 패치 결과

```
/data/local/tmp # cat log
libt loaded...
pty created, file name: /dev/pts/2
can't find: at_send_command_sms
libt _init done.
libt loaded...
pty created, file name: /dev/pts/1
can't find: at_send_command_sms
libt _init done.
libsingi loaded...
pty created, file name: /dev/pts/1
hooking  at send command sms = 40181ab9  hook = 4007e6a8  target:THUMB
37 b5 0d 46 1c 46 00 93 02 21 2b 46 ff f7 68 ff 23 1e 18 bf
30 b4 03 a5 2d 68 02 b0 20 b4 81 b0 37 b5 0d 46 1c 46 00 93 libsingi _init done.
```

It's fuzzing time!

[Demo]

(뻥인지 아닌지 시험!)

감사합니다!

(서로 질문은 없는걸로ㅋㅋㅋ)

- 참고 자료

[1] 쉽게 설명한 3G/4G 이동 통신 시스템/이상근 외 공저/홍릉 과학 출판사

[2] WiBro/WiMAX LTE 모바일 브로드밴드/강철희 외 편저/광문각

[3] <http://www.kandroid.org/online-pdk/guide/telephony.html>

[4]

http://mobiledevdesign.com/tutorials/WIreless_Everywhere_Not_Quite_Yet/index2.html

[5] http://mod-book.ru/files/Gobi2k/Documents/AT_Command_Set_Gobi.pdf

[6] <http://forum.xda-developers.com/showthread.php?t=1471241>

[7] <http://www.mulliner.org/android/>

www.CodeEngn.com | www.RCEHub.com

Windows 8 Exploit

2013. 07.13

서만기 연구원

Code**Engn**

www.CodeEngn.com

2013 CodeEngn Conference 08

AhnLab

Copyright (C) AhnLab, Inc. All rights reserved.

Contents

01 Exploiting 개요

02 Windows 8 Memory Protection

03 Exploit Writing Technique

AhnLab

Copyright (C) AhnLab, Inc. All rights reserved.

01 Exploiting 개요

AhnLab

Copyright (C) AhnLab, Inc. All rights reserved.

Exploit Writing Process

Application
Injection
Vector 파악

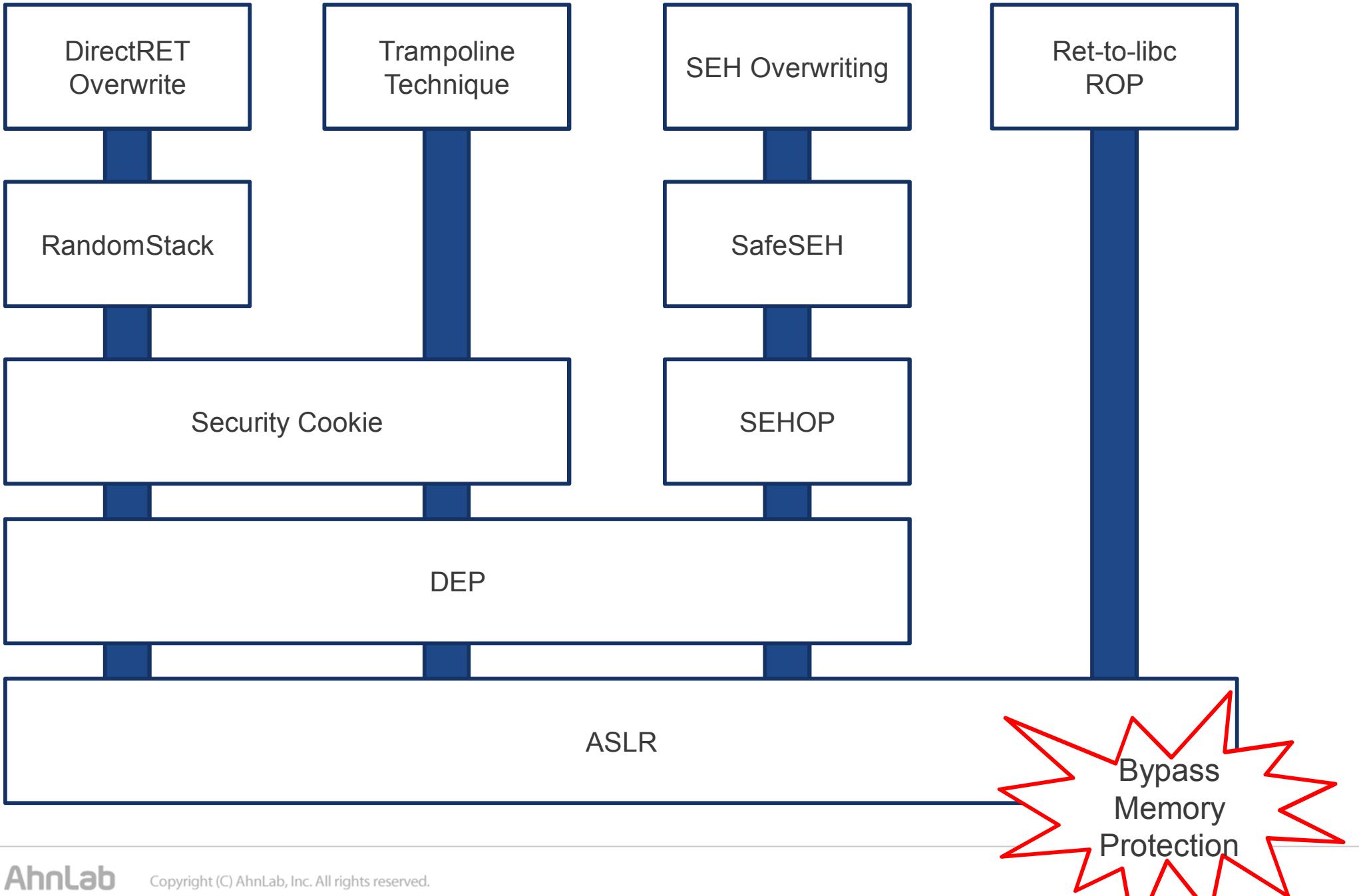
Fuzzing을 통한
Data 주입

Application
Monitoring

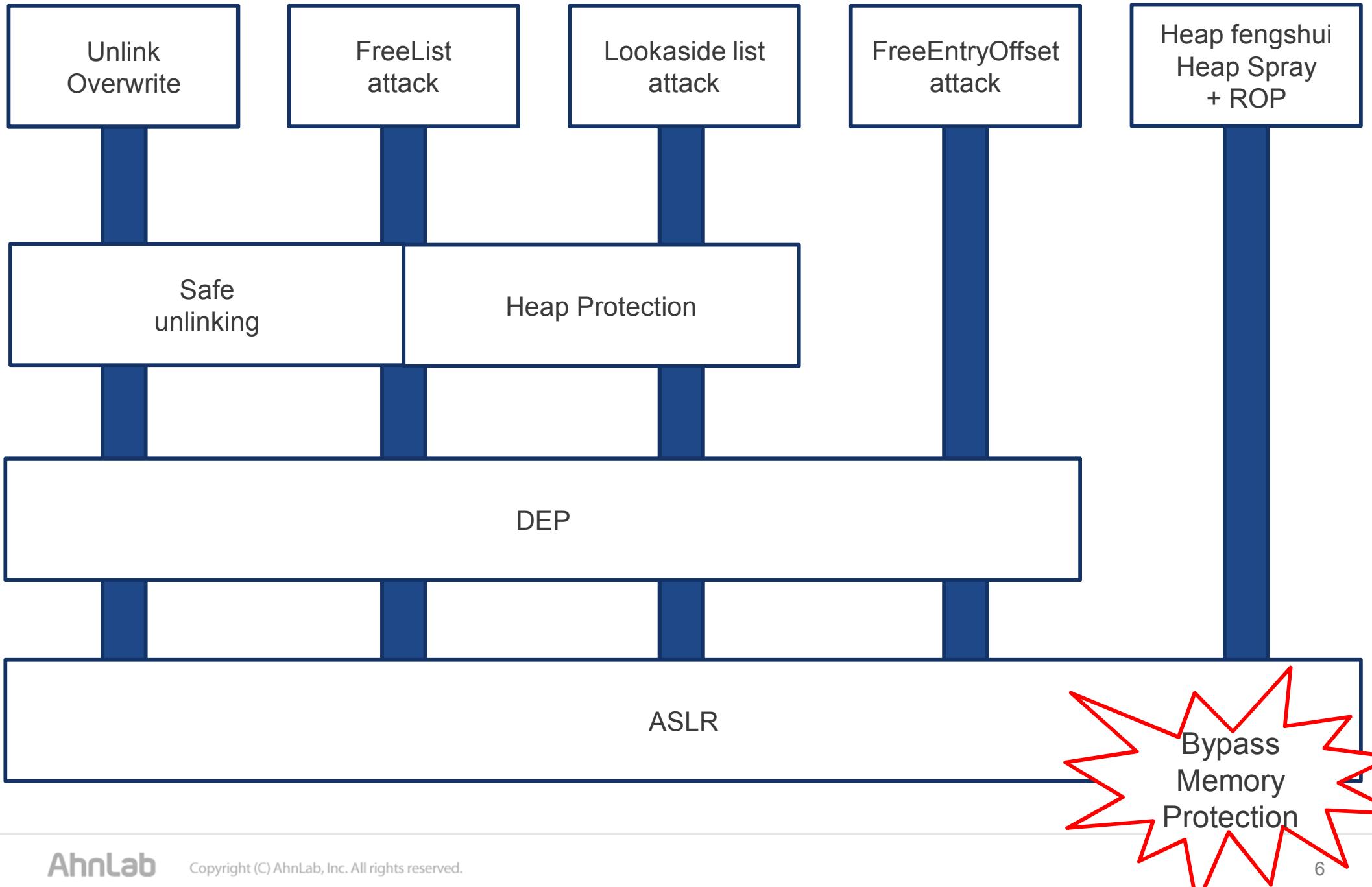
bad
character
Check

Exploit
Code
Design

History Of Stack Exploiting



History Of Heap Exploiting



02 Windows 8 Memory Protection

AhnLab

Copyright (C) AhnLab, Inc. All rights reserved.

❖ ASLR

- ◆ Address Space Layout Randomization의 약자
- ◆ Windows Vista에 처음 활용 되었음
- ◆ PE파일 포맷을 가지는 파일(이미지)이 메모리에 로드 될 때 base 주소를 랜덤하게 생성
- ◆ Image base뿐만 아니라 stack, heap, 프로세스의 메모리 공간 역시 랜덤

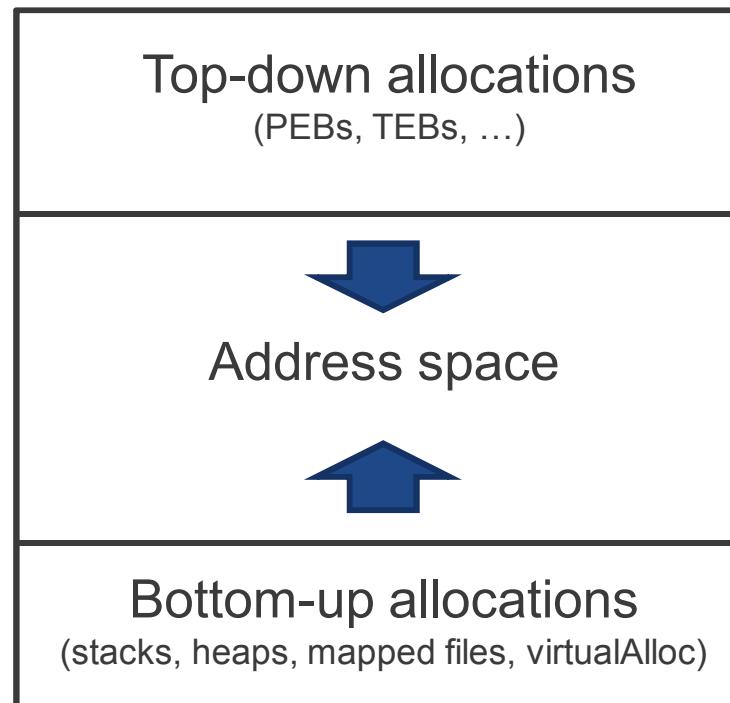
❖ ASLR 우회

- ◆ ASLR 적용되지 않은 모듈 활용
- ◆ Spraying 활용
- ◆ Memory 주소 예측 및 Bruteforcing

❖ Force ASLR

- ◆ Exploit Code가 Non-ASLR Module을 활용한다는 점에서 착안
- ◆ Non-ASLR images를 강제로 Randomized하게 변경
 - ImageBase값을 사용하지 못하는 것 처럼 동작함
 - Bottom-up Randomization

❖ Bottom-up & top-down



❖ DEP?

- ◆ 데이터 실행 방지(Data Execution prevention)
- ◆ stack/stack의 일부분을 non-executable page로 설정하여 stack에서 shellcode가 실행되지 못하게끔 함

❖ DEP 모드

- ◆ Hardware DEP
 - NX bit((No Execute page protection – AMD)
 - XD bit(execute Disable – INTEL)
 - 하지만 지원하지 않는 CPU일 경우 활용할 수 없다.
- ◆ software DEP
 - CPU가 지원하지 못할 경우 Windows DEP는 Software SEP로 동작

❖ DEP 설정 값

◆ OptIn

- 일부 시스템 바이너리와 프로그램에 대해 DEP 적용

◆ OptOut

- DEP가 적용되지 않는 특정 프로그램 목록에 있는 것 외에 모든 프로그램 DEP 적용

◆ AlwaysOn

- DEP제외 목록을 사용할 수 없으며 DEP 시스템 호환성 수정 프로그램이 적용 되지 않는다.

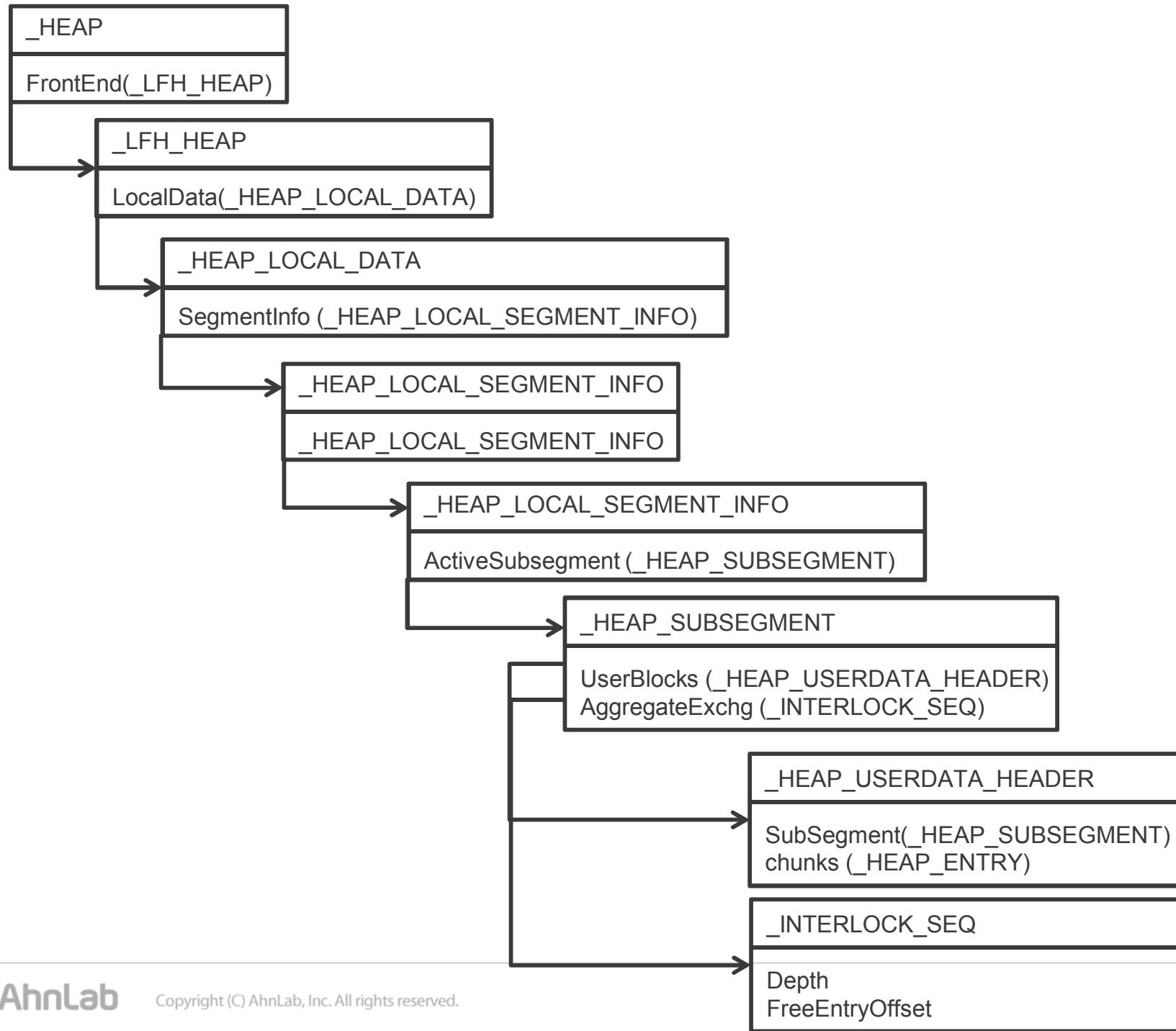
◆ AlwaysOff

- Hardware DEP지원 관계 없이 DEP가 시스템 전체를 보호하지 않음

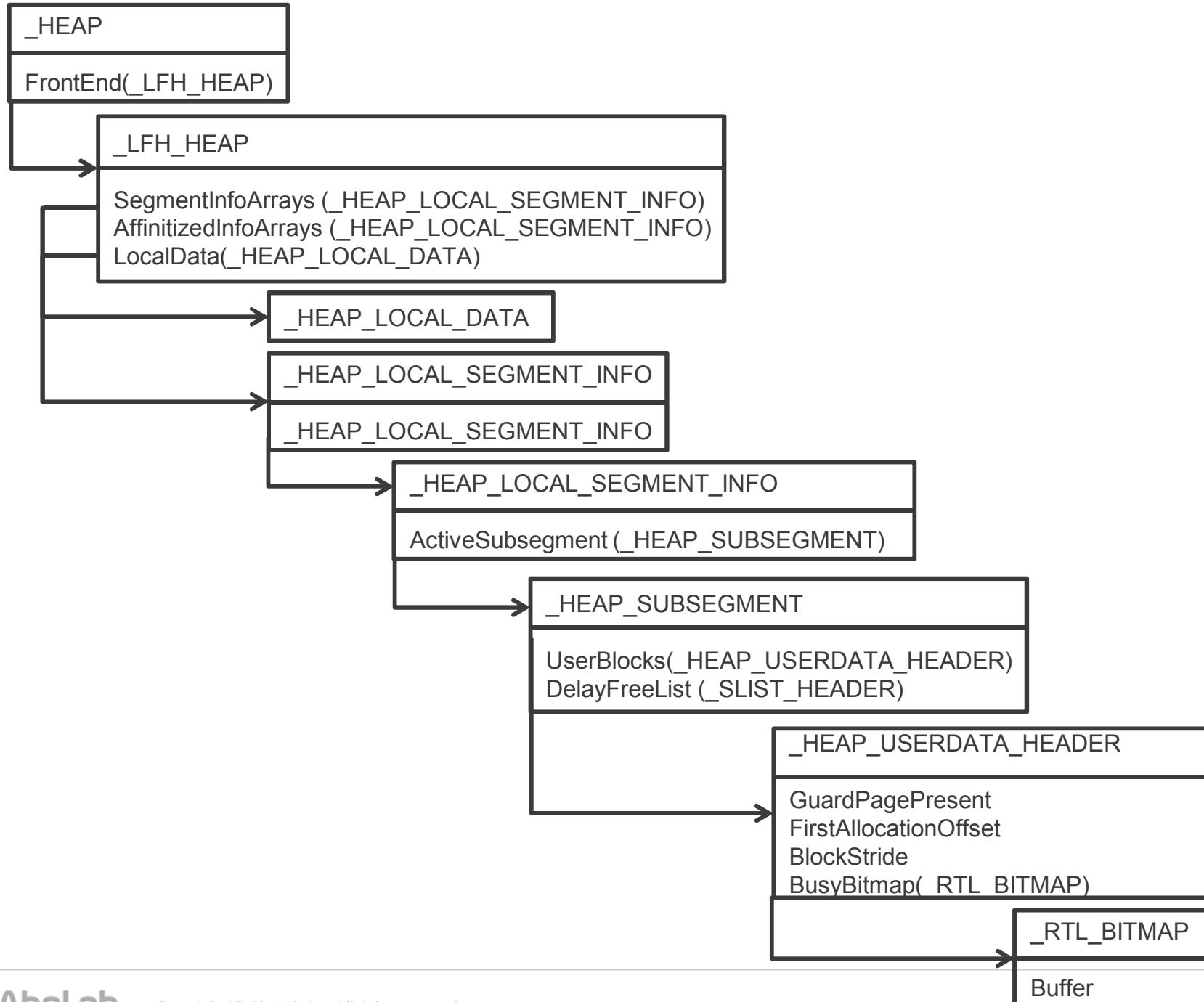
❖ ROP Mitigation

- ◆ Stack Frame의 유효성 여부 확인
 - VirtualProtect(), VirtualAlloc()등 메모리 관련 함수 포함
- ◆ ESP Point 수정을 어렵게 하여 Stack Pivot으로 역할을 하지 못하게 함

Heap Structure – Windows 7



Heap Structure – Windows 8

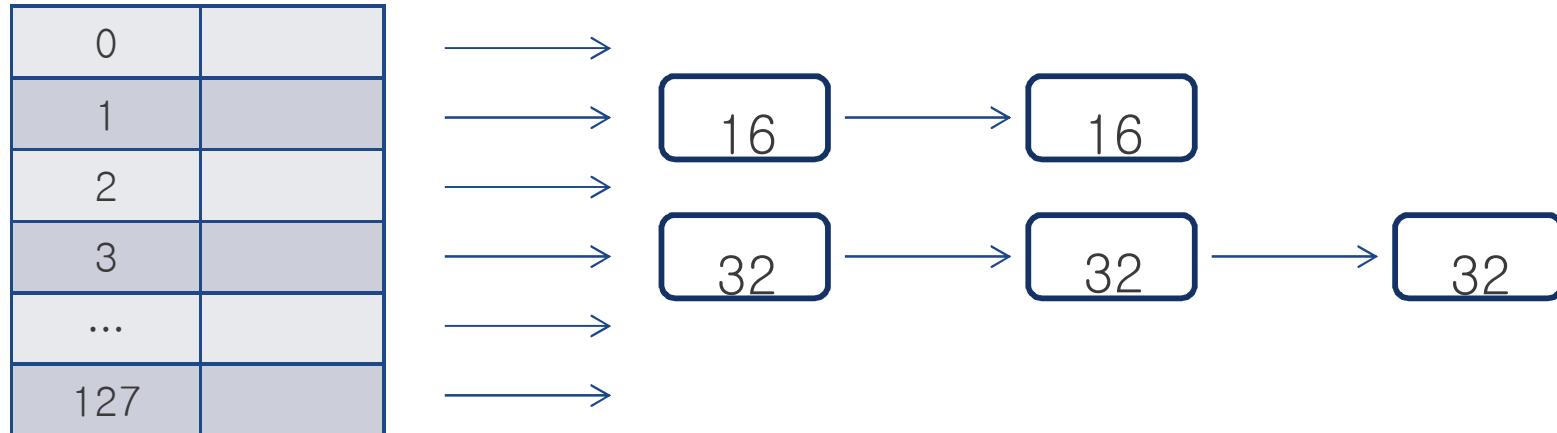


❖ Front End Manager

- ◆ Memory 할당 요청을 효율적으로 처리하기 위해 활용
- ◆ Free Heap 블록을 사이즈 별로 관리
- ◆ Look-aside Lists(활용되지 않음)와 Low-Fragmentation Heap으로 나눔
- ◆ 사용자의 요청을 처리하지 못할 경우 Back End에서 관리함

❖ Front End Manager(cont)

◆ Look-Aside List



◆ Low Fragmentation Heap

Buckets	Granularity	Range
1-32	8	1-256
33-48	16	257-512
49-64	32	513-1024
65-80	64	1025-2048
81-96	128	2049-4096
97-112	256	4097-8192
113-128	512	8193-16384

❖ Front End Manager

_HEAP_ENTRY	<User Data>		_HEAP_ENTRY	<User Data>	
_HEAP_ENTRY	FreeEntryOffset	<User Data>	_HEAP_ENTRY	FreeEntryOffset	<User Data>
_HEAP_ENTRY	FreeEntryOffset	<User Data>	_HEAP_ENTRY	FreeEntryOffset	<User Data>
_HEAP_ENTRY	FreeEntryOffset	<User Data>	_HEAP_ENTRY	FreeEntryOffset	<User Data>

❖ Front End Manager

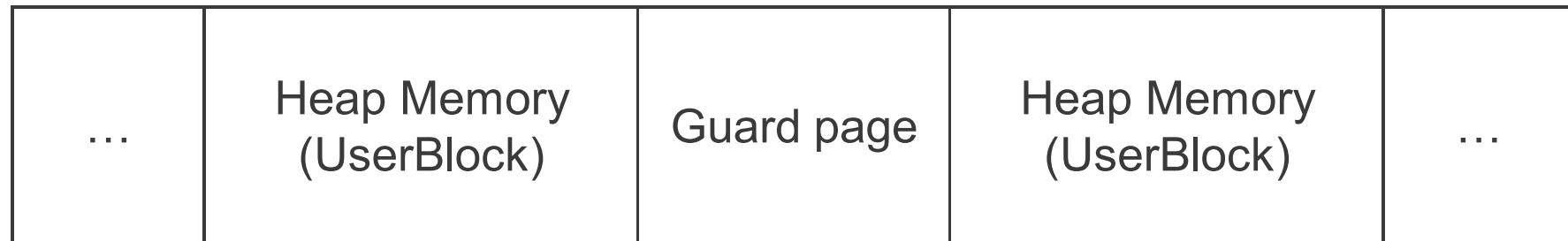
- ◆ Windows 8 Randomization

_HEAP_ENTRY	<User Data>	_HEAP_ENTRY	<User Data>
_HEAP_ENTRY	<User Data>	_HEAP_ENTRY	<User Data>
_HEAP_ENTRY	<User Data>	_HEAP_ENTRY	<User Data>
_HEAP_ENTRY	<User Data>	_HEAP_ENTRY	<User Data>

❖ Front End Manager

◆ Windows 8 Guard page

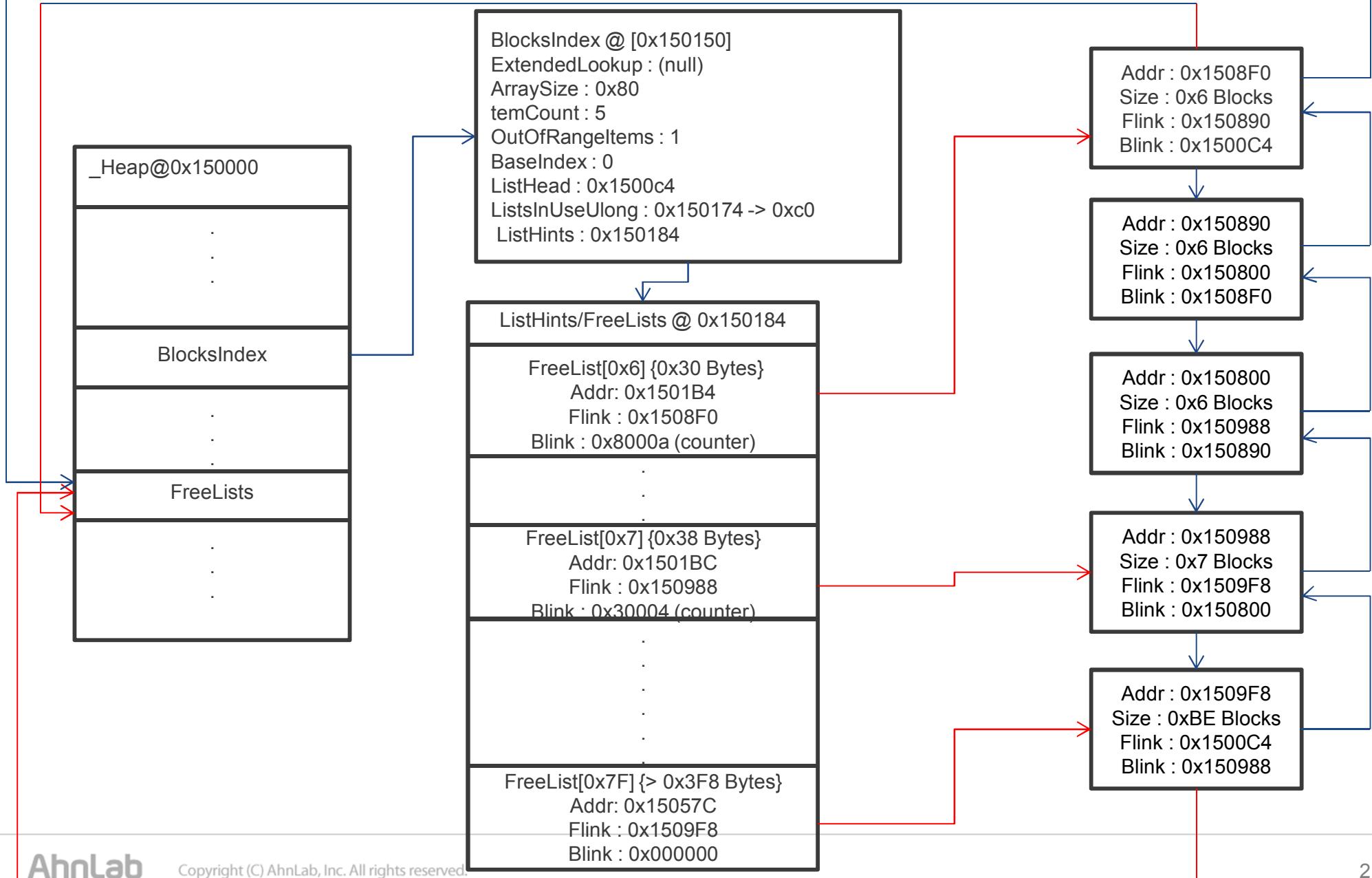
- UserBlock Overflow를 통한 공격을 방어하기 위해서 활용
- Heap Memory의 Partition 역할을 함



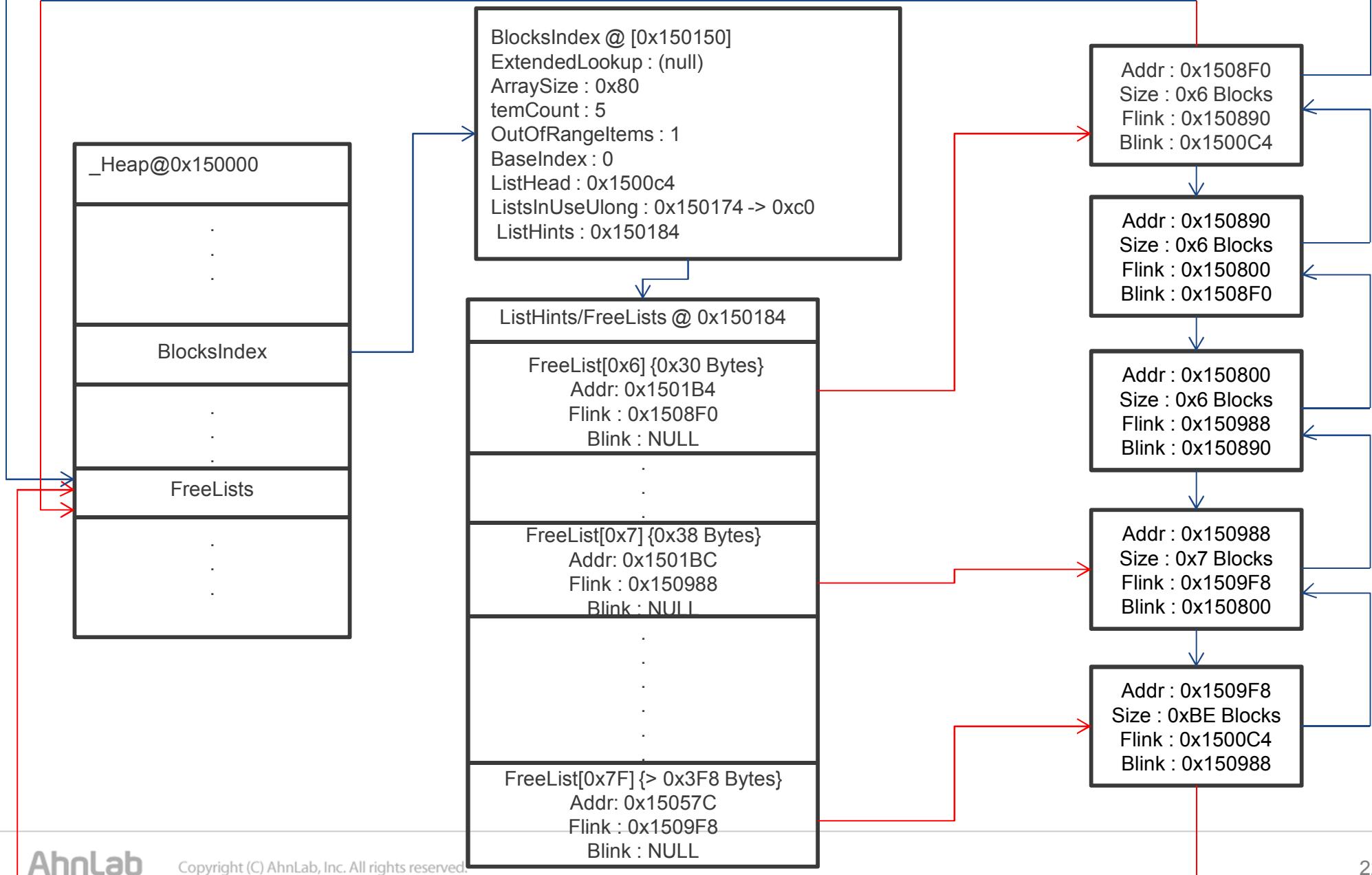
❖ Back End Manager

- ◆ Front-End와 비슷하게 Free List라는 테이블을 통해서 Free Heap 관리
- ◆ double linked list로 Free Heap Block을 관리
- ◆ 특정 크기의 Free Heap Block을 포함하는 128개의 배열로 구성
- ◆ 알맞은 크기의 Heap이 없을 경우 Block Splitting 활용

❖ Free List Architecture



❖ Free List Architecture



Memory Allocation & Free Demo

AhnLab

Copyright (C) AhnLab, Inc. All rights reserved.

03 Exploit Writing Technique

AhnLab

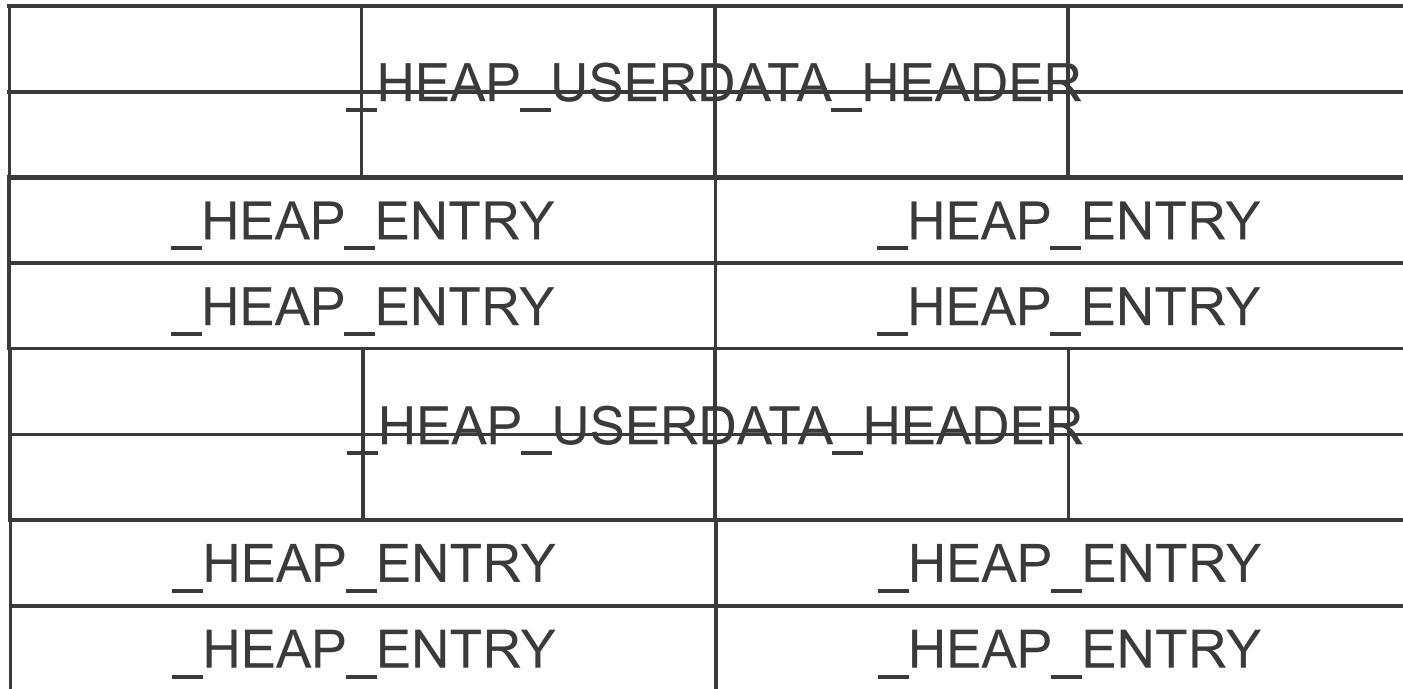
Copyright (C) AhnLab, Inc. All rights reserved.

❖ UserBlocks Overwriting

- ◆ FreeEntryOffset Attack을 더 이상 활용할 수 없음
- ◆ _HEAP_USERDATA_HEADER Structure 공격을 목적으로 함
- ◆ BlockStride는 각각의 Chunk 사이의 간격을 의미 함
- ◆ FreeEntryOffset overwrite와 유사함

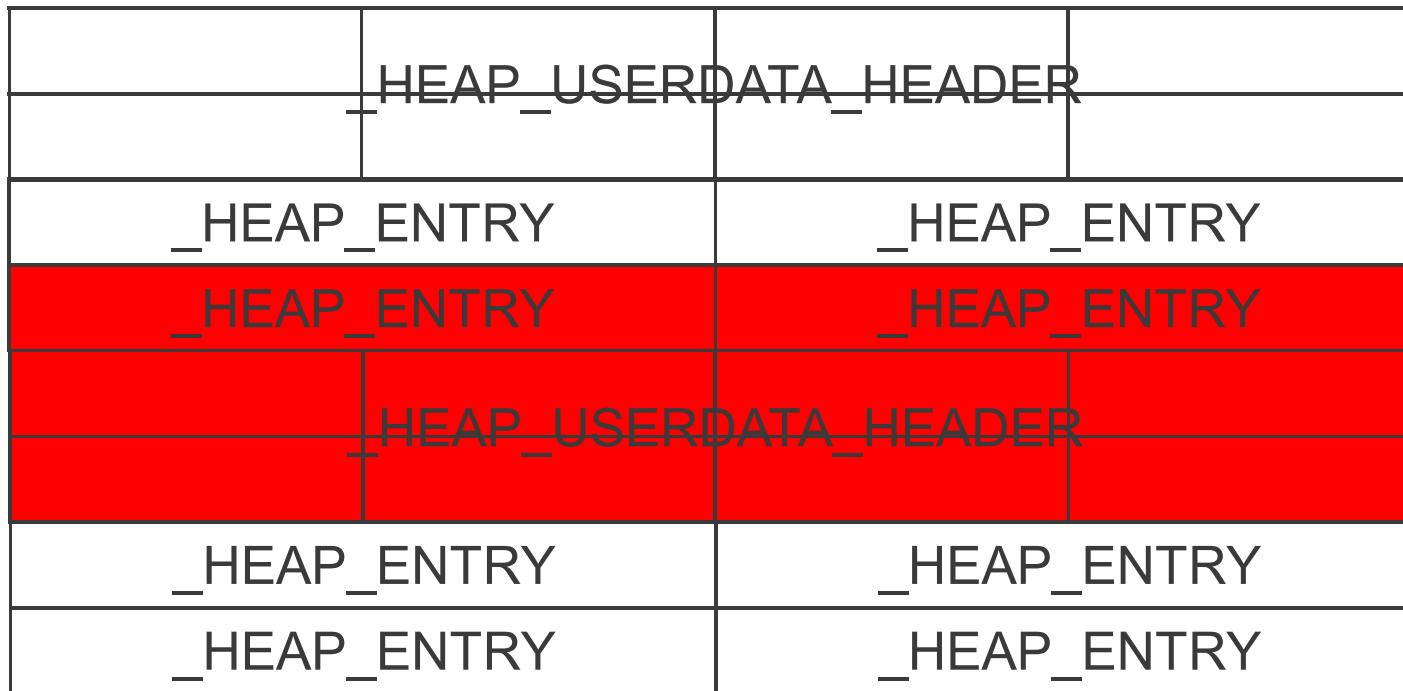
_HEAP_USERDATA_HEADER Attack

❖ UserBlocks Overwriting



_HEAP_USERDATA_HEADER Attack

- ## ❖ UserBlocks Overwriting



❖ Bitmap Flipping 2.0

- ◆ _HEAP_ENTRY.PreviousSize member를 통하여 Bitmap값을 업데이트 함
- ◆ UserBlocks->BusyBitmap->Buffer, Header->PreviousSize
- ◆ PreviousSize값을 변조하게 될 경우 영향을 받을 수 있음

_HEAP_USERDATA_HEADER Attack

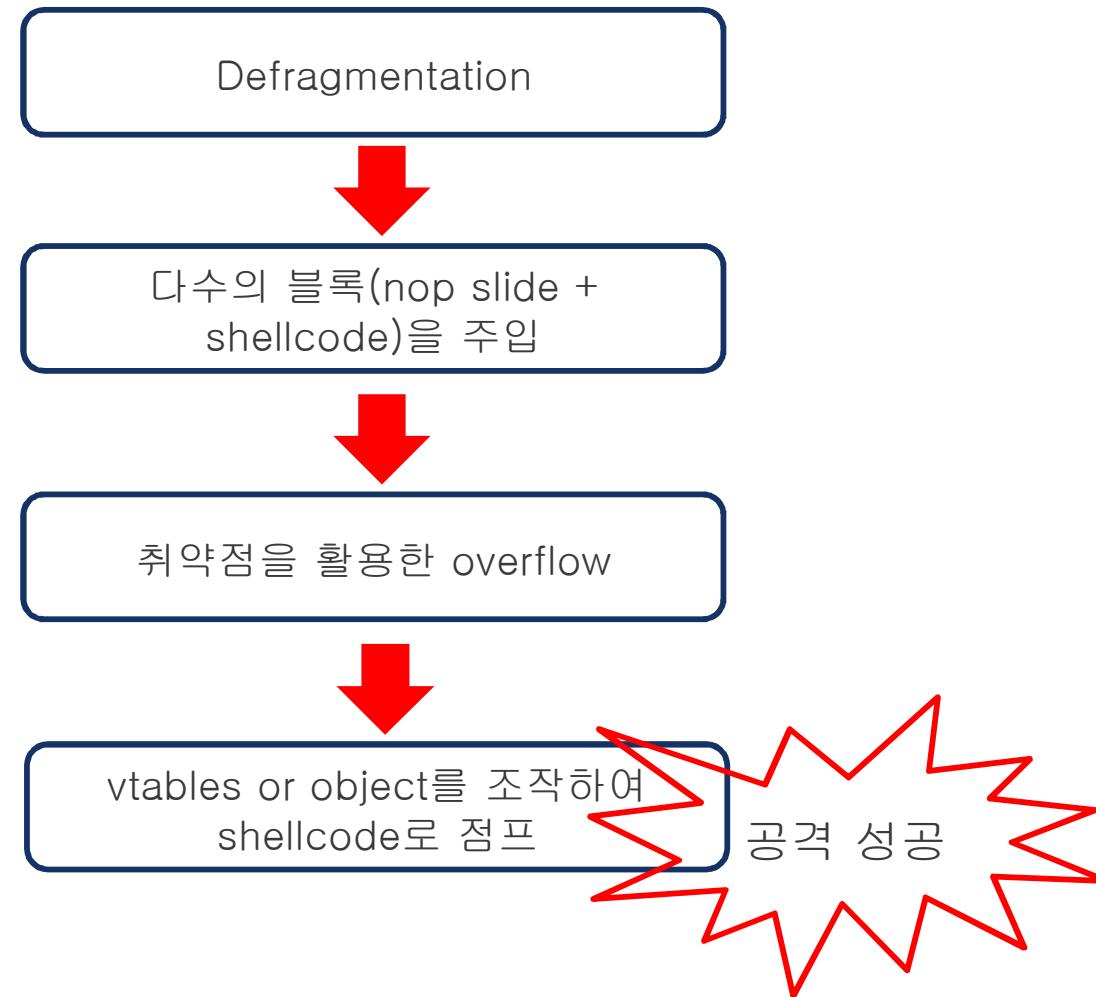
❖ Bitmap Flipping 2.0

_HEAP_ENTRY		_HEAP_ENTRY	_HEAP_ENTRY	_HEAP_ENTRY
_HEAP_ENTRY		_HEAP_ENTRY	_HEAP_ENTRY	_HEAP_ENTRY
_HEAP_ENTRY		_HEAP_ENTRY	_HEAP_ENTRY	_HEAP_ENTRY

❖ Heap Spray

- ◆ 연속된 Heap 영역에 exploit code를 뿌리는 방법
- ◆ 지속적으로 Heap을 할당 받아, 특정 메모리 주소 영역까지 데이터를 덮음
- ◆ 대량의 nop slide와 shellcode로 제작된 block 생성하여 heap에 저장

❖ Heap Spray



❖ Heap Spray

◆ Defragment Heap

- Heap은 alloc/free를 반복하면서 단편화가 생기고, 이 때 다음 할당되는 Heap의 위치를 예상하는 것이 어려움
- 연속적인 공간이 아닐 경우 EIP조작을 통해 점프 하더라도 공격 성공율이 낮음
- NOP slide의 역할을 할 수 있으면서 주소값으로 사용되었을 경우 조작된 Heap블록을 가리킬 수 있어야 하므로 다수의 Heap 블록 할당

```
var dummy = new Array(1000);

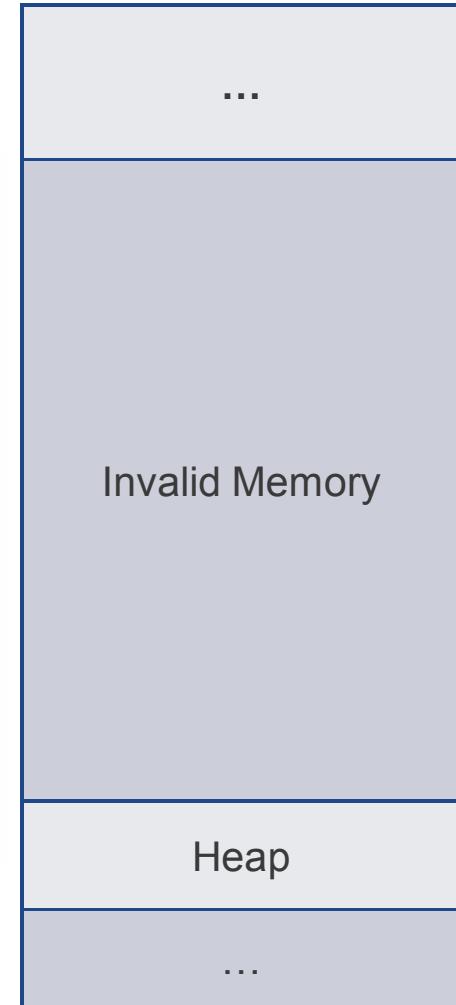
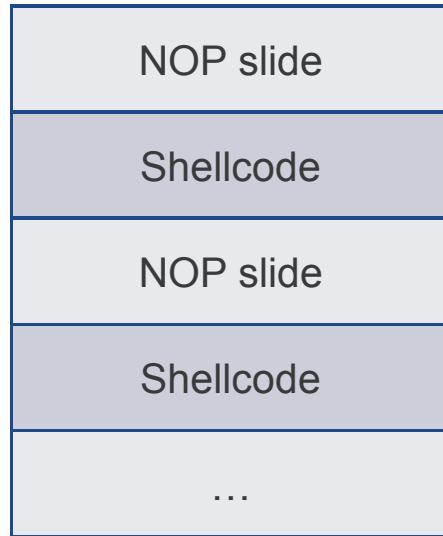
for(i=0; i<1000; i++){

    dummy[i]= new Array(size);

}
```

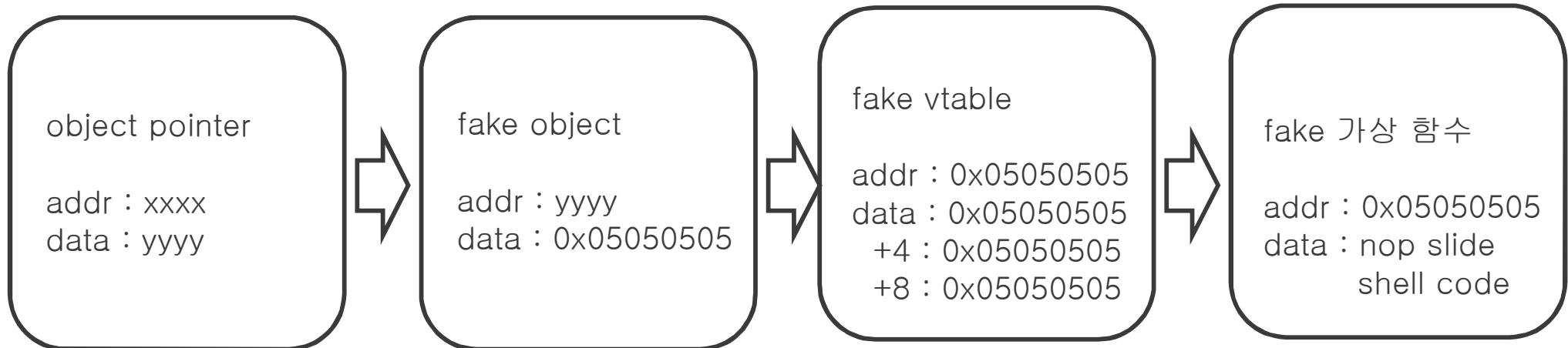
❖ Heap Spray

- ◆ NOP slide + shellcode 주입



❖ Heap Spray

- ◆ vtables or object 조작



❖ Heap Spray

◆ Heap Spray 공격 패턴

```
var shellcode = unescape("shellcode");
var block = unescape("%u0505%0505");

while (nop.length <= 0x100000/2) nop += nop;

var x = new Array();

for (var i = 0; i < 200; i++) {
    x[i] = block + shellcode;
}
```

NOP slide
+
shellcode

특정 주소
지점까지
힘을 할당

Heap Spray In Windows 8 Demo

AhnLab

Copyright (C) AhnLab, Inc. All rights reserved.

?

AhnLab

Copyright (C) AhnLab, Inc. All rights reserved.

thank you.

CodeEngn

www.CodeEngn.com

2013 CodeEngn Conference 08

AhnLab

Copyright (C) AhnLab, Inc. All rights reserved.

www.CodeEngn.com | www.RCEHub.com



2013 CodeEngn Conference 08

2013.07.13 SAT 13:00

www.CodeEngn.com | www.RCEHub.com