맞춤형 침입 규칙

다음 주제에서는 침입 규칙 편집기를 사용하는 방법을 설명합니다.

맞춤형 침입 규칙 개요

침입 규칙은 시스템이 네트워크에서 취약성을 익스플로잇하려는 시도를 탐지하는 데 사용하는 키워드와 인수의 집합입니다. 시스템은 네트워크 트래픽을 분석하면서 패킷을 각 규칙에 지정된 조건과 비교합니다. 패킷 데이터가 규칙에 지정된 모든 조건과 일치하는 경우, 규칙이 트리거됩니다. 규칙이 알림 규칙인 경우, 침입 이벤트를 생성합니다. 규칙이 전달 규칙인 경우, 트래픽을 무시합니다. 인라인 구축의 삭제 규칙의 경우, 시스템은 패킷을 삭제하고 이벤트를 생성합니다. Secure Firewall Management Center 웹 인터페이스에서 침입 이벤트를 보고 평가할 수 있습니다.

Firepower System은 공유 개체 규칙과 표준 텍스트 규칙이라는 두 가지 유형의 침입 규칙을 제공합니다. Talos 인텔리전스 그룹는 공유 개체 규칙을 사용하여 기존의 표준 텍스트 규칙에서는 불가능한 방식으로 취약성에 대한 공격을 탐지할 수 있습니다. 공유 객체 규칙은 생성할 수 없습니다. 자체 침입 규칙을 작성하는 경우, 표준 텍스트 규칙을 생성합니다.

맞춤형 표준 텍스트 규칙을 작성하여 보게 될 이벤트 유형을 조정할 수 있습니다. 이 설명서는 특정 익스플로잇 탐지를 목표로 하는 규칙을 설명하고 있지만, 가장 성공적인 규칙은 알려진 특정 익스플로잇보다는 알려진 취약성을 공격하려고 시도하는 트래픽을 대상으로 합니다. 규칙을 작성하고 규칙의 이벤트 메시지를 지정하여, 공격 및 정책 회피를 나타내는 트래픽을 더욱 쉽게 확인할 수 있습니다.

사용자 지정 침입 정책에서 사용자 지정 표준 텍스트 규칙을 활성화하는 경우, 트래픽이 특정 방법으로 먼저 디코딩 또는 전처리되는 것을 일부 규칙 키워드 및 인수가 요구한다는 점에 유의하십시오. 이 챕터에서는 전처리를 제어하는 네트워크 분석 정책에서 구성해야 하는 옵션을 설명합니다. 필수

전처리기를 비활성화하는 경우, 네트워크 분석 정책 웹 인터페이스에서는 전처리기가 비활성화되어 있더라도 시스템은 자동으로 전처리기를 현재의 설정으로 사용합니다.

⚠

주의 제어 네트워크 환경을 사용하여 프로덕션 환경에서 규칙을 사용하기 전에 작성하는 모든 침입 규칙을 테스트하도록 하십시오. 잘못 작성한 침입 규칙은 시스템의 성능에 심각한 영향을 줄 수 있습니다.

다중 도메인 구축에서 시스템은 현재 도메인에서 생성된 규칙을 표시하며 이러한 규칙은 수정할 수 있습니다. 상위 도메인에서 생성된 규칙도 표시되지만, 이러한 규칙은 수정할 수 없습니다. 하위 도메인에서 생성된 규칙을 보고 수정하려면 해당 도메인으로 전환하십시오. 시스템 제공 침입 규칙은 전역 도메인에 속합니다. 하위 도메인의 관리자는 이러한 시스템 규칙의 편집 가능한 로컬 복사본을 만들 수 있습니다.

침입 규칙 편집기 라이선스 요구 사항

Threat Defense 라이선스

IPS

기본 라이선스

보호

침입 규칙 편집기 요구 사항 및 사전 요건

모델 지원

모두

지원되는 도메인

모든

사용자 역할

- 관리자
- 침입 관리자

규칙 구조

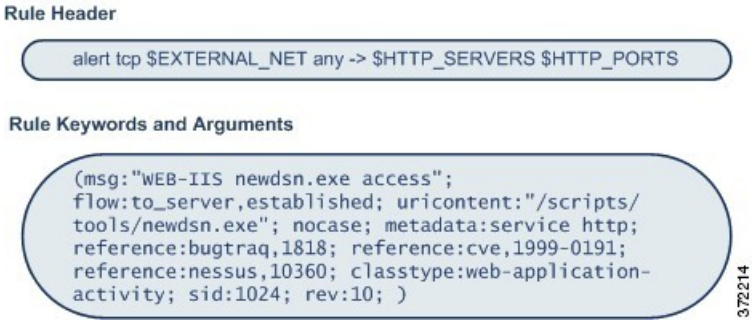
모든 표준 텍스트 규칙은 두 개의 논리적 섹션인 규칙 헤더 및 규칙 옵션을 포함합니다. 규칙 헤더에는 다음이 포함됩니다.

- 규칙의 상태 또는 유형
- 프로토콜
- 소스와 대상 IP 주소 및 넷마스크
- 소스에서 대상에 이르는 트래픽의 흐름을 보여 주는 방향 표시기
- 소스 및 대상 포트

규칙 옵션 섹션에는 다음이 포함됩니다.

- 이벤트 메시지
- 키워드와 매개 변수 및 인수
- 규칙을 트리거하기 위해 패킷 페이로드가 일치해야 하는 패턴
- 규칙 엔진이 패킷의 어느 부분을 검사해야 하는지에 관한 설명서

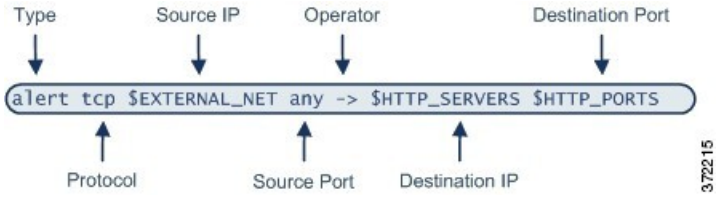
다음 다이어그램은 규칙의 일부를 설명합니다.



규칙의 옵션 섹션은 괄호로 둘러싸인 섹션이라는 점에 유의하십시오. 침입 규칙 편집기는 표준 텍스트 규칙을 구축할 수 있도록 사용하기 쉬운 인터페이스를 제공합니다.

침입 규칙 헤더

모든 표준 텍스트 규칙 및 공유 개체 규칙에는 매개변수와 인수를 포함하는 규칙 헤더가 있습니다. 다음은 규칙 헤더의 일부를 설명합니다.



다음 표는 위에 표시된 규칙 헤더의 각 부분을 나타냅니다.

표 *1:* 규칙 헤더 값

규칙 헤더 구성 요소	예제 값	값
작업	`Alert`	침입 이벤트가 트리거되면 이를 생성합니다.
프로토콜	`tcp`	TCP 트래픽만 테스트합니다.
소스 IP 주소	`$EXTERNAL_NET`	내부 네트워크에 없는 모든 호스트에서 나오는 트래픽을 테스트합니다.
소스 포트	`any`	시작 호스트의 모든 포트에서 나오는 트래픽을 테스트합니다.
연산자	`->`	(네트워크 웹 서버로 가는) 외부 트래픽을 테스트합니다.
대상 IP 주소	`$HTTP_SERVERS`	내부 네트워크에서 웹 서버로 지정된 모든 호스트에 인도되는 트래픽을 테스트합니다.
대상 포트	`$HTTP_PORTS`	내부 네트워크에서 HTTP 포트에 인도되는 트래픽을 테스트합니다.

참고 이전 예제는 대부분의 침입 규칙과 같이 기본 변수를 사용합니다.

관련 항목

변수 세트

침입 규칙 헤더 작업

각 규칙 헤더는 패킷이 규칙을 트리거할 때 시스템이 취할 작업을 지정하는 매개 변수를 포함합니다. 경고로 설정된 작업을 가진 규칙은 규칙을 트리거한 패킷에 대해 침입 이벤트를 생성하고 해당 패킷의 세부 사항을 로깅합니다. 통과로 설정된 작업을 가진 규칙은 규칙을 트리거한 패킷에 대해 이벤트를 생성하거나 해당 패킷의 세부 사항을 로깅하지 않습니다.

참고 인라인 배포에서 *Drop and Generate Events(*이벤트 삭제 및 생성*)*로 설정된 상태의 규칙은 규칙을 트리거한 패킷에 대해 침입 이벤트를 생성합니다. 또한 수동 배포에서 삭제 규칙을 적용할 경우, 규칙은 알림 규칙으로 작동합니다.

기본적으로, 통과 규칙은 경고 규칙을 대체합니다. 특정 상황에서 경고 규칙을 비활성화하는 대신 알림 규칙을 트리거하여 전달 규칙에 정의된 기준을 충족하는 패킷을 방지하는 전달 규칙을 만들 수 있습니다. 예를 들어, "익명" 사용자로 FTP 서버에 로그인을 시도하는 규칙이 활성 상태로 유지되기를 원할 수 있습니다. 하지만, 네트워크에 하나 이상의 적정 익명 FTP 서버가 있는 경우, 특정 서버의 경우, 익명 사용자는 원래 규칙을 트리거하지 않도록 지정하는 규칙을 작성하고 활성화할 수 있습니다.

침입 규칙 편집기 내 **Action(**작업**)** 목록에서 규칙 유형을 선택합니다.

침입 규칙 헤더 프로토콜

각 규칙 헤더에서, 규칙이 검사하는 트래픽의 프로토콜을 지정해야 합니다. 분석을 위해 다음 네트워크 프로토콜을 지정할 수 있습니다.

- ICMP(Internet Control Message Protocol)
- IP(인터넷 프로토콜)

참고 프로토콜이 `ip`로 설정되면 시스템은 침입 규칙 헤더의 포트 정의를 무시합니다.

- TCP(Transmission Control Protocol)
- UDP(User Datagram Protocol)

IP를 프로토콜 유형으로 사용하여 TCP, UDP, ICMP, IGMP 및 더 많은 수를 포함하는 IANA에 의해 할당된 모든 프로토콜을 검토합니다.

참고 지금은 IP 페이로드에서 다음 헤더(예를 들어, TCP 헤더)의 패턴과 일치하는 규칙을 작성할 수 없습니다. 대신, 일치하는 콘텐츠는 마지막 디코딩된 프로토콜과 함께 시작됩니다. 해결 방법으로, 규칙 옵션을 사용하여 TCP 헤더에서 패턴을 일치시킬 수 있습니다.

침입 규칙 편집기 내 **Protocol(**프로토콜**)** 목록에서 프로토콜 유형을 선택합니다.

관련 항목

침입 규칙 헤더 프로토콜, 5 페이지

침입 규칙 헤더 방향

규칙 헤더 내에서, 규칙이 패킷을 검사할 수 있도록 패킷이 이동해야 하는 방향을 지정할 수 있습니다. 다음 표는 이러한 옵션에 대해 설명합니다.

표 ***2:*** 규칙 헤더의 방향 옵션

사용 환경	테스트 대상
방향	특정 소스 IP 주소에서 특정 대상 IP 주소로 이동하는 트래픽에 한정
양방향	특정 소스 및 대상 IP 주소 간에 이동하는 모든 트래픽

침입 규칙 헤더 소스 및 대상 **IP** 주소

특정 IP 주소에서 시작하거나 특정 IP 주소를 대상으로 한 패킷 검사를 제한하면 시스템이 실행해야 하는 패킷 검사량이 줄어듭니다. 이는 또한 규칙을 보다 구체적으로 만들고 소스와 대상 IP 주소가 의심스러운 작업을 표시하지 않는 패킷에 대해 트리거된 규칙 가능성을 제거하여 잘못된 긍정을 줄입니다.

팁 시스템은 IP 주소만 인식하고 소스 또는 대상 IP 주소의 호스트 이름을 수락하지 않습니다.

규칙 편집기의 **Source IPs(**소스 **IP)** 및 **Destination IPs(**대상 **IP)** 필드에서 소스 및 대상 IP 주소를 지정합니다.

표준 텍스트 규칙을 작성할 때 필요에 따라 다양한 방법으로 IPv4 및 IPv6 주소를 지정할 수 있습니다. 단일 IP 주소, `any`(모두), IP 주소 목록, CIDR 표기법, 프리픽스 길이 또는 네트워크 변수를 지정할 수 있습니다. 또한, 특정 IP 주소 또는 IP 주소의 집합을 제외할지 여부를 나타낼 수 있습니다. IPv6 주소를 지정할 때, RFC 4291에 정의된 주소 지정 규칙을 사용할 수 있습니다.

침입 규칙 내 **IP** 주소 구문

다음 표에서는 소스 및 대상 IP 주소를 지정할 수 있는 다양한 방법을 요약합니다.

표 ***3:*** 소스**/**대상 ***IP*** 주소 구문

지정 대상	사용 환경	예
모든 IP 주소	`any`	`any`
특정 IP 주소	해당 IP 주소 동일한 규칙에서 IPv4 및 IPv6 소스와 대상 주소를 혼용하지 않는다는 점에 유의하십시오.	`192.168.1.1` `2001:db8::abcd`
IP 주소 목록	IP 주소를 둘러싸는 괄호(`[]`) 및 IP 주소를 구분하는 쉼표	`[192.168.1.1,192.168.1.15]` `[2001:db8::b3ff, 2001:db8::0202]`
IP 주소 블록	IPv4 CIDR 블록 또는 IPv6 주소 접두사 코멘트	`192.168.1.0/24` `2001:db8::/32`
특정 IP 주소 또는 주소 집합을 제외한 모든 주소	IP 주소 또는 무효화를 원하는 주소 앞에 있는 `!` 문자	`!192.168.1.15` `!2001:db8::0202:b3ff:fe1e`
하나 이상의 특정 IP 주소를 제외한 IP 주소의 블록 내 모든 주소	무효화된 주소 또는 블록 목록이 뒤따르는 주소 블록	`[10.0.0/8, !10.2.3.4, !10.1.0.0/16]` `[2001:db8::/32, !2001:db8::8329, !2001:db8::0202]`

지정 대상	사용 환경	예
네트워크 변수에 정의된 IP 주소	`$`로 시작하는 대문자로 된 변수 이름 전처리기 규칙은 침입 규칙에서 사용되는 네트워크 변수에 의해 정의된 호스트에 관계없이 이벤트를 트리거할 수 있습니다.	`$HOME_NET`
IP 주소 변수에 의해 정의된 주소를 제외한 모든 IP 주소	`!$`로 시작하는 대문자로 된 변수 이름	`!$HOME_NET`

다음 설명은 일부 IP 주소 입력 방법에 대한 추가 정보를 제공합니다.

모든 **IP** 주소

모든 IPv4 또는 IPv6 주소를 나타내는 규칙 소스 또는 대상 IP 주소로 `any`라는 단어를 지정할 수 있습니다.

예를 들어, 다음 규칙은 **`Source IPs`**(소스 **`IP`**) 및 **Destination IPs(**대상 **IP)** 필드의 인수 **any**를 사용하며 모든 IPv4 또는 IPv6 소스와 대상 주소로 패킷을 평가합니다.

```
alert tcp any any -> any any
```

또한 `::`을 지정하여 모든 IPv6 주소를 나타낼 수 있습니다.

여러 **IP** 주소

IP 주소를 쉼표로 구분하여 개별 IP 주소를 나열할 수 있습니다. 원하는 경우, 다음 예에 나온 것처럼 부정되지 않은 목록을 대괄호로 감쌀 수도 있습니다.

```
[192.168.1.100,192.168.1.103,192.168.1.105]
```

IPv4와 IPv6는 다음의 예시에서처럼 개별적으로 또는 조합하여 나열할 수 있습니다.

```
[192.168.1.100,2001:db8::1234,192.168.1.105]
```

이전 소프트웨어 릴리스에서는 괄호로 IP 주소 목록을 포함하는 것이 필요하지만 여기서는 필요하지 않다는 점에 유의하십시오. 선택 사항으로, 각 쉼표 앞이나 뒤에 스페이스로 목록을 입력할 수 있다는 점도 참고하십시오.

참고 무효화된 목록을 괄호로 묶어야 합니다.

또한 IPv4 CIDR(Classless Inter-Domain Routing: 클래스리스 도메인 간 라우팅) 표기법 또는 IPv6 프리픽스 길이를 사용하여 주소 블록을 지정할 수 있습니다. 예를 들면 다음과 같습니다.

- 192.168.1.0/24는 서브넷 마스크 255.255.255.0, 즉, 192.168.1.255를 통한 192.168.1.0으로 192.168.1.0 네트워크에서 IPv4 주소를 지정합니다.

- 2001:db8::/32는 2001:db8:: 네트워크에서 32비트의 접두사 길이로 IPv6 주소를 지정하는데, 이는, 2001:db8:ffff:ffff:ffff:ffff:ffff:ffff를 통한 2001:db8::입니다.

팁 IP 주소 블록을 지정해야 하지만 CIDR 또는 접두사 길이 표기를 사용하여 이를 표시할 수 없는 경우, CIDR 블록 및 IP 주소 내 접두사 길이를 사용할 수 있습니다.

IP 주소 부정

느낌표(!)를 사용하여 지정된 IP 주소를 무효화할 수 있습니다. 즉, 지정된 IP 주소 또는 주소를 제외한 모든 IP 주소와 일치할 수 있습니다. 예를 들어, `! 192.168.1.1`은 192.168.1.1 이외의 모든 IP 주소를 지정하고, `! 2001: db8: ca2e:: fa4c`는 2001: db8: ca2e:: fa4c. 이외의 모든 IP 주소를 지정합니다.

IP 주소 목록을 무효화하려면 괄호로 묶은 IP 주소의 목록 앞에 !를 표시하십시오. 예를 들어, `![192.168.1.1,192.168.1.5]`는 192.168.1.1 또는 192.168.1.5. 이외의 모든 IP 주소를 정의합니다.

참고 IP 주소 목록을 무효화하기 위해서는 괄호를 사용해야 합니다.

IP 주소 목록과 함께 무효화 문자를 사용할 때는 주의하십시오. 예를 들어, 192.168.1.1 또는 192.168.1.5가 아닌 모든 주소를 일치시키기 위해 `[!192.168.1.1,!192.168.1.5]`를 사용하는 경우,시스템은 이 구문을 "192.168.1.1이 아닌 모든 주소, 또는 192.168.1.5가 아닌 모든 주소"로 해석합니다.

192.168.1.5는 192.168.1.1이 아니고 192.168.1.1은 192.168.1.5가 아니므로, 두 IP 주소 모두 IP 주소 `[!192.168.1.1,!192.168.1.5]` 값에 일치하며, 이는 본질적으로 "`any(모두)`"를 사용하는 것과 동일합니다.

이보다는 `![192.168.1.1,192.168.1.5]`를 사용하십시오. 시스템은 이를 "192.168.1.1이 아니며 192.168.1.5도 아닌 주소"로 해석하며, 이는 괄호 사이에 나열된 IP 주소를 제외한 모든 IP 주소에 일치하는 것입니다.

논리적으로 `any(모두)`와 무효화를 함께 사용할 수 없다는 점에 유의하십시오. 함께 사용하는 경우 무효화되면 어떤 주소도 나타내지 못하게 됩니다.

관련 항목

변수 세트

침입 규칙 헤더 소스 및 대상 포트

규칙 편집기의 **Source Port(**소스 포트**)** 및 **Destination Port(**대상 포트**)** 필드에서 소스 포트와 대상 포트를 지정합니다.

침입 규칙 내 포트 구문

Firepower System은 규칙 헤더에 사용되는 포트 번호를 정의하기 위해 특정 유형의 구문을 사용합니다.

참고 프로토콜이 ip로 설정되면 시스템은 침입 규칙 헤더의 포트 정의를 무시합니다.

다음의 예시에서와 같이 쉼표로 포트를 구분하여 나열할 수 있습니다.

```
80, 8080, 8138, 8600-9000, !8650-8675
```

또는, 다음의 예시는 괄호로 포트 목록을 묶는 방법을 보여주는데, 이는 이전의 소프트웨어 버전에서는 필요했지만 더 이상 필요하지 않습니다.

```
[80, 8080, 8138, 8600-9000, !8650-8675]
```

다음의 예시에서처럼 무효화된 포트 목록을 반드시 괄호로 묶어야 한다는 점에 유의하십시오.

```
![20, 22, 23]
```

다음 표는 사용 가능한 구문을 요약한 것입니다.

표 *4:* 소스/대상 포트 구문

지정 대상	사용 환경	예
모든 포트	any	any
특정 포트	포트 번호	80
포트 범위	범위 내 첫 번째 및 마지막 포트 번호 사이의 대시	80-443
특정 포트보다 작거나 같은 모든 포트	포트 번호 앞의 대시	-21
특정 포트보다 같거나 큰 모든 포트	포트 번호 다음의 대시	80-
특정 포트 또는 특정 범위의 포트를 제외한 모든 포트	무효화를 원하는 포트, 포트 목록 또는 범위의 포트 앞의 !문자 논리적으로는 any(모두)를 제외한 모든 포트 지정에 부정을 사용할 수 있다는 점에 유의하십시오. any가 부정되는 경우에는 *no port(*포트 없음*)*가 표시됩니다.	!20
포트 변수에 의해 정의된 모든 포트	$로 시작하는 대문자로 된 변수 이름	$HTTP_PORTS
포트 변수에 의해 정의된 포트를 제외한 모든 포트	!$로 시작하는 대문자로 된 변수 이름	!$HTTP_PORTS

침입 이벤트 세부 정보

표준 텍스트 규칙을 구성할 때 규칙이 익스플로잇 시도에서 탐지하는 취약성을 설명하는 컨텍스트 정보를 포함할 수 있습니다. 또한 취약성 데이터베이스에 외부 참조를 포함하고 사용자 조직에서 이벤트가 가지고 있는 우선 순위를 정의할 수 있습니다. 분석가가 이벤트를 볼 때, 그들은 우선 순위, 공격, 즉시 사용 가능한 알려진 위협 완화에 대한 정보를 가지게 됩니다.

Message

규칙이 트리거될 때 메시지로 표시되는 의미 있는 텍스트를 지정할 수 있습니다. 메시지는 규칙이 공격 시도를 탐지하게 되는 취약성의 속성에 대한 즉각적인 통찰력을 제공해 줍니다. 중괄호({})를 제외한 인쇄 가능한 표준 ASCII 문자를 모두 사용할 수 있습니다. 시스템은 메시지를 완전히 묶고 있는 따옴표를 떼어버립니다.

팁 규칙 메시지를 지정해야 합니다. 또한, 공백, 하나 이상의 따옴표, 하나 이상의 아포스트로피 또는 공백, 따옴표, 아포스트로피로의 조합만으로는 메시지를 구성할 수 없습니다.

침입 규칙 편집기에서 이벤트 메시지를 정의하려면 **Message(**메시지**)** 필드에 이벤트 메시지를 입력합니다.

분류

각 규칙에, 이벤트의 패킷 표시에 나타나는 공격 분류를 지정할 수 있습니다. 다음 표에서는 각 분류의 이름과 번호를 나열합니다.

표 ***5:*** 규칙 분류

번호	분류 이름	설명
1	not-suspicious	의심스럽지 않은 트래픽
2	unknown	알 수 없는 트래픽
3	bad-unknown	잠재적인 악성 트래픽
4	attempted-recon	정보 유출 시도
5	successful-recon-limited	정보 유출
6	successful-recon-largescale	대규모 정보 유출
7	attempted-dos	서비스 거부 시도
8	successful-dos	서비스 거부
9	attempted-user	사용자 권한 획득 시도
10	unsuccessful-user	사용자 권한 획득 실패

번호	분류 이름	설명
11	successful-user	사용자 권한 획득 성공
12	attempted-admin	관리자 권한 획득 시도
13	successful-admin	관리자 권한 획득 성공
14	rpc-portmap-decode	RPC 쿼리 디코드
15	shellcode-detect	실행 가능한 코드가 탐지됨
16	string-detect	의심스러운 문자열이 탐지됨
17	suspicious-filename-detect	의심스러운 파일 이름이 탐지됨
18	suspicious-login	의심스러운 사용자 이름을 사용한 로그인 시도가 탐지됨
19	system-call-detect	시스템 호출이 탐지됨
20	tcp-connection	TCP 연결이 탐지됨
21	trojan-activity	네트워크 트로이 목마가 탐지됨
22	unusual-client-port-connection	클라이언트가 비정상적인 포트를 사용하고 있음
23	network-scan	네트워크 스캔이 탐지됨
24	denial-of-service	DoS(Denial-of-Service) 공격이 탐지됨
25	non-standard-protocol	비표준 프로토콜 또는 이벤트가 탐지됨
26	protocol-command-decode	일반적인 프로토콜 명령 디코드
27	web-application-activity	잠재적으로 취약한 웹 애플리케이션에 액세스
28	web-application-attack	웹 애플리케이션 공격
29	misc-activity	기타 활동
30	misc-attack	기타 공격
31	icmp-event	일반 ICMP 이벤트
32	inappropriate-content	부적절한 콘텐츠가 발견됨
33	policy-violation	잠재적인 기업 개인 정보 보호 위반
34	default-login-attempt	기본 사용자 이름 및 비밀번호로 로그인 시도
35	sdf	중요한 데이터

번호	분류 이름	설명
36	malware-cnc	알려진 악성코드 명령 및 제어 트래픽
37	client-side-exploit	알려진 클라이언트 측 공격 시도
38	file-format	알려진 악성 파일 또는 파일 기반 익스플로잇

맞춤형 분류

사용자가 정의하는 규칙에서 생성된 이벤트의 패킷 표시 설명을 위한 더 많은 맞춤형 콘텐츠를 원할 경우, 맞춤형 분류를 생성할 수 있습니다.

인수	설명
분류 이름	분류의 이름입니다. 40개 이상의 문자를 사용하는 경우, 페이지를 읽기가 어렵습니다. 다음 문자는 지원되지 않습니다: `<> () \ '"&$;` 및 공백 문자.
분류 설명	분류의 설명입니다. 영숫자와 공백을 사용할 수 있습니다. `<> () \ '"&$;` 문자는 지원되지 않습니다.
우선순위	높음, 중간 또는 낮음.

맞춤형 우선 순위

기본적으로, 규칙의 우선 순위는 규칙에 대한 이벤트 분류에서 파생됩니다. 하지만 규칙에 `priority` 키워드를 추가하고 high(높음), medium(중간) 또는 low(낮음) 우선 순위를 선택하여 규칙의 분류 우선 순위를 재정의할 수 있습니다. 예를 들어 웹 애플리케이션 공격을 탐지하는 규칙에 high(높음) 우선 순위를 할당하려면 `priority` 키워드를 규칙에 추가하고 우선 순위로 **high(**높음**)**를 선택합니다.

맞춤형 참조

`reference` 키워드를 사용하여 외부 웹사이트에 참조를 추가하고 이벤트에 대한 자세한 내용을 추가할 수 있습니다. 참조를 추가하면 패킷이 규칙을 트리거한 이유에 대한 확인을 돕기 위해 분석가에게 즉시 사용 가능한 리소스를 제공합니다. 다음 표는 알려진 악성 공격에 관한 데이터를 제공하는 몇 가지 외부 시스템 나열합니다.

표 ***6:*** 외부 공격 식별 시스템

시스템 **ID**	설명	예시 **ID**
`bugtraq`	Bugtraq 페이지	`8550`
`cve`	일반 취약점 및 노출 ID	`2020-9607`
`mcafee`	McAfee 페이지	`98574`

시스템 **ID**	설명	예시 **ID**
`url`	웹사이트 참조	`www.example.com?exploit=14`
`msb`	Microsoft 보안 공지	`MS11-082`
`nessus`	Nessus 페이지	`10039`
`secure-url`	보안 웹사이트 참조(https://...)	`intranet/exploits/exploit=14` 모든 보안 웹사이트로 `secure-url`을 함께 사용할 수 있다는 점을 참고하십시오.

다음과 같이 참조 값을 입력하여 참조를 지정합니다.

```
id_system,id
```

여기서 id_system은 프리픽스로 사용되고 있는 시스템이고 id는 CVE ID 번호, Arachnids ID 또는 URL(`http://` 없음)입니다.

예를 들어 CVE-2020-9607에 설명되어 있는 Adobe Acrobat 및 Reader 문제를 지정하려면 다음 값을 입력합니다.

```
cve,2020-9607
```

규칙에 참조 사항을 추가할 때 다음에 유의하십시오.

- 쉼표 뒤에 스페이스를 사용하지 마십시오.
- 시스템 ID에 대문자를 사용하지 마십시오.

관련 항목

맞춤형 분류 추가, 13 페이지
이벤트 우선 순위 정의, 14 페이지
이벤트 참조 정의, 14 페이지

맞춤형 분류 추가

다중 도메인 구축에서 시스템은 현재 도메인에서 생성된 맞춤형 분류를 표시하며, 사용자는 이러한 분류의 우선 순위를 설정할 수 있습니다. 상위 도메인에서 생성된 맞춤형 분류도 표시되지만 사용자는 이러한 분류의 우선 순위를 설정할 수 없습니다. 하위 도메인에서 생성된 맞춤형 분류를 보고 수정하려면 해당 도메인으로 전환하십시오.

프로시저

단계 **1** 규칙을 만들거나 수정할 때 **Classification(분류)** 드롭다운 목록에서 **Edit Classifications(분류 수정)**을 선택합니다.

View Classifications(분류 보기**)**가 대신 표시되는 경우, 구성이 상위 도메인에 속하거나 구성을 수정할 권한이 없는 것입니다.

단계 **2** 침입 이벤트 세부 정보, 10 페이지에 설명된 대로 **Classification Name(**분류 이름**)**과 **Classification Description(**분류 설명**)**을 입력합니다.

단계 **3** **Priority(**우선 순위**)** 드롭다운 목록에서 분류의 우선 순위를 선택합니다.

단계 **4** **Add(**추가**)**를 클릭합니다.

단계 **5** **Done(**완료**)**을 클릭합니다.

이벤트 우선 순위 정의

프로시저

단계 **1** 규칙을 만들거나 수정할 때 **Detection Options(**탐지 옵션**)** 드롭다운 목록에서 `priority(우선 순위)`를 선택합니다.

단계 **2** **Add Option(**옵션 추가**)**을 클릭합니다.

단계 **3** **priority(**우선 순위**)** 드롭다운 목록에서 값을 선택합니다.

단계 **4** **Save(**저장**)**를 클릭합니다.

이벤트 참조 정의

프로시저

단계 **1** 규칙을 만들거나 수정할 때 **Detection Options(**탐지 옵션**)** 드롭다운 목록에서 `reference(참조)`를 선택합니다.

단계 **2** **Add Option(**옵션 추가**)**을 클릭합니다.

단계 **3** 침입 이벤트 세부 정보, 10 페이지에 설명된 대로 **reference(**참조**)** 필드에 값을 입력합니다.

단계 **4** **Save(**저장**)**를 클릭합니다.

규칙 검색

시스템은 수천 개의 표준 텍스트 규칙을 제공하며, Talos 인텔리전스 그룹는 새로운 취약성과 익스플로잇이 발견됨에 따라 계속 규칙을 추가합니다. 특정 규칙을 손쉽게 검색하여 활성화, 비활성화 또는 수정할 수 있습니다.

프로시저

단계 **1** 다음 방법 중 하나를 사용하여 침입 규칙에 액세스합니다.

- **Policies(**정책**)** > **Access Control(**액세스 제어**)** > **Intrusion(**침입**)**을(를) 선택합니다.

  편집하려는 정책 옆의 **Snort 2 Version(Snort 2** 버전**)**을 클릭하고 **Rules(**규칙**)**를 클릭합니다.

- **Objects(**개체**)** > **Intrusion Rules(**침입 규칙**)**을(를) 선택합니다.

단계 **2** 툴바에서 **Search(**검색**)**를 클릭합니다.

단계 **3** 검색 기준을 추가합니다.

단계 **4** **Search(**검색**)**를 클릭합니다.

침입 규칙에 대한 검색 기준

다음 표에서는 사용 가능한 검색 옵션에 대해 설명합니다.

표 *7:* 규칙 검색 기준

옵션	설명
Signature ID(서명 ID)	Snort ID(SID)를 기반으로 단일 규칙을 검색하려면 SID 번호를 입력합니다. 여러 규칙을 검색하려면 쉼표로 구분된 SID 번호 목록을 입력합니다. 이 필드의 문자 제한은 80자입니다.
Generator ID(생성자 ID)	표준 텍스트 규칙을 검색하려면 **1**을 선택합니다. 공유 개체 규칙을 검색하려면 **3**을 선택합니다.
Message(메시지)	특정 메시지가 있는 규칙을 검색하려면 규칙 메시지의 한 단어를 **Message(**메시지**)** 필드에 입력합니다. 예를 들어 DNS 익스플로잇을 검색하려면 `DNS`를, 버퍼 오버플로 익스플로잇을 검색하려면 `overflow`를 입력합니다.
Protocol(프로토콜)	특정 프로토콜의 트래픽을 평가하는 규칙을 검색하려면 프로토콜을 선택합니다. 프로토콜을 선택하지 않으면 검색 결과에 모든 프로토콜에 대한 규칙이 포함됩니다.
Source Port(소스 포트)	지정된 포트에서 오는 패킷을 검사하는 규칙을 검색하려면 소스 포트 번호 또는 포트 관련 변수를 입력합니다.
Destination Port(대상 포트)	특정 포트로 향하는 패킷을 검사하는 규칙을 검색하려면 대상 포트 번호 또는 포트 관련 변수를 입력합니다.
Source IP(소스 IP)	지정된 IP 주소에서 오는 패킷을 검사하는 규칙을 검색하려면 소스 IP 주소 또는 IP 주소 관련 변수를 입력합니다.
Destination IP(목적지 IP)	지정된 IP 주소로 향하는 패킷을 검사하는 규칙을 검색하려면 대상 IP 주소 또는 IP 주소 관련 변수를 입력합니다.

옵션	설명
Keyword(키워드)	특정 키워드를 검색하려면 키워드 검색 옵션을 사용할 수 있습니다. 키워드를 선택하고 검색할 키워드 값을 입력합니다. 키워드 값 앞에 느낌표(!)를 입력하면 지정된 값 이외의 모든 값을 매칭할 수도 있습니다.
카테고리	특정 범주의 규칙을 검색하려면 **Category(**범주**)** 목록에서 범주를 선택합니다.
Classification(분류)	특정 분류가 있는 규칙을 검색하려면 **Classification(**분류**)** 목록에서 분류를 선택합니다.
Rule State(규칙 상태)	특정 정책 및 규칙 상태 내의 규칙을 검색하려면 첫 번째 **Rule State(**규칙 상태**)** 목록에서 정책을 선택하고 두 번째 목록에서 상태를 선택해 **Generate Events(**이벤트 생성**)**, **Drop and Generate Events(**이벤트 삭제 및 생성**)** 또는 **Disabled(**비활성화**)**로 설정된 규칙을 검색합니다.

침입 규칙 편집기 페이지에서 규칙 필터링

침입 규칙 편집기 페이지에서 규칙을 필터링해 규칙의 하위 집합을 표시할 수 있습니다. 예를 들어, 규칙을 수정하거나 상태를 변경하기를 원하지만 수천 개의 규칙 중에서 이를 찾는 데 어려움이 있는 경우, 이는 유용할 수 있습니다.

필터를 입력하면, 페이지는 적어도 하나의 일치하는 규칙을 포함하는 모든 폴더를 표시하거나, 규칙이 하나도 일치하지 않을 때는 메시지를 표시합니다.

필터링 가이드라인

필터는 특수 키워드 및 해당 인수, 문자열 및 텍스트 문자열, 그리고 여러 필터 상태를 분리하는 스페이스를 포함할 수 있습니다. 필터에는 정규 표현식, 와일드카드 문자 또는 부정 문자(!), 보다 큼 기호(>), 보다 작음 기호(<)와 같은 특별 연산자를 포함할 수 없습니다.

모든 키워드, 키워드 인수 및 문자열은 대소문자를 구분하지 않습니다. `gid` 및 `sid` 키워드를 제외한, 모든 인수 및 문자열은 부분 문자열로 처리됩니다. `gid` 및 `sid`의 인수는 정확히 일치하는 것만 반환합니다.

필터링되지 않은 원래 페이지에서 폴더를 확장할 수 있으며, 후속 필터가 해당 폴더에서 일치 항목을 반환하면 폴더는 확장된 상태로 유지됩니다. 이는 찾고자 하는 규칙이 많은 규칙을 포함하는 폴더에 있을 때 유용할 수 있습니다.

후속 필터로 필터를 제한할 수 없습니다. 입력한 모든 필터는 전체 규칙 데이터베이스를 검색하고 일치하는 규칙을 모두 반환합니다. 페이지가 계속 이전 검색 결과를 표시하고 있는데 필터를 입력하는 경우, 페이지는 이를 지우고 새 필터의 결과로 돌아갑니다.

필터링된 목록이나 필터링되지 않은 목록의 규칙과 같은 기능을 사용할 수 있습니다. 예를 들어 침입 규칙 편집기 페이지에서 필터링된 목록이나 필터링되지 않은 목록의 규칙을 수정할 수 있습니다. 또한 해당 페이지에 대해 상황에 맞는 메뉴에서 모든 옵션을 사용할 수 있습니다.

팁 모든 하위 그룹의 규칙을 포함한 총 규칙 수가 클 경우 규칙이 여러 카테고리에 나타날 수 있기 때문에 고유한 총 규칙 수가 훨씬 적더라도 필터링에 상당한 시간이 소요될 수 있습니다.

키워드 필터링

각 규칙 필터의 형식에는 하나 이상의 키워드를 포함할 수 있습니다.

```
keyword:argument
```

여기서 keyword는 다음 표에 있는 키워드 중 하나이며, argument는 키워드와 관련된 하나 이상의 특정 필드에서 검색할, 대소문자를 구분하지 않는 단일 영숫자 문자열입니다.

`gid` 및 `sid`를 제외한 모든 키워드에 대한 인수는 부분 문자열로 처리됩니다. 예를 들어, 인수 `123`은 `"12345"`, `"41235"`, `"45123"` 등을 반환합니다. `gid` 및 `sid`의 인수는 정확하게 일치하는 경우에만 반환됩니다. 예를 들어, `sid:3080`은 SID 3080만 반환합니다.

팁 하나 이상의 문자열로 필터링하여 부분 SID를 검색할 수 있습니다.

다음 표는 규칙을 필터링하는 데 사용할 수 있는 인수 및 특정 필터링 키워드를 나타냅니다.

표 ***8:*** 규칙 필터링 키워드

키워드	설명	예
`arachnids`	규칙 참조에서 Arachnids ID의 전체 또는 일부에 따라 하나 이상의 규칙을 반환합니다.	`arachnids:181`
`bugtraq`	규칙 참조에서 Bugtraq ID의 전체 또는 일부에 따라 하나 이상의 규칙을 반환합니다.	`bugtraq:2120`
`cve`	규칙 참조에서 CVE 번호의 전체 또는 일부에 따라 하나 이상의 규칙을 반환합니다.	`cve:2003-0109`
`gid`	인수 `1`은 표준 텍스트 규칙을 반환합니다. 인수 `3`은 공유 개체 규칙을 반환합니다.	`gid:3`
`mcafee`	규칙 참조에서 McAfee ID의 전체 또는 일부에 따라 하나 이상의 규칙을 반환합니다.	`mcafee:10566`
`msg`	이벤트 메시지로도 알려진 규칙 Message(메시지) 필드의 전체 또는 일부에 따라 하나 이상의 규칙을 반환합니다.	`msg:chat`
`nessus`	규칙 참조에서 Nessus ID의 전체 또는 일부에 따라 하나 이상의 규칙을 반환합니다.	`nessus:10737`

키워드	설명	예
`ref`	규칙 참조 또는 규칙 Message(메시지) 필드에서 단일 영숫자 문자열의 전체 또는 일부에 따라 하나 이상의 규칙을 반환합니다.	`ref:MS03-039`
`sid`	정확한 Snort ID가 있는 규칙을 반환합니다.	`sid:235`
`url`	규칙 참조에서 URL의 전체 또는 일부에 따라 하나 이상의 규칙을 반환합니다.	`url:faqs.org`

관련 항목

이벤트 참조 정의, 14 페이지
침입 이벤트 세부 정보, 10 페이지

문자열 필터링

각 규칙 필터는 하나 이상의 영숫자 문자열을 포함할 수 있습니다. 문자열은 규칙 **Message(**메시지**)** 필드, Snort ID(SID) 및 생성자 ID(GID)를 검색합니다. 예를 들어, 문자열 `123`은 규칙 메시지에서 문자열 `"Lotus123"`, `"123mania"` 등을 반환하며, 또한 SID 6123, SID 12375 등을 반환합니다.

모든 문자열은 대소문자를 구분하지 않으며 부분 문자열로 처리됩니다. 예를 들어, 문자열 `ADMIN`, `admin` 또는 `Admin`은 모두 `"admin"`, `"CFADMIN"`, `"Administrator"` 등을 반환합니다.

정확히 일치하는 항목을 반환하기 위해 인용구에서 문자열을 묶을 수 있습니다. 예를 들어, 인용구 내 문자열 `"overflow attempt"`는 정확한 문자열만 반환하지만, 인용구가 없는 두 개의 문자열 `overflow` 및 `attempt`로 구성된 필터는 `"overflow attempt"`, `"overflow multipacket attempt"`, `"overflow with evasion attempt"` 등을 반환합니다.

관련 항목

침입 이벤트 세부 정보, 10 페이지

키워드 및 문자열 조합 필터링

키워드, 문자열 또는 둘 다로 이루어진 스페이스로 구분된 문자열의 조합을 입력하여 필터링 결과를 좁힐 수 있습니다. 결과는 필터링 조건과 일치하는 모든 규칙을 포함합니다.

순서에 상관없이 여러 필터 상태를 입력할 수 있습니다. 예를 들어, 다음 필터 각각은 동일한 규칙을 반환합니다.

- `url:at login attempt cve:200`
- `login attempt cve:200 url:at`
- `login cve:200 attempt url:at`

규칙 필터링

Intrusion Rules(침입 규칙) 페이지에서 규칙을 하위 집합으로 필터링하면 특정 규칙을 더 쉽게 찾을 수 있습니다. 그런 다음 컨텍스트 메뉴에서 사용 가능한 기능 선택을 포함하여 원하는 페이지 기능을 사용할 수 있습니다.

규칙 필터링은 수정할 규칙을 찾을 때 특히 유용할 수 있습니다.

프로시저

단계 **1** 다음 방법 중 하나를 사용하여 침입 규칙에 액세스합니다.

- **Policies(**정책**)** > **Access Control(**액세스 제어**)** > **Intrusion(**침입**)**을(를) 선택합니다.

  편집하려는 정책 옆의 **Snort 2 Version(Snort 2** 버전**)**을 클릭하고 **Rules(**규칙**)**를 클릭합니다.

- **Objects(**개체**)** > **Intrusion Rules(**침입 규칙**)**을(를) 선택합니다.

단계 **2** 필터링 전에 다음 옵션을 이용할 수 있습니다.

- 확장할 규칙 그룹을 확장합니다. 일부 규칙 그룹에도 확장할 수 있는 하위 그룹이 있습니다.

  필터링되지 않은 원래 페이지에서 그룹을 확장하면 해당 그룹에 규칙이 있을 것으로 예상되는 경우 유용할 수 있습니다. 후속 필터가 해당 폴더와 일치할 때, 그리고 필터 **Clear(**지우기**)** (✕)을 클릭하여 원래의 필터링되지 않은 페이지로 돌아온 경우 해당 그룹은 확장된 채로 유지됩니다.

- **Group Rules By(**규칙 분류 기준**)** 드롭다운 목록에서 다른 그룹화 방법을 선택합니다.

단계 **3** **Group Rules By(**규칙 그룹화 기준**)** 목록 아래 **Filter(**필터**)** (🔍) 옆에 있는 입력란에 필터 제약 조건을 입력합니다.

단계 **4** Enter를 누릅니다.

참고 **Clear(**지우기**)** (✕)을 클릭하여 필터링된 현재 목록을 삭제합니다.

침입 규칙의 키워드 및 인수

규칙 언어를 사용하여 키워드를 결합함으로써 규칙 작업을 지정할 수 있습니다. 키워드 및 관련 값(일명 인수)은 규칙 엔진이 테스트하는 패킷 및 패킷 관련 값을 시스템이 평가하는 방법을 지시합니다. Firepower System은 현재 콘텐츠 매칭, 프로토콜별 패턴 매칭, 상태별 매칭 등의 검사 기능을 수행할 수 있는 키워드를 지원합니다. 키워드당 최대 100개의 인수를 정의할 수 있고, 매우 특정한 규칙을 만들기 위해 호환성 키워드를 얼마든지 통합할 수 있습니다. 이는 잘못된 긍정 및 잘못된 부정의 가능성을 줄이고 사용자가 수신하는 침입 정보에 집중하는 데 도움을 줍니다.

또한 패시브 구축에서 적응형 프로파일 업데이트을 사용하여 규칙 메타데이터와 호스트 정보에 따라 특정 패킷에 대한 활성 규칙 처리를 동적으로 조정할 수 있습니다.

이 섹션에 설명된 키워드는 규칙 편집기에서 Detection Options(탐지 옵션) 아래 나열됩니다.

관련 항목

적응형 프로파일 정보

content 및 protected_content 키워드

패킷에서 탐지할 내용을 지정하려면 content 키워드 또는 protected_content 키워드를 사용합니다.

사용자는 거의 항상 content 또는 protected_content 키워드를 따라야 합니다. 이 키워드는 콘텐츠를 어디에서 검색해야 할지, 검색에 있어 대소문자 구분이 필요한지 여부, 그리고 다른 옵션을 나타내는 수식어 옆에 있습니다.

모든 콘텐츠 일치는 이벤트를 트리거하는 규칙에 대해 참이어야 한다는 점, 즉, 각 콘텐츠 일치는 다른 항목과 AND 관계를 가지고 있다는 점에 유의하십시오.

또한, 인라인 배포에서 악성 콘텐츠와 일치하고 이를 동일한 길이의 자체 문자열로 대체하는 규칙을 설정할 수 있다는 점에 유의하십시오.

content

content 키워드를 사용하면, 규칙 엔진이 해당 문자열에 대한 패킷 페이로드 또는 스트림을 검색합니다. 예를 들어, content 키워드 중 하나의 값으로 /bin/sh를 입력하면, 규칙 엔진은 문자열 /bin/sh에 대한 패킷 페이로드를 검색합니다.

ASCII 문자열, 16진수 콘텐츠(이진 바이트 코드) 또는 이 둘의 조합 중 하나를 사용하여 콘텐츠에 일치시킵니다. 키워드 값에서 파이프 문자(|)로 16진수 콘텐츠를 묶습니다. 예를 들어, |90C8 C0FF FFFF|/bin/sh와 같은 형태를 사용하여 16진수 콘텐츠 및 ASCII 콘텐츠를 섞을 수 있습니다.

단일 규칙에서 여러 콘텐츠 일치를 지정할 수 있습니다. 이를 위해, content 키워드의 추가 인스턴스를 사용합니다. 각 콘텐츠 일치의 경우, 규칙이 트리거되려면 패킷 페이로드 또는 스트림에서 콘텐츠 일치를 찾아야 한다는 것을 나타낼 수 있습니다.

주의 content 키워드가 하나만 포함된 규칙을 생성하고 키워드에 대해 **Not** 옵션을 선택한 경우 침입 정책을 무효화할 수 있습니다.

protected_content

protected_content 키워드를 사용하면 규칙 인수를 구성하기 전에 검색 내용 문자열을 인코딩할 수 있습니다. 원래 규칙 작성자는 키워드를 구성하기 전에 문자열을 인코딩하기 위해 해시 함수(SHA-512, SHA-256 또는 MD5)를 사용합니다.

content 키워드 대신 protected_content 키워드를 사용하면, 규칙 엔진이 해당 문자열 및 예상한 대로 대부분 키워드 옵션에 대해 패킷 페이로드 또는 스트림을 검색하는 방법은 변경되지 않습니다. 다음 표에는 예외가 요약되어 있습니다. 여기서 protected_content 키워드 옵션은 content 키워드 옵션과 다릅니다.

표 ***9: protected_content*** 옵션 예외

옵션	설명
해시 유형	protected_content 규칙 키워드에 대한 새로운 옵션.
대소문자 구분 안 함	지원되지 않음
내부	지원되지 않음
수준	지원되지 않음
길이	protected_content 규칙 키워드에 대한 새로운 옵션.
빠른 패턴 매처 사용	지원되지 않음
빠른 패턴 매처 한정	지원되지 않음
빠른 패턴 매처 오프셋 및 길이	지원되지 않음

Cisco는 protected_content 키워드를 포함하는 규칙에 최소 하나의 content 키워드를 포함할 것을 권장합니다. 이렇게 하면 규칙 엔진이 빠른 패턴 매처를 사용하여 처리 속도를 높이고 성능을 향상시킬 수 있습니다. 규칙에서 protected_content 키워드 앞에 콘텐츠 keyword를 배치합니다. 규칙에 최소 하나의 content 키워드가 포함될 때, content 키워드 Use Fast Pattern Matcher(빠른 패턴 매처 사용) 인수를 활성화했는지 여부에 상관없이 규칙 엔진이 빠른 패턴 매처를 사용한다는 점에 유의하십시오.

주의 protected_content 키워드가 하나만 포함된 규칙을 생성하고 키워드에 대해 **Not** 옵션을 선택한 경우 침입 정책을 무효화할 수 있습니다.

관련 항목

기본 content 또는 protected_content 키워드 인수, 21 페이지
replace 키워드, 32 페이지

기본 **content** 또는 **protected_content** 키워드 인수

content 또는 protected_content 키워드를 수정하는 파라미터로 콘텐츠 검색 위치 및 대소문자 구분 여부를 제한할 수 있습니다. content 또는 protected_content 키워드를 수정하여 검색할 내용을 지정할 수 있는 옵션을 구성합니다.

대소문자 구분 안 함

참고 protected_content 키워드를 구성할 때 이 옵션은 지원되지 않습니다.

ASCII 문자열의 콘텐츠 일치를 검색할 때 대소문자 구분을 무시하기 위해 규칙 엔진을 검사할 수 있습니다. 검색에서 대소문자를 구분하도록 하려면 콘텐츠 검색을 지정할 때 **Case Insensitive(**대소문자 구분 안 함**)**를 선택합니다.

해시 유형

참고 이 옵션은 `protected_content` 키워드에 한정하여 구성 가능합니다.

Hash Type(해시 유형**)** 드롭다운을 사용하여 검색 문자열을 인코딩하는 데 사용한 해시 함수를 확인합니다. 시스템은 `protected_content` 검색 문자열에 대해 SHA-512, SHA-256 및 MD5 해싱을 지원합니다. 선택한 해시 유형이 해시된 콘텐츠의 길이와 일치하지 않을 경우, 시스템은 규칙을 저장하지 않습니다.

시스템은 Cisco가 설정한 기본값을 자동으로 선택합니다. **Default(**기본값**)**를 선택하면 규칙에 특정 해시 함수가 기록되지 않으며 시스템은 SHA-512를 해시 함수로 가정합니다.

Raw Data

Raw Data(원시 데이터**)** 옵션은 (네트워크 분석 정책에서 디코딩된) 표준화된 페이로드 데이터를 분석하기 전에 원래 패킷 페이로드를 규칙 엔진이 분석하도록 지시하며, 인수 값을 사용하지 않습니다. 텔넷 트래픽을 분석할 때 이 키워드를 사용하여 표준화 전 페이로드에서 텔넷 협상 옵션을 선택할 수 있습니다.

Raw Data 옵션은 동일한 `content` 또는 `protected_content` 키워드에서 HTTP content 옵션과 함께 사용할 수 없습니다.

팁 HTTP 검사 전처리기 **Client Flow Depth(**클라이언트 흐름 수준**)** 및 **Server Flow Depth(**서버 흐름 수준**)** 옵션을 구성하여 원시 데이터가 HTTP 트래픽에서 검사되는지 여부 및 검사할 원시 데이터의 양을 결정할 수 있습니다.

Not

지정된 내용과 일치하지 않는 콘텐츠를 검색하려면 **Not(**아님**)** 옵션을 선택합니다. **Not** 옵션을 선택한 상태로 `content` 또는 `protected_content` 키워드가 포함된 규칙을 생성하는 경우, **Not** 옵션을 선택하지 않은 상태로 하나 이상의 `content` 또는 `protected_content` 키워드를 규칙에 포함해야 합니다.

주의 **Not** 옵션을 선택한 경우 `content` 또는 `protected_content` 키워드가 하나만 포함된 규칙을 생성해서는 안 됩니다. 침입 정책을 무효화할 수 있습니다.

예를 들어 SMTP rule 1:2541:9에는 세 개의 `content` 키워드가 포함되어 있으며, 그중 하나에 대해서만 **Not** 옵션이 선택되어 있습니다. **Not** 옵션이 선택된 하나를 제외하고 모든 `content` 키워드를 제거하면 이 규칙을 기반으로 하는 맞춤형 규칙은 무효가 됩니다. 침입 정책에 이러한 규칙을 추가하면 정책을 무효화할 수 있습니다.

팁 동일한 content 키워드에서 **Not(**아님**)** 체크 박스와 **Use Fast Pattern Matcher(**빠른 패턴 매처 사용**)** 체크 박스를 모두 선택할 수는 없습니다.

content 또는 protected_content 키워드 검색 위치

위치 검색 옵션을 사용하여 지정된 콘텐츠 검색을 시작할 위치 및 검색 범위를 지정할 수 있습니다.

허용된 조합: content 검색 위치 인수

두 개의 content 위치 쌍 중 하나를 사용하여 지정된 콘텐츠 검색을 시작할 위치 및 검색 범위를 다음과 같이 지정할 수 있습니다.

- **Offset(**오프셋**)** 및 **Depth(**수준**)**를 함께 사용하여 패킷 페이로드의 시작과 관련된 항목을 검색합니다.
- **Distance(**영역**)** 및 **Within(**내부**)**을 함께 사용하여 현재 검색 위치와 관련된 항목을 검색합니다.

한 쌍만 지정할 때, 쌍에서 다른 옵션의 기본값이 가정됩니다.

Offset(오프셋**)** 및 **Depth(**수준**)** 옵션을 **Distance(**영역**)** 및 **Within(**내부**)** 옵션과 함께 사용할 수 없습니다. 예를 들어 **Offset** 및 **Within** 쌍은 사용할 수 없습니다. 규칙에서 위치 옵션을 얼마든지 사용할 수 있습니다.

위치가 지정되지 않으면 **Offset** 및 **Depth**의 기본값이 사용됩니다. 즉, 내용 검색은 패킷 페이로드의 처음부터 시작되고 패킷의 끝까지 계속됩니다.

기존의 byte_extract 변수를 사용하여 위치 옵션에 대한 값을 지정할 수 있습니다.

팁 규칙에서 위치 옵션을 얼마든지 사용할 수 있습니다.

관련 항목

byte_extract 키워드, 38 페이지

허용되는 조합: protected_content 검색 위치 인수

지정된 내용에 대한 검색을 어디에서 시작할지, 어디까지 계속해야 할지를 지정하려면 다음과 같이 필수 **Length** protected_content 위치 옵션을 **Offset** 또는 **Distance** 위치 옵션 중 하나와 함께 사용하십시오.

- **Length(**길이**)**와 **Offset(**오프셋**)**을 함께 사용하여 패킷 페이로드의 시작과 관련된 보호된 문자열을 검색합니다.
- **Length(**길이**)**와 **Distance(**영역**)**를 함께 사용하여 현재 검색 위치와 관련된 보호된 문자열을 검색합니다.

팁 단일 키워드 구성에서 **Offset(**오프셋**)**과 **Distance(**영역**)** 옵션을 함께 사용할 수 없지만, 규칙에서 위치 옵션은 얼마든지 사용할 수 있습니다.

어떤 위치도 지정하지 않은 경우, 기본값이 가정됩니다. 즉, 패킷 페이로드가 시작될 때 콘텐츠 검색이 시작되며, 패킷이 종료될 때까지 계속됩니다.

기존의 `byte_extract` 변수를 사용하여 위치 옵션에 대한 값을 지정할 수 있습니다.

관련 항목

byte_extract 키워드, 38 페이지

content 및 **protected_content** 검색 위치 인수

깊이

참고 이 옵션은 `content` 키워드를 구성할 때만 지원됩니다.

최대 콘텐츠 검색 수준을 오프셋 값의 시작부터 바이트 단위로 지정하거나 오프셋이 구성되어 있지 않은 경우 패킷 페이로드의 시작부터 지정합니다.

예를 들어 content 값 `cgi-bin/phf`, `offset` 값 `3`, `depth` 값 `22`의 규칙에서는 규칙 헤더에 지정된 매개 변수를 충족하는 패킷에서 `cgi-bin/phf` 문자열에 대한 일치 검색이 3바이트에서 시작되고 22바이트를 처리한 후(25바이트에서) 중지됩니다.

최대 65535바이트의 지정된 콘텐츠의 길이와 같거나 큰 값을 지정해야 합니다. 값으로 0을 지정할 수 없습니다.

기본 수준은 패킷 끝까지 검색하는 것입니다.

거리

이전의 성공적인 콘텐츠 일치 후에 지정한 바이트 수가 나타나는 다음 콘텐츠 일치를 규칙 엔진이 확인하도록 지시합니다.

영역 카운터가 0바이트에서 시작하므로, 마지막 성공적인 콘텐츠 일치보다 앞으로 이동하기를 원하는 바이트 수보다 하나 적은 수를 지정합니다. 예를 들어, 4를 지정한 경우, 검색은 다섯 번째 바이트에서 시작됩니다.

-65535에서 65535까지 바이트 값을 지정할 수 있습니다. 마이너스 `Distance(영역)` 값을 지정한 경우, 사용자가 찾기 시작한 바이트가 패킷의 초기에 범위 밖으로 밀려날 수 있습니다. 검색은 패킷의 첫 번째 바이트에서 시작하지만, 패킷 외부 바이트를 모두 고려합니다. 예를 들어 패킷의 현재 위치가 5번째 바이트이고 다음 내용 규칙 옵션에서 `Distance` 값 -10이며 `Within` 값이 20이면, 검색은 페이로드의 처음부터 시작되며 `Within` 옵션은 15로 조정됩니다.

기본값이 0인 것은 마지막 콘텐츠 일치 다음의 패킷 내 현재 위치를 의미합니다.

길이

참고 이 옵션은 `protected_content` 키워드를 구성할 때만 지원됩니다.

Length(길이**)** `protected_content` 키워드 옵션은 해시되지 않은 검색 문자열의 길이를 바이트 단위로 나타냅니다.

예를 들어 안전한 해시를 생성하기 위해 내용 `Sample1` 을 사용한 경우 **Length** 값에 `7`을 사용하십시오. 반드시 이 필드에 값을 입력해야 합니다.

Offset(오프셋**)**

패킷 페이로드의 어느 위치에서 패킷 페이로드의 시작과 관련된 콘텐츠 검색을 시작할지 바이트 단위로 지정합니다. -65535에서 65535까지 바이트 값을 지정할 수 있습니다.

오프셋 카운터가 0바이트에서 시작하므로, 패킷 페이로드의 시작에서 앞으로 이동하기를 원하는 바이트 수보다 하나 적은 수를 지정합니다. 예를 들어, 7을 지정한 경우, 검색은 여덟 번째 바이트에서 시작됩니다.

기본 오프셋은 0으로, 패킷의 시작을 의미합니다.

내부

참고 이 옵션은 `content` 키워드를 구성할 때만 지원됩니다.

Within(내부**)** 옵션은 규칙을 트리거하려면 다음 콘텐츠 일치가 마지막 콘텐츠 일치 종료 후 지정한 바이트 수 안에 발생해야 한다는 것을 나타냅니다. 예를 들어 **Within** 값으로 `8`을 지정하면 다음 내용 일치는 패킷 페이로드의 다음 8바이트 이내에 발생해야 합니다. 그렇지 않으면 규칙 트리거 기준이 충족되지 않습니다.

최대 65535바이트의 지정된 콘텐츠의 길이와 같거나 큰 값을 지정할 수 있습니다.

Within의 기본값은 패킷의 끝까지 검색하는 것입니다.

개요: **HTTP content** 및 **protected_content** 키워드 인수

HTTP `content` 또는 `protected_content` 키워드 옵션으로 인해 HTTP 검사 전처리기에서 디코딩된 HTTP 메시지 안에서 콘텐츠 일치를 검색할 위치를 지정할 수 있습니다.

두 가지 옵션은 HTTP 응답의 상태 필드를 검색합니다.

- **HTTP** 상태 코드
- **HTTP** 상태 메시지

규칙 엔진이 원시, 비정규 상태 필드를 검색하더라도 다른 원시 HTTP 필드 및 표준화된 HTTP 필드가 결합되면 이 옵션은 고려해야 할 제한 조건 하에서 설명을 간소화하기 위해 여기에 별도로 나열된다는 점에 유의하십시오.

다섯 가지 옵션은 적절한 수준에서 HTTP 요청이나 응답 또는 둘 다에서 표준화된 필드를 검색합니다.

- **HTTP URI**
- **HTTP** 메서드
- **HTTP** 헤더
- **HTTP** 쿠키
- **HTTP** 클라이언트 본문

세 가지 옵션은 적절한 수준에서 HTTP 요청이나 응답 또는 둘 다에서 원시(표준화되지 않은) 비상태 필드를 검색합니다.

- **HTTP** 원시 **URI**
- **HTTP** 원시 헤더
- **HTTP** 원시 쿠키

HTTP `content` 옵션을 선택할 때 다음 지침을 사용하십시오.

- HTTP `content` 옵션은 TCP 트래픽에만 적용됩니다.
- 성능에 부정적인 영향을 방지하려면, 지정된 내용이 표시될 수 있는 메시지의 해당 부분만 선택합니다.

  예를 들어, 쇼핑 카트 메시지에서처럼 트래픽이 규모가 큰 쿠키를 포함할 가능성이 높을 경우, HTTP 헤더에서 지정된 콘텐츠를 검색할 수 있지만 HTTP 쿠키에서는 검색할 수 없습니다.

- HTTP 검사 전처리기 표준화를 이용하고 성능을 향상시키려면, 사용자가 생성한 모든 HTTP 관련 규칙에 **HTTP URI**, **HTTP Method(HTTP** 메서드**)**, **HTTP Header(HTTP** 헤더**)** 또는 **HTTP Client Body(HTTP** 클라이언트 본문**)** 옵션이 선택된 최소한 하나의 `content` 또는 `protected_content` 키워드가 포함되어야 합니다.

- `replace` 키워드를 HTTP `content` 또는 `protected_content` 키워드 옵션과 함께 사용할 수 없습니다.

단일 표준화된 HTTP 옵션 또는 상태 필드를 지정하거나, 표준화된 HTTP 옵션 및 상태 필드를 어떤 조합에서나 사용하여 일치하는 콘텐츠 영역을 대상으로 할 수 있습니다. 그러나 HTTP 필드 옵션을 사용할 경우, 다음 제한 사항을 참고하십시오.

- **Raw Data** 옵션은 동일한 `content` 또는 `protected_content` 키워드에서 HTTP 옵션과 함께 사용할 수 없습니다.
- 원시 HTTP 필드 옵션(**HTTP Raw URI**, **HTTP Raw Header** 또는 **HTTP Raw Cookie**)을 동일한 `content` 또는 `protected_content` 키워드에서 해당 표준화 옵션(각각 **HTTP URI**, **HTTP Header** 또는 **HTTP Cookie**)과 함께 사용할 수 없습니다.
- 하나 이상의 다음 HTTP 필드 옵션과 함께 **Use Fast Pattern Matcher**를 선택할 수 없습니다.

HTTP Raw URI(HTTP 원시 **URI)**, **HTTP Raw Header(HTTP** 원시 헤더**)**, **HTTP Raw Cookie(HTTP** 원시 쿠키**)**, **HTTP Cookie(HTTP** 쿠키**)**, **HTTP Method(HTTP** 메서드**)**, **HTTP Status Message(HTTP** 상태 메시지**)** 또는 **HTTP Status Code(HTTP** 상태 코드**)**

그러나 다음의 표준화 필드 중 하나를 검색하는 데 역시 fast pattern matcher를 사용하는 `content` 또는 `protected_content` 키워드에는 위의 옵션을 포함할 수 있습니다.

HTTP URI, **HTTP Header(HTTP** 헤더**)** 또는 **HTTP Client Body(HTTP** 클라이언트 본문**)**

예를 들어, **HTTP Cookie(HTTP** 쿠키**)**, **HTTP** 헤더**(HTTP Header)**, 및 **Use Fast Pattern Matcher(**빠른 패턴 매처 사용**)**를 선택한 경우, 규칙 엔진은 HTTP 쿠키 및 HTTP 헤더 모두에서 콘텐츠를 검색하지만 빠른 패턴 매처는 HTTP 헤더에만 적용되고 HTTP 쿠키에는 적용되지 않습니다.

- 제한 옵션과 비제한 옵션을 결합하면 fast pattern matcher는 사용자가 지정한 비제한 필드만 검색하여, 제한된 필드에 대한 평가를 비롯한 완전한 평가를 위해 침입 규칙 편집기로 규칙을 전달할지 여부를 테스트합니다.

관련 항목

content 키워드 빠른 패턴 매처 인수, 30 페이지

HTTP content 및 protected_content 키워드 인수

HTTP URI

이 옵션을 선택하여 표준화된 요청 URI 필드에서 콘텐츠 일치를 검색합니다.

같은 콘텐츠를 검색하기 위해 이 옵션을 `pcre` 키워드 HTTP URI (U) 옵션과 조합하여 사용할 수 없다는 점에 유의하십시오.

참고 파이프라인 방식 HTTP 요청 패킷은 여러 URI를 포함합니다. **HTTP URL**을 선택하여 규칙 엔진이 파이프라인 방식 HTTP 요청 패킷을 탐지하면, 규칙 엔진은 콘텐츠 일치를 위해 패킷 내 모든 URI를 검색합니다.

HTTP 원시 URI

이 옵션을 선택하여 표준화된 요청 URI 필드에서 콘텐츠 일치를 검색합니다.

같은 콘텐츠를 검색하기 위해 이 옵션을 `pcre` 키워드 HTTP URI (U) 옵션과 조합하여 사용할 수 없다는 점에 유의하십시오.

참고 파이프라인 방식 HTTP 요청 패킷은 여러 URI를 포함합니다. **HTTP URL**을 선택하여 규칙 엔진이 파이프라인 방식 HTTP 요청 패킷을 탐지하면, 규칙 엔진은 콘텐츠 일치를 위해 패킷 내 모든 URI를 검색합니다.

HTTP 메서드

이 옵션을 선택하여 요청 메서드 필드에서 콘텐츠 일치를 검색하는데, 이는 URI에서 식별된 리소스를 사용하는 GET 및 POST와 같은 작업을 확인합니다.

HTTP 헤더

이 옵션을 선택하여 표준화된 헤더 필드의 콘텐츠 일치를 검색하는데, HTTP 요청에서는 쿠키를 제외합니다. 또한 HTTP 검사 전처리기 **Inspect HTTP Responses(HTTP** 응답 검사**)** 옵션이 활성화된 응답에서도 쿠키를 제외합니다.

동일한 콘텐츠를 검색하기 위해 이 옵션을 `pcre` 키워드 HTTP 헤더 (H) 옵션과 조합하여 사용할 수 없다는 점에 유의하십시오.

HTTP 원시 헤더

이 옵션을 선택하여 원시 헤더 필드의 콘텐츠 일치를 검색하는데, HTTP 요청에서는 쿠키를 제외합니다. 또한 HTTP 검사 전처리기 **Inspect HTTP Responses(HTTP** 응답 검사**)** 옵션이 활성화된 응답에서도 쿠키를 제외합니다.

동일한 콘텐츠를 검색하기 위해 이 옵션을 `pcre` 키워드 HTTP 원시 헤더(D) 옵션과 조합하여 사용할 수 없다는 점에 유의하십시오.

HTTP 쿠키

표준화된 HTTP 클라이언트 요청 헤더에서 식별된 쿠키에서, 그리고 HTTP Inspect 전처리기 **Inspect HTTP Responses(HTTP** 응답 검사**)** 옵션이 활성화되었을 때 응답 set-cookie 데이터에서 내용 일치를 검색하려면 이 옵션을 선택합니다. 시스템이 본문 내용으로서의 메시지 본문에 포함되는 쿠키를 처리한다는 점에 유의하십시오.

반드시 HTTP 검사 전처리기 **Inspect HTTP Cookies(HTTP** 쿠키 검사**)** 옵션을 활성화하여 일치하는 쿠키만 검색해야 합니다. 그렇지 않을 경우, 규칙 엔진은 쿠키를 포함하는 전체 헤더를 검색합니다.

다음 사항을 참고하십시오.

- 동일한 콘텐츠를 검색하기 위해 이 옵션을 `pcre` 키워드 HTTP 쿠키 (C) 옵션과 조합하여 사용할 수 없습니다.
- `Cookie:` 및 `Set-Cookie:` 헤더 이름, 헤더 행의 주요 스페이스 및 헤더 행을 종료하는 `CRLF`는 헤더의 일부로 검사되지만 쿠키의 일부로는 검사되지 않습니다.

HTTP 원시 쿠키

표준화된 HTTP 클라이언트 요청 헤더에서 식별된 쿠키에서, 그리고 HTTP Inspect 전처리기 **Inspect HTTP Responses(HTTP** 응답 검사**)** 옵션이 활성화되었을 때 응답 set-cookie 데이터에서 내용 일치를 검색하려면 이 옵션을 선택합니다. 시스템은 메시지 본문에 포함된 쿠키를 본문 내용으로 취급합니다.

반드시 HTTP 검사 전처리기 **Inspect HTTP Cookies(HTTP** 쿠키 검사**)** 옵션을 활성화하여 일치하는 쿠키만 검색해야 합니다. 그렇지 않을 경우, 규칙 엔진은 쿠키를 포함하는 전체 헤더를 검색합니다.

다음 사항을 참고하십시오.

- 동일한 콘텐츠를 검색하기 위해 이 옵션을 pcre 키워드 HTTP 원시 쿠키 (K) 옵션과 조합하여 사용할 수 없습니다.
- Cookie: 및 Set-Cookie: 헤더 이름, 헤더 행의 주요 스페이스 및 헤더 행을 종료하는 CRLF는 헤더의 일부로 검사되지만 쿠키의 일부로는 검사되지 않습니다.

HTTP 클라이언트 본문

이 옵션을 선택하여 HTTP 클라이언트 요청에서 메시지 본문의 콘텐츠 일치를 검색합니다.

이 옵션이 작동하도록 하려면, HTTP 검사 전처리기 **HTTP Client Body Extraction Depth(HTTP** 클라이언트 본문 추출 수준**)** 옵션에 대해 0에서 65535까지의 값을 지정해야 합니다.

HTTP 상태 코드

HTTP 응답의 3자리 상태 코드에서 내용 일치를 검색하려면 이 옵션을 선택합니다.

이 옵션이 일치를 되돌릴 수 있도록 HTTP 검사 전처리기 **Inspect HTTP Responses(HTTP** 응답 검사**)** 옵션을 활성화해야 합니다.

HTTP 상태 메시지

이 옵션을 선택하여 HTTP 응답의 상태 코드에 동반되는 본문 설명에서 콘텐츠 일치를 검색합니다.

이 옵션이 일치를 되돌릴 수 있도록 HTTP 검사 전처리기 **Inspect HTTP Responses(HTTP** 응답 검사**)** 옵션을 활성화해야 합니다.

관련 항목

pcre 수식자 옵션, 46 페이지
서버 레벨 HTTP 정상화 옵션

개요: **content** 키워드 빠른 패턴 매처

참고 protected_content 키워드를 구성할 때 이 옵션은 지원되지 않습니다.

빠른 패턴 매처는 패킷을 규칙 엔진에 전달하기 전에 평가할 규칙을 신속하게 결정합니다. 이 초기 결정은 패킷 평가에 사용되는 규칙의 수를 크게 줄여 성능을 개선합니다.

기본적으로, 빠른 패턴 매처는 규칙에 지정된 가장 긴 콘텐츠를 위한 패킷을 검색합니다. 이는 규칙의 필요 없는 평가를 최대한 제거하기 위한 것입니다. 다음의 예제 규칙 조각을 고려하십시오.

```
alert tcp any any -> any 80 (msg:"Exploit"; content:"GET";
http_method; nocase; content:"/exploit.cgi"; http_uri;
nocase;)
```

거의 모든 HTTP 클라이언트 요청은 GET 콘텐츠를 포함하지만, /exploit.cgi 콘텐츠를 포함하는 요청은 거의 없습니다. 빠른 패턴 콘텐츠로 GET을 사용하는 것은 대부분의 경우 규칙 엔진이 이 규칙을 평가하도록 하며 일치로 귀결되는 경우가 거의 없도록 합니다. 그러나 대부분의 클라이언트 GET 요청은 /exploit.cgi를 사용하여 평가되지 않을 것이므로 성능이 증가할 것입니다.

빠른 패턴 매처가 지정된 콘텐츠를 탐지하는 경우에만 규칙 엔진이 규칙에 대해 패킷을 평가합니다. 예를 들어, 규칙 내 하나의 `content` 키워드가 콘텐츠를 `short`로 지정하는데, 다른 키워드는 `longer`로 지정하고, 세 번째 키워드는 `longest`로 지정하는 경우, 빠른 패턴 매처는 `longest` 콘텐츠를 사용하며, 규칙 엔진이 페이로드에서 `longest`로 평가된 경우에만 규칙이 평가됩니다.

content 키워드 빠른 패턴 매처 인수

빠른 패턴 매처 사용

사용할 빠른 패턴 매처에 더 짧은 검색 패턴을 지정하려면 이 옵션을 사용합니다. 원칙적으로, 지정한 패턴은 패킷 내에서 발견될 가능성이 가장 긴 패턴보다 낮으므로 표적 공격을 더욱 구체적으로 식별합니다.

Use Fast Pattern Matcher 및 동일한 `content` 키워드의 다른 옵션을 선택할 때는 다음 제한 사항에 유의하십시오.

- 규칙당 한 번만 **Use Fast Pattern Matcher(**빠른 패턴 매처 사용**)**를 지정할 수 있습니다.
- **Use Fast Pattern Matcher(**빠른 패턴 매처 사용**)**를 **Not(**아님**)**과 함께 선택하는 경우 **Distance(**영역**)**, **Within(**내부**)**, **Offset(**오프셋**)** 또는 **Depth(**수준**)**를 사용할 수 없습니다.
- Use Fast Pattern Matcher(빠른 패턴 매처 사용)는 다음 HTTP 필드 옵션 중 어느 것과도 조합하여 선택할 수 없습니다.

  HTTP Raw URI(HTTP 원시 **URI)**, **HTTP Raw Header(HTTP** 원시 헤더**)**, **HTTP Raw Cookie(HTTP** 원시 쿠키**)**, **HTTP Cookie(HTTP** 쿠키**)**, **HTTP Method(HTTP** 메서드**)**, **HTTP Status Message(HTTP** 상태 메시지**)** 또는 **HTTP Status Code(HTTP** 상태 코드**)**

  그러나 다음의 표준화 필드 중 하나를 검색하는 데 역시 fast pattern matcher를 사용하는 `content` 키워드에는 위의 옵션을 포함할 수 있습니다.

  HTTP URI, **HTTP Header(HTTP** 헤더**)** 또는 **HTTP Client Body(HTTP** 클라이언트 본문**)**

  예를 들어, **HTTP Cookie(HTTP** 쿠키**)**, **HTTP** 헤더**(HTTP Header)**, 및 **Use Fast Pattern Matcher(**빠른 패턴 매처 사용**)**를 선택한 경우, 규칙 엔진은 HTTP 쿠키 및 HTTP 헤더 모두에서 콘텐츠를 검색하지만 빠른 패턴 매처는 HTTP 헤더에만 적용되고 HTTP 쿠키에는 적용되지 않습니다.

  원시 HTTP 필드 옵션(**HTTP Raw URI**, **HTTP Raw Header** 또는 **HTTP Raw Cookie**)을 동일한 `content` 키워드에서 해당 표준화 옵션(각각 **HTTP URI**, **HTTP Header** 또는 **HTTP Cookie**)과 함께 사용할 수 없습니다.

  제한 옵션과 무제한 옵션을 결합할 때, 빠른 패턴 매처는 사용자가 지정하는 무제한 필드만을 검색하여 제한된 필드의 평가를 포함하는 전체 평가를 위한 규칙 엔진으로 패킷을 전달할지 여부를 테스트합니다.

- 선택적으로, **Use Fast Pattern Matcher(**빠른 패턴 매처 사용**)**를 선택하면, **Fast Pattern Matcher Only(**빠른 패턴 매처 한정**)** 또는 **Fast Pattern Matcher Offset and Length(**빠른 패턴 매처 오프셋 및 길이**)** 중 하나를 선택할 수 있지만 둘 다 선택할 수는 없습니다.
- Base64 데이터를 검사할 때는 빠른 패턴 매처를 사용할 수 없습니다.

빠른 패턴 매처 한정

이 옵션을 선택하면 `content` 키워드를 규칙 옵션이 아니라 빠른 패턴 매처 옵션으로만 사용할 수 있습니다. 지정된 콘텐츠의 규칙 엔진 평가가 필요하지 않은 경우 이 옵션을 사용하여 리소스를 유지할 수 있습니다. 예를 들어, 콘텐츠 `12345`가 페이로드 안에서 어디든 있어야 한다고만 요구하는 규칙의 경우를 생각해 보십시오. 빠른 패턴 매처가 패턴을 탐지하면, 패킷이 규칙의 추가 키워드에 대해 평가될 수 있습니다. 패턴 `12345`를 포함하는지 확인하기 위해 패킷을 재평가하는 규칙 엔진은 없어도 됩니다.

규칙이 지정된 콘텐츠에 연결된 다른 조건을 포함할 때는 이 옵션을 사용하지 않습니다. 예를 들어, `abcd`가 `1234` 앞에 나타나는지를 다른 규칙 조건이 확인하는 경우 `1234` 내용을 검색하기 위해 이 옵션을 사용하지 않을 수 있습니다. 이 경우 규칙 엔진은 상대적인 위치를 파악할 수 없습니다. **Fast Pattern Matcher Only**를 지정하면 규칙 엔진은 지정된 내용을 검색하지 않기 때문입니다.

이 옵션을 사용할 때 다음 사항에 유의하십시오:

- 지정된 내용은 위치와 무관합니다. 즉, 페이로드 어디에서나 발생할 수 있습니다. 따라서 위치 옵션(**Distance(**영역**)**, **Within(**내부**)**, **Offset(**오프셋**)**, **Depth(**수준**)**, 또는 **Fast Pattern Matcher Offset and Length(**빠른 패턴 매처 오프셋 및 길이**)**)을 사용할 수 없습니다.
- 이 옵션을 **Not(**아님**)**과 조합하여 사용할 수 없습니다.
- 이 옵션을 **Fast Pattern Matcher Offset and Length(**빠른 패턴 매처 오프셋 및 길이**)**와 조합하여 사용할 수 없습니다.
- 모든 패턴이 빠른 패턴 매처에 대소문자 구분 없이 삽입되므로 지정된 내용에서는 대소문자가 구분되지 않습니다. 이는 자동으로 처리되므로 이 옵션을 선택할 때 **Case Insensitive(**대소문자 구분 안 함**)**를 선택할 필요가 없습니다.
- 사용자는 즉시 다음 키워드와 **Fast Pattern Matcher Only(**빠른 패턴 매처 한정**)** 옵션을 함께 사용하는 `content` 키워드를 따르지 말아야 하며, 이는 현재 검색 위치에 관련된 검색 위치를 설정하는 것입니다.
  - `isdataat`
  - `pcre`
  - **Distance(**영역**)** 또는 **Within(**내부**)**을 선택한 경우 `content`
  - **HTTP URI**를 선택한 경우 `content`
  - `asn1`
  - `byte_jump`
  - `byte_test`
  - `byte_math`
  - `byte_extract`
  - `base64_decode`

빠른 패턴 매처 오프셋 및 길이

Fast Pattern Matcher Offset and Length(빠른 패턴 매처 오프셋 및 길이**)** 옵션을 사용하면 검색할 콘텐츠의 일부를 지정할 수 있습니다. 이는 패턴이 너무 길어 패턴의 일부만으로도 규칙을 가능한 일치로 확인하기에 충분한 경우 메모리 사용량을 줄일 수 있습니다. 규칙이 빠른 패턴 매처에서 선택되면, 규칙에 대해 전체 패턴이 평가됩니다.

다음 구문을 사용하여 검색을 시작할 위치(오프셋) 및 검색 범위(길이)를 바이트 단위로 지정함으로써 빠른 패턴 매처의 일부를 사용하기로 결정합니다.

```
offset,length
```

예를 들어, 다음과 같은 콘텐츠의 경우:

```
1234567
```

오프셋과 길이 바이트 수를 다음과 같이 지정할 경우:

```
1,5
```

빠른 패턴 매처가 콘텐츠 `23456`만 검색합니다.

이 옵션을 **Fast Pattern Matcher Only(**빠른 패턴 매처 한정**)**와 함께 사용할 수 없습니다.

관련 항목

개요: HTTP content 및 protected_content 키워드 인수, 25 페이지
base64_decode 및 base64_data 키워드, 116 페이지

replace 키워드

인라인 구축에서 `replace` 키워드를 사용해 지정된 내용을 교체하거나 Cisco SSL Appliance에 의해 탐지된 SSL 트래픽의 내용을 교체할 수 있습니다.

`replace` 키워드를 사용하려면 `content` 키워드를 사용하여 특정 문자열을 찾는 맞춤형 표준 텍스트 규칙을 작성합니다. `replace` 키워드를 사용하여 콘텐츠를 대체할 문자열을 지정합니다. 대체 값 및 콘텐츠 값은 동일한 길이여야 합니다.

참고 `protected_content` 키워드 내 해시된 콘텐츠를 대체하기 위해 `replace` 키워드를 사용할 수 없습니다.

원하는 경우, 이전 Firepower System 소프트웨어 버전과의 호환을 위해 교체 문자열을 따옴표로 묶을 수 있습니다. 따옴표를 포함하지 않은 경우, 따옴표는 규칙에 자동으로 추가되므로 규칙은 구문적으로 정확합니다. 교체 텍스트의 일부로 앞뒤 따옴표를 포함하려면, 다음의 예시에서 볼 수 있듯이, 백슬래시를 사용하여 이스케이프해야 합니다.

```
"replacement text plus \"quotation\" marks""
```

규칙은 여러 `replace` 키워드를 포함할 수 있지만, `content` 키워드 하나당 하나만 포함됩니다. 규칙에서 찾은 콘텐츠의 첫 번째 인스턴스만 대체됩니다.

다음은 replace 키워드의 사용 예에 대한 설명입니다.

- 시스템이 공격을 포함한 수신 패킷을 탐지하는 경우, 악성 문자열을 무해한 것으로 바꿀 수 있습니다. 때때로 이 기술은 단순히 문제를 일으키는 패킷을 중단하는 것보다 더욱 성공적입니다. 일부 공격 시나리오에서, 공격자는 삭제된 패킷이 네트워크 방어를 무시하거나 네트워크를 초과할 때까지 다시 보냅니다. 패킷을 삭제하는 대신 다른 문자열로 한 문자열을 대체하여, 취약하지 않았던 대상에 대해 공격이 개시되었다고 공격자가 믿도록 속일 수 있습니다.
- 예를 들어, 취약한 버전의 웹 서버를 실행하고 있는지 여부를 알기 위해 시도하는 정찰 공격이 우려되는 경우, 발신 패킷을 탐지하고 본인의 텍스트로 배너를 바꿀 수 있습니다.

참고 대체 규칙을 사용하려는 인라인 침입 정책에서 규칙 상태를 Generate Events(이벤트 생성)로 설정했는지 확인합니다. 규칙을 Drop and Generate events(이벤트 중단 및 생성)로 설정하면 패킷이 중단될 수 있고, 이는 콘텐츠를 대체하는 것을 방지합니다.

문자열 교체 프로세스의 일부로서, 대상 호스트가 패킷을 오류 없이 받을 수 있도록 시스템은 자동으로 패킷 체크섬을 업데이트합니다.

replace 키워드를 HTTP 요청 메시지 content 키워드 옵션과 함께 사용할 수 없다는 점에 유의하십시오.

관련 항목

content 및 protected_content 키워드, 20 페이지
개요: HTTP content 및 protected_content 키워드 인수, 25 페이지

byte_jump 키워드

byte_jump 키워드는 지정된 바이트 세그먼트에서 정의된 바이트 수를 계산한 후 지정하는 옵션에 따라 지정된 바이트 세그먼트의 끝 또는 패킷 페이로드의 시작 또는 끝 또는 마지막 콘텐츠 일치와 관련된 지점에서부터 앞으로 패킷 내 바이트 수를 건너뜁니다. 이는 특정 세그먼트 바이트가 패킷 내 변수 데이터에 포함된 바이트 수를 설명하는 패킷에 유용합니다.

다음 표에서는 byte_jump 키워드에 필요한 인수에 대해 설명합니다.

표 ***10:*** 필수 ***byte_jump*** 인수

인수	설명
바이트	패킷에서 추출할 바이트 수입니다. DCE/RPC 없이 사용하는 경우, 허용되는 값은 0 ~ 10이며 다음 제한 사항이 적용됩니다. • `From End(끝부터)` 인수와 함께 사용하는 경우, bytes는 0일 수 있습니다. Bytes가 0이면 추출되는 값은 0입니다. • 1, 2 또는 4 이외의 바이트 수를 지정하는 경우, 숫자 유형(16진수, 8진수 또는 10진수)을 지정해야 합니다. DCE/RPC와 함께 사용하는 경우, 허용되는 값은 1, 2, 4입니다.
Offset	처리를 시작할 페이로드에 들어가는 바이트 수. `offset` 카운터는 0바이트부터 시작되므로, `offset` 값을 계산할 때는 패킷 페이로드의 처음부터 또는 마지막으로 성공한 내용 일치로부터 앞으로 이동하려는 바이트 수에서 1을 빼야 합니다. -65535에서 65535까지 바이트를 지정할 수 있습니다. 기존의 `byte_extract` 변수 또는 `byte_math` 결과를 사용하여 이 인수의 값을 지정할 수도 있습니다.

다음 표에서는 시스템이 필수 인수에 지정한 값을 해석하는 방식을 정의하는 데 사용할 수 있는 옵션을 설명합니다.

표 ***11:*** 추가 선택 ***byte_jump*** 인수

인수	설명
Relative	오프셋이 마지막으로 성공한 콘텐츠 일치에 포함된 마지막 패턴에 연결되도록 합니다.
Align	변환된 바이트의 수를 다음 32비트 경계로 반올림합니다.
Multiplier	규칙 엔진이 최종 `byte_jump` 값을 얻기 위해 패킷에서 얻은 `byte_jump` 값에 곱해야 하는 값을 나타냅니다. 즉 규칙 엔진은 지정된 바이트 세그먼트에서 정의한 바이트 수를 건너뛰는 대신, Multiplier 인수로 지정한 정수로 곱해진 바이트 수를 건너뜁니다.
Post Jump Offset	다른 `byte_jump` 인수를 적용한 후 앞이나 뒤로 건너뛸 -63535~63535 범위의 바이트 수입니다. 양수 값은 앞으로 건너뛰고 음수 값은 뒤로 건너뜁니다. 필드를 비워 두거나 `0`을 입력하여 비활성화합니다. **DCE/RPC** 인수를 선택할 경우, 일부 `byte_jump` 인수가 적용되지 않습니다.
From Beginning	규칙 엔진이 패킷의 현재 위치부터가 아니라 패킷 페이로드의 처음부터 시작하는 페이로드 내 지정된 바이트 수를 건너뛰어야 한다고 표시합니다.

인수	설명
From End(끝부터)	버퍼의 마지막 바이트 다음에 오는 바이트부터 점프가 시작됩니다.
Bitmask(비트마스크)	AND 연산자를 사용하여 지정된 16진수 비트마스크를 Bytes 인수에서 추출된 바이트에 적용합니다. 비트마스크는 1 ~ 4바이트일 수 있습니다. 결과는 마스크의 후행 0 수와 같은 비트 수만큼 오른쪽 시프트됩니다.

DCE/RPC, **Endian** 또는 **Number Type** 중 하나만 지정할 수 있습니다.

byte_jump 키워드가 바이트를 계산하는 방법을 정의하려는 경우, 다음 표에 설명된 인수 중에서 선택할 수 있습니다. 바이트 순서 인수를 선택하지 않으면 규칙 엔진은 Big Endian 바이트 순서를 사용합니다.

표 ***12:*** 바이트 순서 ***byte_jump*** 인수

인수	설명
Big Endian	빅 엔디언 바이트 순서의 프로세스 데이터. 기본 네트워크 바이트 순서입니다.
Little Endian	리틀 엔디언 순서의 프로세스 데이터
DCE/RPC	DCE/RPC 전처리기에서 처리된 트래픽에 byte_jump 키워드를 지정합니다. DCE/RPC 전처리기는 빅 엔디언 또는 리틀 엔디언 바이트 순서를 결정하고, **Number Type** 및 **Endian** 인수는 적용되지 않습니다. 이 인수를 활성화하면, byte_jump를 다른 DCE/RPC 특정 키워드와 함께 사용할 수 있습니다.

시스템이 다음 표에 있는 인수 중 하나를 사용하여 패킷 내 문자열을 보는 방식을 정의합니다.

표 ***13:*** 번호 유형 인수

인수	설명
Hexadecimal String	변환된 문자열 데이터를 16진수 형태로 나타냅니다.
Decimal String	변환된 문자열 데이터를 십진수 형태로 나타냅니다.
Octal String	변환된 문자열 데이터를 8진수 형태로 나타냅니다.

예를 들어 byte_jump에 대해 설정한 값이 다음과 같으면

- Bytes = 4
- Offset = 12
- Relative 활성화

- Align 활성화

규칙 엔진은 마지막으로 성공한 콘텐츠 일치 후 13바이트를 나타내는 4개의 바이트로 표시된 수를 계산하고, 패킷 내 해당 수의 바이트를 건너뜁니다. 예를 들어, 특정 패킷 내 4개의 계산된 바이트가 `00 00 00 1F`인 경우, 규칙 엔진은 이를 31로 환산합니다. 엔진에 다음 32비트 경계로 이동하도록 지시하는 `align`이 지정되었으므로 규칙 엔진은 패킷에서 32바이트 앞으로 건너뜁니다.

또는 `byte_jump`에 대해 설정한 값이 다음과 같으면

- Bytes = 4
- Offset = 12
- From Beginning 활성화
- Multiplier = 2

규칙 엔진은 패킷의 시작 후 13바이트를 나타내는 4개의 바이트로 표시된 수를 계산합니다. 다음, 엔진은 이 수에 2를 곱하여 건너뛸 총 바이트 수를 얻습니다. 예를 들어, 특정 패킷 내 4개의 계산된 바이트가 `00 00 00 1F`인 경우, 규칙 엔진은 이를 31로 환산한 후 2를 곱하여 62를 얻습니다. From Beginning이 활성화되어 있으므로, 규칙 엔진은 패킷의 처음 63바이트를 건너뜁니다.

관련 항목

byte_extract 키워드, 38 페이지
DCE/RPC 키워드, 73 페이지

byte_test 키워드

`byte_test` 키워드는 Value 인수 및 연산자에 대해 지정된 바이트 세그먼트를 테스트합니다.

다음 표에서는 `byte_test` 키워드에 필요한 인수에 대해 설명합니다.

표 ***14:*** 필수 ***byte_test*** 인수

인수	설명
바이트	패킷에서 계산할 바이트 수. DCE/RPC 없이 사용하는 경우, 허용되는 값은 1 ~ 10입니다. 하지만 1, 2 또는 4 이외의 바이트 수를 지정하는 경우, 숫자 유형(16진수, 8진수 또는 10진수)을 지정해야 합니다. DCE/RPC와 함께 사용하는 경우, 허용되는 값은 1, 2, 4입니다.

인수	설명
Value	해당 연산자를 포함하여 테스트할 값입니다. 지원되는 연산자: <, >, =, !, &, ^, !>, !<, !=, !& 또는 !^. 예를 들어 `!1024`를 지정하면 `byte_test`는 지정된 숫자를 변환하고, 이 숫자가 1024와 같지 않으면 이벤트를 생성합니다(다른 모든 키워드 파라미터가 일치하는 경우). !와 !=는 같다는 점에 유의하십시오. 기존의 `byte_extract` 변수 또는 `byte_math` 결과를 사용하여 이 인수의 값을 지정할 수도 있습니다.
Offset	처리를 시작할 페이로드에 들어가는 바이트 수. `offset` 카운터는 0바이트부터 시작되므로, `offset` 값을 계산할 때는 패킷 페이로드의 처음부터 또는 마지막으로 성공한 내용 일치로부터 앞으로 이동하려는 바이트 수에서 1을 빼야 합니다. 기존의 `byte_extract` 변수 또는 `byte_math` 결과를 사용하여 이 인수의 값을 지정할 수 있습니다.

시스템에서 다음 표에 설명된 인수와 함께 `byte_test` 인수를 사용하는 방법을 더 정의할 수 있습니다.

표 ***15:*** 추가 선택 ***byte_test*** 인수

인수	설명
Bitmask(비트마스크)	AND 연산자를 사용하여 지정된 16진수 비트마스크를 Bytes 인수에서 추출된 바이트에 적용합니다. 비트마스크는 1 ~ 4바이트일 수 있습니다. 결과는 마스크의 후행 0 수와 같은 비트 수만큼 오른쪽 시프트됩니다.
Relative	오프셋이 마지막으로 성공한 패턴 일치와 연결되도록 합니다.

DCE/RPC, **Endian** 또는 **Number Type** 중 하나만 지정할 수 있습니다.

`byte_test` 키워드가 테스트할 바이트를 계산하는 방법을 정의하려면, 다음 표의 인수 중에서 선택합니다. 바이트 순서 인수를 선택하지 않으면 규칙 엔진은 Big Endian 바이트 순서를 사용합니다.

표 ***16:*** 바이트 순서 ***byte_test*** 인수

인수	설명
Big Endian	빅 엔디언 바이트 순서의 프로세스 데이터. 기본 네트워크 바이트 순서입니다.
Little Endian	리틀 엔디언 순서의 프로세스 데이터

인수	설명
DCE/RPC	DCE/RPC 전처리기에서 처리된 트래픽에 byte_test 키워드를 지정합니다. DCE/RPC 전처리기는 빅 엔디언 또는 리틀 엔디언 바이트 순서를 결정하고, **Number Type** 및 **Endian** 인수는 적용되지 않습니다. 이 인수를 활성화하면, byte_test를 다른 DCE/RPC 특정 키워드와 함께 사용할 수 있습니다.

시스템이 다음 표에 있는 인수 중 하나를 사용하여 패킷 내 문자열 데이터를 보는 방식을 정의할 수 있습니다.

표 ***17:*** 번호 유형 ***byte-test*** 인수

인수	설명
Hexadecimal String	변환된 문자열 데이터를 16진수 형태로 나타냅니다.
Decimal String	변환된 문자열 데이터를 십진수 형태로 나타냅니다.
Octal String	변환된 문자열 데이터를 8진수 형태로 나타냅니다.

예를 들어, byte_test에 대한 값이 다음과 같이 지정된 경우:

- Bytes = 4
- Operator 및 Value > 128
- Offset = 8
- Relative 활성화

규칙 엔진은 마지막으로 성공한 내용 일치에서 9바이트 떨어진 곳에 나타나는 4바이트에 설명된 숫자를 계산하고, 계산된 숫자가 128바이트보다 크면 규칙을 트리거합니다.

관련 항목

byte_extract 키워드, 38 페이지
DCE/RPC 키워드, 73 페이지

byte_extract 키워드

byte_extract 키워드를 사용하여 패킷에서 지정한 바이트 수를 변수로 읽을 수 있습니다. 해당 변수는 나중에 동일한 규칙에서 다른 특정 검색 키워드 내 특정 인수에 대한 값으로 사용할 수 있습니다.

예를 들면, 이는 특정 세그먼트 바이트가 패킷 내 데이터에 포함된 바이트 수를 나타내는 패킷에서 데이터 크기를 추출하는 데 유용합니다. 예를 들어, 특정 세그먼트 바이트는 후속 데이터가 4바이트로 구성되어 있다고 표시할 수 있습니다. 사용자는 사용자 변수 값으로 사용하기 위해 4바이트의 데이터 크기를 추출할 수 있습니다.

byte_extract를 사용하여 규칙에서 최대 두 개의 개별 변수를 동시에 만들 수 있습니다. 몇 번이든 byte_extract 변수를 재정의할 수 있습니다. 동일한 변수 이름 및 기타 변수 정의를 새로운 byte_extract 키워드와 함께 입력하면 해당 변수의 이전 정의를 덮어씁니다.

다음 표에서는 byte_extract 키워드에 필요한 인수에 대해 설명합니다.

표 ***18:*** 필수 ***byte_extract*** 인수

인수	설명
Bytes to Extract	패킷에서 추출할 바이트 수. 1, 2 또는 4 이외의 바이트 수를 지정하는 경우, 숫자 유형(16진수, 8진수 또는 10진수)을 지정해야 합니다.
Offset	데이터 추출을 시작할 페이로드 내 바이트 수. -65535에서 65535까지 바이트를 지정할 수 있습니다. 오프셋 카운터는 0바이트에서 시작하므로, 계산에 넣으려는 바이트 수에서 1을 빼 오프셋 값을 계산합니다. 예를 들어, 8바이트를 계산에 넣으려면 7을 지정합니다. 규칙 엔진은 패킷의 페이로드의 처음부터 계산에 넣거나 사용자가 또한 **Relative(**연결**)**를 지정한 경우, 마지막으로 성공한 콘텐츠 일치 후에 계산에 넣습니다. 음수는 **Relative**를 지정한 경우에만 지정할 수 있습니다. 기존의 byte_math 결과를 사용하여 이 인수의 값을 지정할 수 있습니다.
Variable Name	다른 탐지 키워드에 대한 인수에 사용할 변수 이름입니다. 영숫자 문자열이 반드시 문자로 시작되도록 지정할 수 있습니다.

시스템이 추출할 데이터를 찾는 방식을 더욱 자세히 정의하려면 다음 표에 설명된 인수를 사용할 수 있습니다.

표 ***19:*** 추가 옵션 ***byte_extract*** 인수

인수	설명
Multiplier	패킷에서 추출된 값에 대한 승수입니다. 0에서 65535를 지정할 수 있습니다. 승수를 지정하지 않은 경우, 기본값은 1입니다.
Align	추출된 값을 가장 가까운 2바이트 또는 4바이트 경계로 반올림합니다. 또한 **Multiplier**를 선택하는 경우, 시스템은 정렬 전에 승수를 적용합니다.
Relative	**Offset**이 페이로드의 시작 대신 마지막으로 성공한 콘텐츠 일치의 끝에 연결되도록 합니다.
Bitmask(비트마스크)	AND 연산자를 사용하여 지정된 16진수 비트마스크를 Bytes to Extract 인수에서 추출된 바이트에 적용합니다. 비트마스크는 1 ~ 4바이트일 수 있습니다. 결과는 마스크의 후행 0 수와 같은 비트 수만큼 오른쪽 시프트됩니다.

DCE/RPC, **Endian** 또는 **Number Type** 중 하나만 지정할 수 있습니다.

byte_extract 키워드가 테스트할 바이트를 계산하는 방법을 정의하려면, 다음 표의 인수 중에서 선택합니다. 바이트 순서 인수를 선택하지 않으면 규칙 엔진은 Big Endian 바이트 순서를 사용합니다.

표 ***20:*** 바이트 순서 ***byte_extract*** 인수

인수	설명
Big Endian	빅 엔디언 바이트 순서의 프로세스 데이터. 기본 네트워크 바이트 순서입니다.
Little Endian	리틀 엔디언 순서의 프로세스 데이터
DCE/RPC	DCE/RPC 전처리기에서 처리된 트래픽에 byte_extract 키워드를 지정합니다. DCE/RPC 전처리기는 빅 엔디언 또는 리틀 엔디언 바이트 순서를 결정하고, **Number Type** 및 **Endian** 인수는 적용되지 않습니다. 이 인수를 활성화하면, byte_extract를 다른 DCE/RPC 특정 키워드와 함께 사용할 수도 있습니다.

데이터를 ASCII 문자열로 읽으려면 숫자 유형을 지정할 수 있습니다. 다음 표에 있는 인수 중 하나를 선택하여 패킷 내 문자열 데이터를 보는 방식을 정의할 수 있습니다.

표 ***21:*** 번호 유형 ***byte_extract*** 인수

인수	설명
Hexadecimal String	추출된 문자열 데이터를 16진수 형태로 읽습니다.
Decimal String	추출된 문자열 데이터를 십진수 형태로 읽습니다.
Octal String	추출된 문자열 데이터를 8진수 형태로 읽습니다.

예를 들어, byte_extract에 대한 값이 다음과 같이 지정된 경우

- Bytes to Extract = 4
- Variable Name = var
- Offset = 8
- Relative = 활성화

규칙 엔진은 마지막으로 성공한 콘텐츠 일치(에 연결)에서 9바이트 떨어져서 나타나는 네 개의 바이트에 설명된 번호를 var로 명명된 변수로 읽는데, 이는 나중에 규칙에서 특정 키워드 인수에 대한 값으로 지정할 수 있습니다.

다음 표는 byte_extract 키워드에 정의된 변수를 지정할 수 있는 키워드 인수를 나열합니다.

표 *22: byte_extract* 변수를 받아들이는 인수

키워드	인수
content	Depth, Offset, Distance, Within
byte_jump	Offset(오프셋)
byte_test	Offset, Value
byte_math	RValue, Offset
isdataat	Offset(오프셋)

관련 항목

DCE/RPC 전처리기
DCE/RPC 키워드, 73 페이지
기본 content 또는 protected_content 키워드 인수, 21 페이지
byte_jump 키워드, 33 페이지
byte_test 키워드, 36 페이지
패킷 특성, 97 페이지

byte_math 키워드

`byte_math` 키워드는 추출된 값과 지정된 값 또는 기존 변수에서 수학 연산을 수행하고, 결과를 새 결과 변수에 저장합니다. 그런 다음 결과 변수를 다른 키워드에서 인수로 사용할 수 있습니다.

규칙에서 여러 `byte_math` 키워드를 사용하여 여러 byte_math 연산을 수행할 수 있습니다.

다음 표에서는 `byte_math` 키워드에 필요한 인수를 설명합니다.

표 *23:* 필수 *byte_math* 인수

인수	설명
바이트	패킷에서 계산할 바이트 수. DCE/RPC 없이 사용하는 경우, 허용되는 값은 1 ~ 10입니다. • 연산자가 +, -인 경우, 바이트는 1 ~ 10일 수 있습니다. * 또는 /. • 연산자가 << 또는 >>인 경우, 바이트는 1 ~ 4일 수 있습니다. • 1, 2 또는 4 이외의 바이트 수를 지정하는 경우, 숫자 유형(16진수, 8진수 또는 10진수)을 지정해야 합니다. DCE/RPC와 함께 사용하는 경우, 허용되는 값은 1, 2, 4입니다.

인수	설명
Offset	처리를 시작할 페이로드에 들어가는 바이트 수. `offset` 카운터는 0바이트부터 시작되므로, `offset` 값을 계산할 때는 패킷 페이로드의 처음부터 또는 (Relative를 지정한 경우) 마지막으로 성공한 내용 일치로부터 앞으로 이동하려는 바이트 수에서 1을 빼야 합니다. -65535에서 65535까지 바이트를 지정할 수 있습니다. 여기서 `byte_extract` 변수를 지정할 수 있습니다.
연산자	+,-, *, /, <<, 또는 >>
RValue	연산자 다음에 오는 값입니다. 이 값은 부호 없는 정수이거나 `byte_extract`에서 전달된 변수일 수 있습니다.
Result Variable(결과 변수)	`byte_math` 계산 결과가 저장될 변수의 이름입니다. 이 변수를 다른 키워드에서 인수로 사용할 수 있습니다. 이 값은 부호 없는 정수로 저장됩니다. 이 변수의 이름: • 영숫자를 사용해야 합니다 • 숫자로 시작하면 안 됩니다 • Microsoft 파일 이름/변수 이름 규칙이 지원하는 특수 문자가 포함될 수 있습니다 • 특수 문자만으로 구성될 수 없습니다

다음 표에서는 시스템이 필수 인수에 지정한 값을 해석하는 방식을 정의하는 데 사용할 수 있는 옵션을 설명합니다.

표 ***24:*** 추가 옵션 ***byte_math*** 인수

인수	설명
Relative	오프셋이 페이로드의 시작 대신 마지막으로 성공한 콘텐츠 일치에서 발견된 마지막 패턴에 연결되도록 합니다.
Bitmask(비트마스크)	AND 연산자를 사용하여 지정된 16진수 비트마스크를 Bytes 인수에서 추출된 바이트에 적용합니다. 비트마스크는 1 ~ 4바이트일 수 있습니다. 결과는 마스크의 후행 0 수와 같은 비트 수만큼 오른쪽 시프트됩니다.

DCE/RPC, **Endian** 또는 **Number Type** 중 하나만 지정할 수 있습니다.

byte_math 키워드가 바이트를 계산하는 방법을 정의하려는 경우, 다음 표에 설명된 인수 중에서 선택할 수 있습니다. 바이트 순서 인수를 선택하지 않으면 규칙 엔진은 Big Endian 바이트 순서를 사용합니다.

표 ***25:*** 바이트 순서 ***byte_math*** 인수

인수	설명
Big Endian	빅 엔디언 바이트 순서의 프로세스 데이터. 기본 네트워크 바이트 순서입니다.
Little Endian	리틀 엔디언 순서의 프로세스 데이터
DCE/RPC	DCE/RPC 전처리기에서 처리된 트래픽에 byte_math 키워드를 지정합니다. DCE/RPC 전처리기는 빅 엔디언 또는 리틀 엔디언 바이트 순서를 결정하고, **Number Type** 및 **Endian** 인수는 적용되지 않습니다. 이 인수를 활성화하면 byte_math를 다른 특정 DCE/RPC 키워드와 함께 사용할 수 있습니다.

시스템이 다음 표에 있는 인수 중 하나를 사용하여 패킷 내 문자열을 보는 방식을 정의합니다.

표 ***26:*** 번호 유형 인수

인수	설명
Hexadecimal String	문자열 데이터를 16진수 형식으로 나타냅니다.
Decimal String	문자열 데이터를 10진수 형식으로 나타냅니다.
Octal String	문자열 데이터를 8진수 형식으로 나타냅니다.

예를 들어 byte_math에 설정하는 값이 다음과 같은 경우:

- Bytes = 2
- Offset = 0
- Operator = *
- RValue = height
- Result Variable = area

규칙 엔진은 패킷의 처음 2 바이트에 설명된 숫자를 추출하여 RValue(기존 변수 height를 사용하여 새 변수 area를 생성)로 곱합니다.

표 ***27: byte_math*** 변수를 받아들이는 인수

키워드	인수
byte_jump	Offset(오프셋)

키워드	인수
byte_test	Offset, Value
byte_extract	Offset(오프셋)
isdataat	Offset(오프셋)

개요: pcre 키워드

pcre 키워드를 사용하면 PCRE(Perl-compatible regular expression)를 사용하여 패킷 페이로드에서 지정된 내용을 검사할 수 있습니다. PCRE를 사용하여 동일한 콘텐츠의 약간의 차이에 따라 일치하는 여러 규칙을 작성하는 것을 방지할 수 있습니다.

정규 표현식은 다양한 방법으로 표시할 수 있는 콘텐츠를 검색할 때 유용합니다. 콘텐츠는 패킷의 페이로드에서 메시지를 찾는 시도에서 고려할 다른 특성을 가질 수 있습니다.

침입 규칙에서 사용되는 정규 표현식 구문은 전체 정규 표현식의 하위 집합이며, 전체 라이브러리의 지침에 사용되는 구문에서 어느 정도 변경된다는 점에 유의하십시오. 규칙 편집기를 사용하여 pcre 키워드를 추가할 경우, 전체 값을 다음 형식으로 입력합니다.

```
!/pcre/ ismxAEGRBUIPHDMCKSY
```

여기에서 각 항목은 다음을 나타냅니다.

- !는 선택적 무효화입니다(정규 표현식에 일치하지 않는 패턴에 일치시키기를 원할 때 이를 사용합니다).
- /pcre/는 Perl 호환 정규 표현식입니다.
- ismxAEGRBUIPHDMCKSY 수식자 옵션의 모든 조합입니다.

패킷 페이로드에서 특정 콘텐츠를 검색하기 위해 PCRE에서 이들을 사용할 때 규칙 엔진이 정확하게 해석할 수 있도록 하려면 다음 표에 나열된 문자를 이스케이프해야 한다는 점에 유의하십시오.

표 ***28:*** 이스케이프된 ***PCRE*** 문자

이스케이프 대상	백슬래시	헥사 코드
#(해시 부호)	\#	\x23
;(세미콜론)	\;	\x3B
|(세로 선)	\\|	\x7C
:(콜론)	\:	\x3A

또한 /가 아닌 ?가 구분 문자일 때 m?regex?를 사용할 수 있습니다. 정규 표현식에서 슬래시와 일치해야 하는 상황에서 이를 사용하고자 할 수도 있지만 백슬래시로 이스케이프하지 않습니다. 예를 들어

m?regex? ismxAEGRBUIPHDMCKSY를 사용하려는 경우, regex는 Perl 호환 정규식이고 ismxAEGRBUIPHDMCKSY는 수정자 옵션의 조합입니다.

팁 원하는 경우, Perl 호환 정규 표현식을 따옴표로 묶을 수 있습니다(예: pcre_expression 또는 “pcre_expression“) . 따옴표 사용 옵션은 따옴표가 선택 사항이 아닌 필수인 경우 이전 버전에 익숙한 기존 사용자에게 제공됩니다. 침입 규칙 편집기는 보고서를 저장한 후 규칙을 표시할 때 따옴표를 표시하지 않습니다.

pcre 구문

pcre 키워드는 표준 Perl 호환 정규 표현식(PCRE) 구문을 허용합니다. 다음 섹션에서는 각 구문에 대해 설명합니다.

팁 이 섹션은 PCRE에 사용할 수 있는 기본 구문을 설명하지만, 더 많은 고급 정보를 Perl(펄) 및 PCRE에 할애하는 온라인 참조 또는 도서를 참고할 수 있습니다.

메타 문자

메타 문자는 정규 표현식에서 특정 의미가 있는 리터럴 문자입니다. 정규 표현식에서 이를 사용할 때, 이들 앞에 백슬래시를 두어 "이스케이프"해야 합니다.

다음 표에서는 PCRE에 사용할 수 있는 메타 문자를 설명하고 각각의 예를 제공합니다.

표 ***29: PCRE*** 메타 문자

메타 문자	설명	예
.	줄 바꿈을 제외한 모든 문자와 일치합니다. s가 수정 옵션으로 사용되는 경우, 이는 또한 줄 바꿈 문자를 포함합니다.	abc.는 abcd, abc1, abc# 등에 일치시킵니다.
*	0 이상의 문자 또는 표현이 나타나는 것에 일치시킵니다.	abc*는 abc, abcc, abccc, abcccccc 등에 일치시킵니다.
?	0 또는 하나의 문자/표현이 나타나는 것에 일치시킵니다.	abc?는 abc에 일치시킵니다.
+	하나 이상의 문자 또는 표현이 나타나는 것에 일치시킵니다.	abc+는 abc, abcc, abccc, abcccccc 등에 일치시킵니다.
()	표현을 그룹화합니다.	(abc)+는 abc, abcabc, abcabcabc 등에 일치시킵니다.
{}	문자 또는 표현에 일치하는 수에 대한 한계를 지정합니다. 상한 및 하한을 설정하고자 하는 경우, 쉼표로 상한 및 하한을 구분합니다.	a{4,6}은 aaaa, aaaaa 또는 aaaaaa에 일치시킵니다. (ab){2}는 abab에 일치시킵니다.

메타 문자	설명	예
`[]`	문자 클래스를 정의하도록 허용하며, 집합에 설명된 문자의 조합 또는 모든 문자에 일치시킵니다.	`[abc123]`은 `a` 또는 `b` 또는 `c` 등에 일치시킵니다.
`^`	문자열의 시작 지점에 있는 콘텐츠에 일치시킵니다. 문자 클래스 내에서 사용되는 경우에도 무효화에 사용됩니다.	`^in`은 `info` 내 "in"에 일치하지만, `bin`에서는 일치하지 않습니다. `[^a]`은 `a`를 포함하지 않는 모든 문자열에 일치합니다.
`$`	문자열이 끝나는 지점에 있는 콘텐츠에 일치시킵니다.	`ce$`는 `announce` 내 "`ce`"에 일치하지만, `cent`에서는 일치하지 않습니다.
`|`	또는(OR) 표현을 나타냅니다.	`(MAILTO|HELP)`는 `MAILTO` 또는 `HELP`에 일치시킵니다.
`\`	메타 문자를 실제 문자로 사용할 수 있으며, 미리 정의된 문자 클래스를 지정하는 데 사용할 수도 있습니다.	`\.`는 기간에 일치하고, `\*`는 별표에 일치하며, `\\`는 백슬래시에 일치합니다. `\d`는 숫자에 일치하며, `\w`는 영숫자에 일치합니다.

문자 클래스

문자 클래스는 알파벳 문자, 영숫자, 숫자 및 공백 문자를 포함합니다. 괄호 안에서 자체 문자 클래스를 만들 수도 있지만 다른 유형의 문자 유형에 대한 바로 가기로 사전 정의된 클래스를 사용할 수도 있습니다. 추가 수식자 없이 사용할 경우, 문자 클래스는 한 자릿수 또는 문자와 일치됩니다.

다음 표에서는 PCRE가 수용한 사전 정의된 문자 클래스의 예를 설명하고 제공합니다.

표 ***30: PCRE*** 문자 클래스

문자 클래스	설명	문자 클래스 정의
\d	숫자 문자("디지트")에 일치합니다.	[0-9]
\D	숫자 문자가 아닌 모든 문자에 일치합니다.	[^0-9]
\w	영숫자 문자("단어")에 일치합니다.	[a-zA-Z0-9_]
\W	영숫자 문자가 아닌 모든 문자에 일치합니다.	[^a-zA-Z0-9_]
\s	공백, 복귀, 탭, 줄 바꿈 및 서식 이송을 포함하여 공백 문자에 일치시킵니다.	[\r\t\n\f]
\S	공백 문자가 아닌 모든 문자에 일치시킵니다.	[^ \r\t\n\f]

pcre 수식자 옵션

`pcre` 키워드 값의 정규 표현식 구문을 지정한 후 변경 옵션을 사용할 수 있습니다. 이러한 수정자는 Perl, PCRE 및 Snort 관련 처리 기능을 수행합니다. 수식자는 언제나 PCRE 값의 끝에 표시되며, 다음과 같은 형식으로 표시됩니다.

/pcre/ismxAEGRBUIPHDMCKSY

이 경우 ismxAEGRBUPHMC가 다음 표에 나타나는 모든 수정 옵션을 포함할 수 있습니다.

팁 또는, 정규 표현식 및 모든 수정 옵션을 따옴표로 둘러쌀 수 있습니다(예: "/pcre/ismxAEGRBUIPHDMCKSY"). 따옴표 사용 옵션은 따옴표가 선택 사항이 아닌 필수인 경우 이전 버전에 익숙한 기존 사용자에게 제공됩니다. 침입 규칙 편집기는 보고서를 저장한 후 규칙을 표시할 때 따옴표를 표시하지 않습니다.

다음 표에서는 사용자가 Perl 처리 기능을 수행하는 데 사용할 수 있는 옵션을 설명합니다.

표 ***31: Perl*** 관련 게시물 정규 표현식 옵션

옵션	설명
i	정규 표현식에서 대소문자를 구분하지 않도록 합니다.
s	점 문자(.)는 줄 바꿈 또는 \n 문자를 제외한 모든 문자를 설명합니다. "s" 를 옵션으로 사용하여 이를 무시할 수 있으며, 점 문자가 줄 바꿈 문자를 포함한 모든 문자에 일치하도록 할 수 있습니다.
m	기본적으로, 문자열은 문자의 단선으로 처리되며, ^ 및 $는 특정 문자열의 시작 및 끝 지점에 일치됩니다. "m"을 옵션으로 사용할 때, ^ 및 $는 버퍼의 시작 또는 끝부분뿐만 아니라 버퍼에서 모든 줄 바꿈 문자 바로 앞 또는 바로 뒤 콘텐츠에 일치시킵니다.
x	패턴에 나타날 수 있는 공백 데이터 문자를 무시합니다. 이스케이프 되었을 때(앞에 백슬래시가 있을 때) 또는 문자 클래스 안에 포함되었을 때는 제외합니다.

다음 표에서는 정규 표현식 뒤에 사용할 수 있는 PCRE 수식자를 설명합니다.

표 ***32: PCRE*** 관련 게시물 정규 표현식 옵션

옵션	설명
A	패턴은 문자열의 시작 지점과 일치해야 합니다(정규 표현식의 ^를 사용하는 것과 동일).
E	$가 제목 문자열 끝에만 일치하도록 설정합니다. (마지막 문자가 줄바꿈인 경우 E가 없는 $는 또한 마지막 문자 바로 앞에 일치하지만 다른 모든 줄바꿈 문자 앞에서는 일치하지 않습니다).
G	기본적으로, * + 및 ?는 "최대값에 일치"시킵니다. 이는 두 개 이상의 일치가 발견되는 경우, 가장 긴 일치 항목을 선택한다는 것을 의미합니다. 이를 변경하려면 G 문자를 사용하며, 이는 그 뒤에 물음표 문자(?)가 나오지 않는 한 이 문자들이 항상 첫 번째 일치 항목을 선택하도록 하기 위한 것입니다. 예를 들면 *? +? 및 ??는 G 수식자를 사용하는 구조에서 최대값에 일치시킵니다. 그리고 추가적인 물음표가 없는 *, + 또는 ?의 모든 예는 최대값에 일치시키지 않습니다.

다음 표에서는 정규 표현식 뒤에 사용할 수 있는 Snort 특정 수식자를 설명합니다.

.

표 ***33: Snort*** 특정 게시물 정규 표현식 수식자

옵션	설명
R	규칙 엔진으로 발견된 마지막 일치 끝부분과 관련된 일치하는 콘텐츠를 검색합니다.
B	전처리기에서 해독하기 전에 데이터 내의 내용을 검색합니다(이 옵션은 `Raw Data` 인수와 `content` 또는 `protected_content` 키워드를 함께 사용하는 것과 유사합니다).
U	HTTP 검사 전처리기에서 해독된 표준화된 HTTP 요청 메시지에 대해 URI 내의 콘텐츠를 검색합니다. 이 옵션은 `content` 또는 `protected_content` 키워드 **HTTP URI** 옵션과 함께 동일한 내용을 검색하는 데 사용할 수 없습니다. 파이프라인된 HTTP 요청 패킷에는 여러 URI가 포함되어 있습니다. U 옵션을 포함하는 PCRE 표현은 규칙 엔진이 파이프라인 방식 HTTP 요청 패킷의 첫 URI에서만 콘텐츠 일치를 검색하도록 합니다. 패킷의 모든 URI를 검색하려면 U 옵션을 사용하는 동반 PCRE 식의 유무와 상관없이 `content` 또는 `protected_content` 키워드와 함께 **HTTP URI**를 선택하십시오.
I	HTTP 검사 전처리기에서 해독된 원시 HTTP 요청 메시지에 대해 URI 내의 콘텐츠를 검색합니다. 이 옵션은 `content` 또는 `protected_content` 키워드 **HTTP Raw URI** 옵션과 조합하여 동일한 내용을 검색하는 데 사용할 수 없습니다.
P	HTTP 검사 전처리기에서 해독된 표준화된 HTTP 요청 메시지에 대해 본문 내의 콘텐츠를 검색합니다.
H	HTTP 검사 전처리기에서 해독된 HTTP 요청 또는 응답 메시지의 헤더 내 콘텐츠를 검색합니다. 쿠키는 제외합니다. 이 옵션은 `content` 또는 `protected_content` 키워드 **HTTP Header** 옵션과 함께 동일한 내용을 검색하는 데 사용할 수 없습니다.
D	HTTP 검사 전처리기에서 해독된 원시 HTTP 요청 또는 응답 메시지의 헤더 내 콘텐츠를 검색합니다. 쿠키는 제외합니다. 이 옵션은 `content` 또는 `protected_content` 키워드 **HTTP Raw Header** 옵션과 함께 동일한 내용을 검색하는 데 사용할 수 없습니다.
M	HTTP 검사 전처리기에서 해독된 표준화된 HTTP 요청 메시지의 메서드 필드에서 콘텐츠를 검색합니다. 메서드 필드는 URI에서 식별된 리소스를 만들기 위해 GET, PUT, CONNECT 등의 작업을 식별합니다.

옵션	설명
C	HTTP 검사 전처리기 **Inspect HTTP Cookies(HTTP** 쿠키 검사**)** 옵션이 활성화되면 HTTP 요청 헤더의 모든 쿠키 내 표준화된 콘텐츠를 검색합니다. 전처리기 **Inspect HTTP Responses(HTTP** 응답 검사**)** 옵션이 활성화되면 HTTP 응답 헤더의 모든 set-cookie 내 표준화된 콘텐츠도 검색합니다. **Inspect HTTP Cookies(HTTP** 쿠키 검사**)**가 활성화되지 않은 경우, 쿠키 또는 set-cookie 데이터를 포함한 전체 헤더를 검색합니다. 다음 사항을 참고하십시오. • 메시지 본문에 포함된 쿠키는 본문 콘텐츠로 처리됩니다. • 이 옵션은 `content` 또는 `protected_content` 키워드 **HTTP Cookie** 옵션과 함께 동일한 내용을 검색하는 데 사용할 수 없습니다. • `Cookie:` 및 `Set-Cookie:` 헤더 이름, 헤더 행의 주요 스페이스 및 헤더 행을 종료하는 `CRLF`는 헤더의 일부로 검사되지만 쿠키의 일부로는 검사되지 않습니다.
K	HTTP 검사 전처리기 **Inspect HTTP Cookies(HTTP** 쿠키 검사**)** 옵션이 활성화되면 HTTP 요청 헤더의 모든 쿠키 내의 원시 콘텐츠를 검색합니다. 전처리기 **Inspect HTTP Responses(HTTP** 응답 검사**)** 옵션이 활성화되면 HTTP 응답 헤더의 모든 set-cookie 내 표준화된 콘텐츠를 검색합니다. **Inspect HTTP Cookies(HTTP** 쿠키 검사**)**가 활성화되지 않은 경우, 쿠키 또는 set-cookie 데이터를 포함한 전체 헤더를 검색합니다. 다음 사항을 참고하십시오. • 메시지 본문에 포함된 쿠키는 본문 콘텐츠로 처리됩니다. • 이 옵션은 `content` 또는 `protected_content` 키워드 **HTTP Raw Cookie** 옵션과 함께 동일한 내용을 검색하는 데 사용할 수 없습니다. • `Cookie:` 및 `Set-Cookie:` 헤더 이름, 헤더 행의 주요 스페이스 및 헤더 행을 종료하는 `CRLF`는 헤더의 일부로 검사되지만 쿠키의 일부로는 검사되지 않습니다.
S	HTTP 응답에서 3자리 상태 코드를 검색합니다.
Y	HTTP 응답의 상태 코드를 동반하는 텍스트 설명을 검색합니다.

참고 U 옵션을 R 옵션과 함께 사용하지 마십시오. 이는 성능 문제를 야기할 수 있습니다. 또한, U 옵션을 다른 HTTP 콘텐츠 옵션(I, P, H, D, M, C, K, S, 또는 Y)과 함께 사용하지 마십시오.

관련 항목

개요: HTTP content 및 protected_content 키워드 인수, 25 페이지

pcre 예시 키워드 값

다음의 예시는 일치하는 각 예의 설명과 함께 pcre에 입력할 수 있는 값을 보여줍니다.

- **/feedback[(\d{0,1})]?\.cgi/U**

이 예는 feedback 뒤에 0 또는 1개의 숫자 문자와 .cgi가 차례로 나오며 URI 데이터에만 위치하는 feedback에 대한 패킷 페이로드를 검색합니다.

이 예는 다음에 일치됩니다.

- feedback.cgi
- feedback1.cgi
- feedback2.cgi
- feedback3.cgi

이 예는 다음과 일치되지 않습니다.

- feedbacka.cgi
- feedback11.cgi
- feedback21.cgi
- feedbackzb.cgi

- **/^ez(\w{3,5})\.cgi/iU**

이 예제는 패킷 페이로드에서 문자열의 처음에 나오는 ez, 그 뒤에 3~5개의 문자로 구성된 단어, 그리고 그 뒤에 .cgi를 검색합니다. 검색은 대소문자를 구분하지 않으며 URI 데이터만 검색합니다.

이 예는 다음에 일치됩니다.

- EZBoard.cgi
- ezman.cgi
- ezadmin.cgi
- EZAdmin.cgi

이 예는 다음과 일치되지 않습니다.

- ezez.cgi
- fez.cgi
- abcezboard.cgi
- ezboardman.cgi

- **/mail(file|seek)\.cgi/U**

이 예는 mail 뒤에 file 또는 seek이 나오며 URI 데이터에 있는 mail에 대한 패킷 페이로드를 검색합니다.

이 예는 다음에 일치됩니다.

- mailfile.cgi
- mailseek.cgi

이 예는 다음과 일치되지 않습니다.

- MailFile.cgi
- mailfilefile.cgi

- **m?http\\x3a\x2f\x2f.*(\n|\t)+?U**

이 예는 HTTP 요청의 탭 또는 줄 바꿈 문자에 대한 URI 콘텐츠의 패킷 페이로드를 검색하며, 어떤 수의 문자가 앞에 와도 됩니다. 이 예는 m?regex?를 사용하여 표현식에서 http\:\/\/를 사용하는 것을 방지합니다. 콜론은 백슬래시 앞에 위치한다는 점을 참고하십시오.

이 예는 다음에 일치됩니다.

- http://www.example.com?scriptvar=x&othervar=\n\..\..
- http://www.example.com?scriptvar=\t

이 예는 다음과 일치되지 않습니다.

- ftp://ftp.example.com?scriptvar=&othervar=\n\..\..
- http://www.example.com?scriptvar=|/bin/sh -i|

- **m?http\\x3a\x2f\x2f.*=\|.*\|+?sU**

이 예에서는 등호 및 모든 수의 문자를 포함하는 파이프 문자 또는 공백이 뒤따르는 줄 바꿈을 포함하는 모든 문자 수와 함께 URL에 대한 패킷 페이로드를 검색합니다. 이 예는 m?regex?를 사용하여 표현식에서 http\:\/\/를 사용하는 것을 방지합니다.

이 예는 다음에 일치됩니다.

- http://www.example.com?value=|/bin/sh/ -i|
- http://www.example.com?input=|cat /etc/passwd|

이 예는 다음과 일치되지 않습니다.

- ftp://ftp.example.com?value=|/bin/sh/ -i|
- http://www.example.com?value=x&input?|cat /etc/passwd|

- **/[0-9a-f]{2}\:[0-9a-f]{2}\:[0-9a-f]{2}\:[0-9a-f]{2}\:[0-9a-f]{2}\:[0-9a-f]{2}/i**

이 예는 MAC 주소에 대한 패킷 페이로드를 검색합니다. 백슬래시로 콜론 문자를 이스케이프한다는 점에 유의하십시오.

metadata 키워드

`metadata` 키워드를 사용하여 규칙에 자체 설명 정보를 추가할 수 있습니다. `metadata` 키워드를 `service` 인수와 함께 사용하여 네트워크 트래픽에서 애플리케이션과 포트를 식별할 수도 있습니다. 추가하는 정보를 사용하여 필요에 맞게 규칙을 구성하거나 식별할 수 있고, 추가하는 정보와 `service` 인수를 규칙에서 검색할 수 있습니다.

시스템은 인수 형식을 기반으로 메타데이터를 검증합니다.

key value

*key*와*value*가 스페이스로 구분된 결합 설명을 제공합니다. 이것은 Cisco에서 제공하는 규칙에 메타데이터를 추가하기 위해 Talos 인텔리전스 그룹에서 사용하는 형식입니다.

또는, 다음 형식을 사용할 수 있습니다.

key = value

예를 들어 *key value* 형식에서 다음과 같이 카테고리 및 하위 카테고리를 사용하여 작성자와 날짜별로 규칙을 식별할 수 있습니다.

```
author SnortGuru_20050406
```

규칙에서 여러 `metadata` 키워드를 사용할 수 있습니다. 다음 예에 나온 것처럼 쉼표를 사용하여 단일 `metadata` 키워드에서 여러 *key value* 인수를 구분할 수 있습니다.

```
author SnortGuru_20050406, revised_by SnortUser1_20050707,
revised_by SnortUser2_20061003,
revised_by SnortUser1_20070123
```

key value 또는 *key=value* 형식만 사용하도록 제한되지 않습니다. 그러나 이러한 형식 기반의 검증에서 오는 제한 사항에 대해 알고 있어야 합니다.

피해야 할 제한된 문자

다음과 같은 문자 제한 사항을 참고하십시오.

- 세미콜론(;) 또는 콜론(:)을 사용하지 마십시오.
- 시스템은 쉼표를 여러 *key value* 또는 *key=value* 인수의 구분자로 해석합니다. 예를 들면 다음과 같습니다.

  key value, *key value*, *key value*

- 시스템은 등호(=) 문자 또는 공백 문자를 *key* 와 *value* 사이의 구분자로 해석합니다. 예를 들면 다음과 같습니다.

  key value

  key=value

다른 모든 문자는 허용됩니다.

피해야 할 예약된 메타데이터

다음 단어는 Talos에서 사용하도록 예약되어 있으므로 `metadata` 키워드에서 단일 인수나 ***key value*** 인수의 ***key***로 사용하지 마십시오.

```
application
engine
impact_flag
os
policy
rule-type
rule-flushing
soid
```

참고 예상과 다르게 기능했을 수도 있는 로컬 규칙에 제한된 메타데이터를 추가하는 것에 관한 도움을 받으려면 Support(지원부)에 연락하십시오.

영향 레벨 **1**

`metadata` 키워드에서 다음의 예약된 ***key value*** 인수를 사용할 수 있습니다.

```
impact_flag red
```

key value 인수는 가져온 로컬 규칙 또는 침입 규칙 편집기를 사용하여 생성한 맞춤형 규칙에 대해 영향 플래그를 red(레벨 1)로 설정합니다.

Talos가 Cisco에서 제공하는 규칙에 `impact_flag red` 인수를 포함하는 경우, 규칙을 트리거한 패킷이 소스 또는 대상 호스트가 바이러스, 트로이 목마 또는 기타 악성 소프트웨어에 의해 손상되었을 가능성이 있음을 나타내는 것이라고 Talos에서 판단한 것입니다.

서비스 메타데이터

시스템은 네트워크에 있는 호스트에서 실행 중인 애플리케이션을 탐지하고 애플리케이션 프로토콜 정보를 네트워크 트래픽에 삽입합니다. 시스템은 검색 정책의 구성에 관계없이 이 작업을 수행합니다. TCP 또는 UDP 규칙에서 `metadata` 키워드 `service` 인수를 사용하여 네트워크 트래픽에서 애플리케이션 프로토콜 및 포트를 매칭할 수 있습니다. 규칙의 하나 이상의 `service` 애플리케이션 인수를 단일 포트 인수와 조합할 수 있습니다.

서비스 애플리케이션

`metadata` 키워드를 `service`와 함께 ***key***로 사용하고 애플리케이션을 ***value***로 사용하여 패킷을 식별된 애플리케이션 프로토콜과 매칭할 수 있습니다. 예를 들어 다음 `metadata` 키워드 안의 ***key value*** 인수는 HTTP 트래픽과 규칙을 연결합니다.

```
service http
```

쉼표로 구분된 여러 애플리케이션을 식별할 수 있습니다. 예를 들면 다음과 같습니다.

```
service http, service smtp, service ftp
```

주의 침입 규칙이 서비스 메타데이터를 사용하려면 적응형 프로파일 구성에 설명된 대로 적응형 프로파일링이 반드시 활성화되어야 합니다(기본 상태).

다음 표에서는 `service` 키워드와 함께 사용되는 가장 일반적인 애플리케이션 값을 설명합니다.

참고 표에 없는 애플리케이션 식별에 어려움이 있다면 지원 부서에 문의하십시오.

표 ***34:*** 서비스 값

값	설명
`cvs`	Concurrent Versions System
`dcerpc`	분산 컴퓨팅 환경/원격 절차 호출 시스템
`dns`	Domain Name System
`finger`	핑거 사용자 정보 프로토콜
`ftp`	파일 전송 프로토콜
`ftp-data`	파일 전송 프로토콜(데이터 채널)
`http`	Hypertext Transfer Protocol
`imap`	Internet Message Access Protocol
`isakmp`	Internet Security Association and Key Management Protocol
`mysql`	My Structured Query Language
`netbios-dgm`	NETBIOS 데이터그램 서비스
`netbios-ns`	NETBIOS 이름 서비스
`NetBIOS ssn`	NETBIOS 세션 서비스
`nntp`	Network News Transfer Protocol
`oracle`	Oracle Net Services
`shell`	OS 셸
`pop2`	포스트 오피스 프로토콜, 버전 2
`POP3`	포스트 오피스 프로토콜, 버전 3
`smtp`	간단한 메일 전송 프로토콜

값	설명
snmp	Simple Network Management Protocol
ssh	Secure Shell 네트워크 프로토콜
sunrpc	Sun Remote Procedure Call Protocol
telnet	텔넷 네트워크 프로토콜
tftp	Trivial File Transfer Protocol
x11	X 윈도우 시스템

서비스 포트

metadata 키워드를 service와 함께 ***key***로, 지정된 포트 인수를 ***value***로 사용하여 규칙이 애플리케이션과 조합된 포트를 매칭하는 방법을 정의할 수 있습니다.

아래 표에서 규칙당 하나씩 포트 값을 지정할 수 있습니다.

표 ***35:*** 서비스 포트 값

값	설명
else-ports 또는 unknown	다음 조건 중 하나가 충족되는 경우 시스템이 규칙을 적용합니다. • 패킷 애플리케이션이 알려져 있고 규칙 애플리케이션과 일치합니다. • 패킷 애플리케이션이 알려져 있지 않고 패킷 포트가 규칙 포트와 일치합니다. else-ports 및 unknown 값은 service가 포트 수식자 없는 애플리케이션 프로토콜을 지정할 때 시스템이 사용하는 기본 동작을 생성합니다.
and-ports	시스템은 패킷 애플리케이션이 알려져 있고 규칙 애플리케이션과 일치하며 패킷 포트가 규칙 헤더의 포트와 일치하는 경우, 규칙을 적용합니다. 애플리케이션을 지정하지 않는 규칙에서는 and-ports를 사용할 수 없습니다.
or-ports	다음 조건 중 하나가 충족되는 경우 시스템이 규칙을 적용합니다. • 패킷 애플리케이션이 알려져 있고 규칙 애플리케이션과 일치합니다. • 패킷 애플리케이션이 알려져 있지 않고 패킷 포트가 규칙 포트와 일치합니다. • 패킷 애플리케이션이 규칙 애플리케이션과 일치하지 않고 패킷 포트가 규칙 포트와 일치합니다. • 규칙이 애플리케이션을 지정하지 않고 패킷 포트가 규칙 포트와 일치합니다.

다음에 유의하십시오.

- `service` 애플리케이션 인수와 함께 `service and-ports` 인수를 포함해야 합니다.
- 규칙이 위의 표의 값 중 둘 이상을 지정하는 경우, 시스템은 규칙에 표시되는 마지막 값을 적용합니다.
- 포트 인수와 애플리케이션 인수의 순서는 상관없습니다.

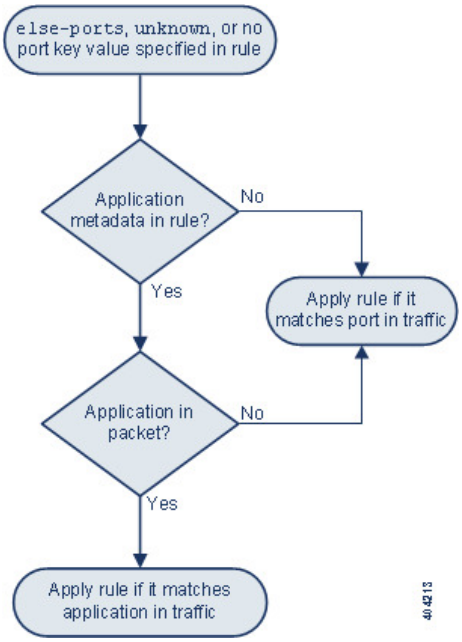
`and-ports` 값을 제외하고 하나 이상의 `service` 애플리케이션 인수와 함께 또는 없이 `service` 포트 인수를 포함시킬 수 있습니다. 예를 들면 다음과 같습니다.

```
service or-ports, service http, service smtp
```

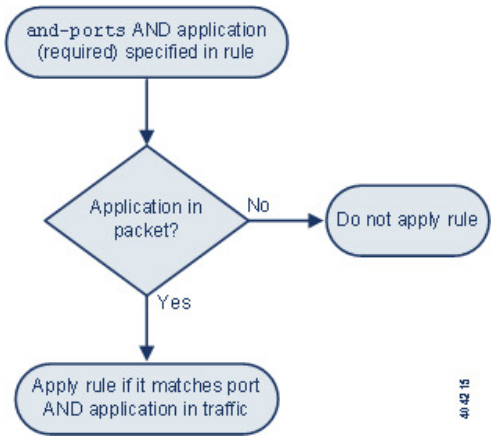
트래픽의 애플리케이션 및 포트

아래 다이어그램은 침입 규칙이 지원하는 애플리케이션 및 포트 조합과 이러한 규칙 제약 조건을 패킷 데이터에 적용한 결과를 보여줍니다.

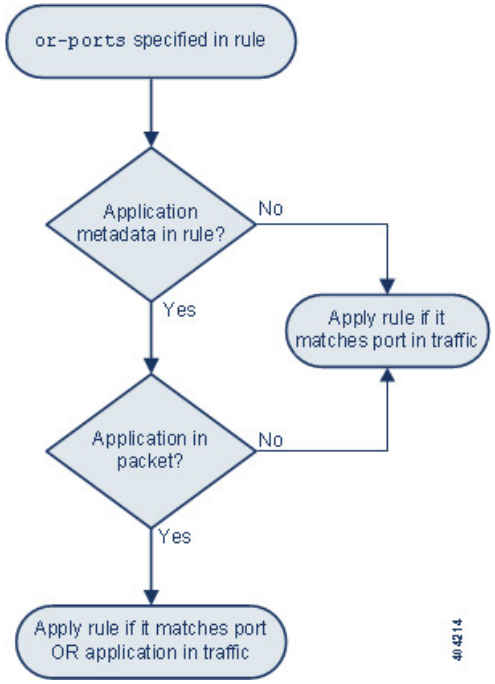
호스트 애플리케이션 프로토콜 **else source/destination** 포트**:**



호스트 애플리케이션 프로토콜 **and source/destination** 포트**:**



호스트 애플리케이션 프로토콜 **or source/destination** 포트**:**



예제 일치

`metadata` 키워드와 함께 `service` 인수를 사용하는 다음 샘플 규칙은 일치하는 데이터 및 일치하지 않는 데이터의 예와 함께 표시되어 있습니다.

- `alert tcp any any -> any [80,8080] (metadata:service and-ports, service http, service smtp;)`

예제 일치	비일치 예
• TCP 포트 80을 통한 HTTP 트래픽 • TCP 포트 8080을 통한 HTTP 트래픽 • TCP 포트 80을 통한 SMTP 트래픽 • TCP 포트 8080을 통한 SMTP 트래픽	• 포트 80 또는 8080의 POP3 트래픽 • 포트 80 또는 8080의 알 수 없는 애플리케이션 트래픽 • 9999 포트의 HTTP 트래픽

- `alert tcp any any -> any [80,8080] (metadata:service or-ports, service http;)`

예제 일치	비일치 예
• 모든 포트의 HTTP 트래픽 • 포트 80의 SMTP 트래픽 • 포트 8080의 SMTP 트래픽 • 포트 80과 8080의 알 수 없는 애플리케이션 트래픽	• 80 또는 8080 외의 포트의 비HTTP 및 비SMTP 트래픽

- 다음 규칙 중 하나:
  - `alert tcp any any -> any [80,8080] metadata:service else-ports, service http;)`
  - `alert tcp any any -> any [80,8080] metadata:service unknown, service http;)`
  - `alert tcp any any -> any [80,8080] metadata:service http;)`

예제 일치	비일치 예
• 모든 포트의 HTTP 트래픽 • 패킷 애플리케이션을 알 수 없는 경우 포트 80 • 패킷 애플리케이션을 알 수 없는 경우 포트 8080	• 포트 80 또는 8080의 SMTP 트래픽 • 포트 80 또는 8080의 POP3 트래픽

메타데이터 검색 가이드라인

`metadata` 키워드를 사용하는 규칙을 검색하려면 규칙 Search(검색) 페이지에서 `metadata` 키워드를 선택하고, 선택적으로 메타데이터의 일부를 입력합니다. 예를 들어 다음을 입력할 수 있습니다.

- `search`를 입력하면 ***key***에 `search`를 사용한 모든 규칙이 표시됩니다.
- `search http`를 입력하면 ***key***에 `search`를 사용하고 ***value***에 `http`를 사용한 모든 규칙이 표시됩니다.

- `author snortguru`를 입력하면 *key*에 `author`를 사용하고 *value*에 `SnortGuru`를 사용한 모든 규칙이 표시됩니다.
- `author s`를 입력하면 *key*에 `author`를 사용하고, *value*에 `SnortGuru` 또는 `SnortUser1` 또는 `SnortUser2` 같은 용어를 사용한 모든 규칙이 표시됩니다.

팁 *key*와 *value*를 모두 검색하는 경우, 검색에서 규칙의 *key value* 인수에 사용되는 것과 동일한 연결 연산자(등호[=] 또는 공백 문자)를 사용하십시오. *key* 뒤에 등호(=)를 사용하는지 공백 문자를 사용하는지에 따라 검색에서 다른 결과가 반환됩니다.

메타데이터를 추가하는 데 사용하는 형식과 상관없이 시스템은 메타데이터 검색 용어를 *key value* 또는 *key=value* 인수의 전체 또는 일부로 해석합니다. 예를 들어 다음은 *key value* 또는 *key=value* 형식을 따르지 않는 유효한 메타데이터일 수 있습니다.

```
ab cd ef gh
```

그러나 시스템은 예제의 각 공백을 *key*와 *value* 사이의 구분 기호로 해석합니다. 따라서 병렬 용어 및 단일 용어에 대해 다음 검색 중 하나를 사용하여 예제 메타데이터를 포함하는 규칙을 성공적으로 찾을 수 있습니다.

```
cd ef
ef gh
ef
```

그러나 다음 검색을 사용하면 규칙을 찾을 수 없습니다. 시스템이 이를 단일 *key value* 인수로 해석하기 때문입니다.

```
ab ef
```

관련 항목

규칙 검색, 14 페이지

IP 헤더 값

키워드를 사용하여 패킷의 IP 헤더 내 가능한 공격 또는 보안 정책 위반을 식별할 수 있습니다.

fragbits

`fragbits` 키워드는 IP 헤더의 단편 및 예약 비트를 검사합니다. Reserved Bit(예약 비트), More Fragments bit(추가 단편 비트)에서 각 패킷을 확인할 수 있으며, 어떤 조합에서나 Don't Fragment bit(단편화 금지 비트)를 확인할 수 있습니다.

표 ***36:*** 단편 비트 인수 값

인수	설명
R	예약 비트

인수	설명
M	추가 단편 비트
D	단편화 금지 비트

fragbits 키워드를 사용하여 규칙을 개선하기 위해 규칙에서 인수 값 다음에 다음 표에 설명된 모든 연산자를 지정할 수 있습니다.

표 ***37:*** 단편 비트 연산자

연산자	설명
더하기 기호(**+**)	패킷은 모든 지정된 비트와 일치해야 합니다.
별표(*****)	패킷은 모든 지정된 비트에 일치할 수 있습니다.
느낌표(**!**)	지정된 비트가 하나도 설정되지 않은 경우 패킷은 기준을 충족합니다.

예를 들어, 설정된 Reserved Bit(예약 비트) (및 가능한 다른 모든 비트)가 있는 패킷에 대한 이벤트를 생성하려면, fragbits 값으로 R+를 사용합니다.

id

ID 키워드는 키워드의 인수에 지정된 값에 대한 IP 헤더 단편 ID 필드를 테스트합니다. 일부 서비스 거부 툴 및 스캐너는 이 필드를 탐지하기 쉬운 특정 번호로 설정합니다. 예를 들어, Synscan 포트스캔을 탐지하는 SID 630에서는 ID 값이 스캐너에서 전송하는 패킷에 ID 번호로 사용된 정적 값 39426으로 설정됩니다.

참고 ID의 값은 숫자여야 합니다.

ipopts

IPopts 키워드를 사용하면 지정된 IP 헤더 옵션을 위한 패킷을 검색할 수 있습니다. 다음 표에는 사용 가능한 인수 값이 나열되어 있습니다.

표 ***38: IPoption*** 인수

인수	설명
rr	레코드 경로
eol	목록 끝
nop	무연산
ts	타임 스탬프

인수	설명
sec	IP 보안 옵션
lsrr	느슨한 소스 라우팅
ssrr	엄격한 소스 라우팅
satid	스트림 식별자

분석가가 가장 자주 살펴보는 것은 엄격한 소스 라우팅과 느슨한 소스 라우팅입니다. 그 이유는 이 옵션이 도용된 소스 IP 주소를 나타낼 수 있기 때문입니다.

ip_proto

`ip_proto` 키워드를 사용하면 키워드 값으로 지정된 IP 프로토콜을 통해 패킷을 식별할 수 있습니다. 0에서 255까지의 번호로 IP 프로토콜을 지정할 수 있습니다. 이 번호를 `<`, `>`, 또는`!` .연산자와 결합할 수 있습니다. 예를 들어, ICMP가 아닌 모든 프로토콜을 통해 트래픽을 검사하려면, `ip_proto` 키워드에 대한 값으로 `!1`을 사용합니다. 또한 단일 규칙에서 `ip_proto` 키워드를 여러 번 사용할 수 있습니다. 하지만, 규칙 엔진은 키워드의 여러 인스턴스를 Boolean AND 관계를 갖는 것으로 해석한다는 점에 유의하십시오. 예를 들어, `ip_proto:!3; ip_proto:!6`을 포함하는 규칙을 생성하는 경우, 규칙은 GGP 프로토콜 및 TCP 프로토콜을 사용하는 트래픽을 무시합니다.

tos

일부 네트워크는 서비스 유형(ToS) 값을 사용하여 네트워크로 이동하는 패킷의 우선 순위를 설정합니다. `tos` 키워드는 키워드의 인수로 지정한 값에 대한 패킷의 IP 헤더 ToS 값을 테스트합니다. `tos` 키워드를 사용하는 규칙은 해당 ToS가 특정 값으로 설정된 패킷과 규칙에서 제시된 기준의 나머지 부분을 충족하는 패킷에서 트리거됩니다.

참고 `tos`의 인수 값은 숫자여야 합니다.

ToS 필드는 IP 헤더 프로토콜에서 더 이상 사용되지 않으며 DSCP(Differentiated Services Code Point) 필드로 대체되었습니다.

ttl

패킷의 ttl(time-to-live) 값은 삭제되기 전에 만들 수 있는 홉의 수를 나타냅니다. `ttl` 키워드를 사용하여 키워드의 인수로 지정한 값 또는 값 범위에 대한 패킷의 IP 헤더 ttl 값을 테스트할 수 있습니다. 낮은 TTL 값은 때로 traceroute 또는 침입 회피 시도를 표시하므로 `ttl` 키워드 매개 변수를 0이나 1과 같은 낮은 값으로 설정하는 것이 도움이 될 수 있습니다. (하지만 이 키워드에 대한 적절한 값은 매니지드 디바이스 배치 및 네트워크 토폴로지에 따라 다르다는 점을 참고하십시오.) 다음과 같이 구문을 사용합니다.

- 0에서 255 사이의 정수를 사용하여 TTL 값으로 특정 값을 설정합니다. 또한 등호(=)로 값을 입력할 수 있습니다(예를 들어 `5` 또는 `=5`를 지정할 수 있습니다).

- 하이픈(-)을 사용하여 TTL 값의 범위를 지정합니다(예를 들어, `0-2` 는 0에서 2까지의 모든 값을, `-5`는 0에서 5까지의 모든 값을, 그리고 `5-`는 5에서 255까지의 모든 값을 지정합니다).
- 부등호(>)를 사용하여 특정 값보다 큰 TTL 값을 지정합니다(예를 들어, `>3`은 3보다 큰 모든 값을 지정합니다).
- 크거나 같음 기호(>=)를 사용하여 특정 값보다 크거나 같은 TTL 값을 지정합니다(예를 들어, `>=3` 은 3보다 크거나 같은 모든 값을 지정합니다).
- 부등호(<)를 사용하여 특정 값보다 작은 TTL 값을 지정합니다(예를 들어, `<3`은 3보다 작은 모든 값을 지정합니다).
- 작거나 같음 기호(<=)를 사용하여 특정 값보다 작거나 같은 TTL 값을 지정합니다(예를 들어, `<=3` 은 3보다 작거나 같은 모든 값을 지정합니다).

ICMP 헤더 값

Firepower System은 ICMP 패킷의 헤더에서 공격 및 보안 정책 위반을 식별하는 데 사용할 수 있는 키워드를 지원합니다. 하지만 대부분의 ICMP 유형 및 코드를 탐지하는 사전 정의된 규칙이 존재한다는 점을 참고하십시오. 기존 규칙을 활성화하거나 기존 규칙에 기반한 로컬 규칙을 생성하는 것을 고려하십시오. ICMP 규칙을 처음부터 구축하면 요구를 충족시키는 규칙을 더욱 신속하게 찾을 수 있습니다.

icmp_id 및 icmp_seq

ICMP ID 및 시퀀스 번호는 ICMP 응답과 ICMP 요구를 연결하는 데 도움이 됩니다. 일반적인 트래픽에서 이 값은 패킷에 동적으로 할당됩니다. 일부 은닉 채널 및 분산서비스거부(DDoS) 프로그램은 정적 ICMP ID 및 시퀀스 값을 사용합니다. 다음 키워드를 사용하면 정적 값으로 ICMP 패킷을 확인할 수 있습니다.

키워드	정의
icmp_id	ICMP 에코 요청 또는 회신 패킷의 ICMP ID 번호를 검사합니다. `icmp_id` 키워드에 대한 인수로 ICMP ID 번호에 해당하는 숫자를 사용합니다.
icmp_seq	`icmp_seq` 키워드는 ICMP 에코 요청 또는 회신 패킷의 ICMP 시퀀스를 검사합니다. `icmp_seq` 키워드에 대한 인수로 ICMP 시퀀스 번호에 해당하는 숫자를 사용합니다.

itype

특정 ICMP 메시지 유형 값으로 패킷을 검색하려면 `itype` 키워드를 사용합니다. 유효한 ICMP 유형 값 또는 유효하지 않은 ICMP 유형 값을 지정하여 다양한 트래픽 유형을 테스트할 수 있습니다. 예를 들어, 공격자는 범위에서 서비스 거부 및 플러딩 공격을 야기할 사정 거리 밖의 ICMP 유형 값을 설정할 수 있습니다.

보다 작음(<) 및 보다 큼(>)을 사용하여 `itype` 인수 값의 범위를 지정할 수 있습니다.

예를 들면 다음과 같습니다.

- <35
- >36
- 3<>55

icode

ICMP 메시지는 때로 목적지에 도달할 수 없는 경우 정보를 제공하는 코드 값을 포함합니다.

특정 ICMP 코드 값으로 패킷을 확인하려면 `icode` 키워드를 사용할 수 있습니다. 유효한 ICMP 코드 값 또는 유효하지 않은 ICMP 코드 값을 지정하여 다양한 트래픽 유형을 테스트할 수 있습니다.

보다 작음(<) 및 보다 큼(>)을 사용하여 `icode` 인수 값의 범위를 지정할 수 있습니다.

예를 들면 다음과 같습니다.

- 35보다 작은 값을 찾으려면, `<35`를 지정합니다.
- 36보다 큰 값을 찾으려면, `>36`을 지정합니다.
- 3에서 55 사이의 값을 찾으려면, `3<>55`를 지정합니다.

팁 `icode` 및 `itype` 키워드를 사용하여 둘 다에 일치하는 트래픽을 식별할 수 있습니다. 예를 들어, ICMP Destination Unreachable(목적지 도달 불가) 코드 유형과 ICMP Port Unreachable(포트 연결 불가) 코드 유형을 포함하는 ICMP 트래픽을 식별하려면 Destination Unreachable(목적지 도달 불가)에 대한 3의 값과 함께 `itype` 키워드를, Port Unreachable(포트 연결 불가)에 대한 3의 값과 함께 `icode` 키워드를 사용합니다.

TCP 헤더 값 및 스트림 크기

Firepower System은 패킷의 TCP 헤더와 TCP 스트림 크기를 사용하여 시도된 공격을 식별하도록 설계된 키워드를 지원합니다.

ack

패킷의 TCP 확인 응답 수에 대한 값을 비교하려면 `ack` 키워드를 사용할 수 있습니다. `ack` 키워드에 지정된 값이 패킷의 TCP 확인 응답 수에 일치하는 경우 규칙이 트리거됩니다.

`ack`의 인수 값은 숫자여야 합니다.

flags

`flags` 키워드를 사용하여 검사한 패킷에 지정되었을 때, 규칙이 트리거되도록 하는 TCP 플래그의 조합을 지정할 수 있습니다.

참고 관례상 `flags` 값으로 `A+`를 사용할 경우, `established` 값으로 `flow` 키워드를 사용해야 합니다. 일반적으로 모든 플래그 조합 탐지를 확인하기 위해 플래그를 사용하는 경우, `stateless` 값으로 `flow` 키워드를 사용해야 합니다.

`flag` 키워드에 대해 다음 표에 설명한 값을 확인하거나 무시할 수 있습니다.

표 ***39:*** 플래그 인수

인수	**TCP** 플래그
Ack	데이터를 확인합니다.
Psh	데이터는 이 패킷에 보내야 합니다.
Syn	새로운 연결입니다.
Urg	패킷은 긴급한 데이터를 포함합니다.
Fin	닫힌 연결입니다.
Rst	중지된 연결입니다.
CWR	ECN 정체 창이 줄었습니다. 이는 이전에 R1 인수였으며 하위 호환성을 위해 계속 지원됩니다.
ECE	ECN 에코입니다. 이는 이전에 R2 인수였으며 하위 호환성을 위해 계속 지원됩니다.

`flags` 키워드를 사용할 때에는 여러 플래그를 기준으로 시스템이 매칭을 수행하는 방법을 나타내기 위해 연산자를 사용할 수 있습니다. 다음 표에서 이 연산자를 설명합니다.

표 ***40:*** 플래그와 함께 사용되는 연산자

연산자	설명	예
all	패킷은 모든 지정된 플래그를 포함해야 합니다.	`Urg` 및 `all`을 선택하여 패킷이 Urgent(긴급) 플래그를 포함해야 하며 다른 모든 플래그를 포함하도록 지정합니다.
any	패킷은 지정된 플래그를 모두 포함할 수 있습니다.	`Ack`, `Psh` 및 `any`를 선택하여 `Ack` 및 `Psh` 플래그 둘 중 하나 또는 둘 다 규칙을 트리거할 수 있도록 설정되어야 하며 모든 플래그가 하나의 패킷에도 설정될 수 있도록 지정합니다.
not	패킷은 지정된 플래그 집합을 포함할 수 없습니다.	`Urg`와 `not`을 선택하여 Urgent(긴급) 플래그가 이 규칙을 트리거하는 패킷에 설정되지 않도록 지정합니다.

flow

flow 키워드를 사용하여 세션 특징에 기반한 규칙에 따라 검사를 위한 패킷을 선택할 수 있습니다. flow 키워드를 사용하면 클라이언트 흐름 또는 서버 흐름에 규칙을 적용하여 규칙이 적용된 트래픽 흐름의 방향을 지정할 수 있습니다. flow 키워드가 패킷을 검사하는 방법을 지정하려면, 분석을 원하는 트래픽의 방향과 검사된 패킷의 상태, 그리고 패킷이 재구축된 스트림의 일부인지 여부를 설정할 수 있습니다.

규칙을 처리할 때 패킷의 상태 저장 검사가 이루어집니다. TCP 규칙이 상태 비저장 트래픽(설정된 세션 컨텍스트가 없는 트래픽)을 무시하기를 원하는 경우, 규칙에 flow 키워드를 추가하고 키워드의 **Established** 인수를 선택해야 합니다. UDP 규칙이 스테이트리스 트래픽을 무시하도록 하려면 규칙에 flow 키워드를 추가하고 **Established** 인수나 방향 인수 또는 둘 모두를 선택해야 합니다. 이것은 TCP 또는 UDP 규칙이 패킷의 상태 저장 검사를 수행하도록 합니다.

방향 인수를 추가하면, 규칙 엔진은 지정된 방향과 일치하는 흐름과 함께 설정한 상태가 있는 해당 패킷만 검사합니다. TCP 또는 UDP 연결이 탐지되었을 때 트리거되는 규칙에 established 인수 및 From Client 인수와 함께 flow 키워드를 추가하는 경우, 규칙 엔진은 클라이언트에서 전송된 패킷만 검사합니다.

팁 최대 성능을 위해, TCP 규칙 또는 UDP 세션 규칙에서 항상 flow 키워드를 포함합니다.

다음 표에서는 flow 키워드에 대해 지정할 수 있는 스트림 관련 인수를 설명합니다.

표 ***41:*** 상태 관련 흐름 인수

인수	설명
Established	연결이 설정된 경우 트리거됩니다.
Stateless	스트림 프로세서의 상태에 상관없이 트리거됩니다.

다음 표에서는 flow 키워드에 대해 지정할 수 있는 방향 옵션을 설명합니다.

표 ***42:*** 흐름 방향 인수

인수	설명
To Client	서버 응답 시 트리거됩니다.
To Server	클라이언트 응답 시 트리거됩니다.
From Client	클라이언트 응답 시 트리거됩니다.
From Server	서버 응답 시 트리거됩니다.

From Server 및 To Client는 같은 기능을 수행하며, To Server 및 From Client도 같은 기능을 수행한다는 점에 유의하십시오. 이 옵션은 규칙에 컨텍스트 및 가독성을 추가하기 위해 존재합니다. 예를 들어 서버에서 클라이언트에 취해지는 공격을 탐지하기 위해 설계된 규칙을 작성하는 경우, From

Server를 사용하십시오. 하지만 클라이언트에서 서버에 취해지는 공격을 탐지하기 위해 설계된 규칙을 작성하는 경우, From Client를 사용합니다.

다음 표에서는 flow 키워드에 대해 지정할 수 있는 스트림 관련 인수를 설명합니다.

표 **43:** 스트림 관련 흐름 인수

인수	설명
Ignore Stream Traffic	재작성한 스트림 패킷에서 트리거되지 않습니다.
Only Stream Traffic	재작성된 스트림 패킷에서만 트리거됩니다.

예를 들어, 스트림 프리프로세서에 의해 리어셈블된, 설정된 세션에서 클라이언트에서 서버로 이동하는 트래픽을 탐지하려면 flow 키워드에 대해 To Server, Established, Only Stream Traffic을 사용할 수 있습니다.

seq

seq 키워드로 정적 시퀀스 번호 값을 지정할 수 있습니다. 시퀀스 번호가 특정 인수에 일치하는 패킷은 키워드를 포함하는 규칙을 트리거합니다. 이 키워드는 거의 사용되지 않지만 정적 시퀀스 번호로 생성된 패킷을 사용한 스캔 네트워크와 공격 식별에 도움이 됩니다.

window

window 키워드를 사용하여 관심 있는 TCP 창 크기를 지정할 수 있습니다. 이 키워드를 포함하는 규칙은 지정된 TCP 창 크기의 패킷이 발생할 때마다 트리거됩니다. 이 키워드는 거의 사용되지 않지만 정적 TCP 창 크기로 생성된 패킷을 사용한 스캔 네트워크와 공격 식별에 도움이 됩니다.

stream_size

다음 형식을 사용하여 stream_size 키워드를 TCP 스트림 바이트의 크기를 결정하는 스트림 전처리기와 함께 사용할 수 있습니다.

```
direction,operator,bytes
```

bytes가 바이트 수인 경우. 쉼표(,)로 인수의 각 옵션을 구분해야 합니다.

다음 표에서는 stream_size 키워드에 대해 지정할 수 있는 대소문자를 구분하지 않는 방향 옵션을 설명합니다.

표 **44: stream_size** 키워드 방향 인수

인수	설명
client	지정된 스트림 크기와 일치하는 클라이언트의 스트림에서 트리거됩니다.
server	지정된 스트림 크기와 일치하는 서버의 스트림에서 트리거됩니다.

인수	설명
both	클라이언트 트래픽 및 지정된 스트림 크기와 일치하는 서버의 트래픽 모두에서 트리거됩니다. 예를 들어, `both, >, 200` 인수는 클라이언트 발신 트래픽이 200바이트보다 큰 경우 및 서버 발신 트래픽이 200바이트보다 큰 경우에 트리거됩니다.
either	무엇이 먼저 발생하든 지정된 스트림 크기와 일치하는 클라이언트 또는 서버의 트래픽에서 트리거됩니다. 예를 들어, `either, >, 200` 인수는 클라이언트 발신 트래픽이 200바이트보다 큰 경우 또는 서버 발신 트래픽이 200바이트보다 큰 경우에 트리거됩니다.

다음 표에서는 `stream_size` 키워드와 함께 사용할 수 있는 연산자를 설명합니다.

표 ***45: stream_size*** 키워드 인수 연산자

연산자	설명
=	같음
!=	같지 않음
>	보다 큼
<	보다 작음
>=	보다 크거나 같음
<=	보다 작거나 같음

예를 들어, `client, >=, 5001216`을 `stream_size` 키워드의 인수로 사용하여 클라이언트에서 5001216 바이트와 같거나 큰 서버로 이동하는 TCP 스트림을 탐지할 수 있습니다.

stream_reassembly 키워드

`stream_reassemble` 키워드를 사용하여 연결 시 검사된 트래픽이 규칙 조건에 일치할 때 단일 연결에 대한 TCP 스트림 리어셈블리를 활성화 및 비활성화할 수 있습니다. 또는, 규칙에서 이러한 키워드를 여러 번 사용할 수 있습니다.

스트림 리어셈블리를 활성화하거나 비활성화하려면 다음 구문을 사용합니다.

```
enable|disable, server|client|both, option, option
```

다음 표에서는 `stream_reassemble` 키워드와 함께 사용할 수 있는 선택적 인수를 설명합니다.

표 ***46: stream_reassemble*** 선택적 인수

인수	설명
noalert	규칙에 지정된 다른 모든 탐지 옵션에 관계없이 어떤 이벤트도 생성하지 않습니다.
fastpath	일치할 때 연결 트래픽의 나머지 부분을 무시합니다.

예를 들어 다음 규칙은 HTTP 응답에서 200 OK 상태 코드가 탐지된 연결에 대해 이벤트를 생성하지 않은 채 TCP 클라이언트 측 스트림 리어셈블리를 비활성화합니다.

```
alert tcp any 80 -> any any (flow:to_client, established; content: “200 OK”;
stream_reassemble:disable, client, noalert
```

SSL 키워드

SSL 규칙 키워드를 사용하여 SSL(Secure Sockets Layer) 전처리기를 호출할 수도 있고 암호화된 세션에서 패킷으로부터 SSL 버전 및 세션 상태에 관한 정보를 추출할 수 있습니다.

클라이언트와 서버가 SSL 또는 TLS(전송 레이어 보안)를 사용하여 암호화된 세션을 설정하기 위해 통신할 때 핸드쉐이크 메시지를 교환합니다. 세션에서 전송되는 데이터는 암호화되지만, 핸드쉐이크 메시지는 그렇지 않습니다.

SSL 전처리기는 특정 핸드쉐이크 필드의 상태 및 버전 정보를 추출합니다. 핸드쉐이크 내 두 필드는 핸드쉐이크의 세션 및 단계를 암호화하는 데 사용된 SSL 또는 TLS 버전을 나타냅니다.

ssl_state

`ssl_state` 키워드는 암호화된 세션에 대한 상태 정보를 비교하는 데 사용될 수 있습니다. 동시에 사용된 2개 이상의 SSL 버전을 확인하려면, 규칙에서 여러 `ssl_version` 키워드를 사용합니다.

규칙이 `ssl_state` 키워드를 사용하면, 규칙 엔진은 SSL 전처리기를 호출하여 SSL 상태 정보에 대한 트래픽을 확인하도록 합니다.

예를 들어, 너무 긴 챌린지 길이 및 너무 많은 데이터를 가진 `ClientHello` 메시지를 보내 서버에 버퍼 오버플로를 일으키는 공격자의 시도를 검색하려면, `client_hello`와 함께 `ssl_state` 키워드를 인수로 사용한 후 비정상적으로 규모가 큰 패킷을 확인할 수 있습니다.

쉼표로 구분된 목록을 사용하여 SSL 상태에 대한 여러 인수를 지정합니다. 여러 인수를 나열할 경우, 시스템은 OR 연산자를 사용하여 이를 평가합니다. 예를 들어, 인수로 `client_hello` 및 `server_hello`를 지정한 경우, 시스템은 `client_hello` 또는 `server_hello`가 있는 트래픽에 대한 규칙을 평가합니다.

또한 모든 인수를 무효화할 수 있습니다. 예를 들면 다음과 같습니다.

```
!client_hello, !unknown
```

연결이 각 상태 집합에 도달했음을 확인하려면 ssl_state 규칙 옵션을 사용하는 여러 규칙이 사용되어야 합니다. `ssl_state` 키워드는 다음 식별자를 인수로 취합니다.

표 47: ssl_state 인수

인수	목적
client_hello	메시지 유형이 ClientHello인 핸드셰이크 메시지에 대해 매칭합니다. 여기서 클라이언트는 암호화된 세션을 요청합니다.
server_hello	메시지 유형이 ServerHello인 핸드셰이크 메시지에 대해 매칭합니다. 여기서 서버는 암호화된 세션에 대한 클라이언트의 요청에 응답합니다.
client_keyx	메시지 유형으로 ClientKeyExchange를 가진 핸드쉐이크 메시지와 비교하며, 서버로부터 키를 수신했음을 확인하기 위해 클라이언트가 서버에 키를 전송합니다.
server_keyx	메시지 유형으로 serverKeyExchange를 가진 핸드쉐이크 메시지와 비교하며, 서버로부터 키를 수신했음을 확인하기 위해 클라이언트가 서버에 키를 전송합니다.
unknown	모든 핸드쉐이크 메시지 유형과 비교합니다.

ssl_version

ssl_version 키워드는 암호화된 세션의 버전 정보와 일치시키는 데 사용될 수 있습니다. 규칙이 ssl_version 키워드를 사용하면, 규칙 엔진은 SSL 전처리기를 호출하여 SSL 버전 정보에 대한 트래픽을 확인하도록 합니다.

예를 들어 SSL 버전 2에 버퍼 오버플로 취약성이 있음을 알고 있는 경우, SSL의 해당 버전을 사용하는 트래픽을 식별하려면 ssl_version 키워드와 sslv2 인수를 사용할 수 있습니다.

쉼표로 구분된 목록을 사용하여 SSL 버전에 대한 여러 인수를 지정합니다. 여러 인수를 나열할 경우, 시스템은 OR 연산자를 사용하여 이를 평가합니다. 예를 들어, SSLv2를 사용하지 않는 모든 암호화된 트래픽을 식별하고자 한다면 규칙에 ssl_version:ssl_v3,tls1.0,tls1.1,tls1.2를 추가할 수 있습니다. 규칙은 SSL 버전 3, TLS 버전 1.0, TLS 버전 1.1, 또는 TLS 버전 1.2를 사용하여 모든 트래픽을 평가합니다.

ssl_version 키워드는 다음 SSL/TLS 버전 식별자를 인수로 취합니다.

표 48: ssl_version 인수

인수	목적
sslv2	SSL(Secure Sockets Layer) 버전 2를 사용하여 인코딩된 트래픽과 비교합니다.
sslv3	SSL(Secure Sockets Layer) 버전 3을 사용하여 인코딩된 트래픽과 비교합니다.
tls1.0	TLS(전송 레이어 보안) 버전 1.0을 사용하여 인코딩된 트래픽과 비교합니다.
tls1.1	TLS(전송 레이어 보안) 버전 1.1을 사용하여 인코딩된 트래픽과 비교합니다.
tls1.2	TLS(전송 레이어 보안) 버전 1.2를 사용하여 인코딩된 트래픽과 비교합니다.

appid 키워드

appid 키워드를 사용하여 패킷에서 애플리케이션 프로토콜, 클라이언트 애플리케이션 또는 웹 애플리케이션을 식별할 수 있습니다. 예를 들어 특정 취약성의 영향을 받기 쉬운 특정 애플리케이션을 대상으로 할 수 있습니다.

침입 규칙의 appid 키워드 내에서 **Configure AppID(AppID** 구성**)**를 클릭하여 탐지할 하나 이상의 애플리케이션을 선택합니다.

사용 가능한 애플리케이션 탐색

조건 만들기를 처음 시작할 때 **Available Application(**사용 가능한 애플리케이션**)** 목록은 제한되지 않은 상태이며 시스템에서 탐지되는 모든 애플리케이션이 페이지당 100개씩 표시됩니다.

- 애플리케이션 페이지를 넘기려면 목록 아래의 화살표를 클릭합니다.
- 애플리케이션의 특성에 대한 요약 정보와 찾아갈 수 있는 인터넷 검색 링크가 포함된 팝업 창을 표시하려면 애플리케이션 옆의 **Information(**정보**)** ( )을 클릭합니다.

애플리케이션 필터 사용

매칭할 애플리케이션을 찾으려면, 다음과 같은 방법으로 **Available Applications** 목록을 제한할 수 있습니다.

- 애플리케이션을 검색하려면 **Search by name** 프롬프트를 클릭한 다음 이름을 입력합니다. 입력을 수행하면 목록이 업데이트되어 일치하는 애플리케이션을 표시합니다.
- 필터를 적용하여 애플리케이션을 제한하려면 **Application Filters(**애플리케이션 필터**)** 목록을 사용합니다. 필터를 적용하면 **Available Applications** 목록이 업데이트됩니다. 사용자의 편의를 위해 시스템은 잠금 취소 아이콘을 사용하여 암호화 트래픽이나 암호화되지 않은 트래픽이 아닌 해독된 트래픽에서만 식별할 수 있는 애플리케이션을 표시합니다.

참고 Application Filters(애플리케이션 필터) 목록에서 하나 이상의 필터를 선택하고 **Available Applications(**사용 가능한 애플리케이션**)** 목록도 검색하는 경우 선택한 항목 및 검색을 통해 필터링한 **Available Applications(**사용 가능한 애플리케이션**)** 목록이 AND 연산을 사용하여 조합됩니다.

애플리케이션 선택

단일 애플리케이션을 선택하려면 애플리케이션을 선택하고 **Add to Rule(**규칙에 추가**)**을 클릭합니다. 현재의 제한된 보기에서 모든 애플리케이션을 선택하려면 마우스 오른쪽 버튼으로 클릭하고 **Select All(**모두 선택**)**을 선택합니다.

애플리케이션 레이어 프로토콜 값

전처리기가 애플리케이션 레이어 프로토콜 값의 표준화 및 검사 작업 대부분을 수행하지만 사용자는 다양한 전처리기 옵션을 사용하여 계속해서 애플리케이션 레이어 값을 검사할 수 있습니다.

RPC 키워드

rpc 키워드는 TCP 또는 UDP 패킷에서 ONC RPC(Open Network Computing Remote Procedure Call) 서비스를 식별합니다. 이를 통해 호스트에서 RPC 프로그램을 확인하려는 시도를 탐지할 수 있습니다. 침입자는 RPC 포트매퍼를 사용하여 네트워크에서 실행되는 RPC 서비스가 공격받을 가능성이 있는지 결정할 수 있습니다. 이들은 포트매퍼를 사용하지 않고 RPC를 실행하는 다른 포트에 액세스를 시도할 수도 있습니다. 다음 표에서는 rpc 키워드가 수용하는 인수를 나열합니다.

표 ***49: rpc*** 키워드 인수

인수	설명
application	RPC 애플리케이션 번호
procedure	호출된 RPC 절차
version	RPC 버전

rpc 키워드에 대한 인수를 지정하려면, 다음 구문을 사용합니다.

```
application,procedure,version
```

application(이 RPC 애플리케이션 번호인 경우, procedure는 RPC 절차 번호이며, version은 RPC 버전 번호입니다. rpc 키워드에 대한 모든 인수를 지정해야 합니다. 인수 중 하나를 지정할 수 없는 경우, 별표(*)로 교체합니다.

예를 들어, 모든 절차 또는 버전으로 RPC 포트매퍼를 검색하려면(번호 100000으로 표시된 RPC 애플리케이션), `100000,*,*`를 인수로 사용합니다.

ASN.1 키워드

asn1 키워드를 사용하면 패킷 또는 패킷의 일부를 해독하여 다양한 악성 인코딩을 찾을 수 있습니다.

다음 표에서는 asn1 키워드의 인수에 대해 설명합니다.

표 ***50: asn.1*** 키워드 인수

인수	설명
Bitstring Overflow	원격으로 공격 가능한 유효하지 않은 비트 스트링 인코딩을 탐지합니다.
Double Overflow	표준 버퍼보다 큰 이중 ASCII 인코딩을 탐지합니다. 이것은 Microsoft Windows에서 악용 가능한 기능으로 알려져 있지만 현재로서는 어떤 서비스가 악용 가능한지 알 수 없습니다.
Oversize Length	제공된 인수보다 큰 ASN.1 유형 길이를 탐지합니다. 예를 들어, Oversize Length로 500을 설정한 경우, 500보다 큰 모든 ASN.1 유형이 규칙을 트리거합니다.
Absolute Offset	패킷 페이로드의 시작 지점으로부터 절대적 오프셋을 설정합니다. (오프셋 카운터가 0바이트부터 시작한다는 점에 유의하십시오.) 예를 들어, SNMP 패킷을 해독할 경우, Absolute Offset은 0으로 설정하고 Relative Offset은 설정하지 않습니다. Absolute Offset은 양수이거나 음수일 수 있습니다.

인수	설명
Relative Offset	이것은 마지막으로 성공한 콘텐츠 일치로부터의 상대적 오프셋이며, pcre 또는 byte_jump입니다. 콘텐츠 "foo" 바로 다음에 있는 ASN.1 시퀀스를 해독하려면, Relative Offset은 0으로 설정하고 Absolute Offset은 설정하지 않습니다. Relative Offset은 양수이거나 음수일 수 있습니다. (오프셋 카운터가 0바이트부터 시작한다는 점에 유의하십시오.)

예를 들어, 버퍼 오버플로를 만드는 Microsoft ASN.1 라이브러리에는 알려진 취약성이 있는데, 공격자가 특별히 만들어진 인증 패킷으로 조건을 공격하도록 허용합니다. 시스템이 asn.1 데이터를 해독할 때 패킷의 익스플로잇 코드가 시스템 수준 권한이 있는 호스트에서 실행되거나 DoS 조건을 일으킬 수 있습니다. 다음 규칙은 asn1 키워드를 사용하여 이 취약성을 이용하려는 시도를 탐지합니다.

```
alert tcp $EXTERNAL_NET any -> $HOME_NET 445
(flow:to_server, established; content:"|FF|SMB|73|";
nocase; offset:4; depth:5;
asn1:bitstring_overflow,double_overflow,oversize_length 100,
relative_offset 54;)
```

위 규칙은 포트 445를 사용하여 $EXTERNAL_NET 변수에 정의된 모든 IP 주소, 모든 포트로부터 $HOME_NET 변수에 정의된 모든 IP 주소로 이동하는 TCP 트래픽에 대한 이벤트를 생성합니다. 또한, 서버로 연결된 TCP 연결에만 규칙을 수행합니다. 규칙은 다음으로 특정 위치에서 특정 내용을 테스트합니다. 마지막으로, 규칙은 asn1 키워드를 사용하여 비트 스트링 인코딩 및 이중 ASCII 인코딩을 탐지하고, 마지막으로 성공한 콘텐츠 일치의 끝 지점에서 55바이트 길이부터 100바이트 이상의 길이를 가진 asn.1 유형을 식별합니다. (오프셋 카운터가 0바이트부터 시작한다는 점에 유의하십시오.)

urilen 키워드

urilen 키워드를 HTTP 검사 전처리기와 함께 사용하여 최대 길이보다는 작고 최소 길이보다는 크며, 특정 범위 안에 있는 특정 길이 URI의 HTTP 트래픽을 검사할 수 있습니다.

HTTP 검사 전처리기가 패킷을 표준화하고 검사한 후, 규칙 엔진은 규칙에 대한 패킷을 평가하고, urilen 키워드가 지정한 길이 상태에 URI가 일치하는지 결정합니다. 이 키워드를 사용하여 URI 길이 취약성을 이용하려고 하는 익스플로잇을 탐지할 수 있습니다. 예를 들면 공격자가 DoS 조건을 야기하거나 시스템 수준 권한이 있는 호스트에서 코드를 실행할 수 있는 버퍼 오버플로를 생성하는 것입니다.

urilen 키워드를 규칙에서 사용할 때 다음 사항에 유의하십시오.

- 실전에서는 항상 urilen 키워드와 flow:established 키워드 및 하나 이상의 다른 키워드를 조합하여 사용합니다.
- 규칙 프로토콜은 항상 TCP입니다.
- 대상 포트는 항상 HTTP 포트입니다.

십진수로 나타낸 바이트 수와 보다 작음(<) 또는 보다 큼(>)을 사용하여 URI 길이를 지정합니다.

예를 들면 다음과 같습니다.

- 5바이트 길이의 URI를 탐지하려면 5를 지정합니다.

- 5바이트 길이보다 짧은 URI를 탐지하려면 `< 5`(스페이스 문자 하나로 분리)를 지정합니다.
- 5바이트 길이보다 긴 URI를 탐지하려면 `> 5`(스페이스 문자 하나로 분리)를 지정합니다.
- 길이가 3에서 5바이트 사이에 들어가는 URI를 탐지하려면 `3 <> 5`(<> 전후에 스페이스 문자 하나씩 표시)를 지정합니다.

예를 들어, eDirectory 버전 8.8과 함께 제공된 Novell의 서버 모니터링 및 진단 유틸리티 iMonitor 버전 2.4에는 알려진 취약성이 있습니다. 과도하게 긴 URI을 포함하는 패킷은 버퍼 오버플로를 생성하며 이로 인해 공격자가 시스템 수준 권한으로 호스트에서 실행하거나 DoS 조건을 야기할 수 있는 특수하게 조작된 패킷이 포함된 조건을 악용할 수 있습니다. 다음 규칙은 `urilen` 키워드를 사용하여 이 취약성을 악용하려는 시도를 탐지합니다.

```
alert tcp $EXTERNAL_NET any -> $HOME_NET $HTTP_PORTS
(msg:"EXPLOIT eDirectory 8.8 Long URI iMonitor buffer
overflow attempt"; flow:to_server,established;
urilen:> 8192; uricontent:"/nds/"; nocase;
classtype:attempted-admin; sid:x; rev:1;)
```

위 규칙은 $HTTP_PORTS 변수에 정의된 포트를 사용하여 $EXTERNAL_NET 변수에 정의된 모든 IP 주소, 모든 포트로부터 $HOME_NET 변수에 정의된 모든 IP 주소로 이동하는 TCP 트래픽에 대한 이벤트를 생성합니다. 또한, 패킷은 서버로 설정된 TCP 연결에서만 규칙에 대해 평가됩니다. 규칙은 `urilen` 키워드를 사용하여 길이가 8192바이트 이상인 모든 URI를 탐지합니다. 마지막으로 규칙은 URI에서 대소문자를 구분하지 않는 특정 콘텐츠 `/nds/`를 검색합니다.

관련 항목

침입 규칙 헤더 프로토콜, 5 페이지
침입 규칙 헤더 소스 및 대상 포트, 8 페이지
사전 정의된 기본 변수

DCE/RPC 키워드

다음 표에 설명된 3개의 DCE/RPC 키워드를 사용하여 DCE/RPC 세션 트래픽에서 익스플로잇을 모니터링할 수 있습니다. 시스템이 이 키워드로 규칙을 처리할 때, DCE/RPC 전처리기를 호출합니다.

표 ***51: DCE/RPC*** 키워드

사용 키워드	사용 방법	탐지 대상
`dce_iface`	단독으로	특정 DCE/RPC 서비스를 식별하는 패킷
`dce_opnum`	`dce_iface`가 앞에 오는 경우	특정 DCE/RPC 서비스 작업을 식별하는 패킷
`dce_stub_data`	`dce_iface` + `dce_opnum`이 앞에 오는 경우	특정 작업 요청 또는 응답을 정의하는 스텁 데이터

표에서 `dce_opnum` 및 `dce_iface`가 항상 앞에 와야 한다는 점과 `dce_stub_data` 및 `dce_iface` + `dce_opnum`이 항상 앞에 와야 한다는 점에 유의하시기 바랍니다.

DCE/RPC 키워드를 다른 규칙 키워드와 조합하여 사용할 수 있습니다. DCE/RPC 규칙의 경우, `byte_jump`, `byte_test`, `and byte_extract` 키워드를 선택한 **DCE/RPC** 인수와 함께 사용합니다.

Cisco는 DCE/RPC 키워드를 포함하는 규칙에 최소 하나의 `content` 키워드를 포함할 것을 권장합니다. 이는 규칙 엔진이 빠른 패턴 매처를 사용하도록 하는 것인데, 이는 처리 속도를 높이고 성능을 향상시킵니다. 규칙에 최소 하나의 `content` 키워드가 포함될 때, `content` 키워드 **Use Fast Pattern Matcher(**빠른 패턴 매처 사용**)** 인수를 활성화했는지 여부에 상관없이 규칙 엔진이 빠른 패턴 매처를 사용한다는 점에 유의하십시오.

다음의 경우 일치 내용으로 DCE/RPC 버전 및 연속되는 헤더 정보를 사용할 수 있습니다.

- 규칙은 다른 `content` 키워드를 포함하지 않습니다
- 규칙은 다른 `content` 키워드를 포함하지만, DCE/RPC 버전 및 연속된 정보는 다른 콘텐츠보다 고유한 패턴을 나타냅니다

  예를 들어, DCE/RPC 버전 및 연속되는 정보는 단일 바이트의 콘텐츠보다 고유할 가능성이 더 높습니다.

다음 버전 중 하나 및 연속되는 정보 콘텐츠 일치로 자격 심사 규칙을 종료해야 합니다.

- 연결 지향 DCE/RPC 규칙은 주요 버전 05, 비주요 버전 00, 요청 PDU(프로토콜 데이터 장치) 유형 00에 대해 콘텐츠 `|05 00 00|`을 사용합니다.
- 연결 없는 DCE/RPC 규칙은 버전 04, 요청 PDU 유형 00에 대해 콘텐츠 `|04 00|`을 사용합니다.

어떤 경우든 DCE/RPC 프리프로세서에 의해 이미 완료된 처리를 반복하지 않고 fast pattern matcher를 호출하려면 버전 및 인접 정보에 대한 `content` 키워드를 규칙의 마지막 키워드로 배치하십시오. `content` 키워드를 규칙의 끝에 두는 것은 fast pattern matcher를 호출하기 위한 디바이스로 사용되는 버전 내용에 적용되며, 규칙의 다른 내용 일치에는 적용할 필요가 없습니다.

관련 항목

DCE/RPC 전처리기
content 및 protected_content 키워드, 20 페이지
content 키워드 빠른 패턴 매처 인수, 30 페이지
개요: byte_jump 및 byte_test 키워드
byte_extract 키워드, 38 페이지

dce_iface

`dce_if ace` 키워드를 사용하여 특정 DCE/RPC 서비스를 확인할 수 있습니다.

선택적으로, `dce_opnum` 및 `dce_stub_data` 키워드와 조합하여 `dce_iface`를 사용해 검사해야 할 DCE/RPC 트래픽을 추가로 제한할 수 있습니다.

고정된 16바이트 UUID(Universally Unique Identifier) 식별자는 각 DCE/RPC 서비스에 할당된 애플리케이션 인터페이스를 식별합니다. 예를 들어 UUID 4b324fc8-670-01d3-1278-5a47bf6ee188은 피어 투 피어 프린터, 파일 및 SMB 명명된 파이프 등을 위한 다양한 관리 기능을 제공하는 DCE/RPC lanmanserver 서비스(srvsvc 서비스라고도 함)를 식별합니다. DCE/RPC 전처리기는 UUID 및 DCE/RPC 세션을 추적하기 위한 관련 헤더 값을 사용합니다.

인터페이스 UUID는 하이픈으로 구분된 5개의 16진수 문자열로 구성됩니다.

```
<4hexbytes>-<2hexbytes>-<2hexbytes>-<2hexbytes>-<6hexbytes>
```

넷로그온 인터페이스에 대해 다음 UUID에 표시된 대로 하이픈을 포함하는 전체 UUID를 입력하여 인터페이스를 지정합니다.

```
12345678-1234-abcd-ef00-01234567cffb
```

빅 엔디언 바이트 순서로 UUID에 처음 3개 문자열을 지정해야 한다는 점에 유의하십시오. 게시된 인터페이스 목록 및 프로토콜 분석기는 일반적으로 UUID를 정확한 바이트 순서로 정렬하지만 이를 입력하기 전에 UUID 바이트 순서를 다시 정렬해야 할 수 있습니다. 경우에 따라 리틀 엔디언 바이트 순서로 처음 3개 문자열과 함께 원시 ASCII 텍스트로 표시될 수 있으므로 다음 메신저 서비스를 보여진 UUID로 생각하십시오.

```
f8 91 7b 5a 00 ff d0 11 a9 b2 00 c0 4f b6 e6 fc
```

다음과 같이 하이픈을 삽입하고 처음 3개 문자열을 빅 엔디언 바이트 순서로 나열함으로써 `dce_iface` 키워드에 동일한 UUID를 지정합니다.

```
5a7b91f8-ff00-11d0-a9b2-00c04fb6e6fc
```

DCE/RPC 세션이 여러 인터페이스에 요청을 포함할 수 있지만, 한 규칙에서 한 개의 `dce_iface` 키워드만 포함해야 합니다. 추가 인터페이스를 검색하려면 추가 규칙을 만듭니다.

DCE/RPC 애플리케이션 인터페이스에는 또한 인터페이스 버전 번호가 있습니다. 선택적으로 동일한 버전, 동일하지 않은 버전, 특정 값보다 적거나 큰 버전을 나타내는 연산자와 인터페이스 버전을 지정할 수 있습니다.

TCP 세그멘테이션 또는 IP 프래그먼트화 외에도 연결 지향 DCE/RPC 및 연결 없는 DCE/RPC를 모두 프래그먼트화할 수 있습니다. 일반적으로, 첫 번째 이외의 다른 DCE/RPC 조각을 지정된 인터페이스와 연결하는 것은 유용하지 않고, 이렇게 하면 그 결과로 많은 양의 잘못된 긍정을 얻을 수 있습니다. 그러나, 유연성을 위해 지정된 인터페이스에 대한 모든 조각을 선택적으로 평가할 수 있습니다.

다음 표에는 `dce_iface` 키워드 인수가 요약되어 있습니다.

표 ***52: dce_iface*** 인수

인수	설명
Interface UUID	하이픈을 포함하는 UUID는 사용자가 DCE/RPC 트래픽에서 탐지할 특정 서비스의 애플리케이션 인터페이스를 식별합니다. 특정 인터페이스에 연결된 모든 요청은 인터페이스 UUID에 일치됩니다.
Version	또는, 0에서 65535까지의 애플리케이션 인터페이스 버전 번호 및 특정 값보다 큰(>), 작은(<), 같은(=), 같지 않은(!) 버전을 탐지할지 여부를 나타내는 연산자.

인수	설명
All Fragments	또는, 모든 관련 DCE/RPC 조각의 인터페이스와 일치하는 것을 가능하게 합니다. 지정되어 있는 경우, 인터페이스 버전에서도 가능합니다. 이 인수는 기본적으로 비활성화되어 있는데, 첫 번째 조각 또는 조각화되지 않은 전체 패킷이 특정 인터페이스와 연결되어 있는 경우에만 키워드 일치를 나타냅니다. 이 인수를 활성화하면 잘못된 긍정을 얻을 수 있다는 점에 유의하십시오.

dce_opnum 키워드

dce_opnum 키워드를 DCE/RPC 전처리기와 함께 사용하여 DCE/RPC 서비스가 제공하는 하나 이상의 특정 작업을 확인하는 패킷을 탐지할 수 있습니다.

클라이언트 함수 호출은 특정 서비스 함수를 요구하는데, 이는 DCE/RPC 사양에서 작업으로 참조됩니다. 작업 번호(opnum)는 DCE/RPC 헤더에서 특정 작업을 식별합니다. 공격이 특정 작업을 대상으로 할 가능성이 높습니다.

예를 들어, UUID 12345678-1234-abcd-ef00-01234567cffb는 여러 다양한 작업을 제공하는 넷로그온 서비스에 대한 인터페이스를 식별합니다. 그중 하나는 작업 6의 NetrServerPasswordSet 작업입니다.

작업을 위한 서비스를 식별하려면 dce_iface 키워드 다음에 dce_opnum 키워드를 입력해야 합니다.

단일 십진수 0에서 65535를 지정하여 특정 작업, 하이픈으로 구분된 작업의 범위 또는 쉼표로 구분된 작업 및 범위의 목록을 어떤 순서로도 나타낼 수 있습니다.

다음의 예시 중 하나를 통해 유효한 넷로그온 작업 번호를 지정합니다.

```
15
15-18
15, 18-20
15, 20-22, 17
15, 18-20, 22, 24-26
```

dce_stub_data 키워드

dce_stub_data 키워드를 DCE/RPC 전처리기와 함께 사용하여 다른 규칙 옵션에 관계없이 스텁 데이터 시작 시 규칙 엔진이 검사를 시작하도록 지정할 수 있습니다. dce_stub_data 키워드를 따르는 패킷 페이로드 규칙 옵션은 스텁 데이터 버퍼에 관련하여 적용됩니다.

DCE/RPC 스텁 데이터는 DCE/RPC 중심의 서비스와 루틴을 제공하는 메커니즘인 클라이언트 절차 호출과 DCE/RPC 런타임 시스템 간 인터페이스를 제공합니다. DCE/RPC 공격은 DCE/RPC 패킷의 스텁 데이터 부분에 표시됩니다. 스텁 데이터는 특정 작업 또는 함수 호출에 연결되어 있으므로, 관련 서비스 및 작업을 식별하려면 dce_stub_data에 앞에 항상 dce_iface 및 dce_opnum을 입력해야 합니다.

dce_stub_data 키워드는 인수를 갖지 않습니다.

SIP 키워드

4개의 SIP 키워드를 사용하면 SIP 세션 트래픽을 모니터링하여 공격을 탐지할 수 있습니다.

SIP 프로토콜은 서비스 거부(DoS) 공격에 취약하다는 점을 참고하십시오. 이러한 공격을 해결할 규칙은 속도 기반 공격 방지를 활용할 수 있습니다.

sip_header 키워드

sip_header 키워드를 사용하여 추출된 SIP 요청 또는 응답 헤더 시작에서 검사를 시작하고 헤더 필드로 검사를 제한할 수 있습니다.

sip_header 키워드는 인수를 갖지 않습니다.

다음의 예제 규칙 조각 SIP 헤더를 가리키고 CSeq 헤더 필드에 일치됩니다.

```
alert udp any any -> any 5060 ( sip_header; content:"CSeq"; )
```

관련 항목

동적 침입 규칙 상태
속도 기반 공격 방지

sip_body 키워드

sip_body 키워드를 사용하여 추출된 SIP 요청 또는 응답 메시지 본문 시작 지점에서 검사를 시작하고 검사를 메시지 본문 검사로 제한할 수 있습니다.

sip_body 키워드는 인수를 갖지 않습니다.

다음의 예제 규칙 조각은 SIP 메시지 본문을 가리키고 추출된 SDP 데이터의 c(연결 정보) 필드에서 특정 IP 주소에 일치됩니다.

```
alert udp any any -> any 5060 ( sip_body; content:"c=IN 192.168.12.14"; )
```

규칙은 SDP 내용을 검색하는 것에 제한되지 않습니다. SIP 전처리기는 전체 메시지 본문을 추출하고 이를 규칙 엔진이 사용할 수 있도록 합니다.

sip_method 키워드

각 SIP 요청의 *method* 필드는 요청의 목적을 확인합니다. sip_method 키워드를 사용하여 특정 메서드에 대한 SIP 요청을 테스트할 수 있습니다. 메서드가 여러 개인 경우 쉼표로 구분하십시오.

다음 중 하나로 현재 정의된 SIP 메서드를 지정할 수 있습니다.

```
ack, benotify, bye, cancel, do, info, invite, join, message, notify, options, prack,
publish, quath, refer, register, service, sprack, subscribe, unsubscribe, update
```

메서드는 대소문자를 구분하지 않습니다. 쉼표로 여러 방법을 분리할 수 있습니다.

새로운 SIP 방법은 향후에 정의될 수도 있기 때문에, 사용자는 또한 사용자 지정 방법, 즉, 현재 정의된 SIP 방법이 아닌 방법을 지정할 수 있습니다. 수락된 필드 값은 RFC 2616에서 정의되는데, =, (, 및)와 같은 제어 문자 및 구분 기호를 제외한 모든 문자를 허용합니다. 제외되는 구분 기호의 전체 목록을 보려면 RFC 2616을 참고하십시오. 시스템은 트래픽의 지정된 사용자 지정 메서드가 발생할 경우, 패킷 헤더를 검사하지만 메시지는 검사하지 않습니다.

시스템은 최대 32개의 메서드까지 지원하는데, 최근 정의된 메서드 21개에 메서드 11개를 추가한 것입니다. 시스템은 사용자가 구성할 수 있는 정의되지 않은 모든 메서드를 무시합니다. 총 32개의 메

서드는 SIP 전처리기 옵션을 **Methods to Check(**확인하는 메서드**)**를 사용하여 지정된 방법을 포함한다는 점에 유의하십시오.

무효화를 사용하면 단 한 개의 방법만 지정할 수 있습니다. 예를 들면 다음과 같습니다.

```
!invite
```

그러나 규칙 내 여러 `sip_method` 키워드는 **AND** 작업으로 연결되어 있다는 점을 참고하십시오. 예를 들어 `invite` 및 `cancel` 외의 모든 추출된 메서드를 테스트하려면 두 개의 부정 `sip_method` 키워드를 사용할 수 있습니다.

```
sip_method: !invite
sip_method: !cancel
```

Cisco는 `sip_method` 키워드를 포함하는 규칙에 최소 하나의 `content` 키워드를 포함할 것을 권장합니다. 이는 규칙 엔진이 빠른 패턴 매처를 사용하도록 하여 처리 속도를 높이고 성능을 향상시키기 위함입니다. 규칙에 최소 하나의 `content` 키워드가 포함될 때, `content` 키워드 **Use Fast Pattern Matcher(**빠른 패턴 매처 사용**)** 인수를 활성화했는지 여부에 상관없이 규칙 엔진이 빠른 패턴 매처를 사용한다는 점에 유의하십시오.

관련 항목

SIP 전처리기 옵션
content 및 protected_content 키워드, 20 페이지
content 키워드 빠른 패턴 매처 인수, 30 페이지

sip_stat_code 키워드

각 SIP 응답에 있는 세 자리의 상태 코드는 요청된 작업의 결과를 나타냅니다. `sip_stat_code` 키워드를 사용하여 특정 상태 코드에 대한 SIP 응답을 테스트할 수 있습니다.

한 자리 응답 유형 번호 1-9, 특정 세 자리 번호 100-999 또는 둘의 어느 조합으로도 쉼표로 구분된 목록을 지정할 수 있습니다. 목록은 목록의 단일 번호가 SIP 응답의 코드에 일치하는 경우 일치합니다.

다음 표에서는 지정할 수 있는 SIP 상태 코드 값을 나타냅니다.

표 ***53: sip_stat_code*** 값

탐지 대상	지정 대상	예시	탐지 대상
특정 상태 코드	3자리 상태 코드	189	189
지정한 단일 숫자로 시작되는 3자리 코드	단일 디지트	1	1xx. 즉, 100, 101, 102 등
값 목록	쉼표로 구분된 특정 코드 및 단일 숫자의 모든 조합	222, 3	222 + 300, 301, 302 등

규칙에 `content` 키워드가 포함되었는지 여부와 상관없이, 규칙 엔진은 `sip_stat_code` 키워드로 지정한 값을 검색하는 데 fast pattern matcher를 사용하지 않습니다.

GTP 키워드

세 개의 GTP(GSRP 터널링 프로토콜) 키워드를 통해 GTP 버전, 메시지 유형 및 정보 요소에 대한 GTP 명령 계통을 검사할 수 있습니다. GTP 키워드는 다른 침입 규칙 키워드(예: `content` 또는 `byte_jump`)와 함께 사용할 수 없습니다. `gtp_info` 또는 `gtp_type` 키워드를 사용하는 각 규칙에서 `gtp_version` 키워드를 사용해야 합니다.

gtp_version 키워드

GTP 버전 0, 1, 2를 위한 GTP 컨트롤 메시지를 검사하려면 `gtp_version` 키워드를 사용할 수 있습니다.

GTP 버전마다 정의하는 메시지 유형 및 정보 요소가 다르기 때문에 `gtp_type` 또는 `gtp_info` 키워드를 사용할 때 반드시 `gtp_version`을 사용해야 합니다. 값 0, 1, 2를 지정할 수 있습니다.

gtp_type 키워드

각 GTP 메시지는 숫자 값 및 문자열 모두로 구성된 메시지 유형으로 식별됩니다. `gtp_type` 키워드를 사용하여 트래픽에서 특정 GTP 메시지 유형을 검사할 수 있습니다. GTP 버전마다 정의하는 메시지 유형 및 정보 요소가 다르기 때문에 `gtp_type` 또는 `gtp_info` 키워드를 사용할 때 반드시 `gtp_version`도 사용해야 합니다.

다음 예에 나온 것처럼 메시지 유형에 대해 정의된 10진수 값, 정의된 문자열, 둘 중 하나 또는 둘 모두의 쉼표로 구분된 조합의 목록을 지정할 수 있습니다.

```
10, 11, echo_request
```

시스템은 OR 연산자를 사용하여 사용자가 나열하는 각 값 또는 문자열에 일치시킵니다. 사용자가 값 및 문자열을 표시하는 순서는 상관없습니다. 목록의 단일 값 또는 문자열과 키워드를 일치시킵니다. 인식되지 않은 문자열 또는 외부 범위 값을 포함하는 규칙을 저장하려고 하는 경우 오류 메시지가 표시됩니다.

동일한 메시지 유형을 위한 서로 다른 값을 사용하는 다양한 GTP 버전이 표에 있다는 점에 유의하십시오. 예를 들어, `sgsn_context_request` 메시지 유형은 GTPv0 및 GTPv1에서 50의 값을 갖지만, GTPv2에서는 130의 값을 갖습니다.

패킷의 버전 번호에 따라 `gtp_type` 키워드는 서로 다른 값에 일치합니다. 위의 예제에서, 키워드는 GTPv0 또는 GTPv1 패킷의 메시지 유형 값 50 및 GTPv2 패킷의 값 130에 일치합니다. 패킷의 메시지 유형 값이 패킷에 지정된 버전에 대해 알려진 값이 아닌 경우 패킷이 키워드에 일치하지 않습니다.

메시지 유형을 위해 정수를 지정한 경우, 키워드의 메시지 유형이 GTP 패킷의 값에 일치하는 경우, 패킷에 지정된 버전에 관계없이 키워드는 일치합니다.

다음 표에서는 각 GTP 메시지 유형에 대해 시스템에서 인식하는 정의된 값 및 문자열을 나열합니다.

표 ***54: GTP*** 메시지 유형

값	버전 **0**	버전 **1**	버전 **2**
1	echo_request	echo_request	echo_request
2	echo_response	echo_response	echo_response
3	version_not_supported	version_not_supported	version_not_supported

값	버전 **0**	버전 **1**	버전 **2**
4	node_alive_request	node_alive_request	해당 없음
5	node_alive_response	node_alive_response	해당 없음
6	redirection_request	redirection_request	해당 없음
7	redirection_response	redirection_response	해당 없음
16	create_pdp_context_request	create_pdp_context_request	해당 없음
17	create_pdp_context_response	create_pdp_context_response	해당 없음
18	update_pdp_context_request	update_pdp_context_request	해당 없음
19	update_pdp_context_response	update_pdp_context_response	해당 없음
20	delete_pdp_context_request	delete_pdp_context_request	해당 없음
21	delete_pdp_context_response	delete_pdp_context_response	해당 없음
22	create_aa_pdp_context_request	init_pdp_context_activation_request	해당 없음
23	create_aa_pdp_context_response	init_pdp_context_activation_response	해당 없음
24	delete_aa_pdp_context_request	해당 없음	해당 없음
25	delete_aa_pdp_context_response	해당 없음	해당 없음
26	error_indication	error_indication	해당 없음
27	pdu_notification_request	pdu_notification_request	해당 없음
28	pdu_notification_response	pdu_notification_response	해당 없음
29	pdu_notification_reject_request	pdu_notification_reject_request	해당 없음
30	pdu_notification_reject_response	pdu_notification_reject_response	해당 없음
31	해당 없음	supported_ext_header_notification	해당 없음
32	send_routing_info_request	send_routing_info_request	create_session_request
33	send_routing_info_response	send_routing_info_response	create_session_response
34	failure_report_request	failure_report_request	modify_bearer_request
35	failure_report_response	failure_report_response	modify_bearer_response
36	note_ms_present_request	note_ms_present_request	delete_session_request
37	note_ms_present_response	note_ms_present_response	delete_session_response

값	버전 0	버전 1	버전 2
38	해당 없음	해당 없음	change_notification_request
39	해당 없음	해당 없음	change_notification_response
48	identification_request	identification_request	해당 없음
49	identification_response	identification_response	해당 없음
50	sgsn_context_request	sgsn_context_request	해당 없음
51	sgsn_context_response	sgsn_context_response	해당 없음
52	sgsn_context_ack	sgsn_context_ack	해당 없음
53	해당 없음	forward_relocation_request	해당 없음
54	해당 없음	forward_relocation_response	해당 없음
55	해당 없음	forward_relocation_complete	해당 없음
56	해당 없음	relocation_cancel_request	해당 없음
57	해당 없음	relocation_cancel_response	해당 없음
58	해당 없음	forward_srns_contex	해당 없음
59	해당 없음	forward_relocation_complete_ack	해당 없음
60	해당 없음	forward_srns_contex_ack	해당 없음
64	해당 없음	해당 없음	modify_bearer_command
65	해당 없음	해당 없음	modify_bearer_failure_indication
66	해당 없음	해당 없음	delete_bearer_command
67	해당 없음	해당 없음	delete_bearer_failure_indication
68	해당 없음	해당 없음	bearer_resource_command
69	해당 없음	해당 없음	bearer_resource_failure_indication
70	해당 없음	ran_info_relay	downlink_failure_indication
71	해당 없음	해당 없음	trace_session_activation
72	해당 없음	해당 없음	trace_session_deactivation
73	해당 없음	해당 없음	stop_paging_indication
95	해당 없음	해당 없음	create_bearer_request

값	버전 **0**	버전 **1**	버전 **2**
96	해당 없음	mbms_notification_request	create_bearer_response
97	해당 없음	mbms_notification_response	update_bearer_request
98	해당 없음	mbms_notification_reject_request	update_bearer_response
99	해당 없음	mbms_notification_reject_response	delete_bearer_request
100	해당 없음	create_mbms_context_request	delete_bearer_response
101	해당 없음	create_mbms_context_response	delete_pdn_request
102	해당 없음	update_mbms_context_request	delete_pdn_response
103	해당 없음	update_mbms_context_response	해당 없음
104	해당 없음	delete_mbms_context_request	해당 없음
105	해당 없음	delete_mbms_context_response	해당 없음
112	해당 없음	mbms_register_request	해당 없음
113	해당 없음	mbms_register_response	해당 없음
114	해당 없음	mbms_deregister_request	해당 없음
115	해당 없음	mbms_deregister_response	해당 없음
116	해당 없음	mbms_session_start_request	해당 없음
117	해당 없음	mbms_session_start_response	해당 없음
118	해당 없음	mbms_session_stop_request	해당 없음
119	해당 없음	mbms_session_stop_response	해당 없음
120	해당 없음	mbms_session_update_request	해당 없음
121	해당 없음	mbms_session_update_response	해당 없음
128	해당 없음	ms_info_change_request	identification_request
129	해당 없음	ms_info_change_response	identification_response
130	해당 없음	해당 없음	sgsn_context_request
131	해당 없음	해당 없음	sgsn_context_response
132	해당 없음	해당 없음	sgsn_context_ack
133	해당 없음	해당 없음	forward_relocation_request

값	버전 0	버전 1	버전 2
134	해당 없음	해당 없음	forward_relocation_response
135	해당 없음	해당 없음	forward_relocation_complete
136	해당 없음	해당 없음	forward_relocation_complete_ack
137	해당 없음	해당 없음	forward_access
138	해당 없음	해당 없음	forward_access_ack
139	해당 없음	해당 없음	relocation_cancel_request
140	해당 없음	해당 없음	relocation_cancel_response
141	해당 없음	해당 없음	configuration_transfer_tunnel
149	해당 없음	해당 없음	detach
150	해당 없음	해당 없음	detach_ack
151	해당 없음	해당 없음	cs_paging
152	해당 없음	해당 없음	ran_info_relay
153	해당 없음	해당 없음	alert_mme
154	해당 없음	해당 없음	alert_mme_ack
155	해당 없음	해당 없음	ue_activity
156	해당 없음	해당 없음	ue_activity_ack
160	해당 없음	해당 없음	create_forward_tunnel_request
161	해당 없음	해당 없음	create_forward_tunnel_response
162	해당 없음	해당 없음	suspend
163	해당 없음	해당 없음	suspend_ack
164	해당 없음	해당 없음	resume
165	해당 없음	해당 없음	resume_ack
166	해당 없음	해당 없음	create_indirect_forward_tunnel_request
167	해당 없음	해당 없음	create_indirect_forward_tunnel_response
168	해당 없음	해당 없음	delete_indirect_forward_tunnel_request
169	해당 없음	해당 없음	delete_indirect_forward_tunnel_response

값	버전 **0**	버전 **1**	버전 **2**
170	해당 없음	해당 없음	release_access_bearer_request
171	해당 없음	해당 없음	release_access_bearer_response
176	해당 없음	해당 없음	downlink_data
177	해당 없음	해당 없음	downlink_data_ack
179	해당 없음	해당 없음	pgw_restart
180	해당 없음	해당 없음	pgw_restart_ack
200	해당 없음	해당 없음	update_pdn_request
201	해당 없음	해당 없음	update_pdn_response
211	해당 없음	해당 없음	modify_access_bearer_request
212	해당 없음	해당 없음	modify_access_bearer_response
231	해당 없음	해당 없음	mbms_session_start_request
232	해당 없음	해당 없음	mbms_session_start_response
233	해당 없음	해당 없음	mbms_session_update_request
234	해당 없음	해당 없음	mbms_session_update_response
235	해당 없음	해당 없음	mbms_session_stop_request
236	해당 없음	해당 없음	mbms_session_stop_response
240	data_record_transfer_request	data_record_transfer_request	해당 없음
241	data_record_transfer_response	data_record_transfer_response	해당 없음
254	해당 없음	end_marker	해당 없음
255	pdu	pdu	해당 없음

gtp_info 키워드

GTP 메시지는 여러 요소 정보를 포함할 수 있는데, 이들 각각은 정의된 숫자 값 및 정의된 문자열 모두에서 확인됩니다. `gtp_info` 키워드를 사용하여 지정된 정보 요소의 시작 부분에서 검사를 시작하고 지정된 정보 요소로 검사를 제한할 수 있습니다. GTP 버전마다 정의하는 메시지 유형 및 정보 요소가 다르기 때문에 이 키워드를 사용할 때 `gtp_version`도 사용해야 합니다.

정보 요소에 대해 정의된 십진수 또는 정의된 문자열을 지정할 수 있습니다. 단일 값 또는 문자열을 지정할 수 있고, 규칙 안에서 여러 `gtp_info` 키워드를 사용하여 여러 요소 정보를 검사할 수 있습니다.

메시지가 동일한 유형의 여러 요소 정보를 포함할 때, 모두가 일치 여부를 위해 검사됩니다. 정보 요소가 잘못된 순서대로 발생하는 경우, 마지막 인스턴스만 검사됩니다.

동일한 정보 요소에 대해 서로 다른 값을 사용하는 다양한 GTP 버전이 있다는 점에 유의하십시오. 예를 들어, `cause` 정보 요소는 GTPv0 및 GTPv1에서 1의 값을 갖지만, GTPv2에서는 2의 값을 갖습니다.

패킷의 버전 번호에 따라 `gtp_info` 키워드는 서로 다른 값에 일치합니다. 위의 예제에서, 키워드는 GTPv0 또는 GTPv1 패킷의 정보 요소 값 1 및 GTPv2 패킷의 값 2에 일치합니다. 패킷의 정보 요소 값이 패킷에 지정된 버전에 대해 알려진 값이 아닌 경우 패킷이 키워드에 일치하지 않습니다.

정보 요소 값에 정수를 지정한 경우, 키워드의 메시지 유형이 GTP 패킷의 값에 일치하는 경우, 패킷에 지정된 버전에 관계없이 키워드에 일치합니다.

다음 표에서는 각 GTP 정보 요소에 대해 시스템에서 인식하는 값 및 문자열을 나열합니다.

표 ***55: GTP*** 정보 요소

값	버전 **0**	버전 **1**	버전 **2**
1	cause	cause	imsi
2	imsi	imsi	cause
3	rai	rai	recovery
4	tlli	tlli	해당 없음
5	p_tmsi	p_tmsi	해당 없음
6	qos	해당 없음	해당 없음
8	recording_required	recording_required	해당 없음
9	authentication	authentication	해당 없음
11	map_cause	map_cause	해당 없음
12	p_tmsi_sig	p_tmsi_sig	해당 없음
13	ms_validated	ms_validated	해당 없음
14	recovery	recovery	해당 없음
15	selection_mode	selection_mode	해당 없음
16	flow_label_data_1	teid_1	해당 없음
17	flow_label_signalling	teid_control	해당 없음
18	flow_label_data_2	teid_2	해당 없음

값	버전 **0**	버전 **1**	버전 **2**
19	ms_unreachable	teardown_ind	해당 없음
20	해당 없음	nsapi	해당 없음
21	해당 없음	ranap	해당 없음
22	해당 없음	rab_context	해당 없음
23	해당 없음	radio_priority_sms	해당 없음
24	해당 없음	radio_priority	해당 없음
25	해당 없음	packet_flow_id	해당 없음
26	해당 없음	charging_char	해당 없음
27	해당 없음	trace_ref	해당 없음
28	해당 없음	trace_type	해당 없음
29	해당 없음	ms_unreachable	해당 없음
71	해당 없음	해당 없음	apn
72	해당 없음	해당 없음	ambr
73	해당 없음	해당 없음	ebi
74	해당 없음	해당 없음	ip_addr
75	해당 없음	해당 없음	mei
76	해당 없음	해당 없음	msisdn
77	해당 없음	해당 없음	indication
78	해당 없음	해당 없음	pco
79	해당 없음	해당 없음	paa
80	해당 없음	해당 없음	bearer_qos
80	해당 없음	해당 없음	flow_qos
82	해당 없음	해당 없음	rat_type
83	해당 없음	해당 없음	serving_network
84	해당 없음	해당 없음	bearer_tft
85	해당 없음	해당 없음	tad

값	버전 0	버전 1	버전 2
86	해당 없음	해당 없음	uli
87	해당 없음	해당 없음	f_teid
88	해당 없음	해당 없음	tmsi
89	해당 없음	해당 없음	cn_id
90	해당 없음	해당 없음	s103pdf
91	해당 없음	해당 없음	s1udf
92	해당 없음	해당 없음	delay_value
93	해당 없음	해당 없음	bearer_context
94	해당 없음	해당 없음	charging_id
95	해당 없음	해당 없음	charging_char
96	해당 없음	해당 없음	trace_info
97	해당 없음	해당 없음	bearer_flag
99	해당 없음	해당 없음	pdn_type
100	해당 없음	해당 없음	pti
101	해당 없음	해당 없음	drx_parameter
103	해당 없음	해당 없음	gsm_key_tri
104	해당 없음	해당 없음	umts_key_cipher_quin
105	해당 없음	해당 없음	gsm_key_cipher_quin
106	해당 없음	해당 없음	umts_key_quin
107	해당 없음	해당 없음	eps_quad
108	해당 없음	해당 없음	umts_key_quad_quin
109	해당 없음	해당 없음	pdn_connection
110	해당 없음	해당 없음	pdn_number
111	해당 없음	해당 없음	p_tmsi
112	해당 없음	해당 없음	p_tmsi_sig
113	해당 없음	해당 없음	hop_counter

값	버전 **0**	버전 **1**	버전 **2**
114	해당 없음	해당 없음	ue_time_zone
115	해당 없음	해당 없음	trace_ref
116	해당 없음	해당 없음	complete_request_msg
117	해당 없음	해당 없음	guti
118	해당 없음	해당 없음	f_container
119	해당 없음	해당 없음	f_cause
120	해당 없음	해당 없음	plmn_id
121	해당 없음	해당 없음	target_id
123	해당 없음	해당 없음	packet_flow_id
124	해당 없음	해당 없음	rab_contex
125	해당 없음	해당 없음	src_rnc_pdcp
126	해당 없음	해당 없음	udp_src_port
127	charge_id	charge_id	apn_restriction
128	end_user_address	end_user_address	selection_mode
129	mm_context	mm_context	src_id
130	pdp_context	pdp_context	해당 없음
131	apn	apn	change_report_action
132	protocol_config	protocol_config	fq_csid
133	gsn	gsn	channel
134	msisdn	msisdn	emlpp_pri
135	해당 없음	qos	node_type
136	해당 없음	authentication_qu	fqdn
137	해당 없음	tft	ti
138	해당 없음	target_id	mbms_session_duration
139	해당 없음	utran_trans	mbms_service_area
140	해당 없음	rab_setup	mbms_session_id

값	버전 0	버전 1	버전 2
141	해당 없음	ext_header	mbms_flow_id
142	해당 없음	trigger_id	mbms_ip_multicast
143	해당 없음	omc_id	mbms_distribution_ack
144	해당 없음	ran_trans	rfsp_index
145	해당 없음	pdp_context_pri	uci
146	해당 없음	addi_rab_setup	csg_info
147	해당 없음	sgsn_number	csg_id
148	해당 없음	common_flag	cmi
149	해당 없음	apn_restriction	service_indicator
150	해당 없음	radio_priority_lcs	detach_type
151	해당 없음	rat_type	ldn
152	해당 없음	user_loc_info	node_feature
153	해당 없음	ms_time_zone	mbms_time_to_transfer
154	해당 없음	imei_sv	throttling
155	해당 없음	camel	arp
156	해당 없음	mbms_ue_context	epc_timer
157	해당 없음	tmp_mobile_group_id	signalling_priority_indication
158	해당 없음	rim_routing_addr	tmgi
159	해당 없음	mbms_config	mm_srvcc
160	해당 없음	mbms_service_area	flags_srvcc
161	해당 없음	src_rnc_pdcp	nmbr
162	해당 없음	addi_trace_info	해당 없음
163	해당 없음	hop_counter	해당 없음
164	해당 없음	plmn_id	해당 없음
165	해당 없음	mbms_session_id	해당 없음
166	해당 없음	mbms_2g3g_indicator	해당 없음

값	버전 **0**	버전 **1**	버전 **2**
167	해당 없음	enhanced_nsapi	해당 없음
168	해당 없음	mbms_session_duration	해당 없음
169	해당 없음	addi_mbms_trace_info	해당 없음
170	해당 없음	mbms_session_repetition_num	해당 없음
171	해당 없음	mbms_time_to_data	해당 없음
173	해당 없음	bss	해당 없음
174	해당 없음	cell_id	해당 없음
175	해당 없음	pdu_num	해당 없음
177	해당 없음	mbms_bearer_capab	해당 없음
178	해당 없음	rim_routing_disc	해당 없음
179	해당 없음	list_pfc	해당 없음
180	해당 없음	ps_xid	해당 없음
181	해당 없음	ms_info_change_report	해당 없음
182	해당 없음	direct_tunnel_flags	해당 없음
183	해당 없음	correlation_id	해당 없음
184	해당 없음	bearer_control_mode	해당 없음
185	해당 없음	mbms_flow_id	해당 없음
186	해당 없음	mbms_ip_multicast	해당 없음
187	해당 없음	mbms_distribution_ack	해당 없음
188	해당 없음	reliable_inter_rat_handover	해당 없음
189	해당 없음	rfsp_index	해당 없음
190	해당 없음	fqdn	해당 없음
191	해당 없음	evolved_allocation1	해당 없음
192	해당 없음	evolved_allocation2	해당 없음
193	해당 없음	extended_flags	해당 없음
194	해당 없음	uci	해당 없음

값	버전 **0**	버전 **1**	버전 **2**
195	해당 없음	csg_info	해당 없음
196	해당 없음	csg_id	해당 없음
197	해당 없음	cmi	해당 없음
198	해당 없음	apn_ambr	해당 없음
199	해당 없음	ue_network	해당 없음
200	해당 없음	ue_ambr	해당 없음
201	해당 없음	apn_ambr_nsapi	해당 없음
202	해당 없음	ggsn_backoff_timer	해당 없음
203	해당 없음	signalling_priority_indication	해당 없음
204	해당 없음	signalling_priority_indication_nsapi	해당 없음
205	해당 없음	high_bitrate	해당 없음
206	해당 없음	max_mbr	해당 없음
251	charging_gateway_addr	charging_gateway_addr	해당 없음
255	private_extension	private_extension	private_extension

SCADA 키워드

규칙 엔진은 Modbus, DNP3, CIP 및 S7Commplus 규칙을 사용하여 특정 프로토콜 필드에 액세스합니다.

Modbus 키워드

Modbus 키워드는 단독으로 또는 `content` 및 `byte_jump`와 같은 다른 키워드와 조합하여 사용할 수 있습니다.

modbus_data

`modbus_data` 키워드를 사용하여 Modbus(모드버스) 요청 또는 응답 내 Data(데이터) 필드의 시작 부분을 나타낼 수 있습니다.

modbus_func

`modbus_func` 키워드를 사용하여 Modbus 애플리케이션 레이어 요청 또는 응답 헤더의 Function Code 필드에 일치시킬 수 있습니다. Modbus(모드버스) 기능 코드에 대해 단일 정의된 십진수 또는 단일 정의된 문자열을 지정할 수 있습니다.

다음 표에서는 Modbus(모드버스) 기능 코드를 위한 시스템에서 인식하는 정의된 값 및 문자열을 나열합니다.

표 ***56:*** 모드버스 기능 코드

값	문자열
1	read_coils
2	read_discrete_inputs
3	read_holding_registers
4	read_input_registers
5	write_single_coil
6	write_single_register
7	read_exception_status
8	diagnostics
11	get_comm_event_counter
12	get_comm_event_log
15	write_multiple_coils
16	write_multiple_registers
17	report_slave_id
20	read_file_record
21	write_file_record
22	mask_write_register
23	read_write_multiple_registers
24	read_fifo_queue
43	encapsulated_interface_transport

modbus_unit

`modbus_unit` 키워드를 사용하여 Modbus(모드버스) 요청 또는 응답 헤더에 Unit ID(장치 ID) 필드에 대한 단일 십진수와 일치시킬 수 있습니다.

DNP3 키워드

DNP3 키워드는 단독으로 또는 다른 키워드(예: `content` 또는 `byte_jump`)와 함께 사용할 수 있습니다.

dnp3_data

리어셈블된 DNP3 애플리케이션 레이어 조각의 시작 부분을 나타낼 때 `dnp3_data` 키워드를 사용할 수 있습니다.

DNP3 전처리기는 연결 레이어 프레임을 애플리케이션 레이어 조각으로 리어셈블합니다. `dnp3_data` 키워드는 각 애플리케이션 레이어 조각의 시작 부분을 나타냅니다. 다른 규칙 옵션은 데이터를 구분하고 16바이트마다 체크섬을 추가할 필요 없이 단편 내 리어셈블된 데이터를 일치시킬 수 있습니다.

dnp3_func

`dnp3_func` 키워드를 사용하여 DNP3 애플리케이션 레이어 요청 또는 응답 헤더의 Function Code(기능 코드) 필드에 일치시킬 수 있습니다. DNP3 기능 코드에 대해 단일 정의된 십진수 또는 단일 정의된 문자열을 지정할 수 있습니다.

다음 표에서는 DNP3 기능 코드의 시스템에서 인식하는 정의된 값 및 문자열을 나열합니다.

표 ***57: DNP3*** 기능 코드

값	문자열
0	confirm
1	read
2	write
3	select
4	operate
5	direct_operate
6	direct_operate_nr
7	immed_freeze
8	immed_freeze_nr
9	freeze_clear
10	freeze_clear_nr
11	freeze_at_time
12	freeze_at_time_nr
13	cold_restart
14	warm_restart

값	문자열
15	initialize_data
16	initialize_appl
17	start_appl
18	stop_appl
19	save_config
20	enable_unsolicited
21	disable_unsolicited
22	assign_class
23	delay_measure
24	record_current_time
25	open_file
26	close_file
27	delete_file
28	get_file_info
29	authenticate_file
30	abort_file
31	activate_config
32	authenticate_req
33	authenticate_err
129	response
130	unsolicited_response
131	authenticate_resp

dnp3_ind

`dnp3_ind` 키워드를 사용하여 DNP3 애플리케이션 레이어 응답 헤더의 Internal Indications(내부 표시) 필드의 플래그에 일치시킬 수 있습니다.

다음 예와 같이 알려진 단일 플래그 또는 쉼표로 구분된 플래그 목록에 대한 문자열을 지정할 수 있습니다.

```
class_1_events, class_2_events
```

여러 플래그를 지정할 때, 키워드는 목록의 모든 플래그에 일치시킵니다. 플래그 조합을 검색하려면, 규칙에서 `dnp3_ind` 키워드를 여러 번 사용합니다.

다음 목록은 정의된 DNP3 내부 표시 플래그를 시스템에서 인식하는 문자열 구문을 제공합니다.

```
class_1_events
class_2_events
class_3_events
need_time
local_control
device_trouble
device_restart
no_func_code_support
object_unknown
parameter_error
event_buffer_overflow
already_executing
config_corrupt
reserved_2
reserved_1
```

dnp3_obj

`dnp3_obj` 키워드를 사용하여 요청 또는 응답의 DNP3 개체 헤더에 일치시킬 수 있습니다.

DNP3 데이터는 아날로그 입력 및 이진 입력 등과 같은 다양한 유형의 일련의 DNP3 개체로 구성됩니다. 각 유형은 아날로그 입력 그룹, 이진 입력 그룹과 같이 그룹으로 식별되는데, 이들 각각은 십진수 값으로 식별됩니다. 각 그룹의 개체는 16비트 정수, 32비트 정수, 짧은 부동 소수점 등과 같은 개체 변수에 의해 추가 식별되며 이들 각각은 개체의 데이터 형식을 지정합니다. 각 유형의 개체 변수는 또한 십진수로 식별할 수 있습니다.

개체 헤더 그룹 유형에 대한 십진수 및 개체 변수 유형에 대한 십진수를 지정하여 개체 헤더를 식별합니다. 이 둘의 조합은 DNP3 개체의 특정 유형을 정의합니다.

CIP 및 ENIP 키워드

다음 키워드를 단독으로 또는 함께 사용하여 CIP 전처리기에 의해 식별된 CIP 및 ENIP 트래픽에 대해 공격을 식별하는 맞춤형 침입 규칙을 생성할 수 있습니다. 구성 가능한 키워드의 경우, 허용되는 범위 내에서 단일 정수를 지정합니다. 자세한 내용은 CIP 전처리기를 참조하십시오.

표 ***58:***

키워드	일치 항목	범위
`cip_attribute`	CIP 메시지의 개체 클래스/인스턴스 속성 필드입니다. 정의된 단일 정수 값을 지정합니다.	0 - 65535
`cip_class`	CIP 메시지의 개체 클래스 필드입니다. 정의된 단일 정수 값을 지정합니다.	0 - 65535
`cip_conn_path_class`	연결 경로의 개체 클래스입니다. 단일 정수 값을 지정합니다.	0 - 65535

키워드	일치 항목	범위
cip_instance	CIP 메시지의 인스턴스 ID 필드입니다. 단일 정수 값을 지정합니다.	0 - 4284927295
cip_req	서비스 요청 메시지입니다.	해당 없음
cip_rsp	서비스 응답 메시지입니다.	해당 없음
cip_service	CIP 서비스 요청 메시지의 서비스 필드입니다. 단일 정수 값을 지정합니다.	0 - 127
cip_status	CIP 서비스 응답 메시지의 상태 필드입니다. 단일 정수 값을 지정합니다.	0 - 255
enip_command	EthNet/IP 헤더의 명령 코드입니다. 단일 정수 값을 지정합니다.	0 - 65535
enip_req	EthNet/IP 요청 메시지입니다.	해당 없음
enip_rsp	EthNet/IP 응답 메시지입니다.	해당 없음

S7Commplus 키워드

S7Commplus 키워드를 단독으로 또는 함께 사용하여 CS7Commplus 전처리기에 의해 식별된 트래픽에 대해 공격을 식별하는 맞춤형 침입 규칙을 생성할 수 있습니다. 구성 가능한 키워드의 경우, 허용되는 범위 내에서 단일 정수를 지정합니다. 자세한 내용은 S7Commplus 전처리기를 참고하십시오.

다음 사항을 참고하십시오.

- 동일한 규칙의 여러 S7commplus 키워드는 AND-ed입니다.
- 동일한 규칙에서 여러 s7commplus_func 또는 s7commplus_opcode 키워드를 사용하면 규칙이 무효화되며 트래픽과 일치하지 않습니다. 이러한 키워드로 여러 값을 검색하려면 여러 규칙을 생성합니다.

s7commplus_content

S7Commplus 침입 규칙에서 content 또는 protected_content 키워드를 사용하기 전에 s7commplus_content 키워드를 사용하여 패킷 페이로드의 시작 부분에 커서를 놓습니다. 자세한 내용은 content 및 protected_content 키워드, 20 페이지를 참조하십시오.

s7commplus_func

s7commplus_func 키워드를 사용하여 S7Commplus 헤더의 다음 값 중 하나와 일치시킵니다.

- explore
- create object
- delete object

- setvariable
- getlink
- setmultivar
- getmultivar
- beginsequence
- endsequence
- invoke
- getvarsubstr
- 0x0 ~ 0xFFF

  숫자 식은 추가 값을 허용합니다.

s7commplus_opcode

`s7commplus_opcode` 키워드를 사용하여 S7Commplus 헤더의 다음 값 중 하나와 일치시킵니다.

- request
- response
- notification
- response2
- 0x0 ~ 0xFF

  숫자 식은 추가 값을 허용합니다.

패킷 특성

특정 패킷 특성을 가진 패킷에 대해서만 이벤트를 생성하는 규칙을 작성할 수 있습니다.

dsize

`dsize` 키워드는 패킷 페이로드 크기를 테스트합니다. 이 키워드와 보다 큼 연산자 및 보다 작음 연산자(`<`와 `>`)를 사용하여 값의 범위를 지정할 수 있습니다. 다음 구문을 사용하여 범위를 지정할 수 있습니다.

```
>number_of_bytes
<number_of_bytes
number_of_bytes<>number_of_bytes
```

예를 들어, 400바이트보다 큰 패킷 크기를 나타내려면, `dtype` 값으로 `>400`을 사용합니다. 500바이트 미만의 패킷 크기를 나타내려면, `<500`을 사용합니다. 400바이트와 500바이트 사이의 모든 패킷에 대한 규칙 트리거를 포함하는 것으로 지정하려면, `400<>500`을 사용합니다.

⚠

주의 `dsize` 키워드는 모든 전처리기가 해독하기 전에 패킷을 테스트합니다.

isdataat

`isdataat` 키워드는 데이터가 페이로드의 특정 위치에 있다는 것을 규칙 엔진이 확인하도록 지시합니다.

다음 표에서는 `isdataat` 키워드와 함께 사용할 수 있는 인수를 나열합니다.

표 ***59: isdataat*** 인수

인수	유형	설명
Offset(오프셋)	필수	페이로드에서 특정 위치입니다. 예를 들어, 패킷 페이로드에서 데이터가 50바이트에 나타난다는 것을 테스트하려면, 오프셋 값으로 `50`을 지정합니다. `!` 수식자는 `isdataat` 테스트의 결과를 무효화합니다. 페이로드에 일정량의 데이터가 나타나지 않으면 경고합니다. 기존의 `byte_extract` 변수 또는 `byte_math` 결과를 사용하여 이 인수의 값을 지정할 수도 있습니다.
Relative	선택 사항	위치가 마지막으로 성공한 콘텐츠 일치와 연결되도록 합니다. 상대적 위치를 지정한 경우, 카운터가 0바이트에서 시작하므로, 계산하는 마지막으로 성공한 콘텐츠 일치에서 앞으로 이동할 바이트 수에서 1를 빼 위치를 계산한다는 점에 유의하십시오. 예를 들어, 데이터가 마지막으로 성공한 콘텐츠 일치 후 아홉 번째 바이트에 표시되도록 지정하려면, `8`의 상대적 오프셋을 지정합니다.
Raw Data	선택 사항	Firepower System 전처리기에서 암호 해독 또는 애플리케이션 레이어 표준화가 이루어지기 전에 데이터가 원래 패킷 페이로드에 위치하도록 지정합니다. 이전 콘텐츠 일치 항목이 원시 데이터 패킷에 있는 경우 이 인수를 **Relative(**연결**)**와 함께 사용할 수 있습니다.

예를 들어, 콘텐츠 `foo`를 검색하는 규칙에서 `isdataat`에 대한 값이 다음과 같이 지정될 경우

- Offset =!10
- Relative = 활성화

페이로드가 끝나기 전 `foo` 다음에 오는 10바이트를 규칙 엔진이 탐지하지 않는 경우 시스템에서 경고합니다.

sameip

`sameip` 키워드는 패킷의 소스와 대상 IP 주소가 동일한지 테스트합니다. 이 키워드는 인수를 취하지 않습니다.

fragoffset

`fragoffset` 키워드는 단편화된 패킷의 오프셋을 테스트합니다. 일부 익스플로잇(예: WinNuke DoS 공격)은 특정 오프셋이 있는, 수동으로 생성된 패킷 프래그먼트를 사용하므로 이 키워드가 유용합니다.

예를 들어, 단편화된 패킷의 오프셋이 31337바이트인지 여부를 테스트하려면, `fragoffset` 값으로 `31337`을 지정합니다.

`fragoffset` 키워드에 인수를 지정할 때 다음 연산자를 사용할 수 있습니다.

표 ***60: fragoffset*** 키워드 인수 연산자

연산자	설명
!	아님
>	보다 큼
<	보다 작음

not 연산자(`!`)를 `<` 또는 `>`와 조합하여 사용할 수 없음에 유의하십시오.

cvs

`cvs` 키워드는 잘못된 형식의 CVS 항목에 대해 Concurrent Versions System(공동 버전 시스템, CVS) 트래픽을 테스트합니다. 공격자는 잘못된 형식의 항목을 사용하여 힙 오버플로를 강제하고 CSV 서버에서 악성 코드를 실행할 수 있습니다. 이 키워드는 두 개의 알려진 CVS 취약성에 대한 공격을 식별하는 데 사용될 수 있습니다. CVE-2004-0396(CVS 1.11 x~1.11.15, 1.12 x~1.12.7) 및 CVS-2004-0414(CVS 1.12 x~1.12.8, 1.11 x~1.11.16). `cvs` 키워드는 올바른 형식의 항목을 점검하고 잘못된 형식의 항목이 탐지되면 알림을 생성합니다.

사용자 규칙은 CVS 실행 포트를 포함해야 합니다. 또한, 트래픽이 발생할 수 있는 모든 포트는 CVS 세션을 위해 상태가 유지될 수 있도록 TCP 정책 내 스트림 리어셈블리에 대한 포트 목록에 추가해야 합니다. TCP 포트 2401(`pserver`) 및 514(`rsh`)는 스트림 리어셈블리가 발생하는 클라이언트 포트 목록에 포함됩니다. 그러나 사용자 서버가 `xinetd` 서버(즉, pserver)로 실행되는 경우, 이는 모든 TCP 포트에서 실행될 수 있다는 점에 유의하십시오. 스트림 리어셈블리 **Client Ports(**클라이언트 포트**)** 목록에 모든 비표준 포트를 추가합니다.

관련 항목

byte_extract 키워드, 38 페이지
TCP 스트림 전처리 옵션

활성 응답 키워드

resp와 **react** 키워드는 활성 응답을 시작하는 2가지 방법을 제공합니다. 키워드 하나를 포함하는 침입 규칙은 패킷이 규칙을 트리거하면 단일 활성 응답을 시작합니다. 활성 응답 키워드는 트리거된 TCP 규칙에 대한 응답으로 TCP 연결을 종료하거나 트리거된 UDP 규칙에 대한 응답으로 UDP 세션을 닫는 활성 응답을 시작할 수 있습니다. 침입 삭제 규칙에서의 활성 응답의 내용을 참조하십시오.

활성 응답은 방화벽을 대신하지는 않습니다. 다양한 이유가 있는데, 대표적인 이유는 공격자가 활성 응답을 무시하거나 우회하도록 선택할 수 있다는 것입니다.

활성 응답은 (라우티드 또는 투명을 포함하는) 인라인 구축에서 지원됩니다. 예를 들어 인라인 배포의 `react` 키워드에 대한 응답으로, 시스템은 연결의 각 첨단 트래픽에 TCP 재설정(RST) 패킷을 직접 삽입할 수 있는데, 이는 보통 연결을 종료하게 합니다. 활성 응답은 패시브 구축에는 지원되지 않으며 적합하지도 않습니다.

활성 응답이 다시 라우팅될 수 있기 때문에, 시스템은 TCP 재설정이 TCP 재설정을 시작하도록 허용하지 않습니다. 이는 활성 응답의 무한 시퀀스를 방지합니다. 시스템은 또한 표준 관행에 따라 ICMP에서 연결할 수 없는 패킷이 ICMP에서 연결할 수 없는 패킷을 시작하도록 허용하지 않습니다.

침입 규칙이 활성 응답을 시작한 후 TCP 스트림 전처리기를 구성하여 TCP 연결에서 추가 트래픽을 탐지할 수 있습니다. 전처리기가 추가 트래픽을 탐지하면, 연결 또는 세션의 양쪽 끝에 지정된 최대 값까지 추가 활성 응답을 보냅니다. 고급 전송/네트워크 전처리기 옵션의 **Maximum Active Responses(**최대 활성 응답**)** 및 **Minimum Response Seconds(**최소 응답 시간**(**초**))**를 참조하십시오.

관련 항목

침입 삭제 규칙에서의 활성 응답

resp 키워드

규칙 헤더에서 TCP 또는 UDP 프로토콜 지정 여부에 따라 `resp` 키워드를 사용하여 TCP 연결 또는 UDP 세션에 적극적으로 응답할 수 있습니다.

키워드 인수를 통해 패킷 방향을 지정하고 TCP 재설정(RST) 패킷 또는 ICMP에서 연결할 수 없는 패킷을 활성 응답으로 사용할지 여부를 지정할 수 있습니다.

TCP 재설정(RST) 패킷 또는 ICMP에서 연결할 수 없는 패킷 중 무엇이든 사용하여 TCP 연결을 종료할 수 있습니다. UDP 세션을 종료하려면 ICMP에서 연결할 수 없는 인수만 사용해야 합니다.

다양한 TCP 재설정 인수를 통해 패킷 소스, 대상 또는 둘 다에 대한 활성 응답을 대상으로 할 수도 있습니다. 모든 ICMP에서 연결할 수 없는 인수는 패킷 소스를 대상으로 하며, 이러한 인수를 사용하여 ICMP 네트워크, 호스트, 포트에서 연결할 수 없는 패킷 또는 셋 모두를 사용할지 여부를 지정할 수 있습니다.

다음 표에는 규칙이 트리거될 때 Firepower System이 수행할 작업을 정확히 지정하기 위해 `resp` 키워드와 함께 사용할 수 있는 인수가 나열되어 있습니다.

표 ***61: resp*** 인수

인수	설명
reset_source	TCP 재설정 패킷을 규칙을 시작했던 패킷을 전송한 엔드 포인트에 보냅니다. 또는, 이전 버전과의 호환성을 위해 지원되는 `rst_snd`를 지정할 수 있습니다.
reset_dest	TCP 재설정 패킷을 규칙을 트리거한 패킷의 해당 대상 엔드 포인트에 보냅니다. 또는, 이전 버전과의 호환성을 위해 지원되는 `rst_rcv`를 지정할 수 있습니다.
reset_both	TCP 재설정 패킷을 전송 및 수신 엔드 포인트 모두에 보냅니다. 또는, 이전 버전과의 호환성을 위해 지원되는 `rst_all`을 지정할 수 있습니다.

인수	설명
icmp_net	ICMP 네트워크에서 연결할 수 없는 메시지를 발신자에게 보냅니다.
icmp_host	ICMP 호스트에서 연결할 수 없는 메시지를 발신자에게 보냅니다.
icmp_port	ICMP 포트에서 연결할 수 없는 메시지를 발신자에게 보냅니다. 이 인수는 UDP 트래픽을 종료하는 데 사용됩니다.
icmp_all	다음 ICMP 메시지를 발신자에게 보냅니다. • 네트워크에서 연결할 수 없음 • 호스트에서 연결할 수 없음 • 포트에서 연결할 수 없음

예를 들어 규칙이 트리거될 때 연결의 양쪽을 재설정하도록 규칙을 구성하려면 `resp` 키워드에 대한 값으로 `reset_both`를 사용합니다.

쉼표로 구분된 목록을 사용하여 다음과 같이 여러 인수를 지정할 수 있습니다.

```
argument,argument,argument
```

react 키워드

`react` 키워드를 사용하여 패킷이 규칙을 트리거할 때 TCP 연결 클라이언트에 기본 HTML 페이지를 보낼 수 있습니다. HTML 페이지를 보낸 후, 시스템은 TCP 재설정 패킷을 사용하여 연결 양쪽 끝에 활성 응답을 시작합니다. `react` 키워드는 UDP 트래픽에 대한 활성 응답을 시작하지 않습니다.

선택적으로, 다음 인수를 지정할 수 있습니다.

```
msg
```

패킷이 `msg` 인수를 사용하는 `react` 규칙을 트리거할 때, HTML 페이지는 규칙 이벤트 메시지를 포함합니다.

`msg` 인수를 지정하지 않으면 HTML 페이지에 다음 메시지가 포함됩니다.

```
You are attempting to access a forbidden site.
Consult your system administrator for details.
```

참고 활성 응답이 다시 라우팅될 수 있으므로, HTML 페이지가 `react` 규칙을 시작하지 않도록 확인합니다. 이는 활성 응답의 무한 시퀀스를 방지합니다. Cisco는 프로덕션 환경에서 이를 활성화하기 전에 `react` 규칙을 광범위하게 테스트하는 것을 권장합니다.

관련 항목

규칙 구조, 2 페이지

detection_filter 키워드

지정된 패킷 수가 지정된 시간 안에 규칙을 트리거하지 않는 한 `detection_filter` 키워드를 사용하여 이벤트 생성에서 규칙을 방지할 수 있습니다. 이를 통해 규칙이 조기에 이벤트를 생성하는 것을 중지할 수 있습니다. 예를 들어, 몇 초 동안 둘 또는 세 번의 로그인 실패는 예상된 작업일 수 있지만, 동일한 시간 동안 많은 시도가 있었다면 무차별 암호 대입 공격(brute force attack)을 나타낼 수 있습니다.

`detection_filter` 키워드에는 시스템이 소스 또는 대상 IP 주소를 추적하는지 여부, 이벤트를 트리거하기 전에 탐지 기준이 충족해야 하는 횟수, 횟수를 세는 기간을 정의하는 인수가 필요합니다.

이벤트 트리거를 연기하려면 다음 구문을 사용합니다.

```
track by_src/by_dst, count count, seconds number_of_seconds
```

`track` 인수는 규칙의 탐지 기준에 맞는 패킷 수를 셀 때 패킷의 소스 또는 대상 IP 주소를 사용할지 여부를 지정합니다. 시스템이 이벤트 인스턴스를 추적하는 방식을 지정하려면 다음 표에 설명된 인수 값에서 선택합니다.

표 ***62: detection_filter*** 추적 인수

인수	설명
by_src	소스 IP 주소로 세는 탐지 기준입니다.
by_dst	대상 IP 주소로 세는 탐지 기준입니다.

`count` 인수는 규칙이 이벤트를 생성하기 전에 지정된 시간 내에 지정된 IP 주소에 대한 규칙을 트리거해야 하는 패킷 수를 지정합니다.

`seconds` 인수는 규칙이 이벤트를 생성하기 전에 패킷의 지정된 수가 규칙을 트리거해야 하는 초를 지정합니다.

콘텐츠 `foo`에 대한 패킷을 검색하고 다음 인수와 함께 `detection_filter` 키워드를 사용하는 규칙의 사례를 고려해 보십시오.

```
track by_src, count 10, seconds 20
```

예제에서, 규칙이 특정 소스 IP 주소에서 20초 내에 10개 패킷에서 `foo`를 탐지할 때까지 이벤트를 생성하지 않습니다. 시스템이 처음 20초 내에 `foo`를 포함하는 패킷을 7개만 검색할 경우, 어떤 이벤트도 생성되지 않습니다. 그러나, 처음 20초 안에 `foo`가 40번 발생하면 규칙은 30개의 이벤트를 생성하고, 20초가 경과할 때 카운트가 다시 시작됩니다.

임계값과 **detection_filter** 키워드 비교

`detection_filter` 키워드는 더 이상 사용되지 않는 `threshold` 키워드를 대체합니다. `threshold` 키워드는 하위 버전 호환성을 계속 지원하며 침입 정책 안에서 설정한 임계값과 동일하게 실행합니다.

`detection_filter` 키워드는 패킷이 규칙을 트리거하기 전에 적용되는 탐지 기능입니다. 규칙은 지정된 패킷을 카운트하기 전에 탐지된 패킷 트리거에 대한 이벤트를 생성하지 않으며, 인라인 배포에서 규칙이 패킷을 삭제하도록 설정된 경우라도 해당 패킷을 삭제하지 않습니다. 반대로, 규칙은 규칙을

트리거하고 지정된 패킷 수 후에 발생하는 이벤트를 생성하며, 인라인 배포에서 규칙이 패킷을 삭제하도록 설정된 경우라면 해당 패킷을 삭제합니다.

임계값 설정은 탐지 작업으로 귀결되지 않는 이벤트 알림 기능입니다. 이는 패킷이 이벤트를 트리거한 후 적용됩니다. 인라인 배포에서 패킷을 삭제하도록 설정된 규칙은 규칙 임계값과는 별개로 규칙을 트리거하는 모든 패킷을 삭제합니다.

침입 정책에서 침입 이벤트 임계값 지정, 침입 이벤트 억제, 속도 기반 공격 방지 기능을 원하는 대로 조합하여 `detection_filter` 키워드를 사용할 수 있습니다. 또한 더 이상 사용되지 않는 `threshold` 키워드를 침입 정책의 침입 이벤트 임계값 설정 기능과 함께 사용하는, 가져온 로컬 규칙을 활성화한 경우 정책 인증이 실패할 수 있다는 점을 참고하십시오.

관련 항목

- 침입 이벤트 임계값
- 침입 정책 삭제 구성
- 규칙 페이지에서 동적 규칙 상태 설정

tag 키워드

`tag` 키워드를 사용하여 시스템이 호스트 또는 세션에 대한 추가 트래픽을 로깅하도록 지시합니다. `tag` 키워드를 사용하여 캡처할 트래픽 유형 및 볼륨을 지정할 때 다음 구문을 사용합니다.

```
tagging_type, count, metric, optional_direction
```

다음 3개의 표에서는 사용 가능한 기타 인수를 설명합니다.

태그 지정의 2가지 유형을 선택할 수 있습니다. 다음 표에서는 태그 지정의 2가지 유형을 나타냅니다. 침입 규칙에 규칙 헤더 옵션만 구성한 경우 세션 태그 인수 유형은 다른 세션에서 가져온 것과 동일한 세션에서 가져온 패킷을 시스템이 로깅한다는 점에 유의하십시오. 동일한 세션에서 가져온 패킷을 그룹화하려면, 동일한 침입 규칙 내에서 하나 이상의 규칙 옵션(`flag` 키워드 또는 `content` 키워드)을 구성합니다.

표 ***63:*** 태그 인수

인수	설명
session	규칙을 트리거한 세션의 패킷을 로깅합니다.
host	규칙을 트리거한 패킷을 전송한 호스트의 패킷을 로깅합니다. 방향 수식자를 추가하여 호스트에서 가져오는 트래픽만 로깅(`src`)하거나 호스트로 이동하는 트래픽만 로깅(`dst`)할 수 있습니다.

트래픽을 얼마나 로깅할지 나타내려면 다음 인수를 사용합니다.

표 ***64: count*** 인수

인수	설명
count	규칙이 트리거된 후 로깅하려는 패킷 또는 시간(초)입니다. 이 측정 단위는 count 인수 다음에 오는 메트릭 인수로 지정됩니다.

다음 표에 설명된 메트릭 중 시간 단위나 트래픽 볼륨으로 로깅하려는 메트릭을 선택하십시오.

⚠

주의 높은 대역폭 네트워크는 초당 수천 개의 패킷을 볼 수 있고, 많은 수의 패킷을 태그하는 것은 성능에 심각한 영향을 미치게 될 수도 있으므로, 네트워크 환경에 맞춰 이 설정을 조정하십시오.

표 ***65:*** 로깅 메트릭 인수

인수	설명
packets	규칙이 트리거된 후 메트릭에 의해 지정된 패킷 수를 로깅합니다.
seconds	규칙이 트리거된 후 메트릭에 의해 지정된 시간(초)을 로깅합니다.

예를 들어, 다음 `tag` 키워드 값을 가진 규칙이 트리거되면

```
host, 30, seconds, dst
```

다음 30초 동안 클라이언트에서 호스트로 전송된 모든 패킷이 로깅됩니다.

flowbits 키워드

`flowbits` 키워드를 사용하여 세션에 상태 이름을 지정합니다. 이전에 표시된 상태에 따라 세션의 후속 패킷을 분석함으로써, 시스템은 단일 세션에서 여러 패킷을 포함하는 공격을 탐지하고 경고할 수 있습니다.

`flowbits` 상태 이름은 세션의 특정 부분에서 패킷에 할당된 사용자가 정의한 레이블입니다. 경고하기를 원하지 않는 패킷과 악성 패킷을 구별할 수 있도록 패킷 내용에 따라 상태 이름으로 패킷을 표시할 수 있습니다. 매니지드 디바이스 당 최대 1024개의 상태 이름을 정의할 수 있습니다. 예를 들어, 성공적인 로그인 후에만 발생하는 악성 패킷을 경고하려는 경우, `flowbits` 키워드를 사용하여 초기 로그인 시도를 구성하는 패킷을 제거할 수 있으므로 악성 패킷에만 집중할 수 있습니다. 먼저 `logged_in` 상태로 설정된 로그인이 있는 세션의 모든 패킷에 표시하는 규칙을 생성한 다음 `flowbits`가 첫 번째 규칙에 설정한 상태를 가진 패킷을 확인하고 해당 패킷에만 작동하는 두 번째 규칙을 생성하여 이 작업을 수행할 수 있습니다.

선택적 그룹 이름을 사용하면 상태 그룹에 상태 이름을 포함할 수 있습니다. 상태 이름은 여러 그룹에 속할 수 있습니다. 그룹과 연관되지 않은 상태는 상호 배타적이지 않으며, 따라서 그룹과 연관되지 않은 상태를 트리거하고 설정하는 규칙은 다른 현재 설정 상태에 영향을 주지 않습니다.

flowbits 키워드 옵션

다음 표에서는 flowbits 키워드에서 사용할 수 있는 연산자, 상태 및 그룹의 다양한 조합에 대해 설명합니다. 상태 이름은 영숫자 문자, 점(.), 밑줄(_), 대시(-)를 포함할 수 있습니다.

표 ***66: flowbits*** 옵션

연산자	상태 옵션	그룹	설명
set	state_name	선택 사항	패킷에 대해 지정된 상태를 설정합니다. 그룹이 정의된 경우 지정된 그룹의 상태를 설정합니다.
set	state_name&state_name	선택 사항	패킷에 대해 지정된 상태를 설정합니다. 그룹이 정의된 경우 지정된 그룹의 상태를 설정합니다.
setx	state_name	필수	패킷에 대해 지정된 그룹에서 지정한 상태를 설정하고, 그룹의 다른 모든 상태를 해제합니다.
setx	state_name&state_name	필수	패킷에 대해 지정된 그룹에서 지정한 상태를 설정하고, 그룹의 다른 모든 상태를 해제합니다.
unset	state_name	그룹 없음	패킷에 대해 지정된 상태를 해제합니다.
unset	state_name&state_name	그룹 없음	패킷에 대해 지정된 상태를 해제합니다.
unset	all	필수	지정된 그룹의 모든 상태를 해제합니다.
toggle	state_name	그룹 없음	설정된 경우 지정된 상태를 해제하고, 해제된 경우 지정된 상태를 설정합니다.
toggle	state_name&state_name	그룹 없음	설정된 경우 지정된 상태를 해제하고, 해제된 경우 지정된 상태를 설정합니다.
toggle	all	필수	지정된 그룹에 설정된 모든 상태를 해제하고, 지정된 그룹에서 해제된 모든 상태를 설정합니다.
isset	state_name	그룹 없음	지정된 상태가 패킷에서 설정되었는지 확인합니다.
isset	state_name&state_name	그룹 없음	지정된 상태가 패킷에서 설정되었는지 확인합니다.
isset	state_name|state_name	그룹 없음	지정된 상태 중 무엇이든 패킷에서 설정되었는지 확인합니다.
isset	any	필수	어느 상태든 지정된 그룹에서 설정되었는지 확인합니다.
isset	all	필수	모든 상태가 지정된 그룹에서 설정되었는지 확인합니다.

연산자	상태 옵션	그룹	설명
isnotset	state_name	그룹 없음	지정된 상태가 패킷에 설정되지 않았는지 확인합니다.
isnotset	state_name&state_name	그룹 없음	지정된 상태가 패킷에 설정되지 않았는지 확인합니다.
isnotset	state_name|state_name	그룹 없음	지정된 상태 중 무엇이든 패킷에 설정되지 않았는지 확인합니다.
isnotset	any	필수	어느 상태든 패킷에 설정되지 않았는지 확인합니다.
isnotset	all	필수	모든 상태가 패킷에 설정되지 않았는지 확인합니다.
reset	(상태 없음)	선택 사항	모든 패킷에 대한 모든 상태를 해제합니다. 그룹이 지정된 경우 그룹의 모든 상태를 해제합니다.
noalert	(상태 없음)	그룹 없음	이를 이벤트 발생을 억제할 다른 모든 연산자와 함께 사용합니다.

flowbits 키워드 사용 가이드라인

flowbits 키워드를 사용할 때 다음 사항에 유의하십시오.

- setx 연산자를 사용할 때, 지정된 상태는 지정된 그룹에만 속하며, 다른 그룹에는 속할 수 없습니다.
- setx 연산자는 여러 번 정의할 수 있으며, 각 인스턴스로 다른 상태와 동일 그룹을 지정합니다.
- setx 연산자를 사용하여 그룹을 지정할 때, 지정된 해당 그룹에서 set, toggle 또는 unset 연산자를 사용할 수 없습니다.
- isset 및 isnotset 연산자는 상태가 그룹 내에 있는지 여부에 상관없이 지정된 상태를 평가합니다.
- 침입 정책이 저장하는 동안 침입 정책은 다시 적용되며 액세스 제어 정책은 적용됩니다. (액세스 제어 정책이 하나의 침입 정책 또는 여러 침입 정책을 참조하는지 여부는 상관 없습니다.) 지정된 그룹 없이 isset 또는 isnotset 연산자를 포함하는 규칙을 활성화하고, 해당하는 상태 이름 및 프로토콜에 대한 flowbits 할당(set, setx, unset, toggle)에 영향을 주는 최소 하나의 규칙을 활성화하지 않는 경우, 해당하는 상태 이름의 flowbits 할당에 영향을 미치는 모든 규칙이 활성화됩니다.
- 침입 정책을 저장하고 침입 정책을 다시 적용하고 액세스 제어 정책을 적용하는 동안(액세스 제어 정책이 침입 정책 하나를 참조하든 여러 개를 참조하든 상관없이), 지정된 그룹과 함께 isset 또는 isnotset 연산자가 포함된 규칙을 활성화하면, flowbits 할당(set, setx, unset, toggle)에 영향을 주고 해당 그룹 이름을 정의하는 모든 규칙도 활성화됩니다.

flowbits 키워드 예시

이 섹션에서는 flowbits 키워드를 사용하는 세 가지 예를 제공합니다.

flowbits 키워드 예시: state_name을 사용한 구성

이것은 state_name을 사용한 flowbits 구성의 예입니다.

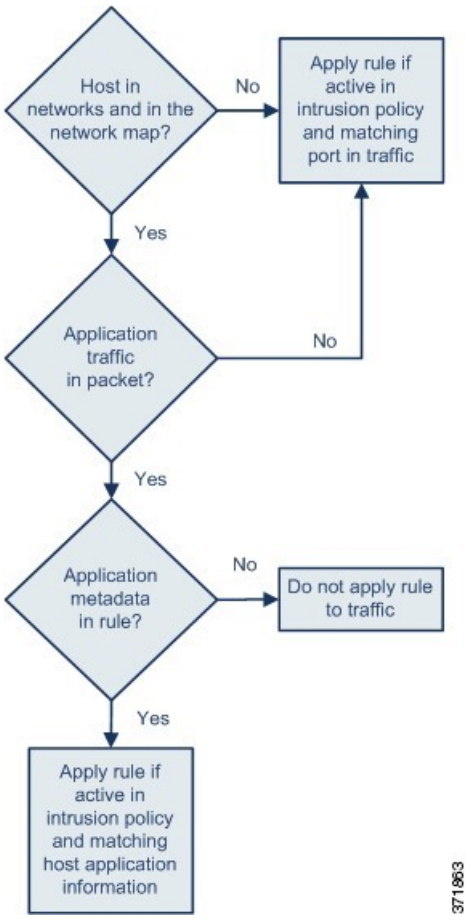
CVE ID 2000-0284에 설명된 IMAP 취약성을 고려하십시오. 이러한 취약성은 IMAP 구현에서 존재하는데, 특히 LIST, LSUB, RENAME, FIND 및 COPY 명령에 존재합니다. 그러나, 취약성을 사용하려면 공격자는 IMAP 서버에 로그인해야 합니다. IMAP 서버로부터의 LOGIN 확인 및 이를 따르는 익스플로잇은 서로 다른 패킷에 있어야 하므로 이러한 익스플로잇을 포착하는 비 플로우 기반(non-flow-based) 규칙을 작성하기는 어렵습니다. flowbits 키워드를 사용하여 사용자가 IMAP 서버에 로그인되어 있는지 여부를 추적하는 일련의 규칙을 구성할 수 있으며, 로그인되어 있는 경우, 공격 중 하나가 탐지되면 이벤트를 생성할 수 있습니다. 사용자가 로그인되어 있지 않은 경우, 공격은 취약성을 사용할 수 없고 어떤 이벤트도 생성되지 않습니다.

다음 두 가지 규칙 조각이 이 예를 보여줍니다. 첫 번째 규칙 조각은 IMAP 서버에서 IMAP 로그인 인증을 검색합니다.

```
alert tcp any 143 -> any any (msg:"IMAP login"; content:"OK

LOGIN"; flowbits:set,logged_in; flowbits:noalert;)
```

다음 다이어그램은 앞의 조각 규칙에서 flowbits 키워드의 영향에 대해 설명합니다.



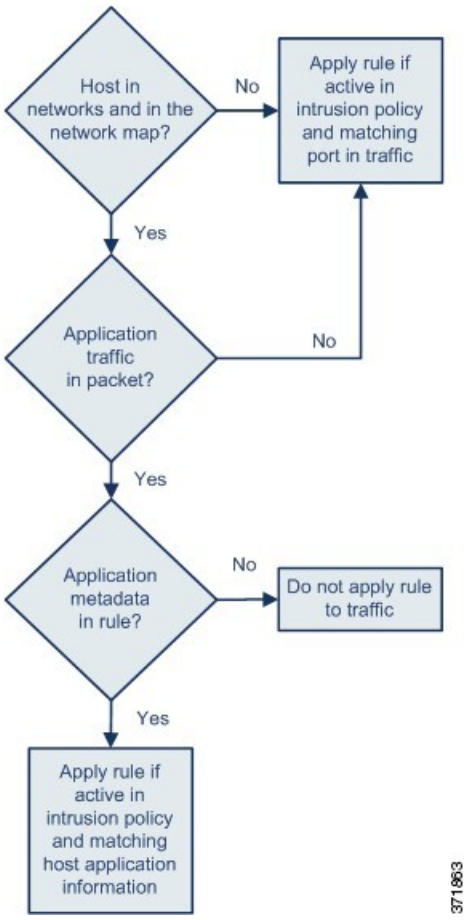
IMAP 서버에서 여러 무해한 로그인 세션을 볼 가능성이 크기 때문에 `flowbits:set`는 `logged_in`의 상태를 설정하는 반면, `flowbits:noalert`는 경고를 억제한다는 점에 유의하십시오.

logged_in 상태가 세션에서 일부의 이전 패킷의 결과로 설정되어 있지 않는 한 다음 규칙 조각은 LIST 문자열을 검색하지만 이벤트를 생성하지 않습니다:

```
alert tcp any any -> any 143 (msg:"IMAP LIST";

content:"LIST"; flowbits:isset,logged_in;)
```

다음 다이어그램은 앞의 조각 규칙에서 `flowbits` 키워드의 영향에 대해 설명합니다.



이 경우, 이전 패킷이 첫 번째 조각을 포함하는 규칙을 트리거하도록 야기하는 경우, 두 번째 조각을 포함하는 규칙이 트리거되고 이벤트를 생성합니다.

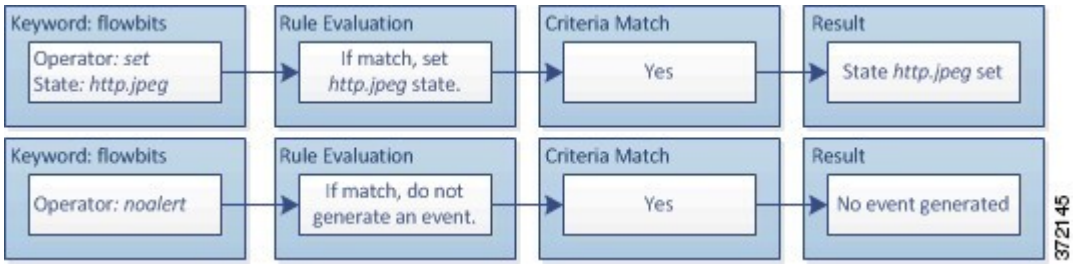
flowbits 키워드 예시: 오탐 이벤트의 구성 결과

그룹의 다양한 규칙에 설정된 여러 상태 이름을 포함하는 것은 뒤따르는 패킷의 콘텐츠가 더 이상 유효하지 않은 상태의 규칙에 일치할 때 발생할 가능성이 있는 잘못된 긍정 이벤트를 방지할 수 있습니다. 다음의 예제는 그룹의 여러 상태 이름을 포함하지 않을 때 잘못된 긍정을 얻을 수 있는 방법에 대해 설명합니다.

여기서 단일 세션 동안 표시된 다음 세 가지 규칙 조각이 순서대로 트리거하는 케이스를 고려해 보십시오.

```
(msg:"JPEG transfer";
content:"image/";pcre:"/^Content-?Type\x3a(\s*|\s*\r?\n\s+)image\x2fp?jpe?g/smi";
?flowbits:set,http.jpeg; flowbits:noalert;)
```

다음 다이어그램은 앞의 조각 규칙에서 `flowbits` 키워드의 영향에 대해 설명합니다.

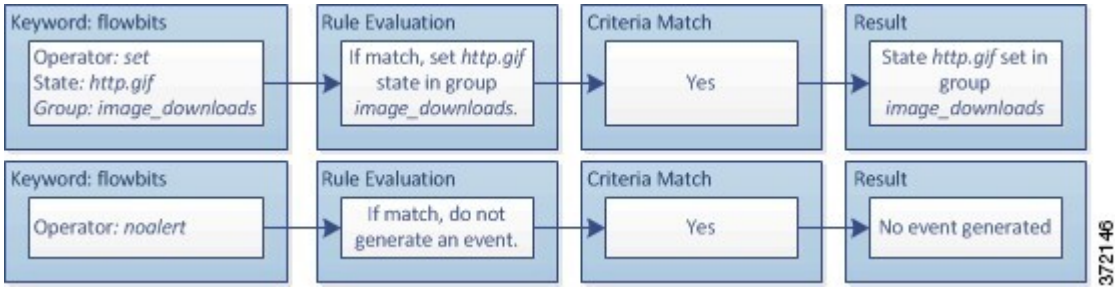


첫 번째 규칙 프래그먼트의 `content` 및 `pcre` 키워드는 JPEG 파일 다운로드를 매칭하고, `flowbits:set,http.jpeg`는 `http.jpeg flowbits` 상태를 설정하며, `flowbits:noalert`는 규칙의 이벤트 생성을 중지합니다. 규칙의 목적은 파일 다운로드를 탐지하고 `flowbits` 상태를 설정하는 것이며 하나 이상의 동반 규칙이 악성 콘텐츠와 조합된 상태 이름을 테스트하여 악성 콘텐츠가 탐지되면 이벤트를 생성할 수 있도록 하는 것이므로 어떤 이벤트도 생성되지 않습니다.

다음 규칙 조각은 위의 JPEG 파일 다운로드 다음에 일어나는 GIF 파일 다운로드를 탐지합니다.

```
(msg:"GIF transfer"; content:"image/";
pcre:"/^Content-?Type\x3a(\s*|\s*\r?\n\s+)image\x2fgif/smi";
?flowbits:set,http.jpg,image_downloads; flowbits:noalert;)
```

다음 다이어그램은 앞의 조각 규칙에서 `flowbits` 키워드의 영향에 대해 설명합니다.

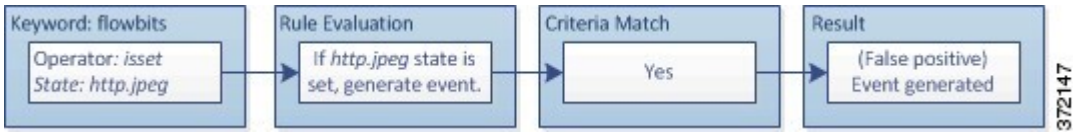


두 번째 규칙의 `content` 및 `pcre` 키워드는 GIF 파일 다운로드를 매칭하고, `flowbits:set,http.tif`는 `http.tif flowbits` 상태를 설정하며, `flowbits:noalert`는 규칙의 이벤트 생성을 중지합니다. 첫 번째 규칙 조각에 의해 설정된 `http.jpeg` 상태는 더 이상 필요 없더라도 여전히 설정되어 있다는 점에 유의하십시오. 이는 후속 GIF 다운로드가 발견되는 경우 JPEG 다운로드가 반드시 완료되어 있어야 하기 때문입니다.

세 번째 규칙 조각은 첫 번째 규칙 조각에 사용됩니다.

```
(msg:"JPEG exploit";?flowbits:isset,http.jpeg;content:"|FF|";
pcre:"?/\xFF[\xE1\xE2\xED\xFE]\x00[\x00\x01]/";)
```

다음 다이어그램은 앞의 조각 규칙에서 `flowbits` 키워드의 영향에 대해 설명합니다.



세 번째 규칙 프래그먼트에서 `flowbits:isset,http.jpeg`는 이제 관련이 없는 `http.jpeg` 상태가 설정되었는지, 그리고 `content` 및 `pcre`가 JPEG 파일에서는 악성이지만 GIF 파일에서는 악성이 아닌 내용

과 일치하는지를 확인합니다. 세 번째 규칙 조각은 JPEG 파일에는 없는 공격을 위한 잘못된 긍정 이벤트로 귀결됩니다.

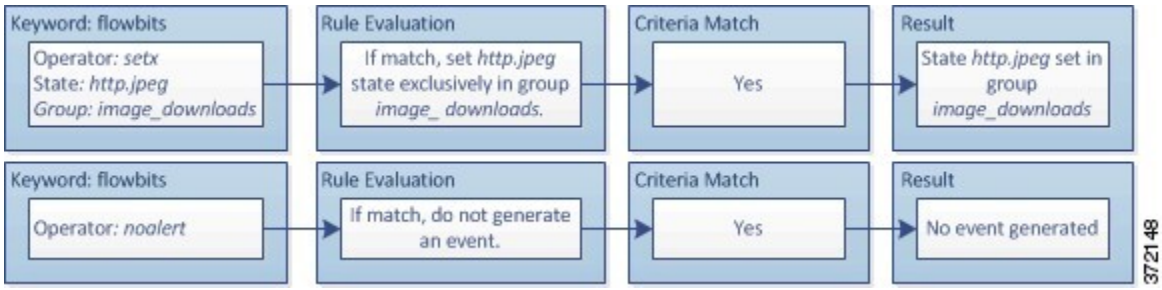
flowbits 키워드 예시: 오탐 이벤트 방지 구성

다음의 예시는 그룹에 상태 이름을 포함하고 `setx` 연산자를 사용하여 잘못된 긍정을 차단할 수 있는 방법에 대해 설명합니다.

이전 예와 동일한 경우를 고려하십시오. 이제는 동일한 상태 그룹에서 자신의 두 가지 상태 이름을 포함하는 처음 두 개의 규칙은 제외합니다.

```
(msg:"JPEG transfer";
content:"image/";pcre:"/^Content-?Type\x3a(\s*|\s*\r?\n\s+)image\x2fp?jpe?g/smi";
?flowbits:setx,http.jpeg,image_downloads; flowbits:noalert;)
```

다음 다이어그램은 앞의 조각 규칙에서 `flowbits` 키워드의 영향에 대해 설명합니다.

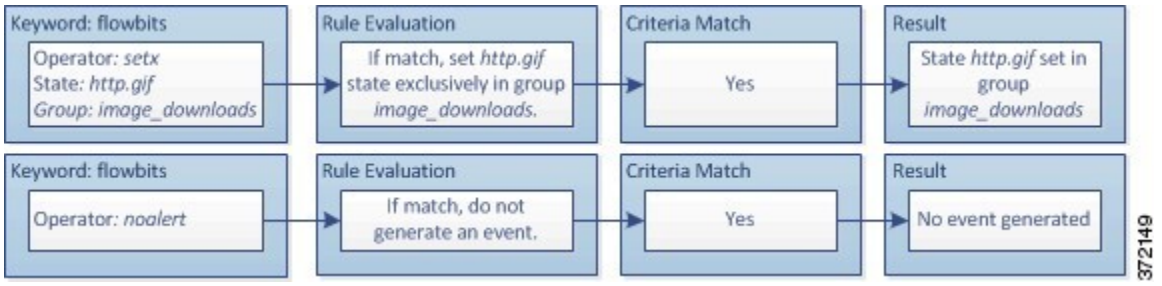


첫 번째 규칙 조각이 JPEG 파일 다운로드를 탐지하면, `flowbits:setx,http.jpeg,image_downloads` 키워드는 `flowbits` 상태를 `http.jpeg`로 설정하고 `image_downloads` 그룹의 상태를 포함합니다.

다음 규칙은 후속 GIF 파일 다운로드를 탐지합니다.

```
(msg:"GIF transfer"; content:"image/";
pcre:"/^Content-?Type\x3a(\s*|\s*\r?\n\s+)image\x2fgif/smi";
?flowbits:setx,http.jpg,image_downloads; flowbits:noalert;)
```

다음 다이어그램은 앞의 조각 규칙에서 `flowbits` 키워드의 영향에 대해 설명합니다.

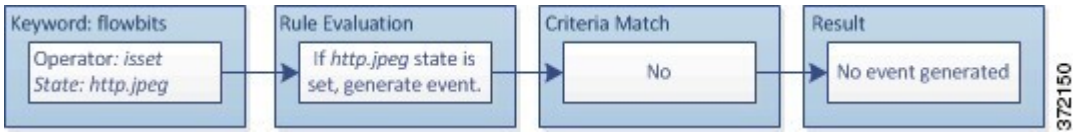


두 번째 규칙 프래그먼트가 GIF 다운로드와 일치하면 `flowbits:setx,http.tif,image_downloads` 키워드는 `http.jpg flowbits` 상태를 설정하고 그룹의 다른 상태인 `http.jpeg`를 설정 취소합니다.

세 번째 규칙 조각이 잘못된 긍정으로 귀결되지 않습니다.

```
(msg:"JPEG exploit"; ?flowbits:isset,http.jpeg;content:"|FF|";
pcre:"/?\xFF[\xE1\xE2\xED\xFE]\x00[\x00\x01]/";)
```

다음 다이어그램은 앞의 조각 규칙에서 `flowbits` 키워드의 영향에 대해 설명합니다.



`flowbits:isset,http.jpeg`가 거짓이므로, 규칙 엔진은 규칙 처리를 중지하고 어떤 이벤트도 생성되지 않으며, 따라서 GIF 파일 내 콘텐츠가 JPEG 파일에 대한 공격 콘텐츠에 일치하는 경우에도 잘못된 긍정이 차단됩니다.

http_encode 키워드

`http_encode` 키워드를 사용하여 표준화 전에 HTTP 요청 또는 응답의 인코딩 유형에서 이벤트를 생성할 수 있습니다. HTTP URI, HTTP 헤더의 비쿠키 데이터, HTTP 요청 헤더의 쿠키 또는 HTTP 응답의 set-cookie 데이터 중 하나에서 가능합니다.

`http_encode` 키워드를 사용하여 규칙에 대한 일치 항목을 반환하는 HTTP 응답 및 HTTP 쿠키를 검사하는 HTTP Inspect(HTTP 검사) 전처리기를 구성해야 합니다.

또한 HTTP Inspect(HTTP 검사) 전처리기 구성에서 각 특정 인코딩 유형의 디코딩 및 알림 옵션을 활성화해야 침입 규칙의 `http_encode` 키워드가 해당 인코딩 유형에서 이벤트를 트리거할 수 있습니다.

다음 표에서는 이 옵션이 HTTP URI, 헤더, 쿠키 및 set-cookie에서 이벤트를 생성할 수 있는 인코딩 유형을 설명합니다.

표 ***67: http_encode*** 인코딩 유형

인코딩 유형	설명
utf8	이 인코딩 유형이 HTTP Inspect(HTTP 검사) 전처리기에 의한 디코딩에 대해 활성화되는 경우 지정된 위치에서 UTF - 8 인코딩을 탐지합니다.
double_encode	이 인코딩 유형이 HTTP Inspect(HTTP 검사) 전처리기에 의한 디코딩에 대해 활성화되는 경우 지정된 위치에서 이중 인코딩을 탐지합니다.
non_ascii	비ASCII 문자가 검색되었지만 탐지된 인코딩 유형이 활성화되지 않은 경우 지정된 위치에서 비ASCII 문자를 탐지합니다.
uencode	이 인코딩 유형이 HTTP Inspect(HTTP 검사) 전처리기에 의한 디코딩에 대해 활성화되는 경우 지정된 위치에서 Microsoft %u 인코딩을 탐지합니다.
bare_byte	이 인코딩 유형이 HTTP Inspect(HTTP 검사) 전처리기에 의한 디코딩에 대해 활성화되는 경우 지정된 위치에서 베어 바이트 인코딩을 탐지합니다.

관련 항목

서버 레벨 HTTP 정상화 옵션

HTTP 검사 전처리기

http_encode 키워드 구문

인코딩 위치

HTTP URI, 헤더 또는 집합 쿠키를 포함하는 쿠키에서 지정된 인코딩 유형을 검색할지 여부를 지정합니다.

인코딩 유형

다음 형식 중 하나를 사용하여 하나 이상의 인코딩 유형을 지정합니다.

```
encode_type
encode_type|encode_type|encode_type...
```

여기서 `encode_type`은 다음 중 하나입니다.

```
utf8
double_encode
non_ascii
uencode
bare_byte.
```

부정 연산자(!)와 OR 연산자(|)를 함께 사용할 수 없음에 유의하십시오.

http_encode 키워드 예시: 2개의 http_endcode 키워드를 사용하여 2개의 인코딩 검색

다음 예는 같은 규칙에서 2개의 `http_encode` 키워드를 사용하여 HTTP URI에서 UTF-8 AND Microsoft IIS %u 인코딩을 검색합니다.

첫 번째, `http_encode` 키워드:

- **Encoding Location(**인코딩 위치**)**: `HTTP URI`
- **Encoding Type(**인코딩 유형**)**: `utf8`

그런 다음 추가적인 `http_encode` 키워드:

- **Encoding Location(**인코딩 위치**)**: `HTTP URI`
- **Encoding Type(**인코딩 유형**)**: `uencode`

개요: file_type 및 file_group 키워드

`file_type` 및 `file_group` 키워드를 사용하면 FTP, HTTP, SMTP, IMAP, POP3, 그리고 해당 유형 및 버전에 따라 NetBIOS ssn(SMB)을 통해 전송되는 파일을 탐지할 수 있습니다. 단일 침입 규칙에서 하나 이상의 `file_type` 또는 `file_group` 키워드를 사용하지 마십시오.

팁 VDB(취약성 데이터베이스)를 업데이트하면 침입 규칙 편집기가 최신 파일 형식, 버전 및 그룹으로 채워집니다.

참고 시스템은 file_type 및 file_group 키워드를 수용하기 위해 전처리기를 자동으로 활성화하지 않습니다.

file_type 또는 file_group 키워드와 일치하는 트래픽에 대한 이벤트를 생성하고, 인라인 구축에서 문제가 되는 패킷을 삭제합니다.을 위해서는 특정 전처리기를 활성화해야 합니다.

표 ***68: file_type*** 및 ***file_group*** 침입 이벤트 생성

프로토콜	필수 전처리기 또는 전처리기 옵션
FTP	FTP/텔넷 전처리기 및 **Normalize TCP Payload(TCP** 페이로드 표준화**)** 인라인 표준화 전처리기 옵션
HTTP	HTTP 트래픽에서 침입 이벤트를 생성하기 위한 HTTP 검사 전처리기
SMTP	HTTP 트래픽에서 침입 이벤트를 생성하기 위한 SMTP 전처리기
IMAP	IMAP 전처리기
POP3	POP 전처리기
Netbios-ssn(SMB)	DCE/RPC 전처리기 및 **SMB File Inspection(SMB** 파일 검사**)** DCE/RPC 전처리기 옵션

관련 항목

FTP/텔넷 디코더
인라인 정상화 전처리기
HTTP 검사 전처리기
SMTP 전처리기
IMAP 전처리기
POP 전처리기
DCE/RPC 전처리기

file_type 및 file_group 키워드

file_type

file_type 키워드를 사용하면 트래픽에서 탐지된 파일의 파일 유형 및 버전을 지정할 수 있습니다. 파일 유형 인수(예를 들어, **JPEG** 및 **PDF**)는 트래픽에서 찾을 파일의 형식을 식별합니다.

참고 file_type 키워드를 동일한 침입 규칙의 또 다른 file_type 또는 file_group 키워드와 사용해서는 안 됩니다.

시스템은 기본적으로 **Any Version(**모든 버전**)**을 선택하지만, 일부 파일 유형을 사용하면 트래픽에서 찾을 특정 파일 유형 버전을 식별할 수 있도록 버전 옵션을 선택할 수 있습니다(예를 들어, PDF 버전 **1.7**).

file_group

`file_group` 키워드를 사용하면 트래픽에서 Cisco가 정의한 비슷한 파일 유형 그룹을 선택할 수 있습니다(예를 들어 멀티미디어 또는 오디오). 파일 그룹은 또한 그룹 내 각 파일 유형에 대해 Cisco가 정의한 버전을 포함합니다.

참고 `file_group` 키워드를 동일한 침입 규칙의 또 다른 `file_group` 또는 `file_type` 키워드와 사용해서는 안 됩니다.

file_data 키워드

`file_data` 키워드는 `content`, `byte_jump`, `byte_test` 및 `pcre`와 같은 다른 키워드에 대해 사용 가능한 위치 인수를 위한 참고 사항으로 기능하는 포인터를 제공합니다. 탐지된 트래픽은 `file_data` 키워드가 나타내는 데이터 유형을 확인합니다. `file_data` 키워드를 사용하여 다음 페이로드 유형의 시작을 가리킬 수 있습니다.

- HTTP 응답 본문

  HTTP 응답 패킷을 검사하려면, HTTP Inspect(HTTP 검사) 전처리기를 활성화해야 하고 HTTP 응답을 검사하는 전처리기를 구성해야 합니다. HTTP Inspect(HTTP 검사) 전처리기가 HTTP 응답 본문 데이터를 탐지할 경우 `file_data` 키워드가 일치됩니다.

- 미압축 gzip 파일 데이터

  HTTP 응답 본문에서 압축되지 않은 gzip 파일을 검사하려면 HTTP Inspect(HTTP 검사) 전처리기를 활성화해야 하며, HTTP 응답을 검사하고 HTTP 응답 본문의 gzip 압축 파일을 압축 해제하도록 전처리기를 구성해야 합니다. 자세한 내용은 **Inspect HTTP Responses(HTTP** 응답 검사**)**와 **Inspect Compressed Data(**압축 데이터 검사**)** Server-Level HTTP Normalization(서버 수준 HTTP 표준화) 옵션을 참고하십시오. HTTP Inspect(HTTP 검사) 전처리기가 HTTP 응답 본문 내 미압축 gzip 데이터를 탐지할 경우 `file_data` 키워드가 일치됩니다.

- 표준화된 Javascript

  표준화된 Javascript 데이터를 검사하려면, HTTP Inspect(HTTP 검사) 전처리기를 활성화해야 하고 HTTP 응답을 검사하는 전처리기를 구성해야 합니다. HTTP Inspect(HTTP 검사) 전처리기가 HTTP 응답 본문 데이터 내 Javascript를 탐지할 경우 `file_data` 키워드가 일치됩니다.

- SMTP 페이로드

  SMTP 페이로드를 검사하려면, SMTP 전처리기를 활성화해야 합니다. `file_data` 키워드는 SMTP 프리프로세서가 SMTP 데이터를 탐지하는지 여부를 매칭합니다.

- SMTP, POP 또는 IMAP 트래픽의 인코딩된 이메일 첨부 파일

SMTP, POP 또는 IMAP 트래픽의 인코딩된 이메일 첨부 파일을 검사하려면 SMTP, POP 또는 IMAP 전처리기를 각각 활성화해야 하며 단독으로 또는 조합하여 활성화합니다. 다음, 각 활성화된 전처리기를 위해, 해독할 각 첨부 파일 인코딩 유형을 디코딩하기 위해 전처리기가 구성되어 있는지 확인해야 합니다. 사용자가 각 전처리기에 대해 구성할 수 있는 첨부 파일 디코딩 옵션은 다음과 같습니다: **Base64 Decoding Depth(**베이스**64** 디코딩 수준**)**, **7-Bit/8-Bit/Binary Decoding Depth(7**비트**/8**비트**/**이진 디코딩 수준**)**, **Quoted-Printable Decoding Depth(**발췌되어 인쇄 가능한 디코딩 수준**)** 그리고 **Unix-to-Unix Decoding Depth(**유닉스 투 유닉스 디코딩 수준**)**.

규칙에서 여러 `file_data` 키워드를 사용할 수 있습니다.

관련 항목

HTTP 검사 전처리기
서버 레벨 HTTP 정상화 옵션
SMTP 전처리기
IMAP 전처리기

pkt_data 키워드

`pkt_data` 키워드는 `content`, `byte_jump`, `byte_test` 및 `pcre`와 같은 다른 키워드에 대해 사용 가능한 위치 인수를 위한 참고 사항으로 기능하는 포인터를 제공합니다.

표준화된 FTP, 텔넷 또는 SMTP 트래픽이 발견되면, `pkt_data` 키워드는 표준화된 패킷 페이로드의 시작을 나타냅니다. 다른 트래픽이 발견되면 `pkt_data` 키워드는 원시 TCP 또는 UDP 페이로드의 시작을 나타냅니다.

다음 표준화 옵션은 시스템이 침입 규칙에 의한 검사를 위해 해당 트래픽을 표준화할 수 있도록 활성화되어야 합니다.

- 검사할 FTP 트래픽을 표준화하려면 FTP 및 텔넷 전처리기 **Detect Telnet Escape codes within FTP commands(FTP** 명령 내에서 텔넷 이스케이프 코드 탐지**)**를 활성화합니다.
- 검사할 텔넷 트래픽을 표준화하려면 FTP 및 텔넷 전처리기 **Normalize(**표준화**)** 텔넷 옵션을 활성화합니다.
- 검사할 SMTP 트래픽을 표준화하려면 SMTP 전처리기 **Normalize(**표준화**)** 옵션을 활성합니다.

규칙에서 여러 `pkt_data` 키워드를 사용할 수 있습니다.

관련 항목

클라이언트 레벨 FTP 옵션
텔넷 옵션
SMTP 전처리기 옵션

base64_decode 및 base64_data 키워드

`base64_decode` 및 `base64_data` 키워드를 함께 사용하여 규칙 엔진이 Base64 데이터로 지정된 데이터를 해독하고 검사하도록 지시할 수 있습니다. 이 방법은 예를 들면 HTTP PUT 및 POST 요청에서

Base64로 인코딩된 HTTP 인증 요청 헤더 및 Base64로 인코딩된 데이터를 검사하는 경우 유용할 수 있습니다.

이 키워드는 특히 HTTP 요청의 Base64 데이터를 해독하고 검사하는 데 유용합니다. 그러나, 또한 이 키워드를 여러 행 위의 길이가 긴 헤더 행을 확장하기 위해 HTTP가 스페이스 및 탭 문자를 사용하는 것과 같은 방식으로 이 문자를 사용하는 SMTP와 같은 모든 프로토콜과 함께 사용할 수 있습니다. 폴딩으로 알려진 이와 같은 행 확장이 이를 사용하는 프로토콜에 나타나지 않는 경우, 검사는 스페이스 또는 탭이 뒤따르지 않는 모든 복귀 또는 라인 피드에서 끝납니다.

base64_decode

`base64_decode` 키워드는 규칙 엔진이 패킷 데이터를 Base64 데이터로 해독하도록 지시합니다. 선택적 인수는 디코딩할 바이트 수와 디코딩을 시작할 데이터 내 위치를 지정하도록 합니다.

규칙에서 `base64_decode` 키워드를 한 번 사용할 수 있습니다. 이는 최소한 `base64_data` 키워드의 인스턴스 하나를 선행해야 합니다.

Base64 데이터를 해독하기 전에, 규칙 엔진은 여러 행을 가로질러 접힌 긴 헤더를 펼칩니다. 규칙 엔진에 다음이 발생할 때 디코딩은 종료됩니다.

- 헤더 행 끝
- 디코딩할 지정된 바이트 수
- 패킷 종료

다음 표에서는 `base64_decode` 키워드와 함께 사용할 수 있는 인수를 설명합니다.

표 ***69:*** 선택적 ***base64_*** 디코딩 인수

인수	설명
바이트	디코딩할 바이트 수를 지정합니다. 지정되지 않은 경우, 디코딩은 헤더 행 끝 또는 패킷 페이로드의 끝 중 먼저 오는 것까지 계속합니다. 0이 아닌 양수 값을 지정할 수 있습니다.
Offset	패킷 페이로드의 처음을 기준으로, 또는 **Relative**를 지정한 경우 현재 검사 위치를 기준으로 오프셋을 결정합니다. 0이 아닌 양수 값을 지정할 수 있습니다.
Relative	현재 검사 위치에 관련된 검사를 지정합니다.

base64_data

`base64_data` 키워드는 `base64_decode` 키워드를 사용하여 디코딩된 Base64 데이터 검사를 위한 참고 자료를 제공합니다. `base64_data` 키워드는 디코딩된 Base64 데이터 시작 시 검사가 시작되도록 설정합니다. 선택적으로, 검사할 위치를 추가로 지정하려면 `content` 또는 `byte_test`와 같은 기타 키워드에서 사용 가능한 위치 인수를 사용할 수 있습니다.

`base64_decode` 키워드를 사용한 후 `base64_data` 키워드를 최소한 한 번 사용해야 합니다. 선택적으로, `base64_data`를 여러 번 사용하여 디코딩된 Base64 데이터의 시작 부분으로 돌아갈 수 있습니다.

Base64 데이터를 검사할 때 다음에 유의하십시오.

- 빠른 패턴 매처를 사용할 수 없습니다.
- 개입하는 HTTP 콘텐츠 인수로 규칙에서 Base64 검사를 중지하는 경우, Base64 데이터의 상세 검사 전에 규칙에 다른 `base64_data` 키워드를 삽입해야 합니다.

관련 항목

개요: HTTP content 및 protected_content 키워드 인수, 25 페이지
content 키워드 빠른 패턴 매처 인수, 30 페이지