악성코드로 파일 정책에 대한 액세스 제어 활성화

목차

소개

이 문서에서는 탐지된 파일에서 SHA 조회를 수행하기 위해 SFDataCorrelator 프로세스를 사용하여 snort에 할당하는 방법에 대해 설명합니다.

사전 요구 사항

- 보호 및 악성코드 라이센스
- 악성코드를 사용하는 파일 정책

요구 사항

- 5.3.0 이상
- ASA(모든 모델)
- 7000 및 8000 series("AMP" 어플라이언스 제외)
- ASA에서 실행 중인 FTD
- FXOS 섀시에서 실행되는 FTD

사용되는 구성 요소

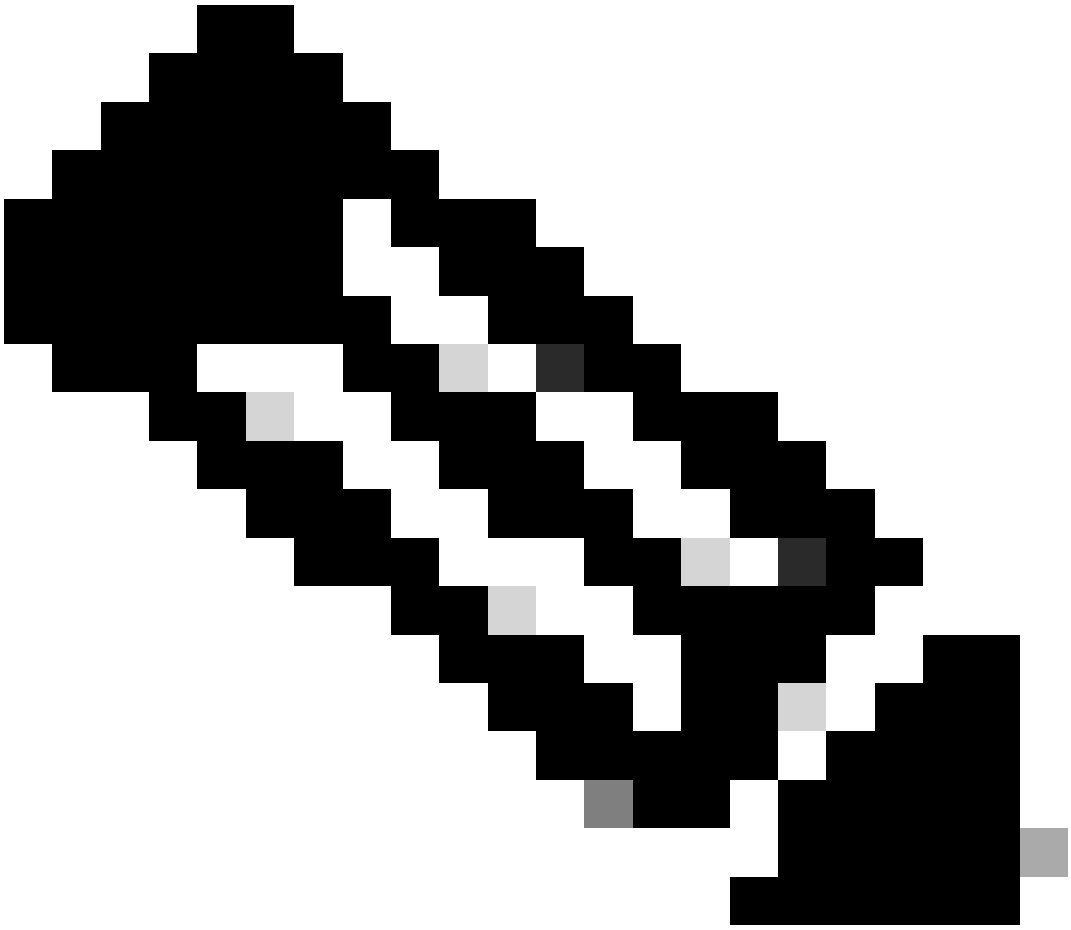
- 악성코드

이 문서의 정보는 특정 랩 환경의 디바이스를 토대로 작성되었습니다. 이 문서에 사용된 모든 디바이스는 초기화된(기본) 컨피그레이션으로 시작되었습니다. 현재 네트워크가 작동 중인 경우 모든 명령의 잠재적인 영향을 미리 숙지하시기 바랍니다.

배경 정보

Malware 작업 또는 "Store Files" 옵션을 사용하는 File(파일) 정책으로 액세스 제어 정책을 활성화하면 CPU(또는 대규모 모델의 경우 두 개)를 snort에서 제거할 수 있습니다.

성능에 미치는 영향



참고: 하위 리소스 어플라이언스에서 악성코드를 활성화하면 성능에 미치는 영향이 더 큽니다.

- 대기 시간
- 삭제
- 높은 CPU
- 낮은 처리량

문제 해결

AC 정책에서 파일 정책을 제거하거나 파일 정책을 사용하여 AC 규칙을 비활성화합니다. 그런 다음

AC 정책을 다시 적용하여 사용 가능한 모든 CPU 코어에 snort를 할당합니다.

ASA

```
root@Sourcefire3D:~# grep "SW\|MODEL" /etc/sf/ims.conf
SWVERSION=5.3.1
SWBUILD=152
MODEL_CLASS="3D Sensor"
MODELNUMBER=72
MODEL="ASA5545"
MODEL_TYPE=Sensor
MODELID=H

root@Sourcefire3D:~# pmtool show affinity
Received status (0):
Affinity Status
System CPU Affinity: 08 (desired: 08)
Process CPU Affinity:
Node 0:
CPU 0:
CPU 1:
SFDataCorrelator (/usr/local/sf/bin/SFDataCorrelator) (2, desired: 2)
CPU 2:
d54fff2a-37f7-11e4-a1bd-d47ac274f5bf-d01 (/var/sf/detection_engines/d54fff2a-37f7-11e4-a1bd-d47ac274f5b
CPU 3:
CPU 4:
d54fff2a-37f7-11e4-a1bd-d47ac274f5bf-d02 (/var/sf/detection_engines/d54fff2a-37f7-11e4-a1bd-d47ac274f5b
CPU 5:
d54fff2a-37f7-11e4-a1bd-d47ac274f5bf-d03 (/var/sf/detection_engines/d54fff2a-37f7-11e4-a1bd-d47ac274f5b
Device Affinity (0 PENDING):
kvm_ivshmem (desired: 01):
10: kvm_ivshmem (01)
Process Affinity:
SFDataCorrelator (desired: 02, actual: 02)
```

7000 및 8000시리즈(Series)

```
root@8250a-sftac:~# grep "SW\|MODEL" /etc/sf/ims.conf
SWVERSION=5.3.0
SWBUILD=571
MODEL_CLASS="3D Sensor"
MODELNUMBER=63
MODEL="3D8250"
MODEL_TYPE=Sensor
MODELID=C
root@8250a-sftac:~# pmtool show affinity
Received status (0):
Affinity Status
System CPU Affinity: fffff0 (desired: fffff0)
Process CPU Affinity:
Node 0:
CPU 0:
CPU 2:
SFDataCorrelator (/usr/local/sf/bin/SFDataCorrelator) (c, desired: c)
```

```
CPU 4:
3a3b8424-c8d3-11e4-98f5-1d2068538813-d01 (/var/sf/detection_engines/3a3b8424-c8d3-11e4-98f5-1d206853881
CPU 6:
3a3b8424-c8d3-11e4-98f5-1d2068538813-d03 (/var/sf/detection_engines/3a3b8424-c8d3-11e4-98f5-1d206853881
CPU 8:
3a3b8424-c8d3-11e4-98f5-1d2068538813-d05 (/var/sf/detection_engines/3a3b8424-c8d3-11e4-98f5-1d206853881
CPU 10:
3a3b8424-c8d3-11e4-98f5-1d2068538813-d07 (/var/sf/detection_engines/3a3b8424-c8d3-11e4-98f5-1d206853881
CPU 12:
3a3b8424-c8d3-11e4-98f5-1d2068538813-d09 (/var/sf/detection_engines/3a3b8424-c8d3-11e4-98f5-1d206853881
CPU 14:
3a3b8424-c8d3-11e4-98f5-1d2068538813-d10 (/var/sf/detection_engines/3a3b8424-c8d3-11e4-98f5-1d206853881
CPU 16:
3a3b8424-c8d3-11e4-98f5-1d2068538813-d02 (/var/sf/detection_engines/3a3b8424-c8d3-11e4-98f5-1d206853881
CPU 18:
3a3b8424-c8d3-11e4-98f5-1d2068538813-d04 (/var/sf/detection_engines/3a3b8424-c8d3-11e4-98f5-1d206853881
CPU 20:
3a3b8424-c8d3-11e4-98f5-1d2068538813-d06 (/var/sf/detection_engines/3a3b8424-c8d3-11e4-98f5-1d206853881
CPU 22:
3a3b8424-c8d3-11e4-98f5-1d2068538813-d08 (/var/sf/detection_engines/3a3b8424-c8d3-11e4-98f5-1d206853881
Node 1:
CPU 1:
CPU 3:
SFDataCorrelator (/usr/local/sf/bin/SFDataCorrelator) (c, desired: c)
CPU 5:
3a3b8424-c8d3-11e4-98f5-1d2068538813-d11 (/var/sf/detection_engines/3a3b8424-c8d3-11e4-98f5-1d206853881
CPU 7:
3a3b8424-c8d3-11e4-98f5-1d2068538813-d12 (/var/sf/detection_engines/3a3b8424-c8d3-11e4-98f5-1d206853881
CPU 9:
3a3b8424-c8d3-11e4-98f5-1d2068538813-d13 (/var/sf/detection_engines/3a3b8424-c8d3-11e4-98f5-1d206853881
CPU 11:
3a3b8424-c8d3-11e4-98f5-1d2068538813-d14 (/var/sf/detection_engines/3a3b8424-c8d3-11e4-98f5-1d206853881
CPU 13:
3a3b8424-c8d3-11e4-98f5-1d2068538813-d15 (/var/sf/detection_engines/3a3b8424-c8d3-11e4-98f5-1d206853881
CPU 15:
3a3b8424-c8d3-11e4-98f5-1d2068538813-d16 (/var/sf/detection_engines/3a3b8424-c8d3-11e4-98f5-1d206853881
CPU 17:
3a3b8424-c8d3-11e4-98f5-1d2068538813-d17 (/var/sf/detection_engines/3a3b8424-c8d3-11e4-98f5-1d206853881
CPU 19:
3a3b8424-c8d3-11e4-98f5-1d2068538813-d18 (/var/sf/detection_engines/3a3b8424-c8d3-11e4-98f5-1d206853881
CPU 21:
3a3b8424-c8d3-11e4-98f5-1d2068538813-d19 (/var/sf/detection_engines/3a3b8424-c8d3-11e4-98f5-1d206853881
CPU 23:
3a3b8424-c8d3-11e4-98f5-1d2068538813-d20 (/var/sf/detection_engines/3a3b8424-c8d3-11e4-98f5-1d206853881
Endpoint CPUs:
c0e1: 0 (desired: -1)
c1e1: 1 (desired: -1)
Process Affinity:
SFDataCorrelator (desired: 0c, actual: 0c)
```

FTD

모든 FTD 플랫폼에서 SSH 액세스 후 초기 '>' 프롬프트에서 이전pmtool show affnity 명령을 실행할 수 있습니다. 예를 들면 다음과 같습니다.

Cisco Fire Linux OS v6.2.1 (build 6)
Cisco Firepower 2110 Threat Defense v6.2.1 (build 327)

> pmtool show affinity
Received status (0):

Affinity Status

System CPU Affinity: 0 (desired: 0)

Process CPU Affinity:
CPU 0:
CPU 1:
65a99306-360b-11e7-a8f4-5671cccf5a71-d01 (/ngfw/var/sf/detection_engines/65a99306-360b-11e7-a8f4-5671cccf5a71/snort) (?, desired: 1,5)
CPU 2:
65a99306-360b-11e7-a8f4-5671cccf5a71-d02 (/ngfw/var/sf/detection_engines/65a99306-360b-11e7-a8f4-5671cccf5a71/snort) (?, desired: 2,6)
CPU 3:
65a99306-360b-11e7-a8f4-5671cccf5a71-d03 (/ngfw/var/sf/detection_engines/65a99306-360b-11e7-a8f4-5671cccf5a71/snort) (?, desired: 3,7)
CPU 4:
CPU 5:
65a99306-360b-11e7-a8f4-5671cccf5a71-d01 (/ngfw/var/sf/detection_engines/65a99306-360b-11e7-a8f4-5671cccf5a71/snort) (?, desired: 1,5)
CPU 6:
65a99306-360b-11e7-a8f4-5671cccf5a71-d02 (/ngfw/var/sf/detection_engines/65a99306-360b-11e7-a8f4-5671cccf5a71/snort) (?, desired: 2,6)
CPU 7:
65a99306-360b-11e7-a8f4-5671cccf5a71-d03 (/ngfw/var/sf/detection_engines/65a99306-360b-11e7-a8f4-5671cccf5a71/snort) (?, desired: 3,7)

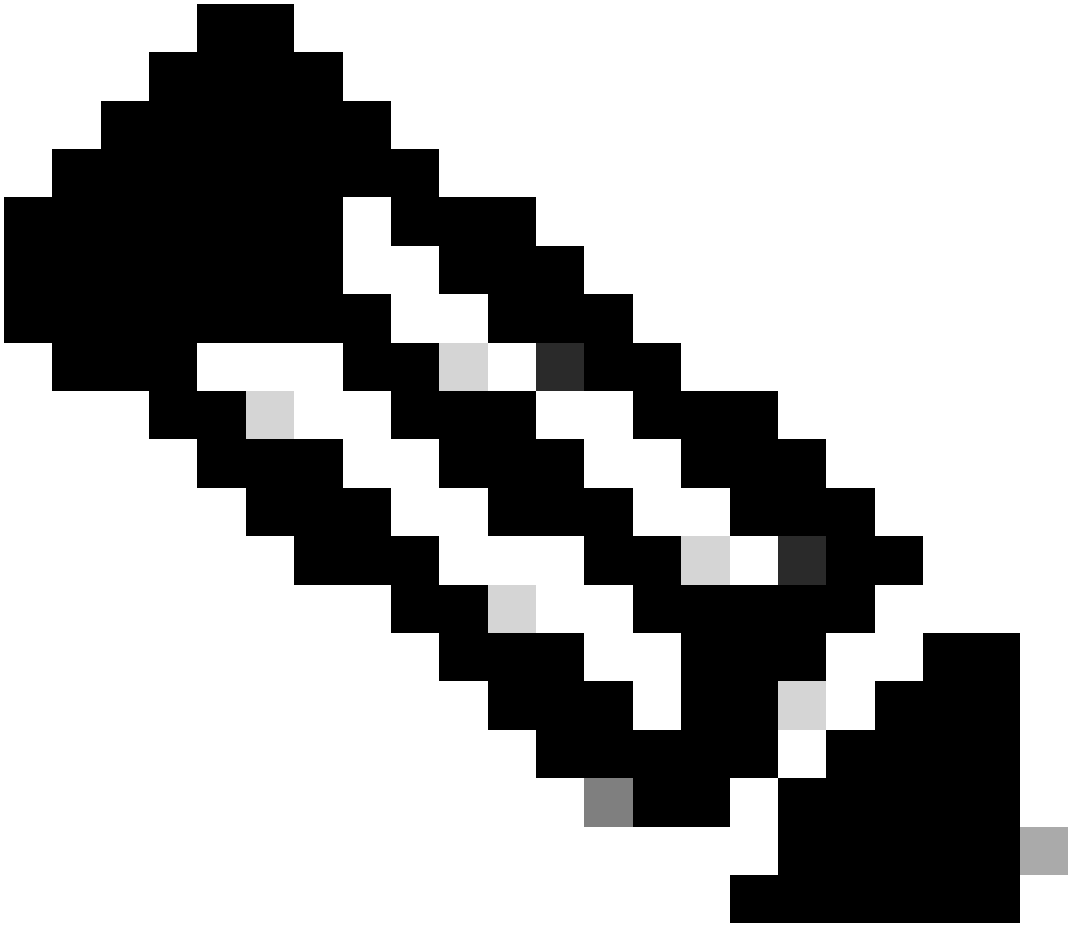
문제 해결 파일의 pmtool show affnity 명령 출력은 command-outputs 디렉토리에 있습니다. 파일 이름은 usr-**local-sf-bin-pmtool show affinity.output입니다.**

대형 어플라이언스의 문제 해결 시 실행하는 경우 출력이 상당히 길 수 있습니다. 다음은 snort 및 SFDataCorrelator 프로세스에 할당되는 CPU 수를 명확하게 나타내는 몇 가지 grep 명령입니다.

[user@tex command-outputs]$ grep snort usr-local-sf-bin-pmtool\ show\ affinity.output |wc -l
46

[user@tex command-outputs]$ grep "/SFDataC" usr-local-sf-bin-pmtool\ show\ affinity.output |wc -l
2

이전 출력은 현재 가장 큰 장치(FPR-9300 SM-44)에서 얻은 것입니다. 보시는 것처럼, CPU가 46개가 Snort에 할당되어 있고 2개가 SFDataCorrelator에 할당되어 있습니다(악성코드 정책이 활성화됨).



참고: TS Analysis(TS 분석)에서는 이러한 시나리오에서 전체 DE 성능 그래프를 올바르게 표시할 수 없습니다

이 번역에 관하여

Cisco는 전 세계 사용자에게 다양한 언어로 지원 콘텐츠를 제공하기 위해 기계 번역 기술과 수작업 번역을 병행하여 이 문서를 번역했습니다. 아무리 품질이 높은 기계 번역이라도 전문 번역가의 번역 결과물만큼 정확하지는 않습니다. Cisco Systems, Inc.는 이 같은 번역에 대해 어떠한 책임도 지지 않으며 항상 원본 영문 문서(링크 제공됨)를 참조할 것을 권장합니다.