Secure Firewall Threat Defense 7.4에서 AppID Early Packet Detection 구성

목차

소개

이 문서에서는 Cisco Secure Firewall 7.4에서 AppID Early Packet Detection을 구성하는 방법에 대해 설명합니다.

배경 - 문제(고객 요구 사항)

- Deep Packet Inspection을 통한 애플리케이션 탐지는 트래픽을 식별하는 데 둘 이상의 패킷이 걸릴 수 있습니다.
- 애플리케이션 서버의 IP 및/또는 포트를 알고 있는 경우 추가 패킷 검사를 피할 수 있습니다.

새로운 기능

- IP 주소, 포트 및 프로토콜을 각각에 매핑할 수 있는 새로운 Snort 기반 Lua AppID API가 생성되었습니다.
  - 애플리케이션 프로토콜(서비스 appid),
  - 클라이언트 애플리케이션(클라이언트 appid) 및
  - 웹 애플리케이션(페이로드 appid).
- 애플리케이션 탐지를 위해 이 API를 사용하여 FMC에서 사용자 지정 애플리케이션 탐지기를 생성할 수 있습니다.
- 이 탐지기가 활성화되면 이 새로운 API를 통해 세션의 첫 번째 패킷에서 애플리케이션을 식별할 수 있습니다.

기능 개요

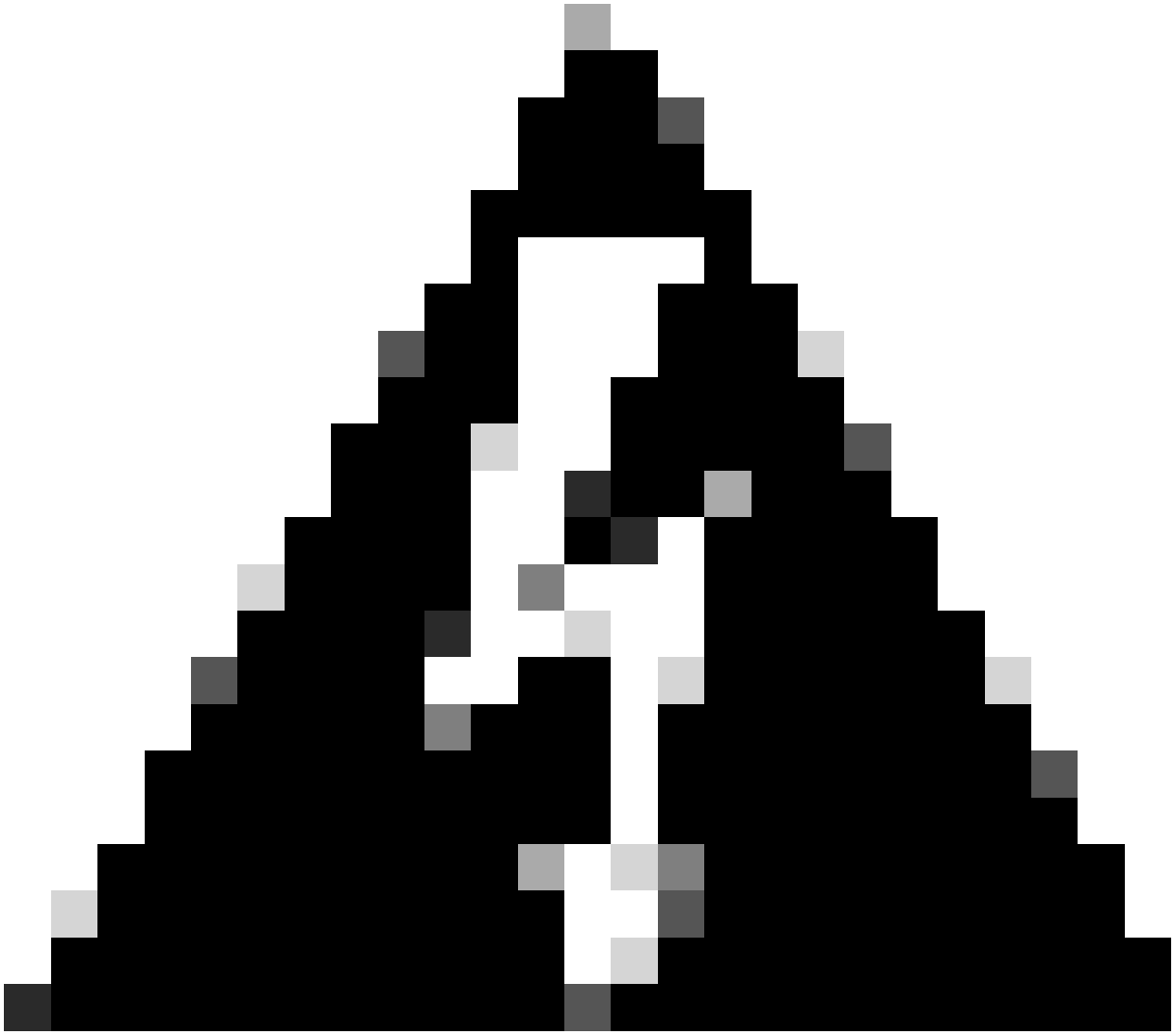
- API는 다음과 같이 식별됩니다.
  - **addHostFirstPktApp** (protocol_appId, client_appId, payload_appId, IP 주소, 포트, 프로토콜, 재검사)

- 맞춤형 앱 탐지기에서 생성된 모든 매핑에 대해 캐시 항목이 생성됩니다.
- 모든 수신 세션의 첫 번째 패킷을 검사하여 캐시에서 일치하는 항목이 있는지 확인합니다.
- 일치하는 항목이 발견되면 해당 세션에 대한 해당 appid를 할당하고 앱 검색 프로세스를 중지합니다.
- 사용자는 API에서 일치하는 항목을 찾은 후에도 트래픽을 재검사할 수 있습니다.
- reinspect 인수는 첫 번째 패킷에 있는 애플리케이션을 재검사할 필요가 있는지 여부를 나타내는 부울 값입니다.
- 재검사가 참이면 API에서 일치하는 항목을 발견하더라도 앱 검색이 계속됩니다.
- 이 경우 첫 번째 패킷에 할당된 appid가 변경될 수 있습니다.

사전 요구 사항, 지원되는 플랫폼, 라이센싱

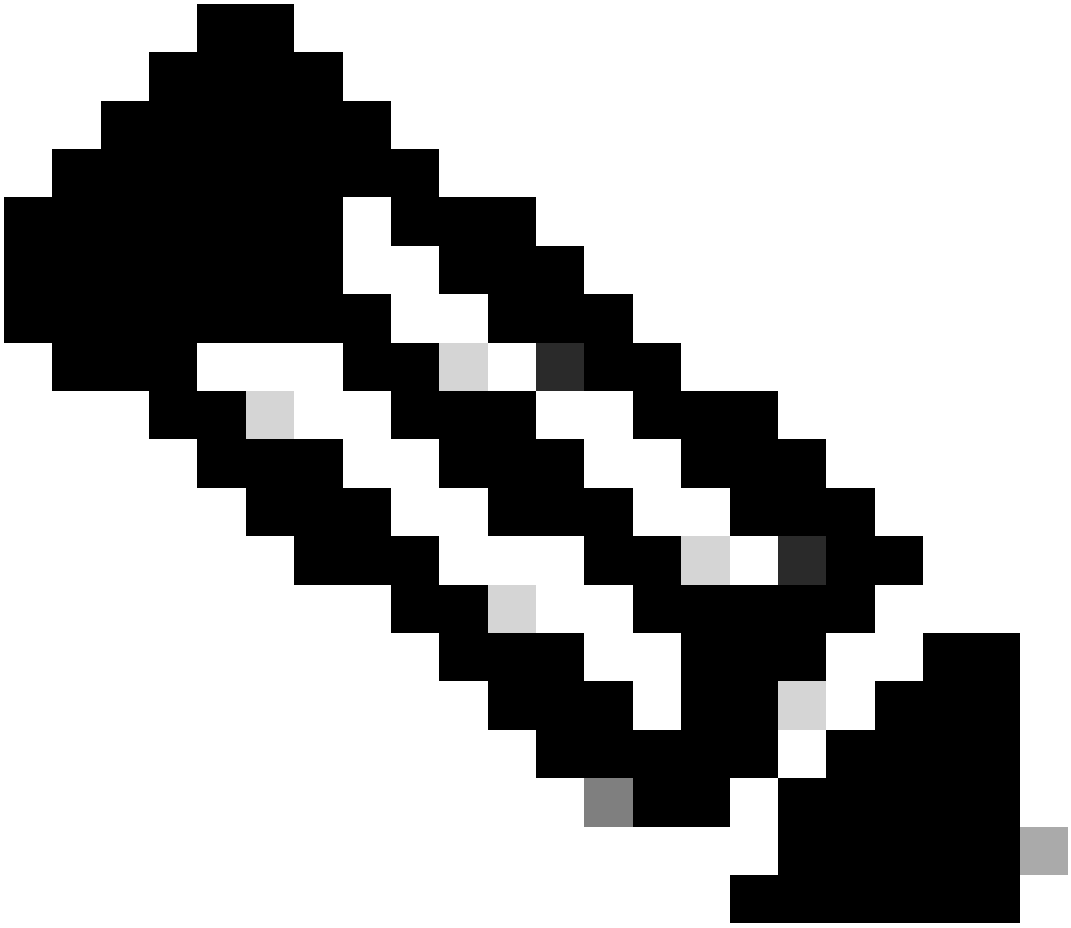
최소 소프트웨어 및 하드웨어 플랫폼

애플리케이션 및 최소 버전	지원되는 관리 플랫폼 및 버전	관리자	참고
Secure Firewall 7.4 Snort3 사용	FTD 7.4를 지원하는 모든 플랫폼	FMC 온프레미스 + FTD	이는 디바이스측 기능입니다. FTD는 7.4에 있어야 합니다.



경고: Snort 2는 이 API를 지원하지 않습니다.

Snort 3, 다중 인스턴스, HA/클러스터링 지원



참고: Snort 3이 탐지 엔진이어야 합니다.

FTD	
다중 인스턴스가 지원됩니까?	예
HA 디바이스에서 지원됨	예
클러스터링된 디바이스에서 지원됩니	예

까?	

사용되는 구성 요소

이 문서의 정보는 다음 소프트웨어 및 하드웨어 버전을 기반으로 합니다.

· Cisco Firepower Threat Defense 7.4 이상

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.

기능 세부사항

기능 기능 설명

이번 릴리스에 대한 이전 버전 대비

Secure Firewall 7.3 이하	Secure Firewall의 새로운 기능 7.4
· 알려진 IP/포트/프로토콜 조합에 대한 애플리케이션 탐지는 다른 모든 애플리케이션 탐지 메커니즘이 소진된 후에만 대체 옵션으로 사용할 수 있습니다. · 기본적으로 세션의 첫 번째 패킷에 대한 탐지는 지원되지 않습니다.	· 새로운 lua 탐지기 API는 다른 앱 탐지 메커니즘보다 먼저 평가됩니다. · 따라서 7.4에서는 세션의 첫 번째 패킷에 대한 탐지를 지원합니다.

운영 방식

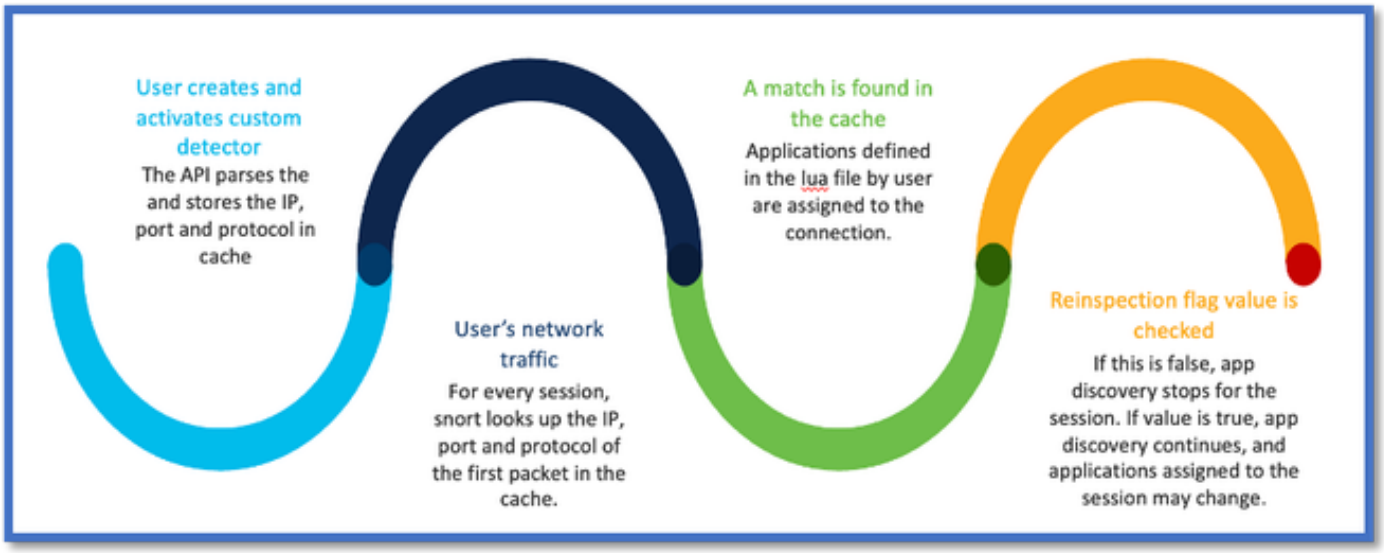
- lua 파일 만들기: 파일이 lua 템플릿에 있는지 확인합니다(구문 오류 없음). 또한 파일의 API에 제공된 인수가 올바른지 확인합니다.

- 새 사용자 지정 탐지기 생성: FMC에서 새 사용자 지정 탐지기를 생성하고 lua 파일을 업로드합니다. 탐지기를 활성화합니다.

- Run traffic(트래픽 실행): 맞춤형 앱 탐지기에 정의된 IP/포트/프로토콜 조합과 일치하는 트래픽을 디바이스로 보냅니다.

- Check connection events(연결 이벤트 확인): FMC에서 IP 및 포트로 필터링된 연결 이벤트를 확인합니다. 사용자 정의 애플리케이션이 식별됩니다.

AppID Early Packet Detection API 워크플로



사용자 지정 탐지기의 API 필드 설명 예

gDetector:addHostFirstPktApp

(gAppIdProto, gAppIdClient, gAppId, 0, "192.0.2.1", 443, DC.ipproto.tcp);

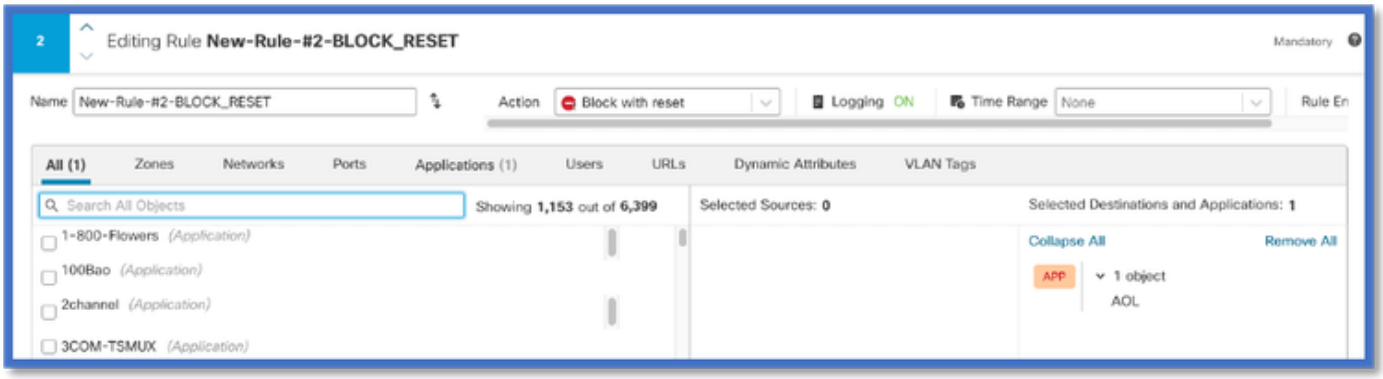
- 강조 표시된 인수는 reinspect 플래그, IP 주소, 포트 및 프로토콜에 대한 사용자 정의 값입니다.
- 0은 와일드카드를 나타냅니다.

인수	설명	예상 값
Reinspect 플래그	사용자가 IP/Port/Protocol을 기반으로 방화벽 작업을 수행하는 대신 트래픽을 검사하려는 경우 reinspect 플래그 값을 1로 활성화할 수 있습니다.	0 = 재검사 사용 안 함 또는 1 = 재검사 사용
IP 주소	서버의 대상 IP(단일 또는 서브넷의 IP 범위)입니다. 세션의 첫 번째 패킷의 대상 IP	192.168.4.198 또는 192.168.4.198/24 또는 2a03:2880:f103:83:face:b00c:0:25de 또는

		2a03:2880:f103:83:face:b00c:0:25de/32
포트	세션의 첫 번째 패킷의 대상 포트.	0~65535
프로토콜	네트워크 프로토콜	TCP/UDP/ICMP

활용 사례: 더 신속하게 트래픽을 차단하는 방법

- 정책 보기: 애플리케이션 "AOL"에 대한 차단 규칙



- curl을 사용하여 트래픽 테스트: curl https://www.example.com v/s curl https://192.0.2.1/(TEST IP 주소 중 하나)

<#root>

> curl https://www.example.com/

curl: (35) OpenSSL SSL_connect: SSL_ERROR_SYSCALL in connection to www.example.com:443

> curl https://192.0.2.1/

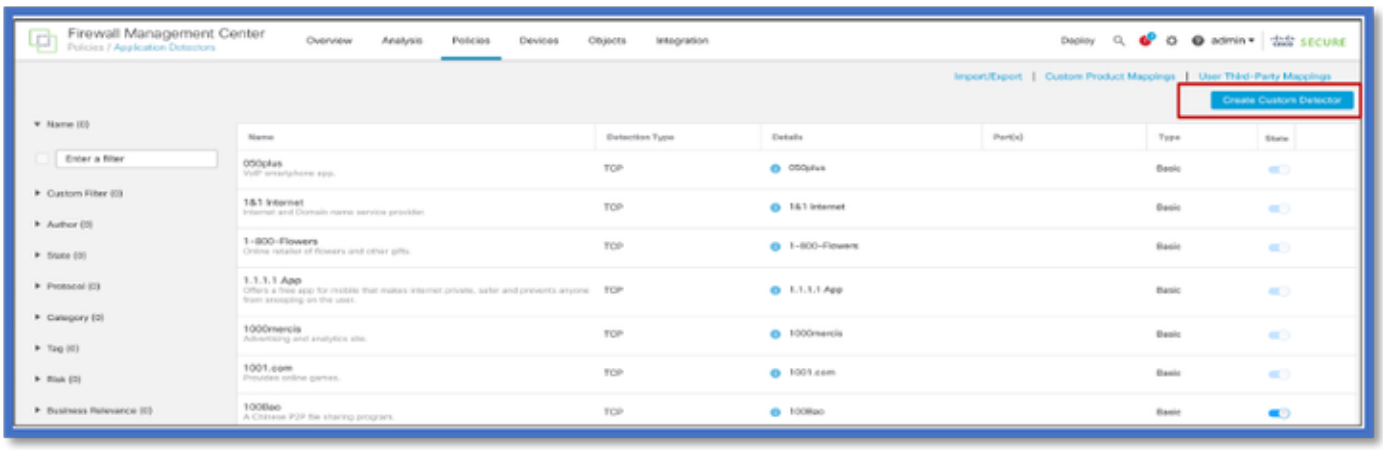
```
curl: (7) Failed to connect to 192.0.2.1 port 443: Connection refused
```

방화벽 관리 센터 연습

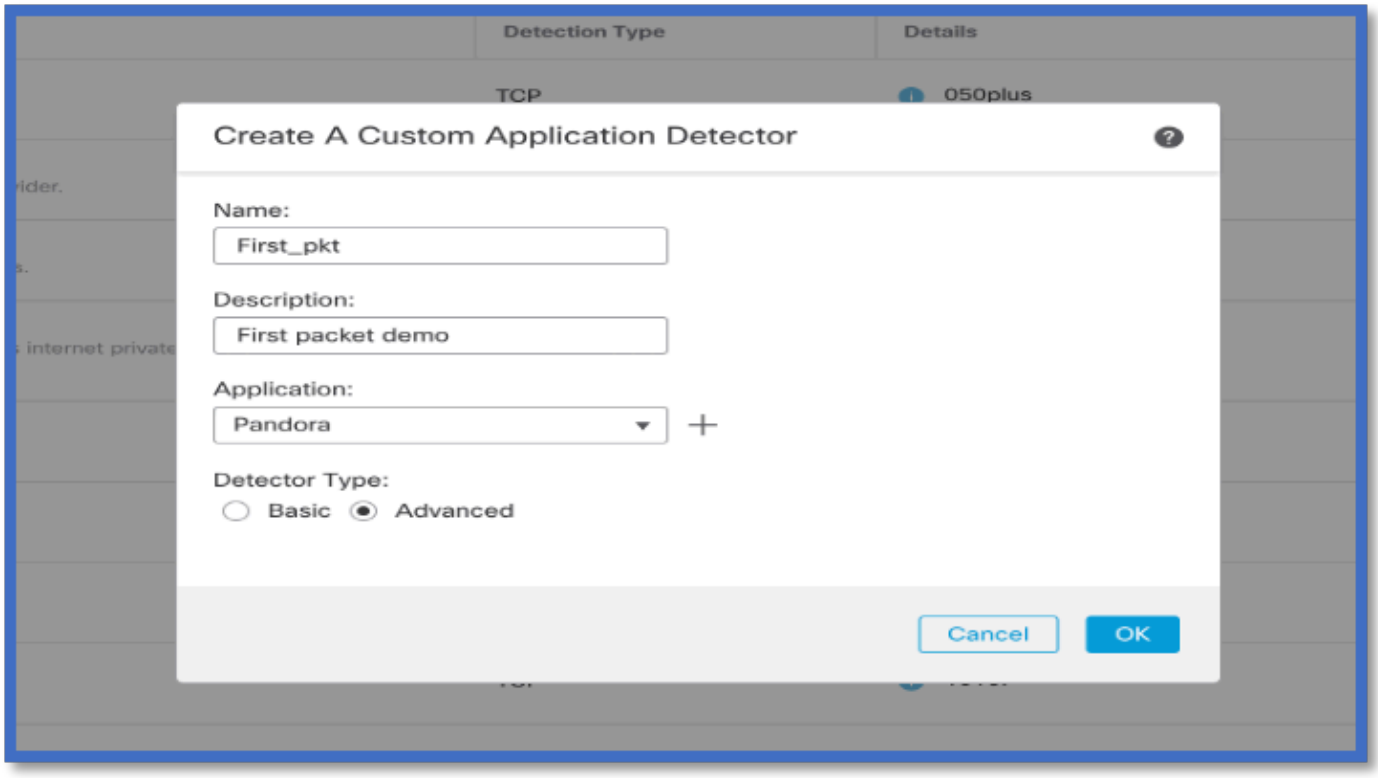
API를 사용하여 맞춤형 탐지기를 생성하는 단계

다음 위치에서 FMC에 새 사용자 지정 탐지기를 생성합니다.

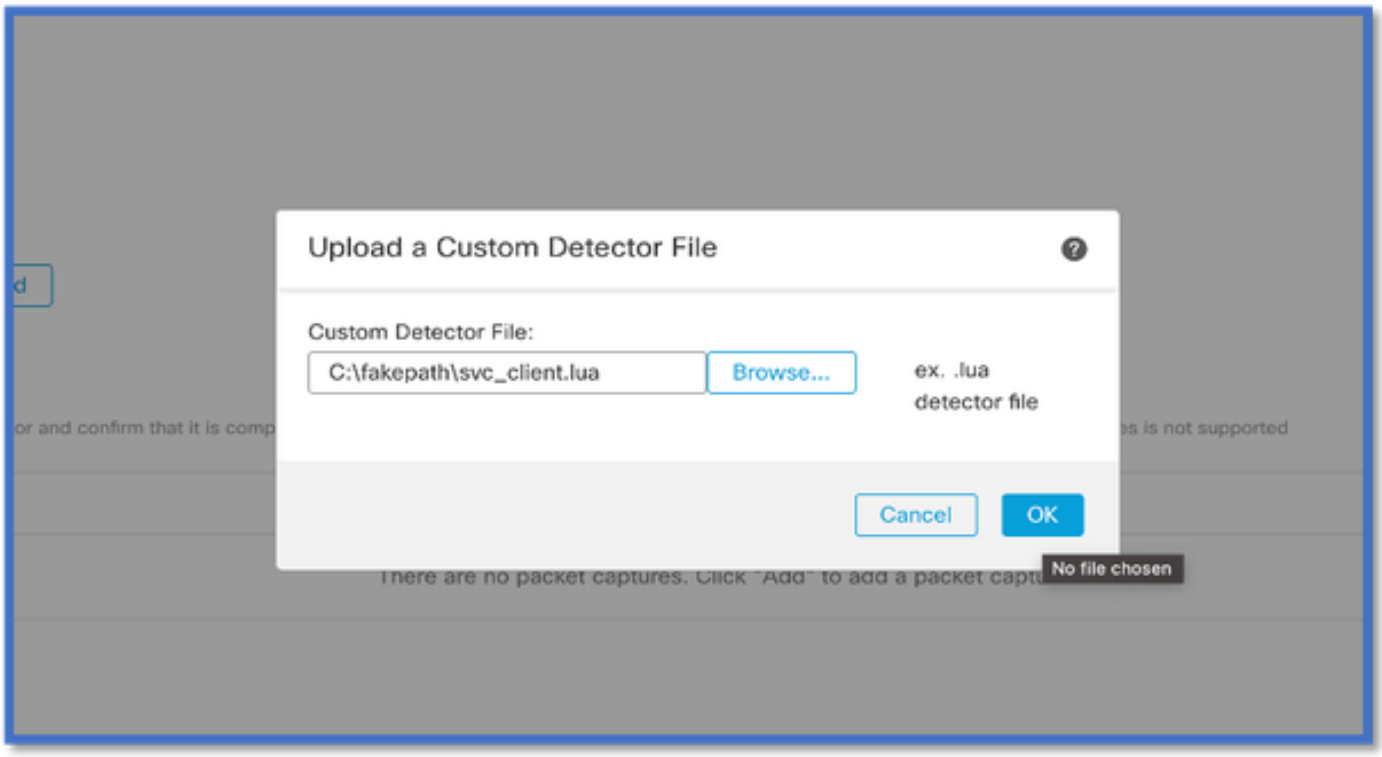
- Policies > Application Detectors > Create Custom Detector .



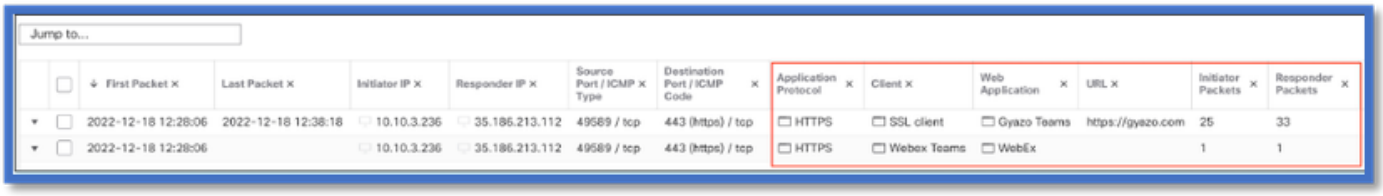
- 이름 및 설명을 정의합니다.
  - 드롭다운 메뉴에서 애플리케이션을 선택합니다.
  - Advanced Detector Type을 선택합니다.



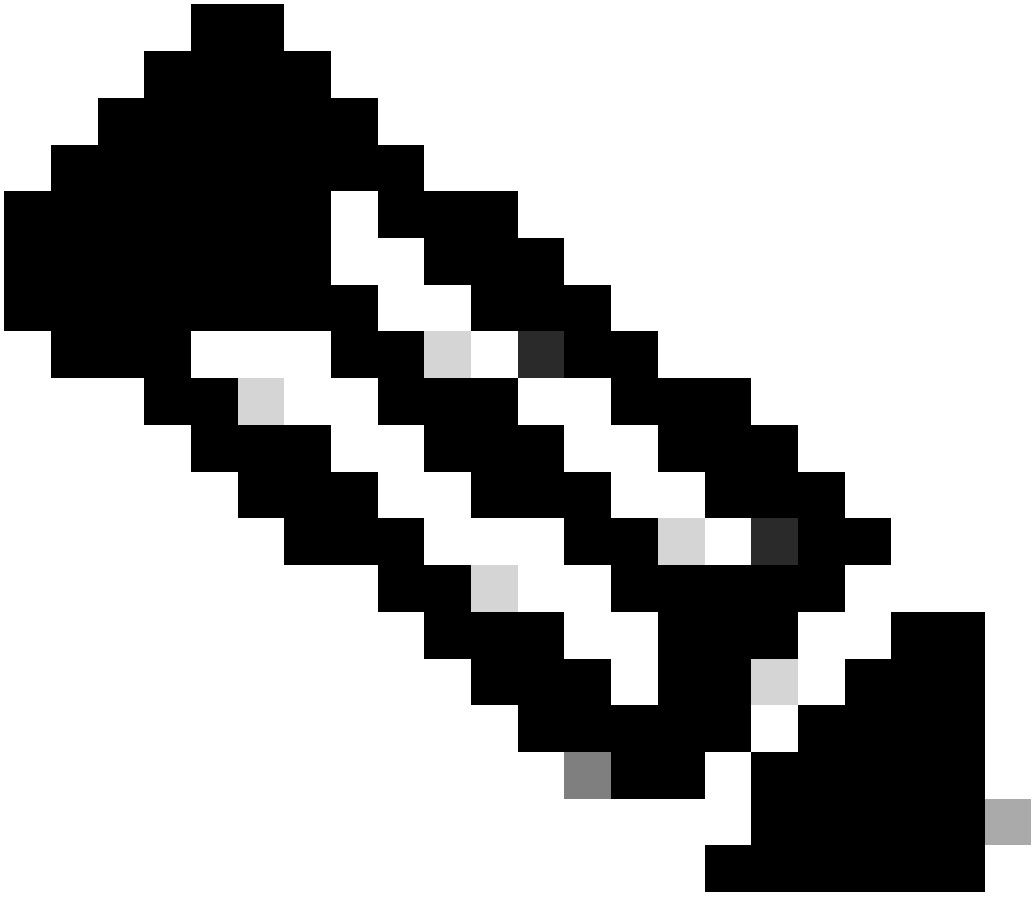
- Detection Criteria(탐지 기준) 아래에 Lua 파일을 업로드합니다. 탐지기를 저장하고 활성화합니다.



Reinspect Enabled v/s Disabled(재검사 활성화 v/s 비활성화)



- 두 이벤트는 재검사가 활성화된 경우 연결 시작 v/s와 연결 종료를 보여줍니다.



참고: 참고 사항:

1. 'HTTPS, Webex 및 Webex 팀'은 연결 시작 시 API에 의해 식별됩니다. 재검사가 참이므로 앱 검색이 계속되고 appIds가 'HTTPS, SSL Client and Gyazo Teams'로 업데이트됩니다.

2. initiator 및 responder 패킷의 수를 확인합니다. 일반적인 앱 탐지 방식에는 API보다 훨씬 많은 패킷이 필요합니다.

문제 해결/진단

진단 개요

- 첫 번째 패킷 탐지 API에 의해 발견된 애플리케이션이 있는지 여부를 나타내기 위해 시스템 지원 애플리케이션 식별 디버그에 새 로그가 추가됩니다.
- 사용자가 트래픽의 재검사를 선택했는지도 로그에 표시됩니다.
- 사용자가 업로드한 lua 탐지기 파일의 내용은 아래의 FTD에서 확인할 수 /var/sf/appid/custom/lua/<UUID> 있습니다.
- lua 파일의 모든 오류는 탐지기 활성화 시 /var/log/messages 파일의 FTD에 덤프됩니다.

CLI: 시스템 지원 application-identification-debug

<#root>

192.0.2.1 443 -> 192.168.1.16 51251 6 AS=4 ID=0 New AppId session

192.0.2.1 443 -> 192.168.1.16 51251 6 AS=4 ID=0 Host cache match found on first packet, service: HTTPS(1

```
192.168.1.16 51251 -> 192.0.2.1 443 6 AS=4 ID=0 app event with client changed, service changed, payload
192.168.1.16 51251 -> 192.0.2.1 443 6 AS=4 ID=0 New firewall session
192.168.1.16 51251 -> 192.0.2.1 443 6 AS=4 ID=0 Starting with minimum 2, 'New-Rule-#1-MONITOR', and Src
192.168.1.16 51251 -> 192.0.2.1 443 6 AS=4 ID=0 match rule order 2, 'New-Rule-#1-MONITOR', action Audit
```

192.168.1.16 51251 -> 192.0.2.1 443 6 AS=4 ID=0 match rule order 3, 'New-Rule-#2-BLOCK_RESET', action Re

```
192.168.1.16 51251 -> 192.0.2.1 443 6 AS=4 ID=0 MidRecovery data sent for rule id: 268437504, rule_acti
192.168.1.16 51251 -> 192.0.2.1 443 6 AS=4 ID=0 Generating an SOF event with rule_id = 268437504 ruleAc
```

192.168.1.16 51251 -> 192.0.2.1 443 6 AS=4 ID=0 reset action

192.0.2.1 443 > 192.168.1.16 51251 6 AS-4 ID=0 New AppId session
192.0.2.1 443 > 192.168.1.16 51251 6 AS=4 ID=0 Host cache match found on first packet, service:
HTTPS (1122), client: AOL(1419), payload: AOL (1419), reinspect: False
192.168.1.16 51251 > 192.0.2.1 443 6 AS=4 ID=0 app event with client changed, service changed, payload changed, referred no change, miss no change, Mad no change, fas host no change, bits 0x1D 192.168.1.16 51251 > 192.0.2.1 443 6 AS=4 ID=0 New firewall session
192.168.1.16 51251 > 192.0.2.1 443 6 AS=4 ID=0 Starting with minimum 2, 'New-Rule-#1-MONITOR', and Saclone first with zones 1 →> 1, geo 0(xff0) →> 0, yan 0, sae, sgt; 0, sag sat, type: unknown, det sat: 0, det sat type: unknown, sve 1122, payload 1419, client 1419, mise 0, user 9999997, no Mad or host, no xff
192.168.1.16 51251 > 192.0.2.1 443 6 AS=4 ID=0 match rule order 2, 'New-Rule-#1-MONITOR', action Audit
192.168.1.16 51251 > 192.0.2.1 443 6 AS=4 ID=0 match rule order 3, 'New-Rule-#2-BLOCK_
_RESET', action
Reset
192.168.1.16 51251 > 192.0.2.1 443 6 AS-4 ID=0 MidRecovery, data sent for rule id: 268437504, rule_action:5, rev id:3558448739, Eule_match flag:0x1
192.168.1.16 51251 > 192.0.2.1 443 6 AS-4 ID-0 Generating an SOF event with zuleid - 268437504
ruleAction = 5 ruleReason = 0

AppID Lua 탐지기 콘텐츠의 위치

이 새 API가 포함된 Lua 탐지기가 디바이스/FTD에 있는지 확인하려면 addHostFirstPktApp API가 다음 2개의 애플리케이션 탐지기 폴더에서 사용되고 있는지 확인할 수 있습니다.

1. VDB 애플리케이션 ID 탐지기 -/var/sf/appid/odp/lua

2. 사용자 지정 탐지기 -/var/sf/appid/custom/lua

예:grep addHostFirstPktApp * 각 폴더에서

샘플 문제:

- 문제: FMC에서 사용자 지정 Lua 탐지기가 활성화되지 않았습니다.

확인할 위치: /var/sf/appid/custom/lua/

예상 결과: FMC에서 활성화된 모든 사용자 지정 앱 탐지기에 대해 하나의 파일이 여기에 있어야 합니다. 내용이 업로드된 lua 파일과 일치하는지 확인합니다.

- 문제: 업로드된 lua 탐지기 파일에 오류가 있습니다.

확인할 파일: /var/log/messages on FTD

오류 로그:

<#root>

Dec 18 14:17:49 intel-x86-64 SF-IMS[15741]:

Error - appid: can not set env of Lua detector /ngfw/var/sf/appid/custom/lua/6698fbd6-7ede-11ed-972c-d12

문제 해결 단계

문제: 사용자 정의 IP 주소 및 포트로 이동하는 트래픽에 대해 애플리케이션이 올바르게 식별되지 않았습니다.

문제 해결 단계:

- lua 탐지기가 FTD에서 올바르게 정의되고 활성화되었는지 확인합니다.
    - FTD에서 lua 파일의 내용을 확인하고 활성화 시 오류가 표시되지 않는지 확인합니다.
- 트래픽 세션에서 첫 번째 패킷의 목적지 IP, 포트 및 프로토콜을 확인합니다.
    - lua 탐지기에 정의된 값을 매칭할 수 있습니다.
- system-support-application-identification-debug를 확인합니다.
    - 행을 찾습니다. Host cache match found on first packet. 이 항목이 없으면 API에서 일치하는 항목을 찾을 수 없음을 나타냅니다.

제한 사항 세부 정보, 일반적인 문제 및 해결 방법

7.4에서는 API를 사용할 UI가 없습니다. UI 지원은 향후 릴리스에서 추가될 예정입니다.

개정 이력

개정	게시 날짜	의견

1.0	2024년 7월 18일	최초 릴리스

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.