침해사고 대응 알림 서비스
(Infringement Incident Response Notification Service)

목표ㄱㅋKG

팀원 소개

은정욱

팀장

총괄
윈도우 서비스 개발

박형준

팀원

윈도우 서버 구축
윈도우 서비스 개발

김두형

팀원

리눅스 서버 구축
리눅스 서비스 개발

프로젝트 배경

< 한국인터넷진흥원(KISA) 침해사고 신고 건수 통계(건) >

구분 \ 연도	2021년		2022년		2023년
	상반기	하반기	상반기	하반기	상반기
건수	298	342	473	669	664
합계	640		1,142		664

(단위:건)

구분	2019년	2020년	2021년	2022년	2023년 8월
DDoS 공격	155	213	123	122	159
악성코드 감염·유포	59	140	234	347	218
시스템 해킹	204	250	283	673	513
합계	**418**	**603**	**640**	**1,142**	**890**

(자료: 한국인터넷진흥원, 2023.8월말 기준)

침해사고 신고 건수는 매년 증가

보안을 위협하는 공격은 계속해서 진화하고 발전함

보안 감시 및 위협 대응 능력 향상

소규모, 저자본 보안성 강화

J

프로젝트 개요

slack

+

GRR
RAPID RESPONSE

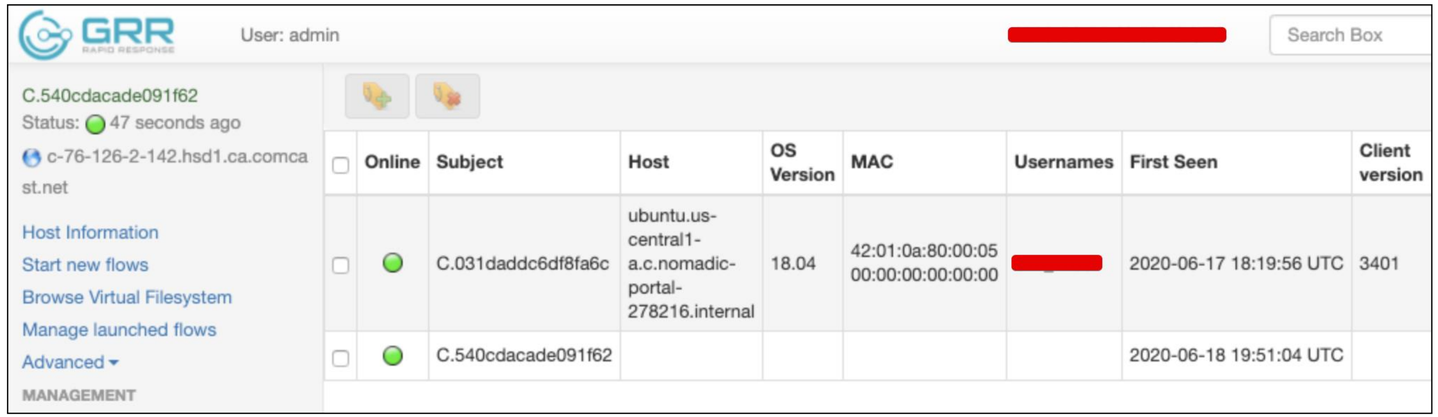
+

VirusTotal

프로젝트 개요 – GRR

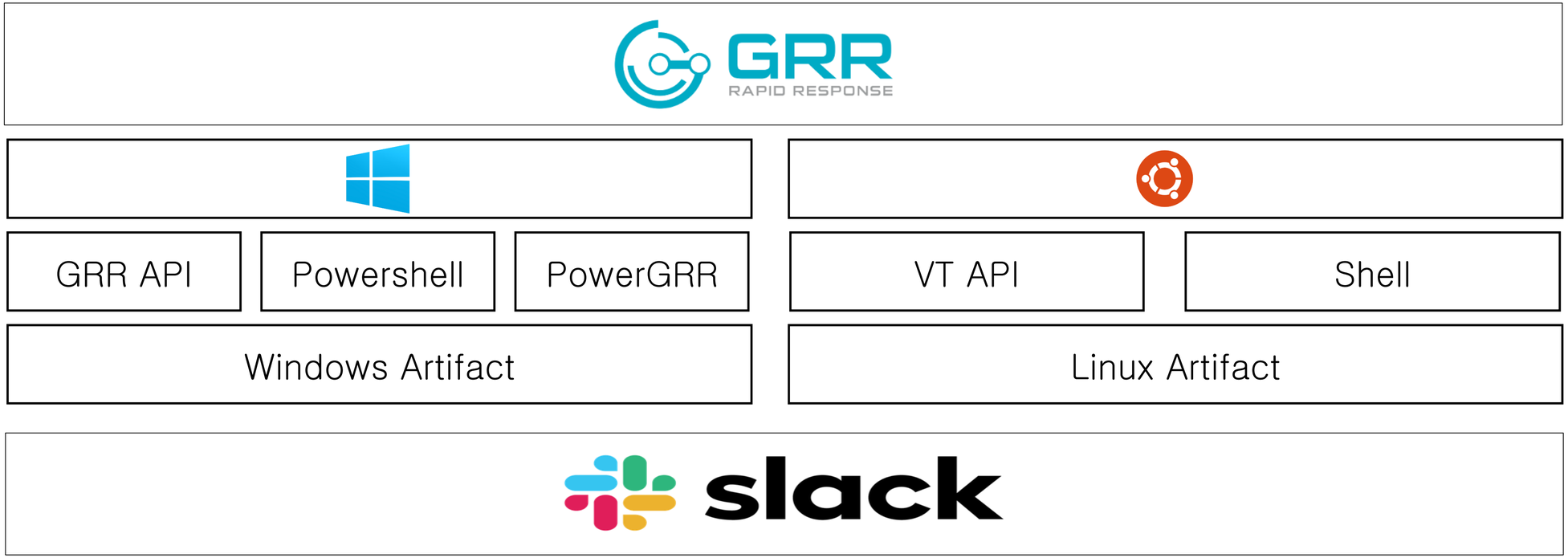
GRR(Google Rapid Response)

- 원격 라이브 포렌식에 중점을 둔 침해사고 대응 프레임워크이다.
- Linux, Mac OS 및 클라이언트에 대한 크로스 플랫폼 지원한다.



서비스 요약

침해사고 대응 알림 서비스 서비스 요약



서비스 요약

Windows

Attacker Use Path, File Path, Network Info

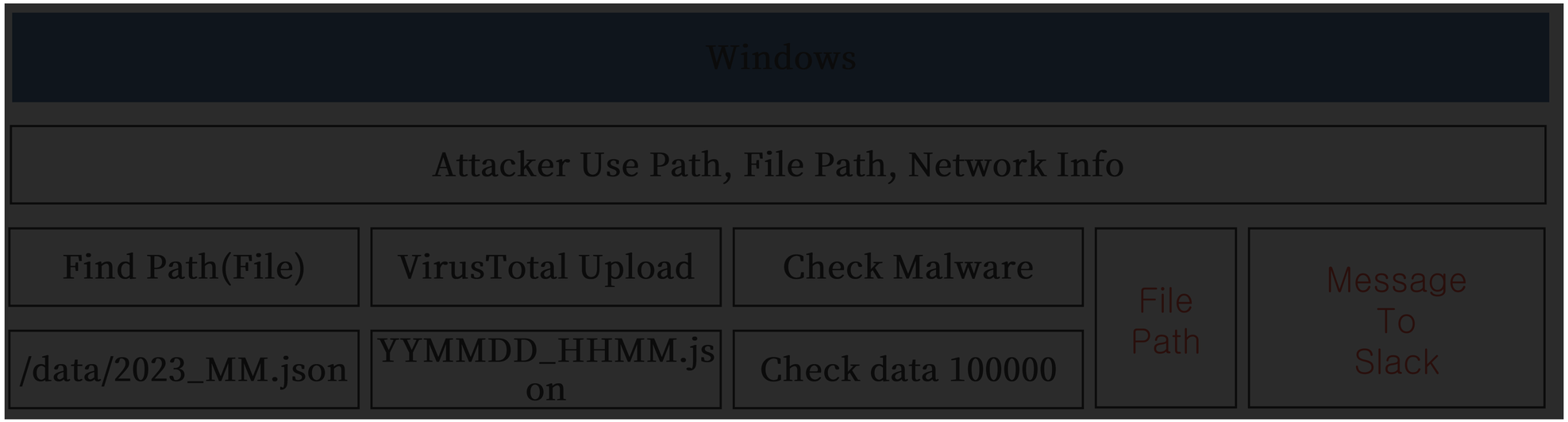
Find Path(File)	VirusTotal Upload	Check Malware	File Path	Message To Slack
/data/2023_MM.json	YYMMDD_HHMM.json	Check data 100000		

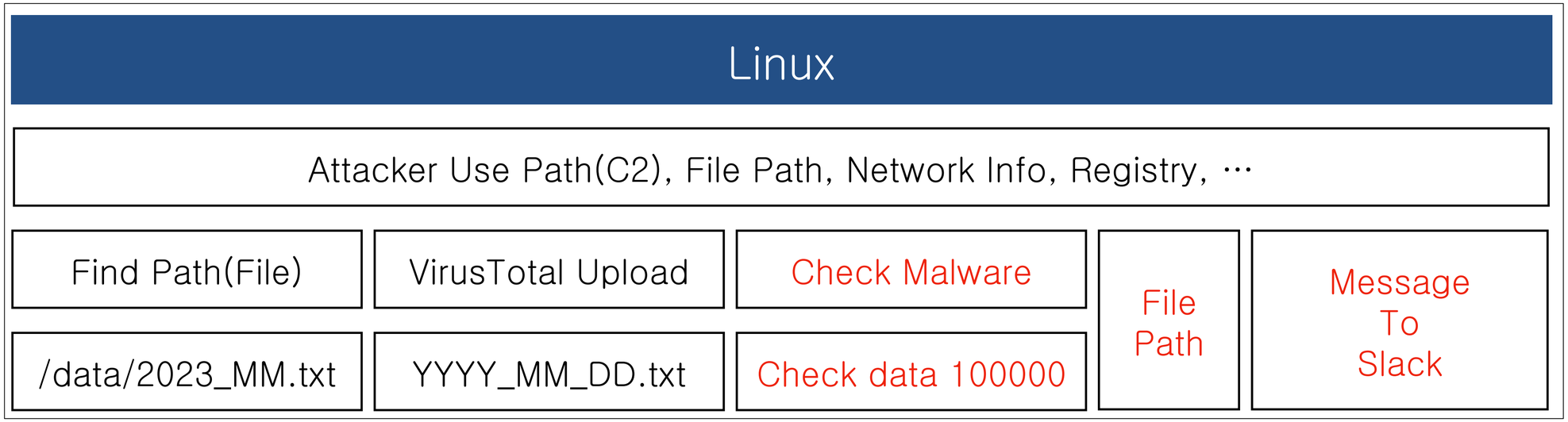
Linux

Attacker Use Path(C2), File Path, Network Info, Registry, …

Find Path(File)	VirusTotal Upload	Check Malware	File Path	Message To Slack
/data/2023_MM.txt	YYYY_MM_DD.txt	Check data 100000		

서비스 요약


Windows
Attacker Use Path, File Path, Network Info
Find Path(File)
VirusTotal Upload
Check Malware
File Path
Message To Slack
/data/2023_MM.json
YYMMDD_HHMM.json
Check data 100000


Linux
Attacker Use Path(C2), File Path, Network Info, Registry, …
Find Path(File)
VirusTotal Upload
Check Malware
File Path
Message To Slack
/data/2023_MM.txt
YYYY_MM_DD.txt
Check data 100000

취업 담지 시나리오 - 리눅스

ɔ.

- 매월 ɔ일 정보들을 수집하여 YYYYMM_main.txt로 저장
- MIRTE ATT&CK와 보안 회사 리포트를 기반으로 리눅스 정보 수집
- Cʃ 서버로 사용될 경우, 기본 경로, 네트워크 정보, 파일 시스템 등
- YYYYMM_main과 YYYYMMDD_HHMM 파일을 비교하여 /diff_date/YYYYMM_HHMM 저장

```
srw------- 1 plitoo plitoo 0 Jun 11 05:11 agent.2397
total 0
srwxrwxrwx 1 plitoo plitoo 0 Jun 11 05:11 X0
srwxrwxr-x 1 gdm    gdm    0 Jun 11 05:10 X1024
total 0
total 0
total 0
srwxrwxrwx 1 gdm    gdm    0 Jun 11 05:10 1214
srwxrwxrwx 1 plitoo plitoo 0 Jun 11 05:11 2397
total 32
-rw------- 1 logstash logstash 32768 Jun 19 07:52 7711
total 0
total 0
total 20
-rw-r--r-- 1 root root 10918 Jun 18 10:56 index.html
-rw-r--r-- 1 root root   612 Apr  6 23:20 index.nginx-debian.html
total 0
*Normally Path info*
```

date/ɟoɟpMM_main.txt

```
+total 0
+srwxrwxrwx 1 gdm    gdm    0 Jun 11 05:10 1214
+srwxrwxrwx 1 plitoo plitoo 0 Jun 11 05:11 2397
 total 32
--rw------- 1 logstash logstash 32768 Aug 31 06:51 91464
+-rw------- 1 logstash logstash 32768 Jun 19 07:52 7711
 total 0
 total 0
-total 5360
--rwxr-xr-x 1 root root 5455872 Jun 19 09:53 Exaramel-Linux
--rw-r--r-- 1 root root   10918 Jun 18 10:56 index.html
--rw-r--r-- 1 root root     612 Apr  6 23:20 index.nginx-debian.html
--rw-r--r-- 1 root root     294 Jun 19 09:53 obfuscated_webShell.php
--rw-r--r-- 1 root root       5 Jun 18 10:57 ple.exe
--rwxr-xr-x 1 root root    2760 Jun 19 09:53 test_webshell.py
--rw-r--r-- 1 root root     844 Jun 19 09:53 webShell.php
+total 20
+-rw-r--r-- 1 root root 10918 Jun 18 10:56 index.html
+-rw-r--r-- 1 root root   612 Apr  6 23:20 index.nginx-debian.html
 total 0
 *Normally Path info*
```

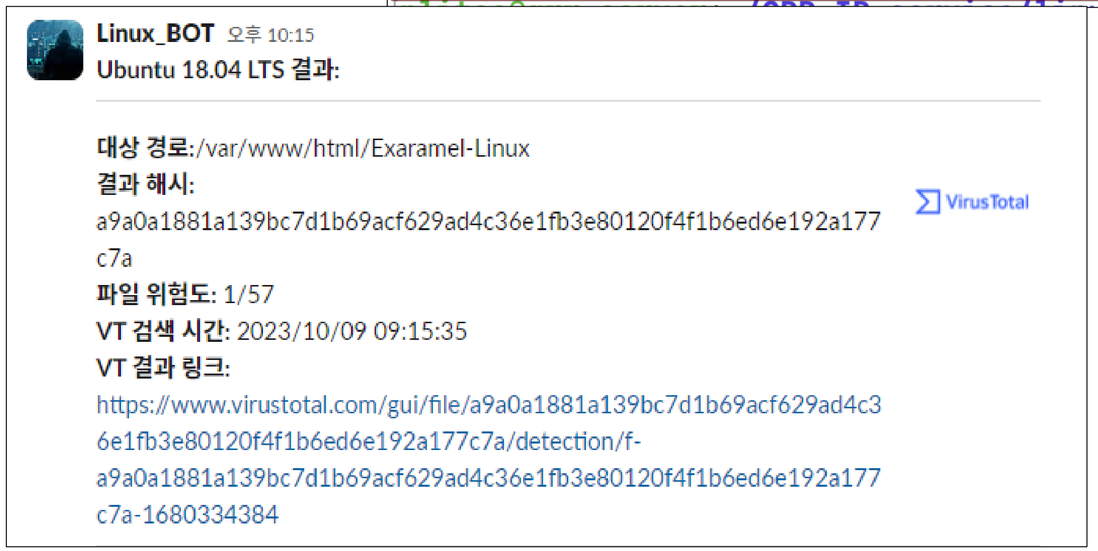
data/ɟoɟpMMDD_HHMM.txt

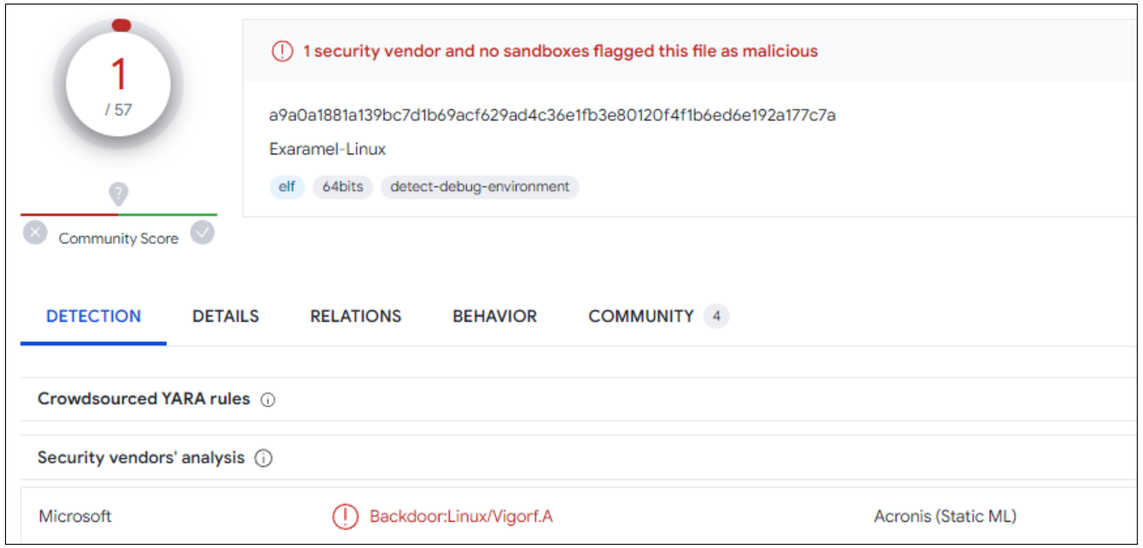
위협 탐지 시나리오 - 리눅스

J.

- /diff_date/YYYYMM_HHMM 에서 ɔooooooo이상의 파일이 있다면 Hash로 변환한다.
- 변환한 정보는 /diff_hash/filename.hash 로 저장
- 해당 해시를 VirusTotal API를 이용하여 업로드 후 검사

```
plitoo@grr_server:~/GRR_IR_service/linux/diff_hash$ ls -a
.  ..  Exaramel-Linux.hash  hermez.hash
plitoo@grr_server:~/GRR_IR_service/linux/diff_hash$ cat Exaramel-Linux.hash hermez.hash
a9a0a1881a139bc7d1b69acf629ad4c36e1fb3e80120f4f1b6ed6e192a177c7a
8d3f68b16f0710f858d8c1d2c699260e6f43161a5510abb0e7ba567bd72c965b
```





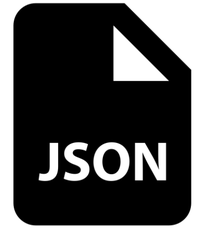
위협 탐지 시나리오 - 윈도우

윈도우 환경

GRR Client 배포

Slack 채널 연동

GRR 진단 원격 실행

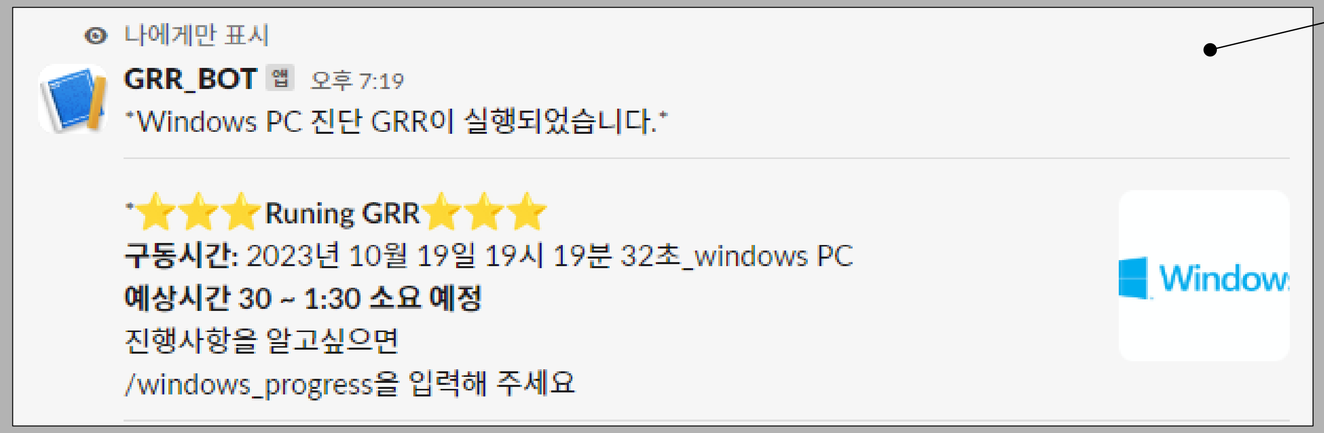


진단 내역 JSON 파일로 정재 및 저장

바이러스 토탈 연동 및 스캔

진단 결과 Slack 전송

위협 탐지 시나리오 - 윈도우



Slack Start 메시지

GRR 진단 powershell 스크립트
진단 시작 후 중간 내용 process.txt에 저장

```
Import-Module C:\PowerGRR-0.12.0\PowerGRR-0.12.0\PowerGRR.psd1 -force
$GRRCredential = Microsoft.PowerShell.Security\get-credential -UserName admin -Message "GRR 인증 password 입력중 기다리세요"
    #컴퓨터이름 출력
$Client_ComputerName = hostname
    #연동된 client를 모두 보는법(clientid만 출력하게 만듬.)
$ALL_Clientid = (Get-GRRClientIdFromComputerName $Client_ComputerName).Clientid
    #클라이언트ID에 라벨 붙이는법(client host name별)
Set-GRRLabel -ComputerName $Client_ComputerName -Label Park
    #클라이언트 ID를 라벨로 찾는법
$clients = Find-GRRClientByLabel -SearchString Park
    #프로세스 헌팅 준비 + huntid만 출력 변수에 저장
$Huntid_system1 = New-GRRHunt -HuntDescription "file system path hash save1" -Flow FileFinder -RuleType Label -Label Park -ActionType Hash -Mode ALL_HITS -Path 'C:\
$Huntid_system1.hunt_id > huntid.txt
    #GRR헌트 시작
Start-GRRHunt -HuntId $Huntid_system1.hunt_id
```

GRR RAPID RESPONSE

GRR Client 배포

위협 탐지 시나리오 - 윈도우

```powershell
# 작업 액션을 생성합니다. 실행할 Python 스크립트 파일을 지정합니다.
$action = New-ScheduledTaskAction -Execute 'python.exe' -Argument 'C:\PowerGRR-0.12.0\PowerGRR-0.12.0\py_slackbot_test\ret_grr.py'

# 작업 트리거를 생성합니다. 현재 실행 시간에서 30분 후에 시작하도록 설정합니다.
$trigger = New-ScheduledTaskTrigger -Once -At ($currentTime.AddMinutes(30))

# 작업을 등록합니다. 작업 이름과 트리거를 지정합니다.
Register-ScheduledTask -TaskName 'grr_ret' -Trigger $trigger -Action $action

# 작업에 설명을 추가합니다.
$task = Get-ScheduledTask -TaskName 'grr_ret'
$task.Description = "grr 결과값 받아오는 작업 실행중"

# 작업 트리거를 수정하여 30분 간격으로 반복하도록 설정합니다.
$trigger.RepetitionInterval = "PT30M"  # 30분 간격
```

GRR 진단 결과 자동업로드를 위해 스케줄러에 등록

```python
# PowerShell 스크립트를 생성하고 파일로 저장
ps_script = f"""
Import-Module C:\\PowerGRR-0.12.0\\PowerGRR-0.12.0\\PowerGRR.psd1 -force;
$GRRCredential = Microsoft.PowerShell.Security\\get-credential -UserName admin -Message 'GRR 인증 password 입력중 기다리세
Get-GRRHuntResult -HuntId {huntid}_system1 -ShowJSON > C:\PowerGRR-0.12.0\PowerGRR-0.12.0\py_slackbot_test\json_ret\\tes
"""

# 스크립트를 파일로 저장
with open("C:\\PowerGRR-0.12.0\\PowerGRR-0.12.0\\ret_grr.ps1", "w") as script_file:
    script_file.write(ps_script)

# PowerShell 스크립트 실행
ps_script_path = "C:\\PowerGRR-0.12.0\\PowerGRR-0.12.0\\ret_grr.ps1"
username = "admin"
password = "park1004"

process = subprocess.Popen(
    ['powershell', '-ExecutionPolicy', 'Bypass', '-File', ps_script_path],
    stdin=subprocess.PIPE,
    stdout=subprocess.PIPE,
    stderr=subprocess.PIPE,
    text=True
)
```

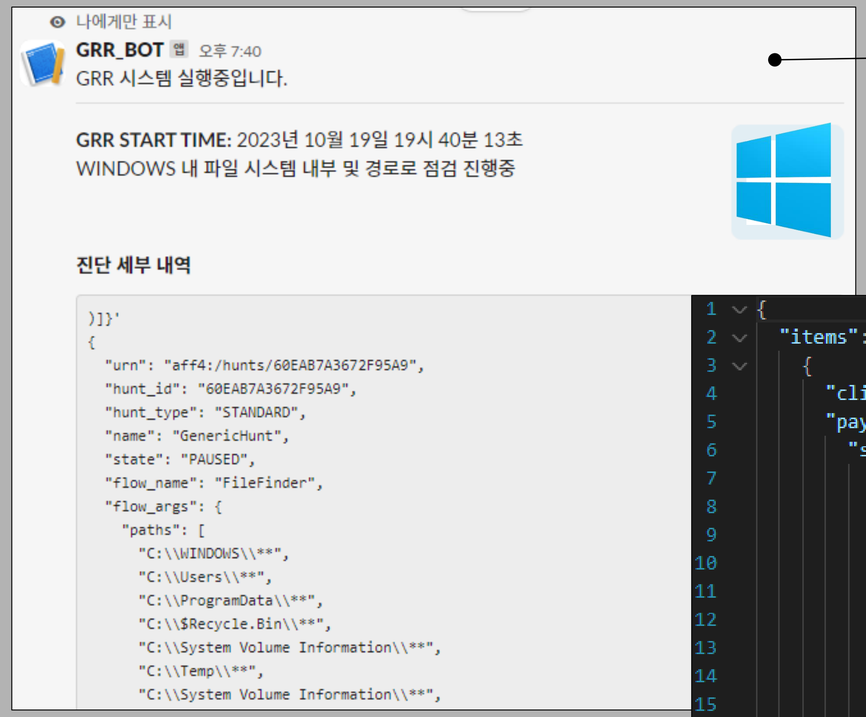
스케줄러 등록된 실행 .py 파일
결과 값을 json형태로 가져온다.

```python
# 각 파일에서 hash 값을 추출합니다.
hash_set1 = set(entry["payload"]["hash_entry"]["sha256"] for entry in data1["items"])
hash_set2 = set(entry["payload"]["hash_entry"]["sha256"] for entry in data2["items"])

# 두 세트를 비교하여 중복되지 않는 항목을 찾습니다.
unique_hashes = hash_set1.symmetric_difference(hash_set2)

# 중복되지 않는 hash 값을 가진 항목을 리스트로 저장합니다.
unique_entries = []

# index를 1부터 시작하도록 초기화합니다.
index = 1

for entry in data1["items"] + data2["items"]:
    if entry["payload"]["hash_entry"]["sha256"] in unique_hashes:
        sha256 = entry["payload"]["hash_entry"]["sha256"]
        sha1 = entry["payload"]["hash_entry"]["sha1"]
        md5 = entry["payload"]["hash_entry"]["md5"]
        path = entry["payload"]["stat_entry"]["pathspec"]["path"]
        timestamp = entry["timestamp"]
        unique_entry = {
            "index": index,
            "sha256": sha256,
            "sha1": sha1,
            "md5": md5,
            "path": path,
            "timestamp": timestamp
        }
        unique_entries.append(unique_entry)
        # 다음 항목을 위해 index를 증가시킵니다.
        index += 1
```

결과값 json 파일
초기 원본파일과 대조 추가된 값 추출
해쉬값 추출 및 정재

GRR 진단 원격 실행

위협 탐지 시나리오 - 윈도우



Slack상 진행사항 과정


```
{
  "items": [
    {
      "client_id": "C.2ff1ac46dbeb95ba",
      "payload": {
        "stat_entry": {
          "st_mode": 33206,
          "st_ino": 281474976774258,
          "st_dev": 2319326956
          "st_nlink": 2,
          "st_uid": 0,
          "st_gid": 0,
          "st_size": 802,
          "st_atime": 1575730704,
          "st_mtime": 1555613340,
          "st_ctime": 1575730704,
          "st_flags_osx": 0,
          "st_flags_linux": 0,
          "pathspec": {
            "pathtype": "OS",
            "path": "/C:/Windows/addins/FXSEXT.ecf",
            "path_options": "CASE_LITERAL"
          }
        },
        "hash_entry": {
          "sha256": "63R9h8c5vigYloSoSvz2Am+D4QxXJJIJOx4u/Fc1
          "sha1": "b1l8hegHyI1Lv13ueZb/sd+Ggls=",
          "md5": "GFFfjdruJ1DIHXaPLA5xFw==",
          "num_bytes": 802
        }
      },
      "payload_type": "FileFinderResult",
      "timestamp": 1696711791093263
```


결과 json파일 필요값만 추출


```
[
    {
        "index": 1,
        "sha1": "gk5sg8B5BzE5IRzI531As3IVbSc=",
        "sha256": "V6AnnjtrFYQEz79QcOM7QZY+KIp7sjA4Kx0/YRYpnHU=",
        "md5": "gguXQp5BU6dDcIs3aAfuaQ==",
        "path": "/C:/Windows/bfsvc.exe",
        "timestamp": 1696732898053408
    },
    {
        "index": 2,
        "sha1": "A/eB5V2PTW3YirIIhR8OKcziQdk=",
        "sha256": "7iVE42tmq0ZaQxFthd65eX2tduFVOtN9dBGT6yQL6x8=",
        "md5": "4zp5VsdvaXqkYZid3FwQHg==",
        "path": "/C:/Windows/bootstat.dat",
        "timestamp": 1696732898053485
    },
```


JSON

진단 내역 JSON 파일로 정제 후 저장

위협 탐지 시나리오 - 윈도우

```
entry = unique_entries[0]
sha256_hash = entry["sha256"]
print(sha256_hash)
file_path = entry["path"]

scan_url = f"https://www.virustotal.com/api/v3/files/{sha256_hash}"
headers = {
    "x-apikey": api_key
}

response = requests.get(scan_url, headers=headers)
```

해쉬값을 이용 V.T 스캔

```
if total_engines > 0:
    file_risk = f"파일 위험도: {malicious_engines}/{total_engines}"
else:
    file_risk = "파일 위험도: 알 수 없음"

formatted_date = korea_time.strftime("%Y-%m-d %H:%M:%S")

current_time = datetime.now()
formatted_time = current_time.strftime("%Y-%m-d %H:%M:%S")

file_status = scan_result["data"]["attributes"]["last_analysis_stats"]["harmless"]
if file_status > 0:
    status_message = "스캔 결과: 악성 파일일 수 있습니다. 관리자에게 문의하세요."
else:
    status_message = "스캔 결과: 안전한 파일입니다."

vt_link = f"VirusTotal 파일 정보 링크: https://www.virustotal.com/gui/file/{sha256_ha
```

위험도, 스캔 시간, 자세한 링크로 스캔결과를 가져옴

VirusTotal

바이러스 토탈 연동 및 스캔

위협 탐지 시나리오 - 윈도우

@GRR_BOT /windows_result

GRR_BOT 앱 오후 8:10

Windows PC GRR 결과:

대상 경로: /C:/app_apk/1945-Air-Force-v11.17-mod.apk
결과 해시: 376200CC2375E5BFBD994DA7D2CA8967608AFD0F
파일 위험도: 파일 위험도: 2/75
스캔 결과: 파일은 안전해 보입니다.
VT 서버 시간: 2022-11-d 12:55:17
작동 시간: 2023-10-d 20:10:41

VirusTotal

자세한 내용을 보고싶으면 옆에 링크버튼을 클릭하세요

V.T 링크

Linux_BOT 오후 10:15

Ubuntu 18.04 LTS 결과:

대상 경로:/var/www/html/Exaramel-Linux
결과 해시:
a9a0a1881a139bc7d1b69acf629ad4c36e1fb3e80120f4f1b6ed6e192a177c7a
파일 위험도: 1/57
VT 검색 시간: 2023/10/09 09:15:35
VT 결과 링크:
https://www.virustotal.com/gui/file/a9a0a1881a139bc7d1b69acf629ad4c36e1fb3e80120f4f1b6ed6e192a177c7a/detection/f-a9a0a1881a139bc7d1b69acf629ad4c36e1fb3e80120f4f1b6ed6e192a177c7a-1680334384

VirusTotal
VirusTotal
VirusTotal

b27e46e231b107d12659ad25193d177a952ca9652dc27d96978327e8610395eb

Sign in

2 / 64

2 security vendors and no sandboxes flagged this file as malicious

Reanalyze Similar More

b27e46e231b107d12659ad25193d177a952ca9652dc27d969783...
-1945-Air-Force-v11.17.apk

Size 157.95 MB

Last Analysis Date 11 months ago

APK

android apk contains-elf runtime-modules reflection telephony

Community Score

DETECTION DETAILS RELATIONS BEHAVIOR COMMUNITY

Join the VT Community and enjoy additional community insights and crowdsourced detections, plus an API key to automate checks.

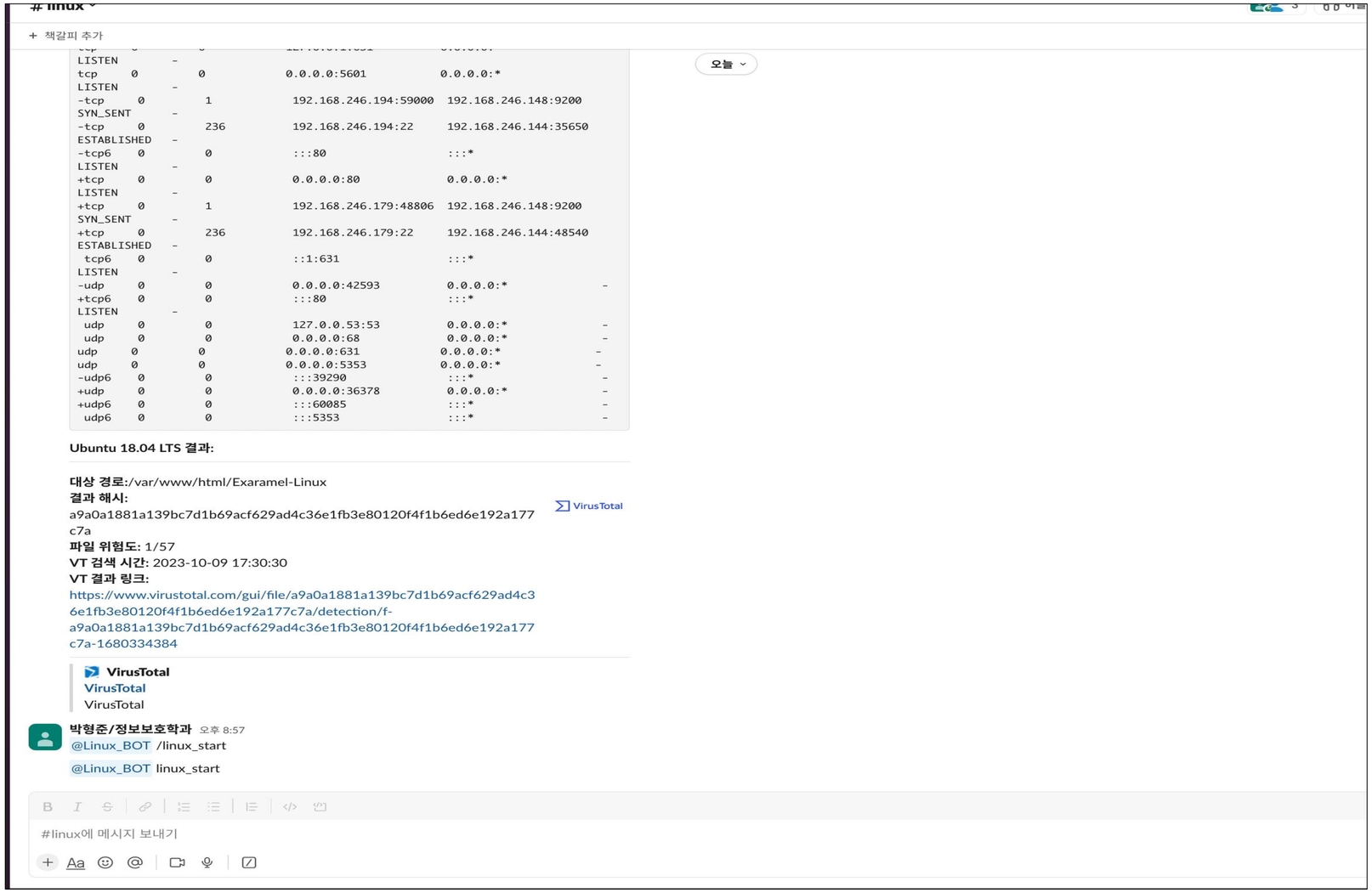
Security vendors' analysis

Do you want to automate checks?

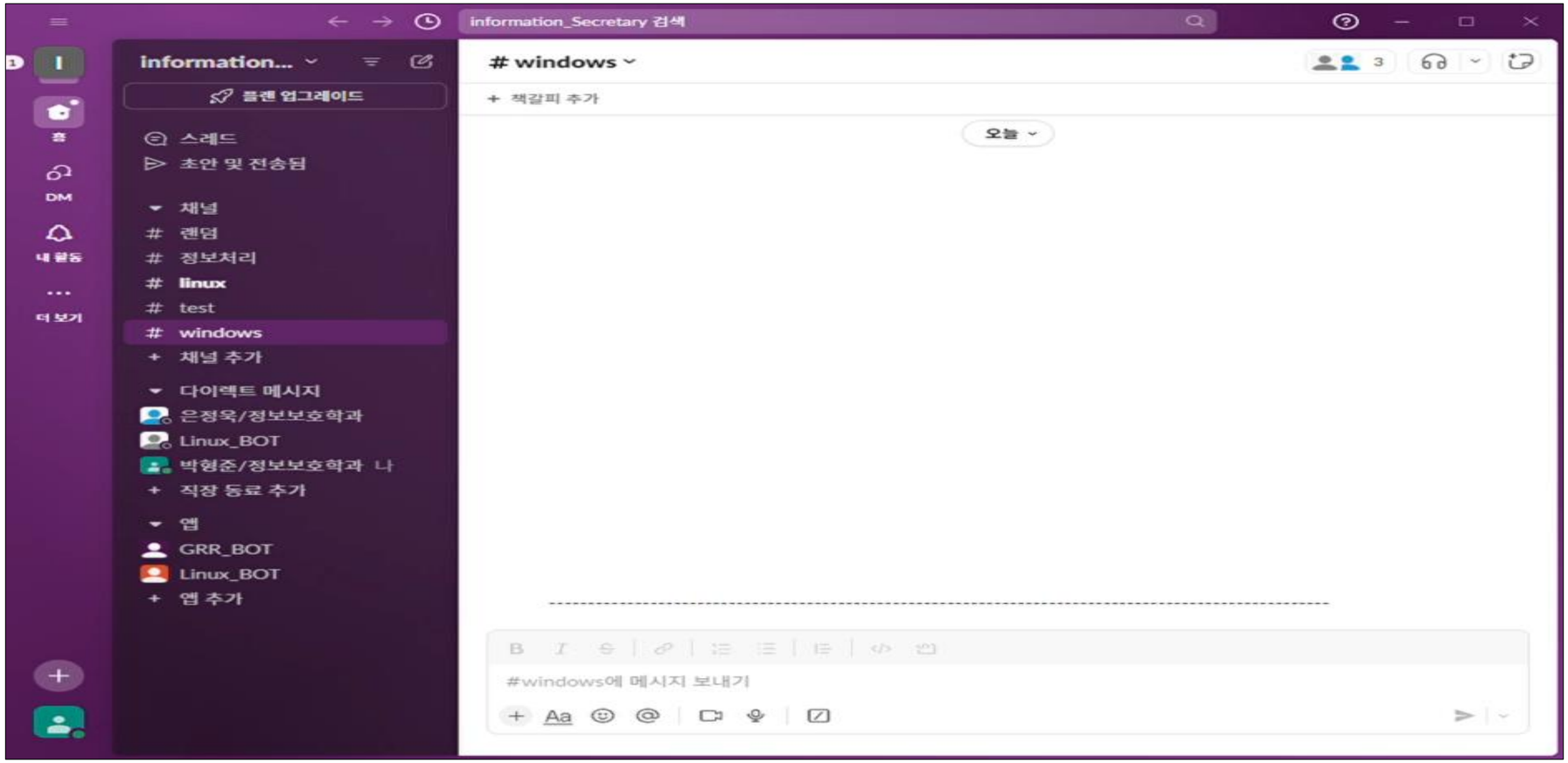
Fortinet	Android/Agent.JDU!tr	Trustlook	Android.PUA.DebugKey
Acronis (Static ML)	Undetected	Ad-Aware	Undetected
AhnLab-V3	Undetected	Alibaba	Undetected
ALYac	Undetected	Antiy-AVL	Undetected
Arcabit	Undetected	Avast	Undetected
Avast-Mobile	Undetected	AVG	Undetected

진단 결과 Slack 전송

시연 영상 - 리눅스


linux
책갈피 추가
오늘
LISTEN -
tcp 0 0 0.0.0.0:5601 0.0.0.0:*
LISTEN -
-tcp 0 1 192.168.246.194:59000 192.168.246.148:9200
SYN_SENT -
-tcp 0 236 192.168.246.194:22 192.168.246.144:35650
ESTABLISHED -
-tcp6 0 0 :::80 :::*
LISTEN -
+tcp 0 0 0.0.0.0:80 0.0.0.0:*
LISTEN -
+tcp 0 1 192.168.246.179:48806 192.168.246.148:9200
SYN_SENT -
+tcp 0 236 192.168.246.179:22 192.168.246.144:48540
ESTABLISHED -
tcp6 0 0 ::1:631 :::*
LISTEN -
-udp 0 0 0.0.0.0:42593 0.0.0.0:* -
+tcp6 0 0 :::80 :::*
LISTEN -
udp 0 0 127.0.0.53:53 0.0.0.0:* -
udp 0 0 0.0.0.0:68 0.0.0.0:* -
udp 0 0 0.0.0.0:631 0.0.0.0:* -
udp 0 0 0.0.0.0:5353 0.0.0.0:* -
-udp6 0 0 :::39290 :::* -
+udp 0 0 0.0.0.0:36378 0.0.0.0:* -
+udp6 0 0 :::60085 :::* -
udp6 0 0 :::5353 :::* -
Ubuntu 18.04 LTS 결과:
대상 경로:/var/www/html/Exaramel-Linux
결과 해시:
a9a0a1881a139bc7d1b69acf629ad4c36e1fb3e80120f4f1b6ed6e192a177c7a
VirusTotal
파일 위험도: 1/57
VT 검색 시간: 2023-10-09 17:30:30
VT 결과 링크:
https://www.virustotal.com/gui/file/a9a0a1881a139bc7d1b69acf629ad4c36e1fb3e80120f4f1b6ed6e192a177c7a/detection/f-a9a0a1881a139bc7d1b69acf629ad4c36e1fb3e80120f4f1b6ed6e192a177c7a-1680334384
VirusTotal
VirusTotal
VirusTotal
박형준/정보보호학과 오후 8:57
@Linux_BOT /linux_start
@Linux_BOT linux_start
#linux에 메시지 보내기

시연 영상 - 윈도우


information_Secretary 검색
information...
플랜 업그레이드
스레드
초안 및 전송됨
채널
랜덤
정보처리
linux
test
windows
채널 추가
다이렉트 메시지
은정욱/정보보호학과
Linux_BOT
박형준/정보보호학과 나
직장 동료 추가
앱
GRR_BOT
Linux_BOT
앱 추가
windows
3
책갈피 추가
오늘
#windows에 메시지 보내기
홈
DM
내 활동
더 보기

결론 및 기대 효과

결론

- 유료 도구들을 사용하지 않고 서버들의 위협을 탐지하는 서비스를 개발하였다.
- 불필요한 정보들은 제외하며 관리자가 원하는 정보만 수집하였다.
- 모든 사항을 Slack으로 확인함으로 사용자 친화적으로 확인할 수 있다.

기대 효과

- 값비싼 솔루션을 사용하지 못 하는 기업들의 비용을 아낄 수 있다.
- APT 그룹들의 목표를 가기 위한 C_J 서버와 Host에 대한 위험 부담을 줄일 수 있다.

감사합니다.

목표ᄀᄀKG