



自定义入侵规则

以下主题介绍如何使用入侵规则编辑器：

- [自定义入侵规则概述，第 1 页](#)
- [入侵规则编辑器的许可证要求，第 2 页](#)
- [入侵规则编辑器的要求和必备条件，第 2 页](#)
- [规则剖析，第 2 页](#)
- [自定义规则创建，第 14 页](#)
- [搜索规则，第 19 页](#)
- [入侵规则编辑器页面上的规则过滤，第 20 页](#)
- [入侵规则中的关键字和参数，第 24 页](#)

自定义入侵规则概述

入侵规则是系统用于检测利用网络漏洞企图的一组关键字和参数。当系统分析网络流量时，它会将数据包与每条规则中指定的条件进行比较。如果数据包数据与规则中指定的所有条件都匹配，则会触发此规则。如果规则是警报规则，将生成入侵事件。如果是通过规则，将忽略流量。对于内联部署中的丢弃规则，系统将丢弃数据包并生成事件。可以通过Cisco Secure Firewall Management Center 或网络界面查看和评估入侵事件。

系统提供两种入侵规则：共享对象规则和标准文本规则。Talos 情报小组 可以使用共享对象规则来检测传统的标准文本规则无法检测到的漏洞攻击。您无法创建共享对象规则。在自行编写入侵规则时，您可以创建标准文本规则。

您可以编写自定义标准文本规则，以调整可能出现的事件类型。请注意，虽然本文档有时讨论以检测特定漏洞为目标的规则，但最成功的规则是以检测可能试图利用已知漏洞的流量为目标，而不是以检测特定已知漏洞为目标。通过编写规则和指定规则的事件消息，可以更轻松地识别可能存在攻击和策略逃避行为的流量。

当在自定义入侵策略中启用自定义标准文本规则时，请记住，某些规则关键字和参数要求首先以特定方式对流量进行解码或预处理。本章说明在用于管理预处理的网络分析策略中必须配置的选项。请注意，如果禁用所需的预处理器，系统会自动采用其当前设置使用该预处理器，尽管该预处理器在网络分析策略网络界面中保持禁用状态。



注意 将编写的入侵规则用于生产环境之前，请务必使用受控网络环境测试这些规则。编写错误的入侵规则可能会严重影响系统性能。



注释 您可以使用 Snort 创建自定义入侵规则。但是，目前不支持对这些规则进行调整和故障排除。

入侵规则编辑器的许可证要求

威胁防御 许可证

IPS

经典许可证

保护

入侵规则编辑器的要求和必备条件

型号支持

任意。

支持的域

任意

用户角色

- 管理员
- 入侵管理员 (Intrusion Admin)

规则剖析

所有标准文本规则均包含两个逻辑部分：规则报头和规则选项。规则报头包含：

- 规则的操作或类型
- 协议
- 源 IP 地址、目标 IP 地址和子网掩码

- 方向指示符（显示从源到目标的流量流动方向）
- 源端口和目标端口

规则选项部分包含：

- 事件消息
- 关键字及其参数
- 模式（数据包负载必须与之匹配才能触发规则）
- 规范（规定规则引擎应检查数据包的哪些部分）

下图说明规则的组成部分：

Rule Header

```
alert tcp $EXTERNAL_NET any -> $HTTP_SERVERS $HTTP_PORTS
```

Rule Keywords and Arguments

```
(msg:"WEB-IIS newdsn.exe access";
flow:to_server,established; uricontent:"/scripts/
tools/newdsn.exe"; nocase; metadata:service http;
reference:bugtraq,1818; reference:cve,1999-0191;
reference:nessus,10360; classtype:web-application-
activity; sid:1024; rev:10; )
```

372214

请注意，括号里的是规则选项部分。入侵规则编辑器提供了一个易于使用的界面来帮助您构建标准文本规则。

入侵规则报头

每个标准文本规则和共享对象规则都有一个包含参数的规则报头。下面说明规则报头的组成部分：



372215

下表介绍了规则报头的上述各个部分。

表 1: 规则报头值

规则报头组成部分	示例值	示例值的作用...
操作	alert	如果触发，将会生成事件。
协议	tcp	仅测试 TCP 流量。
源 IP 地址	\$EXTERNAL_NET	测试来自不在内部网络上的任何主机的流量。

规则报头组成部分	示例值	示例值的作用...
源端口	any	测试来自发起主机上任何端口的流量。
Operator	->	测试外部流量（流向网络上的 Web 服务器）。
目标 IP 地址:	\$HTTP_SERVERS	测试将要传送到内部网络上被指定为 Web 服务器的任何主机的流量。
目标端口	\$HTTP_PORTS	测试传送到内部网络上 HTTP 端口的流量。



注释 与大多数入侵规则一样，以上示例使用默认变量。

相关主题

[变量集](#)

入侵规则报头操作

每个规则报头都包含一个用于指定数据包触发规则时系统应采取的操作的参数。操作设置为 *alert* 的规则将会针对触发规则的数据包生成入侵事件并记录该数据包的详细信息。操作设置为 *pass* 的规则不会针对触发规则的数据包生成入侵事件，也不会记录该数据包的详细信息。



注释 在内联部署中，规则状态设置为“丢弃并生成事件”的规则会针对触发规则的数据包生成入侵事件。此外，如果在被动部署中应用丢弃规则，该规则将会充当警报规则。

默认情况下，通过规则会覆盖警报规则。可以创建通过规则来防止符合通过规则中定义的条件数据包在特定情况下触发警报规则，而无需禁用预警规则。例如，您可能希望使检测尝试作为“匿名”用户登录 FTP 服务器这种情况的规则保持活动状态。但是，如果网络有一个或多个合法的匿名 FTP 服务器，您可以编写并激活一个通过规则，在其中指明匿名用户不会对那些特定服务器触发原始规则。

在入侵规则编辑器中，可以从**操作 (Action)** 列表中选择规则类型。

入侵规则报头协议

在每个规则报头中，必须指定规则检查的流量的协议。可以指定以下网络协议用于分析：

- ICMP（互联网控制消息协议）
- IP（互联网协议）



注释 如果协议设置为 *ip*，系统将忽略入侵规则报头中的端口定义。

- TCP（传输控制协议）
- UDP（用户数据报协议）

请使用 **IP** 作为协议类型，以检查 IANA 分配的所有协议（包括 TCP、UDP、ICMP、IGMP 等等）。



注释 目前不能编写与 IP 负载中下一个报头（例如 TCP 报头）模式匹配的规则。相反，内容匹配从上一个解码的协议开始。要解决这个问题，可以使用规则选项来匹配 TCP 报头中的模式。

在“入侵规则” (Intrusion Rules) 编辑器中，可以从**协议 (Protocol)** 列表中选择协议类型。

相关主题

[入侵规则报头协议](#)，第 4 页

入侵规则报头方向

在规则报头中，可以指定数据包为使规则对其进行检查而必须传播的方向。下表介绍了这些选项。

表 2: 规则报头中的方向选项

使用.....	以测试...
定向天线	仅测试从指定源 IP 地址流向指定目标 IP 地址的流量
双向	测试指定的源 IP 地址和目标 IP 地址之间的所有流量

入侵规则报头源和目标 IP 地址

通过将数据包检查限制为仅对来自或发往特定 IP 地址的数据包，可以减少系统必须执行的数据包检查工作。这样做还可以令规则更加具体，并消除规则针对源和目标 IP 地址未指示可疑行为的数据包进行触发的可能性，从而减少误报。



提示 系统只能识别 IP 地址，不接受源或目标 IP 地址的主机名。

在入侵规则编辑器中，可以在**源 IP (Source IPs)** 和**目标 IP (Destination IPs)** 字段中指定源 IP 地址和目标 IP 地址。

编写标准文本规则时，可以根据自身需求以多种方法指定 IPv4 和 IPv6 地址。可以指定单个 IP 地址、any、IP 地址列表、CIDR 表示法、前缀长度或网络变量。此外，还可以指明要排除的特定 IP 地址或 IP 地址集。指定 IPv6 地址时，可使用 RFC 4291 中定义的任何寻址约定。

入侵规则中的 IP 地址语法

下表总结了可用于指定源 IP 地址和目标 IP 地址的各种方法。

表 3: 源/目标 IP 地址语法

指定的对象	使用.....	示例
任意 IP 地址	any	any
特定 IP 地址	IP 地址 请注意，不能在同一规则中混合使用 IPv4 和 IPv6 源地址和目标地址。	192.168.1.1 2001:db8::abcd
IP 地址列表	使用方括号 ([]) 将地址括起来，并使用逗号分隔各个 IP 地址	[192.168.1.1,192.168.1.15] [2001:db8::b3ff, 2001:db8::0202]
IP 地址块	IPv4 CIDR 块或 IPv6 地址前缀表示法	192.168.1.0/24 2001:db8::/32
除特定 IP 地址或地址集以外的任何项	! 字符，后接要否定的 IP 地址	!192.168.1.15 !2001:db8::0202:b3ff:fe1e
IP 地址块中除一个或多个特定 IP 地址以外的任何 IP 地址	在地址块后加上被否定地址或地址块的列表	[10.0.0/8, !10.2.3.4, !10.1.0.0/16] [2001:db8::/32, !2001:db8::8329, !2001:db8::0202]
网络变量定义的 IP 地址	前面带有 \$ 的大写字母形式的变量名称 请注意，无论入侵规则中使用的网络变量定义的主机如何，预处理器规则都可以触发事件。	\$HOME_NET
除 IP 地址变量定义的所有 IP 地址以外的所有 IP 地址	前面带有 !\$ 的大写字母形式的变量名称	!\$HOME_NET

以下描述提供了有关某些 IP 地址输入方法的其他信息。

任意 IP 地址

可以指定 `any` 这个词作为规则的源或目标 IP 地址，以指示 IPv4 或 IPv6 地址。

例如，以下规则在源 IP 和目标 IP 字段中使用参数 `any` 来评估具有 IPv4 或 IPv6 源地址或目标地址的数据包：

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

还可以指定 `::` 以指示 IPv6 地址。

多个 IP 地址

可以列出多个 IP 地址，地址之间用逗号分隔，如有需要，还可以用方括号将非否定地址列表括起来，如以下示例所示：

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

可以单独或以任意组合列出 IPv4 和 IPv6 地址，如以下示例所示：

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

请注意，现在不再要求用方括号将 IP 地址列表括起来（旧版软件要求这样做）。另请注意，输入列表时，可以在每个逗号前后添加一个空格。



注释 必须用方括号将否定列表括起来。

也可以使用 IPv4 无类域间路由选择 (CIDR) 表示法或 IPv6 前缀长度来指定地址块。例如：

- 192.168.1.0/24 指定子网掩码为 255.255.255.0 的 192.168.1.0 网络中的 IPv4 地址，即，192.168.1.0 至 192.168.1.255。
- 2001:db8::/32 指定前缀长度为 32 位的 2001:db8:: 网络中的 IPv6 地址，即，2001:db8:: 至 2001:db8:ffff:ffff:ffff:ffff:ffff:ffff。



提示 如果需要指定 IP 地址块，但仅以 CIDR 或前缀长度表示法无法表示出该地址块，可以在 IP 地址列表中使用 CIDR 块和前缀长度。

IP 地址否定

可以使用感叹号 (!) 否定指定 IP 地址。也就是说，可以匹配除指定 IP 地址以外的所有 IP 地址。例如，!192.168.1.1 指定除 192.168.1.1 以外的任何 IP 地址，!2001:db8:ca2e::fa4c 指定除 2001:db8:ca2e::fa4c 以外的任何 IP 地址。

要否定某个 IP 地址列表，请用方括号将该 IP 地址列表括起来，并在其前面加上 !。例如，![192.168.1.1,192.168.1.5] 将定义除 192.168.1.1 和 192.168.1.5 以外的任何 IP 地址。



注释 要否定 IP 地址列表，必须使用方括号。

对 IP 地址列表使用否定字符时务必要小心。例如，如果使用 ![192.168.1.1,!192.168.1.5] 匹配不是 192.168.1.1 和 192.168.1.5 的任何地址，系统会将此语法解释为“非 192.168.1.1 的任何地址，或非 192.168.1.5 的任何地址”。

由于 192.168.1.5 不是 192.168.1.1，且 192.168.1.1 不是 192.168.1.5，因此，这两个 IP 地址都与 ![192.168.1.1,!192.168.1.5] 的 IP 地址值匹配；此语法实质上与使用“any”相同。

应该使用 `![192.168.1.1,192.168.1.5]`。系统会将此语法为“非 192.168.1.1 且非 192.168.1.5”，这意味着，与方括号中所列地址以外的任何 IP 地址匹配。

请注意，从逻辑上讲，不能对 `any` 进行否定（如果它被否定，将表示无地址）。

相关主题

[变量集](#)

入侵规则报头源和目标端口

在入侵规则编辑器中，可以在源端口 (Source Port) 和目标端口 (Destination Port) 字段中指定源端口和目标端口。

入侵规则中的端口语法

Firepower 系统使用特定类型的语法来定义规则报头中使用的端口号。



注释 如果协议设置为 `ip`，系统将忽略入侵规则报头中的端口定义。

可以列出多个端口，端口之间用逗号分隔，如下列示例所示：

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

或者，以下示例显示如何用方括号将端口列表括起来（先前软件版本中要求如此，但现在不再有此要求）：

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

请注意，**必须**用方括号将求反端口列表括起来，如下列示例所示：

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

下表总结了可使用的语法：

表 4: 源/目标端口语法

指定的对象	使用	示例
任意端口	<code>any</code>	<code>any</code>
特定端口	端口号	<code>80</code>
端口范围	范围内第一个和最后一个端口号之间使用破折号	<code>80-443</code>
所有小于或等于指定端口号的端口	在端口号前面加上破折号	<code>-21</code>
所有大于或等于指定端口号的端口	在端口号后面加上破折号	<code>80-</code>

指定的对象	使用	示例
除特定端口或端口范围以外的所有端口	要求反的端口、端口列表或端口范围前面的 ! 字符 请注意，在逻辑上可以对除 any（如果求反，将指示无端口）以外的所有端口指定使用求反。	!20
端口变量定义的所有端口	前面带有 \$ 的大写字母形式的变量名称	\$HTTP_PORTS
除端口变量定义的端口以外的所有端口	前面带有 !\$ 的大写字母形式的变量名称	!\$HTTP_PORTS

入侵事件详细信息

构建标准文本规则时，可以包含说明规则在攻击尝试中检测到的漏洞的情景信息。也可以在其中纳入对漏洞数据库的外部参考，以及定义入侵事件在贵公司中具有的最高优先级。这样，如果分析师发现入侵事件，他们可随时获取有关优先级、漏洞和已知缓解措施的信息。

消息

可以指定规则触发时以消息形式显示的有意义的文本。这类消息使您可以及时了解规则检测的漏洞的性质。可以使用除花括号 ({}) 以外的所有可打印标准 ASCII 字符。系统将移除将消息完全引起来的引号。



提示 必须指定规则消息。此外，消息不能只包含空白字符、一个或多个引号、一个或多个撇号或者仅由空白字符、引号或撇号组成的任意组合。

要在入侵规则编辑器中定义事件消息，请在**消息 (Message)** 字段中输入事件消息。

分类

对于每个规则，可以指定事件数据包显示中出现的攻击分类。下表列出了每种分类的名称和编号。

表 5: 规则分类

编号	分类名称	说明
1	not-suspicious	非可疑流量
2	unknown	未知流量
3	bad-unknown	潜在不良流量
4	attempted-recon	尝试信息泄露
5	successful-recon-limited	信息泄露

编号	分类名称	说明
6	successful-recon-largescale	大规模信息泄露
7	attempted-dos	尝试拒绝服务
8	successful-dos	拒绝服务
9	attempted-user	尝试获取用户权限
10	unsuccessful-user	未成功获取用户权限
11	successful-user	未成功获取用户权限
12	attempted-admin	尝试获取管理员权限
13	successful-admin	成功获取管理员权限
14	rpc-portmap-decode	解码 RPC 查询
15	shellcode-detect	检测到可执行代码
16	string-detect	检测到可疑字符串
17	suspicious-filename-detect	检测到可疑文件名
18	suspicious-login	检测到尝试使用可疑用户名的登录
19	system-call-detect	检测到系统调用
20	tcp-connection	检测到 TCP 连接
21	trojan-activity	检测到网络木马
22	unusual-client-port-connection	客户端使用异常端口
23	network-scan	检测网络扫描
24	denial-of-service	检测拒绝服务攻击
25	non-standard-protocol	检测非标准协议或事件
26	protocol-command-decode	通用协议命令解码
27	web-application-activity	访问可能易受攻击的 Web 应用
28	web-application-attack	Web 应用攻击
29	misc-activity	其他活动
30	misc-attack	其他攻击
31	icmp-event	般 ICMP 事件

编号	分类名称	说明
32	inappropriate-content	检测到不当内容
33	policy-violation	可能违反公司隐私策略
34	default-login-attempt	尝试使用默认用户名和密码登录
35	sdf	敏感数据
36	malware-cnc	已知恶意软件命令和控制流量
37	client-side-exploit	已知客户端攻击尝试
38	file-format	已知的恶意文件或基于文件的攻击

自定义分类

如果您希望事件的数据包显示说明的更多自定义内容由您定义的规则生成，则可以创建自定义分类。

参数	说明
分类名称	分类的名称。如果使用超过 40 个字符，则页面难以读取。不支持以下字符：<>()\'\"&\$; 以及空格字符。
分类说明 (Classification Description)	分类的说明。可使用字母数字字符和空格。不支持以下字符：<>()\'\"&\$;
优先级	高、中或低。

自定义优先级

默认情况下，规则的优先级来源于其事件分类。但是，可以通过向规则中添加 `priority` 关键字并选择高、中或低优先级来覆盖规则的分类优先级。例如，要为检测 Web 应用攻击的规则分配高优先级，请向该规则中添加 `priority` 关键字，并选择高 (**high**) 作为优先级。

自定义参考

可以使用 `reference` 关键字添加对外部网站以及对关于事件的其他信息的参考。添加参考使分析师可以随时获得所需的资源，从而帮助他们确定数据包触发规则的原因。下表列出了一些可提供关于已知漏洞和攻击的数据的外部系统。

表 6: 外部攻击识别系统

系统 ID	说明	示例 ID
bugtraq	Bugtraq 页面	8550
cve	常见漏洞和风险 ID	2020-9607

系统 ID	说明	示例 ID
mcafee	McAfee 页面	98574
url	网站参考	www.example.com?exploit=14
msb	Microsoft 安全公告	MS11-082
nessus	Nessus 页面	10039
secure-url	安全网站参考 (https://...)	intranet/exploits/exploit=14 请注意，可以对任何安全网站使用 secure-url。

通过输入参考值指定参考，如下所示：

```
id_system,id
```

其中，`id_system` 是用作前缀的系统，`id` 是 CVE ID 编号、Arachnids ID 或 URL（不包含 `http://`）。

例如，要指定 CVE-2020-9607 中记录的 Adobe Acrobat 和 Reader 问题，请输入值：

```
cve,2020-9607
```

向规则添加参考时应注意以下几点：

- 逗号后不能有空格。
- 系统 ID 不能是大写字母。

相关主题

[添加自定义分类](#)，第 12 页

[定义事件优先级](#)，第 13 页

[定义事件引用](#)，第 13 页

添加自定义分类

过程

步骤 1 当创建或编辑规则时，请从 **分类** 下拉列表 (**对象 > 入侵规则 > 创建规则 > 编辑分类**) 中选择 **编辑分类**。

如果系统显示**查看分类 (View Classifications)**，则配置属于祖先域，或者您没有修改配置的权限。

步骤 2 输入分类名称 (**Classification Name**) 和分类说明 (**Classification Description**)，如[入侵事件详细信息](#)，第 9 页中所述。

步骤 3 从**优先级 (Priority)** 下拉列表中为分类选择优先级。

步骤 4 点击 **Add**。

步骤 5 点击 **Done**。

下一步做什么

- 继续创建或编辑规则。有关详细信息，请参阅[编写新规则](#)，第 15 页或[修改现有规则](#)，第 16 页。

相关主题

[自定义规则创建](#)，第 14 页

定义事件优先级

过程

步骤 1 当创建或编辑规则时，从**检测选项 (Detection Options)** 下拉列表中选择优先级 (priority)。

步骤 2 点击添加选项 (**Add Option**)。

步骤 3 从**优先级 (priority)** 下拉列表中选择值。

步骤 4 单击保存。

下一步做什么

- 继续创建或编辑规则。有关详细信息，请参阅[编写新规则](#)，第 15 页或[修改现有规则](#)，第 16 页。

相关主题

[自定义规则创建](#)，第 14 页

定义事件引用

过程

步骤 1 在创建或编辑规则时，从**检测选项 (Detection Options)** 下拉列表中选择 `reference`。

步骤 2 点击添加选项 (**Add Option**)。

步骤 3 在 **reference** 字段中输入值，如[入侵事件详细信息](#)，第 9 页中所述。

步骤 4 单击保存。

下一步做什么

- 继续创建或编辑规则。有关详细信息，请参阅[编写新规则](#)，第 15 页或[修改现有规则](#)，第 16 页。

相关主题

[自定义规则创建](#)，第 14 页

自定义规则创建

您可以通过以下方式创建自定义入侵规则：

- 创建您自己的标准文本规则
- 将现有标准文本规则保存为新规则
- 将系统提供的共享对象规则保存为新规则
- 导入本地规则文件

无论您使用哪种创建方法，系统都会将自定义规则保存在本地规则类别中。

当您创建自定义入侵规则时，系统会为它分配唯一的规则编号（其格式为 GID:SID:Rev）。此编号的元素如下：

GID

生成器 ID。对于所有标准文本规则，此值为 1（全局域或旧式 GID）或 1000-2000（子代域）。对于您保存为新规则的所有共享对象规则，此值为 1。

SID

Snort ID。指示规则是否为系统规则的本地规则。当您创建新规则时，系统会为本地规则分配下一个可用 SID。

本地规则的 Snort ID 号从 1000000 开始，且每个本地新规则的 SID 号以 1 递增。

Rev

修订号。对于新规则，修订号为 1。每修改一次自定义规则，修订号就增加 1。

在自定义标准文本规则中，可以设置规则报头设置、规则关键字和规则参数。您可以通过规则报头设置将规则设置为仅匹配使用特定协议以及发往或来自特定 IP 地址或端口的流量。

在系统提供的自定义标准文本规则或共享对象规则中，您只能修改规则报头信息，例如，源端口、目标端口、源 IP 地址和目标 IP 地址。您无法修改规则关键字或规则参数。

为共享对象规则修改报头信息并保存更改将会为该规则创建生成器 ID (GID) 为 1（全局域）或 1000-2000（子代域）的新实例，并会为自定义规则创建下一个可用 SID。系统将共享对象规则的新实例链接到保留的 `soid` 关键字，该关键字会将创建的规则映射到 Talos 情报小组 所创建的规则。您可以删除您创建的共享对象规则的实例，但无法删除 Talos 创建的共享对象规则。

编写新规则

过程

步骤 1 选择对象 > 入侵规则。

步骤 2 点击 **Create Rule**。

步骤 3 在消息 (**Message**) 字段中输入值。

步骤 4 从以下每个下拉列表中选择值：

- 分类
- 操作
- 协议
- **Direction**

步骤 5 在以下字段中输入值：

- 源 IP (**Source IPs**)
- 目标 IP
- 源端口
- 目的端口

如果没有为这些字段指定值，请系统会使用值 `any`。

步骤 6 从检测选项 (**Detection Options**) 下拉列表中选择值。

步骤 7 点击添加选项 (**Add Option**)。

步骤 8 输入所添加的关键字的任何参数。

步骤 9 或者，重复步骤 6 至步骤 8。

步骤 10 如果添加了多个关键字，则可以执行以下操作：

- 对关键字重新排序 - 点击要移动的关键字旁边的向上或向下箭头。
- 删除关键字 - 点击该关键字旁边的 **X**。

步骤 11 点击另存为新。

下一步做什么

- 在相应的入侵策略中启用新的或已更改的规则；请参阅[查看入侵策略中的入侵规则](#)。
- 部署配置更改；请参阅 [部署配置更改](#)。

修改现有规则

可以将系统提供的规则以及属于祖先域的规则另存为本地规则类别中的新自定义规则，随后可对这些新规则进行修改。

过程

步骤 1 使用以下任一方法访问入侵规则：


- 选择策略 > 访问控制 > 入侵。


点击要编辑的策略旁边的 **Snort 2 版本 (Snort 2 Version)**，然后点击规则 (**Rules**)。

- 选择对象 > 入侵规则。

步骤 2 找到要修改的规则。有以下选项可供选择：

- 浏览文件夹以查找规则。
- 搜索规则；请参阅[搜索规则](#)，第 19 页。
- 过滤规则所属的组；请参阅[过滤规则](#)，第 23 页。

步骤 3 点击规则旁边的 **编辑** ()，如果是搜索结果，则点击规则消息。

如果显示视图 ()，则表明配置属于祖先域，或者您没有修改配置的权限。

步骤 4 针对规则类型相应地修改规则。

注释 请勿修改共享对象规则的协议；此修改将会致使规则无效。

步骤 5 有以下选项可供选择：

- 如果编辑的是自定义规则并要覆盖该规则的当前版本，请点击**保存 (Save)**。
- 如果编辑的是系统提供的规则或属于祖先域的任何规则，或者如果编辑的是自定义规则并要将更改另存为新规则，请点击**另存为新项目 (Save As New)**。

下一步做什么

- 如果要使用规则的本地修改而不是系统提供的规则，请通过使用[入侵规则状态](#)中的程序停用系统提供的规则并激活本地规则。
- 部署配置更改；请参阅[部署配置更改](#)。

相关主题

[搜索规则](#)，第 19 页

[入侵规则编辑器页面上的规则过滤](#)，第 20 页

查看规则文档

在“规则编辑”页面中，您可以查看Talos 情报小组 提供的规则文档。查看时，您可以点击 **规则文档** 和其他 外部参考来查看 Talos 提供的附加信息。也可以点击**情景管理器**来查看由该规则生成的事件的情景信息。

过程

步骤 1 使用以下任一方法访问入侵规则：


- 选择策略 > 访问控制 > 入侵。


点击要编辑的策略旁边的 **Snort 2 版本 (Snort 2 Version)**，然后点击规则 (**Rules**)。

- 选择对象 > 入侵规则。

步骤 2 找到要查看的规则。有以下选项可供选择：

- 浏览文件夹以查找规则。
- 搜索规则；请参阅[搜索规则](#)，第 19 页。
- 过滤规则所属的组；请参阅[过滤规则](#)，第 23 页。

步骤 3 点击规则旁边的 **编辑** ()，如果是搜索结果，则点击规则消息。

如果显示视图 ()，则表明配置属于祖先域，或者您没有修改配置的权限。

步骤 4 点击**查看文档**。

步骤 5 或者，点击以下任一链接：

- **规则文档** - 查看详细的规则具体信息。
- 其他外部参考 - 参阅[关键字过滤](#)，第 21 页和[入侵事件详细信息](#)，第 9 页中的自定义参考获取有关可用外部参考的信息。
- **情景管理器 (Context Explorer)** - 请参阅《[Cisco Secure Firewall Management Center 管理指南](#)》中的入侵信息部分，获取有关在情景管理器中查看规则的情景数据的信息。

提示 选择外部链接会关闭文档弹出窗口；要退出规则编辑页面而不修改规则，请选择任意菜单路径。

将注释添加到入侵规则

可以向任何入侵规则添加注释。这种注释可有助于提供有关规则及其识别出的漏洞或策略违规的额外情景和信息。

过程

步骤 1 使用以下任一方法访问入侵规则：


- 选择策略 > 访问控制 > 入侵。


点击要编辑的策略旁边的 **Snort 2 版本 (Snort 2 Version)**，然后点击规则 (**Rules**)。

- 选择对象 > 入侵规则。

步骤 2 找到要添加注释的规则。有以下选项可供选择：

- 浏览文件夹以查找规则。
- 搜索规则；请参阅[搜索规则](#)，第 19 页。
- 过滤规则所属的组；请参阅[过滤规则](#)，第 23 页。

步骤 3 点击规则旁边的 **编辑** ()，如果是搜索结果，则点击规则消息。

如果规则旁显示**视图** ()，则表明规则属于祖先策略，或者您没有修改规则的权限。

步骤 4 点击 **Rule Comment**。

步骤 5 在文本框中输入注释。

步骤 6 点击添加注释 (**Add Comment**)。

提示 也可以在入侵事件的数据包视图中添加和查看规则注释。

下一步做什么

- 继续创建或编辑规则。有关详细信息，请参阅[编写新规则](#)，第 15 页或[修改现有规则](#)，第 16 页。

相关主题

[搜索规则](#)，第 19 页

删除自定义规则

可以删除当前未在入侵策略中启用的自定义规则。无法删除系统提供的标准文本规则或共享对象规则。

系统将删除的规则存储在删除的类别中，您可以使用删除的规则作为新规则的依据。入侵策略中的 Rules 页面不显示删除的类别，因此您不能启用删除的自定义规则。



提示 自定义规则包括使用修改后的报头信息保存的共享对象规则。系统还会将这些保存在本地规则类别中，并将其与 GID 为 1（全局域或旧式 GID）或 1000 - 2000（子代域）一起列出。您可以删除修改后的共享对象规则版本，但无法删除原始共享对象规则。

过程

步骤 1 使用以下任一方法访问入侵规则：

- 选择策略 > 访问控制 > 入侵。

点击要编辑的策略旁边的 **Snort 2 版本 (Snort 2 Version)**，然后点击规则 (**Rules**)。

- 选择对象 > 入侵规则。

步骤 2 您有两种选择：

- 删除所有本地规则 - 点击删除本地规则 (**Delete Local Rules**)，然后点击确定 (**OK**)。
- 删除单条规则 - 从规则分组方法 (**Group Rules By**) 下拉列表中选择 **Local Rules**，点击要删除的规则旁边的 删除 ()，然后点击确定 (**OK**) 以确认删除。

相关主题

[入侵规则状态](#)

搜索规则

系统提供数千个标准文本规则，并且Talos 情报小组 会在发现新漏洞和攻击时继续添加规则。您可以轻松搜索您想要激活、禁用或编辑的特定规则。

过程

步骤 1 使用以下任一方法访问入侵规则：

- 选择策略 > 访问控制 > 入侵。

点击要编辑的策略旁边的 **Snort 2 版本 (Snort 2 Version)**，然后点击规则 (**Rules**)。

- 选择对象 > 入侵规则。

步骤 2 点击工具栏上的 **Search**。

步骤 3 添加搜索条件。

步骤 4 点击 **Search**。

下一步做什么

- 如果要查看或编辑找到的规则（或系统规则的副本），请点击超链接规则消息。有关详细信息，请参阅[编写新规则](#)，第 15 页或[修改现有规则](#)，第 16 页。

入侵规则的搜索条件

下表介绍了可用的搜索选项：

表 7: 规则搜索条件

选项	说明
签名 ID (Signature ID)	要根据 Snort ID (SID) 搜索单个规则，请输入一个 SID 号。要搜索多个规则，请输入以逗号分隔的 SID 号列表。此字段中最多可输入 80 个字符。
生成器 ID	要搜索标准文本规则，请选择 1 。要搜索共享对象规则，请选择 3 。
消息	要搜索带有特殊消息的规则，请在消息 (Message) 字段中输入规则消息中的一个字。例如，要搜索 DNS 攻击，可输入 DNS；要搜索缓冲区溢出攻击，可输入溢出 (overflow)。
协议	要搜索评估特定协议的流量的规则，请选择该协议。如果不选择协议，搜索结果将包含适用于所有协议的规则。
源端口	要搜索检查来自指定端口的数据包规则，请输入源端口号或端口相关变量。
目标端口	要搜索检查发往特定端口的数据包规则，请输入目标端口号或端口相关变量。
源 IP	要搜索检查来自指定 IP 地址的数据包规则，请输入源 IP 地址或 IP 地址相关变量。
目标 IP	要搜索检查发往指定 IP 地址的数据包规则，请输入目标 IP 地址或 IP 地址相关变量。
关键字	要搜索特定关键字，可以使用关键字搜索选项。可以选择要搜索的关键字并输入关键字值。也可以在关键字值前面加上感叹号 (!) 以匹配任何未指定的值。
类别 (Category)	要搜索特定类别中的规则，请从类别 (Category) 列表中选择该类别。
分类 (Classification)	要搜索具有特定分类的规则，请从分类 (Classification) 列表中选择该分类名称。
规则状态 (Rule State)	要在特定策略和特定规则状态中搜索规则，请从第一个规则状态 (Rule State) 列表中选择策略，并从第二个列表中选择状态，以搜索状态设置为生成事件 (Generate Events)、丢弃并生成事件 (Drop and Generate Events) 或已禁用 (Disabled) 的规则。

入侵规则编辑器页面上的规则过滤

您可以对入侵规则编辑器页面上的规则进行过滤以显示其中一组规则。例如，如果想要修改某个规则或更改其状态，但是难以在成千上万个可用规则中找到该规则，这个过滤功能可能很有用。

当您输入过滤器时，页面将显示至少包含一条匹配规则或消息（如果没有匹配规则）的文件夹。

过滤准则

过滤器可以包含特殊关键字及其参数、字符串和用引号引起来的文字字符串，多个过滤器条件之间用空格隔开。过滤器不能包含正则表达式、通配符或任何特殊运算符，例如取反字符(!)、大于号(>)和小于号(<)等。

所有关键字、关键字参数和字符串都不区分大小写。除关键字 `gid` 和 `sid` 之外，所有参数和字符串都被视为部分字符串。`gid` 和 `sid` 的参数只会返回完全匹配项。

您可以在未过滤的原始页面上展开某个文件夹，如果后续过滤器返回该文件夹中的匹配项，该文件夹将会保持展开。这对于在包含大量规则的文件夹中搜索规则可能有用。

不能使用后续过滤器限制任何过滤器。输入的任何过滤器都会搜索整个规则数据库并返回所有匹配的规则。当您在页面仍显示上一过滤器的结果时输入过滤器，页面将清空，转而返回新过滤器的结果。

您可以对已过滤或未过滤列表中的规则使用相同的功能。例如，可以编辑入侵规则编辑器页面上经过过滤或未经过滤的列表中的规则。您也可以使用该页面上上下文菜单中的任何选项。



提示 如果所有子组中的总规则数量很大，过滤所需的时间可能大大增加，因为规则显示在多个类别中，即使唯一规则的总数少很多也是如此。

关键字过滤

每个规则过滤器都可以包含一个或多个关键字，其格式如下：

```
keyword:argument
```

其中，关键字是下表中的其中一个关键字，参数是要在与该关键字相关的一个或多个指定字段中搜索的一个字母数字字符串，不区分大小写。

除 `gid` 和 `sid` 之外，所有关键字的参数都会被视为部分字符串。例如，参数 `123` 将返回 `"12345"`、`"41235"`、`"45123"` 等。`gid` 和 `sid` 的参数只会返回完全匹配项；例如，`sid:3080` 只会返回结果 `SID 3080`。



提示 使用一个或多个字符串来进行过滤可以搜索部分 `SID`。

下表介绍了可以用于过滤规则的特定过滤关键字和参数。

表 8: 规则过滤器关键字

关键字	Description	示例
<code>arachnids</code>	根据规则引用中的完整或部分 <code>Arachnids ID</code> 返回一个或多个规则。	<code>arachnids:181</code>

关键字	Description	示例
bugtraq	根据规则引用中的完整或部分 Bugtraq ID 返回一个或多个规则。	bugtraq:2120
cve	根据规则引用中的完整或部分 CVE 编号返回一个或多个规则。	cve:2003-0109
gid	参数 1 将返回标准文本规则。参数 3 将返回共享对象规则。	gid:3
mcafee	根据规则引用中的完整或部分 McAfee ID 返回一个或多个规则。	mcafee:10566
msg	根据规则的完整或部分 Message 字段（又称为事件消息）返回一个或多个规则。	msg:chat
nessus	根据规则引用中的完整或部分 Nessus ID 返回一个或多个规则。	nessus:10737
ref	根据规则引用或规则 Message 字段中一个完整的字母数字字符串或其一部分返回一个或多个规则。	ref:MS03-039
sid	返回带有完全匹配的 Snort ID 的规则。	sid:235
url	根据规则引用中的完整或部分 URL 返回一个或多个规则。	url:faqs.org

相关主题

[定义事件引用](#)，第 13 页

[入侵事件详细信息](#)，第 9 页

字符串过滤

每个规则过滤器可以包含一个或多个字母数字字符串。字符串将搜索规则的消息字段、Snort ID (SID) 和生成器 ID (GID)。例如，字符串 123 会返回规则消息中的 "Lotus123"、"123mania" 等字符串，也会返回 SID 6123、SID 12375 等。

所有字符串都不区分大小写并被视为部分字符串。例如，字符串 ADMIN、admin 或 Admin 中的任意一个都会返回 "admin"、"CFADMIN"、"Administrator" 等等。

用引号将字符串引起来可以返回完全匹配项。例如，用引号引起来的原义字符串 "overflow attempt" 只会返回完全匹配的该字符串，而由 overflow 和 attempt 这两个字符串组成的未加引号的过滤器则会返回 "overflow attempt"、"overflow multipacket attempt"、"overflow with evasion attempt" 等结果。

相关主题

[入侵事件详细信息](#)，第 9 页

组合关键字和字符串过滤

输入关键字、字符串或这二者的任意组合并以空格分隔可以缩小过滤结果的范围。结果包括符合所有过滤条件的任意规则。

可以按照任意顺序输入多个过滤条件。例如，以下每个过滤器返回的规则相同：

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

过滤规则

在“入侵规则” (Intrusion Rules) 页面上，可以将规则过滤为子集，以便可以更轻松地查找特定规则。然后，可以使用任何页面功能，包括选择情景菜单中可用的任何功能。

规则过滤尤其适用于查找要编辑的特定规则。

过程

步骤 1 使用以下任一方法访问入侵规则：

- 选择策略 > 访问控制 > 入侵。

点击要编辑的策略旁边的 **Snort 2 版本 (Snort 2 Version)**，然后点击规则 (**Rules**)。

- 选择对象 > 入侵规则。

步骤 2 在过滤之前，您有以下选择：

- 展开要展开的任何规则组。某些规则组还具有可展开的子组。

如果您预计规则可能在某个组中，则在未经过滤的原始页面上展开该组可能有用。如果后续过滤器返回该文件夹中的匹配项，当您点击过滤器 **清除** (✕) 返回到未经过滤的原始页面时，该组将会保持展开。

- 从规则分组方法 (**Group Rules By**) 下拉列表中选择其他分组方法。

步骤 3 在规则分组方法 (**Group Rules By**) 列表下的 **过滤器** (🔍) 旁边的文本框中输入过滤器限制。

步骤 4 按下 Enter 键。

注释 通过点击过滤器 **清除** (✕) 清除当前已过滤的列表。

入侵规则中的关键字和参数

借助规则语言，可以通过组合关键字来指定规则行为。关键字及其相关值（称为参数）规定系统如何评估规则引擎测试的数据包和数据包相关值。Firepower 系统当前支持允许您执行检查功能（例如内容匹配、协议特定模式匹配和状态特定匹配）的关键字。在每个关键字中最多可以定义 100 个参数，还可以组合任意数量的兼容关键字来创建非常具体的规则。这有助于降低出现误报和漏报的可能性，使您可以重点关注接收到的入侵信息。

请注意，您也可以被动部署中使用自适应配置文件，以根据规则元数据和主机信息动态调整对特定数据包的主动规则处理。

本节所述的关键字列于规则编辑器中的“检测选项” (Detection Options) 下。

相关主题

[关于自适应配置文件](#)

content 和 protected_content 关键字

使用 `content` 关键字或 `protected_content` 关键字可以指定要在数据包中检测的内容。

大多数情况下，应始终在 `content` 或 `protected_content` 关键字后面加上修饰符，指示对内容进行搜索的位置、搜索是否区分大小写及其他选项。

请注意，要使规则触发事件，所有内容匹配必须为真，也就是说，每项内容匹配与其他匹配之间都存在 AND 关系。

另请注意，在内联部署中，可以将规则设置为匹配恶意内容并将其更换为您自定义的等长文本字符串。

content

当使用 `content` 关键字时，规则引擎在数据包负载或数据流中搜索该字符串。例如，如果输入 `/bin/sh` 作为其中一个 `content` 关键字的值，则规则引擎会在数据包负载中搜索字符串 `/bin/sh`。

可以使用 ASCII 字符串、十六进制内容（二进制字节代码）或这两者的组合来匹配内容。可以在关键字值中将十六进制内容放在两条竖线 (|) 之间。例如，可以混合使用十六进制内容和 ASCII 内容，例如，`|90C8 C0FF FFFF|/bin/sh`。

可以在一个规则中指定多项内容匹配。要这样做，请使用 `content` 关键字的其他实例。对于各项内容匹配，可以指明必须在数据包负载或数据流中发现内容匹配才可触发规则。



注意 如果创建的规则只包含一个 `content` 关键字，但没有为该关键字选择 **Not** 选项，可能会使入侵策略无效。

protected_content

protected_content 关键字使您可以在配置规则参数前对搜索内容字符串进行编码。原始规则作者在配置关键字前使用哈希函数（SHA-512、SHA-256 或 MD5）对字符串进行编码。

如果使用 protected_content 关键字而不使用 content 关键字，规则引擎在数据包负载或数据流中搜索字符串的方式并不会改变，且大多数关键字选项将起到预期作用。下表总结了 protected_content 关键字选项与 content 关键字选项存在差异的例外情况。

表 9: protected_content 选项例外

选项	说明
散列类型 (Hash Type)	protected_content 规则关键字的新增选项。
区分大小写	不支持
Within	不支持
Depth	不支持
Length	protected_content 规则关键字的新增选项。
使用快速模式匹配程序 (Use Fast Pattern Matcher)	不支持
仅快速模式匹配程序 (Fast Pattern Matcher Only)	不支持
快速模式匹配程序偏移和长度 (Fast Pattern Matcher Offset and Length)	不支持

思科建议在包含 protected_content 关键字的规则中至少包含一个 content 关键字，以确保规则引擎使用快速模式匹配程序，从而加快处理速度和提高性能。在规则中，content 关键字应置于 protected_content 关键字之前。请注意，如果规则包含至少一个 content 关键字，无论您是否启用 content 关键字的“使用快速模式匹配程序” (Use Fast Pattern Matcher) 参数，规则引擎都会使用快速模式匹配程序。



注意 如果创建的规则只包含一个 protected_content 关键字，但没有为该关键字选择 **Not** 选项，可能会使入侵策略无效。

相关主题

[自定义规则创建](#)，第 14 页

[基本 content 和 protected_content 关键字参数](#)，第 26 页

[replace 关键字](#)，第 35 页

基本 content 和 protected_content 关键字参数

可以通过修饰 content 或 protected_content 关键字的参数来限制内容搜索的位置以及大小写。配置用于修饰 content 或 protected_content 关键字的选项可以指定要搜索的内容。

区分大小写



注释 配置 protected_content 关键字时不支持此选项。

可以指示规则引擎在搜索 ASCII 字符串内容匹配时忽略大小写。要使搜索不区分大小写，请在指定内容搜索时选择不区分大小写 (**Case Insensitive**)。

散列类型



注释 此选项仅对于 protected_content 关键字可配置。

使用**散列类型 (Hash Type)** 下拉列表确定用于编码搜索字符串的散列函数。系统支持对 protected_content 搜索字符串进行 SHA-512、SHA-256 和 MD5 散列处理。如果散列内容的长度与所选的散列类型不匹配，系统将不会保存规则。

系统自动选择思科设置的默认值。如果选择了 **Default**，将不会向规则写入特定哈希函数，且系统将假设 SHA-512 为哈希函数。

原始数据

Raw Data 选项指示规则引擎在分析规范化负载数据（由网络分析策略解码）之前分析原始数据包负载，并且此选项不使用参数值。进行规范化之前，可以在分析 telnet 流量时使用此关键字在负载中检查 telnet 协商选项。

不能在同一个 content 或 protected_content 关键字中同时使用 **Raw Data** 选项和任何 HTTP 内容选项。



提示 可以配置 HTTP 检查预处理器 **Client Flow Depth** 和 **Server Flow Depth** 选项，以确定是否在 HTTP 流量中检查原始数据以及检查的原始数据量。

不

选择不选项可搜索与指定内容不匹配的内容。如果创建包含已选择不选项的 content 或 protected_content 关键字的规则，则必须在该规则中至少包含另一个未选择 **Not** 选项的 content 或 protected_content 关键字。



注意 请勿创建仅包含一个已选择 **Not** 选项的 `content` 或 `protected_content` 关键字的规则。否则，可能会使入侵策略无效。

例如，SMTP 规则 1:2541:9 包含三个 `content` 关键字，其中一个选择了**不匹配 (Not)** 选项。如果移除除选择了**不匹配 (Not)** 选项的关键字以外的其他 `content` 关键字，基于该规则的自定义规则将无效。将此类规则添加到入侵策略可能会导致该策略失效。



提示 不能对同一个 `content` 关键字同时选择 **Not** 复选框和 **Use Fast Pattern Matcher** 复选框。

content 和 protected_content 关键字搜索位置

可以使用搜索位置选项指定开始搜索指定内容的位置以及继续搜索的深度。

允许的组合：content 搜索位置参数

可以使用两个 `content` 位置对指定开始搜索指定内容的位置以及继续搜索的深度，如下所述：

- 同时使用**偏移量 (Offset)** 和**深度 (Depth)** 选项可相对于数据包负载起点进行搜索。
- 同时使用**距离 (Distance)** 和**范围内 (Within)** 可相对于当前搜索位置进行搜索。

如果仅指定选项对中的其中一个选项，系统将会假设另一个选项使用默认值。

不能将**偏移量**和**深度**选项与**距离**和**范围内**选项混合使用。例如，不能将**偏移量 (Offset)** 和**范围内 (Within)** 这两个选项配合使用。可以在规则中使用任意数量的位置选项。

如果未指定位置，系统将假设**偏移量 (Offset)** 和**深度 (Depth)** 选项为默认值；也就是说，将从数据包负载起点开始进行内容搜索，直至数据包终点。

还可以使用现有 `byte_extract` 变量指定位置选项的值。



提示 可以在规则中使用任意数量的位置选项。

相关主题

[byte_extract 关键字](#)，第 41 页

允许的组合：protected_content 搜索位置参数

将必填的**长度** `protected_content` 位置选项与**偏移**或**距离**位置选项结合使用，可指定开始搜索指定内容的位置以及继续搜索的深度，如下所示：

- 同时使用**长度 (Length)** 和**偏移 (Offset)** 选项可相对于数据包负载起点搜索受保护字符串。
- 同时使用**长度 (Length)** 和**距离 (Distance)** 选项可相对于当前搜索位置搜索受保护字符串。



提示 不能在单个关键字配置中同时使用**偏移 (Offset)** 和**距离 (Distance)** 选项，但可以在规则内使用任意数量的位置选项。

如果未指定位置，系统将假设使用默认值；也就是说，将从数据包负载起点开始进行内容搜索，直至数据包终点。

还可以使用现有 `byte_extract` 变量指定位置选项的值。

相关主题

[byte_extract 关键字](#)，第 41 页

content 和 protected_content 搜索位置参数

Depth



注释 此选项**仅在配置 content 关键字时可用**。

指定最大内容搜索深度（以字节为单位），从偏移量值起点开始计算，如果没有配置偏移量，则从数据包负载起点开始计算。

例如，如果规则的内容值为 `cgi-bin/phf`，`offset` 值为 3，`depth` 值为 22，规则将从字节 3 开始搜索 `cgi-bin/phf` 字符串内容匹配，并在处理完符合规则报头指定参数的数据包中的 22 个字节（字节 25）后停止。

必须指定一个大于或等于指定内容长度的数值，最多 65535 字节。不能指定值 0。

默认深度是搜索至数据包终点。

Distance

指示规则引擎识别在上一次成功内容匹配后出现指定数量字节的后续内容匹配。

由于偏移量计数器从字节 0 开始计算，因此，应指定比所需字节数小 1 的值，以便从上一次成功内容匹配开始继续搜索。例如，如果指定 4，搜索将从第五个字节开始。

可指定 -65535 到 65535 字节之间的值。如果在距离中指定负值，开始搜索的字节可能位于数据包开头以外。所有计算都会将数据包以外的字节考虑在内，尽管搜索实际上从数据包的第一个字节开始。例如，如果数据包当前位置是第五个字节，下一个内容规则选项指定距离值为 -10，内部值为 20，搜索将从负载起点开始，且内部选项将调整为 15。

默认距离是 0，表示继上一次内容匹配之后数据包中的当前位置。

Length



注释 此选项**仅在配置 protected_content 关键字时可用**。

长度 (Length) `protected_content` 关键字选项表示非散列搜索字符串的长度（以字节为单位）。

例如，如果使用了内容 `Sample1` 生成安全散列值，请将**长度 (Length)** 值设置为 7。必须在该字段中输入一个值。

Offset

指定数据包负载中开始内容搜索的位置与数据包负载起点之间的距离（以字节为单位）。可指定 65535 到 65535 字节之间的值。

由于偏移量计数器从字节 0 开始计算，因此，应指定比所需字节数小 1 的值，以便从数据包负载起点开始继续搜索。例如，如果指定 7，搜索将从第八个字节开始。

默认偏移量是 0，表示数据包起点。

Within



注释 此选项仅在配置 `content` 关键字时可用。

范围内 (Within) 选项指明，要触发规则，下一次内容匹配必须发生在上一次成功内容匹配结束之后指定数量的字节内。例如，如果将**范围内 (Within)** 值指定为 8，下一次内容匹配必须出现在数据包负载中接下来的八个字节之内，否则将无法触发规则的标准。

可以指定一个大于或等于指定内容长度的数值，最多 65535 字节。

范围内 (Within) 的默认设置是搜索至数据包终点。

概述: HTTP content 和 protected_content 关键字参数

通过 `HTTP content` 或 `protected_content` 关键字选项，可以在 HTTP 检查预处理器解码的 HTTP 消息中指定搜索内容匹配项的位置。

以下两个选项搜索 HTTP 响应中的状态字段：

- **HTTP 状态代码**
- **HTTP 状态消息 (HTTP Status Message)**

请注意，尽管规则引擎搜索未规范化的原始状态字段，但这里分别列出这些选项，以方便在下文解释将其他原始 HTTP 字段与规范化 HTTP 字段结合使用时应考虑的限制。

以下五个选项根据情况搜索 HTTP 请求和/或响应中的规范化字段：

- **HTTP URI**
- **HTTP 方法**
- **HTTP 报头 (HTTP Header)**
- **HTTP Cookie**
- **HTTP 客户端正文 (HTTP Client Body)**

以下三个选项根据情况搜索 HTTP 请求和/或响应中的原始（未规范化）非状态字段：

- **HTTP 原始 URI (HTTP Raw URI)**
- **HTTP 原始报头 (HTTP Raw Header)**
- **HTTP 原始 Cookie (HTTP Raw Cookie)**

选择 HTTP content 选项时，请遵循以下准则：

- HTTP content 选项仅适用于 TCP 流量。
- 为避免对性能造成负面影响，应只选择消息中那些可能出现指定内容的部分。

例如，如果流量可能包含大型 cookie（例如，购物车消息中的 cookie），可以在 HTTP 报头中搜索指定内容，而不是在 HTTP cookie 中搜索。

- 为利用 HTTP 检查预处理器规范化以及提高性能，所创建的任何 HTTP 相关规则应包含至少一个已选择 **HTTP URI**、**HTTP Method**、**HTTP Header** 或 **HTTP Client Body** 选项的 content 或 protected_content 关键字。
- 不能将 replace 关键字与 HTTP content 或 protected_content 关键字选项配合使用。

可以指定单个规范化 HTTP 选项或状态字段，或者使用规范化 HTTP 选项与状态字段的任意组合，以指向要匹配的内容区域。但在使用 HTTP 字段选项时，请注意以下限制：

- 不能在同一个 content 或 protected_content 关键字中同时使用**原始数据 (Raw Data)** 选项和任何 HTTP 选项。
- 不能在同一个 content 或 protected_content 关键字中同时使用原始 HTTP 字段选项（**HTTP 原始 URI [HTTP Raw URI]**、**HTTP 原始报头 [HTTP Raw Header]** 或 **HTTP 原始 Cookie [HTTP Raw Cookie]**）及其对应的规范化选项（分别是 **HTTP URI**、**HTTP 报头 [HTTP Header]** 或 **HTTP Cookie**）。
- 不能同时选择使用快速模式匹配程序 (**Use Fast Pattern Matcher**) 和以下一个或多个 HTTP 字段选项：

HTTP 原始 URI (HTTP Raw URI)、**HTTP 原始报头 (HTTP Raw Header)**、**HTTP 原始 Cookie (HTTP Raw Cookie)**、**HTTP Cookie**、**HTTP 方法 (HTTP Method)**、**HTTP 状态消息 (HTTP Status Message)** 或 **HTTP 状态代码 (HTTP Status Code)**

但是，可以在也使用快速模式匹配程序搜索以下其中一个规范化字段的 content 或 protected_content 关键字中包含上述选项：

HTTP URI、**HTTP 报头 (HTTP Header)** 或 **HTTP 客户端正文 (HTTP Client Body)**

例如，如果选择 **HTTP Cookie**、**HTTP 报头 (HTTP Header)** 和使用快速模式匹配程序 (**Use Fast Pattern Matcher**)，规则引擎将会在 HTTP cookie 和 HTTP 报头中搜索内容，但快速模式匹配程序仅适用于 HTTP 报头，而不适用于 HTTP cookie。

- 将受限选项和不受限选项结合使用时，快速模式匹配程序将仅搜索您指定的不受限字段，以测试是否要将规则传递到入侵规则编辑器来完成评估，包括受限字段的评估。

相关主题

[content 关键字快速模式匹配程序参数](#)，第 33 页

HTTP content 和 protected_content 关键字参数

HTTP URI

选择此选项将会在规范化的请求 URI 字段中搜索内容匹配。

请注意，不能将此选项与 `pcre` 关键字 HTTP URI (U) 选项结合使用来搜索相同的内容。



注释 管道化 HTTP 请求数据包包含多个 URI。如果选择了 **HTTP URI**，且规则引擎检测到管道化 HTTP 请求数据包，规则引擎将会搜索数据包中的所有 URI 以进行内容匹配。

HTTP Raw URI

选择此选项将会在规范化的请求 URI 字段中搜索内容匹配。

请注意，不能将此选项与 `pcre` 关键字 HTTP URI (U) 选项结合使用来搜索相同的内容。



注释 管道化 HTTP 请求数据包包含多个 URI。如果选择了 **HTTP URI**，且规则引擎检测到管道化 HTTP 请求数据包，规则引擎将会搜索数据包中的所有 URI 以进行内容匹配。

HTTP 方法

选择此选项将会在请求方法字段中搜索内容匹配，该字段确定要对 URI 中识别出的资源执行的操作（例如 GET 和 POST）。

HTTP Header

选择此选项将会在 HTTP 请求内的规范化报头字段（cookie 除外）中搜索内容匹配；如果 HTTP 检查预处理器的**检查 HTTP 响应 (Inspect HTTP Responses)**选项已启用，还会在响应中搜索内容匹配。

请注意，不能将此选项与 `pcre` 关键字 HTTP 报头 (H) 选项结合使用来搜索相同的内容。

HTTP Raw Header

选择此选项将会在 HTTP 请求内的原始报头字段（cookie 除外）中搜索内容匹配；如果 HTTP 检查预处理器的**检查 HTTP 响应 (Inspect HTTP Responses)**选项已启用，还会在响应中搜索内容匹配。

请注意，不能将此选项与 `pcre` 关键字 HTTP 原始报头 (D) 选项结合使用来搜索相同的内容。

HTTP Cookie

选择此选项将会在规范化 HTTP 客户端请求报头内识别出的任何 cookie 中搜索内容匹配；如果 HTTP 检查预处理器的**检查 HTTP 响应 (Inspect HTTP Responses)**选项已启用，还会在响应 set-cookie 数据中搜索内容匹配。请注意，系统将消息正文中包含的 cookie 看作正文内容。

若要仅对 cookie 进行内容匹配搜索，必须启用 HTTP 检查预处理器的**检查 HTTP Cookie (Inspect HTTP Cookies)** 选项；否则，规则引擎将搜索包括 cookie 在内的整个报头。

请注意以下提示：

- 不能将此选项与 pcre 关键字 HTTP cookie (C) 选项结合使用来搜索相同的内容。
- Cookie: 和 Set-Cookie: 报头名称、标题行中的前导空格以及终止标题行的 CRLF 将作为报头的一部分而非 cookie 的一部分进行检查。

HTTP Raw Cookie

选择此选项将会在原始 HTTP 客户端请求报头内识别出的任何 cookie 中搜索内容匹配；如果 HTTP 检查预处理器的**检查 HTTP 响应 (Inspect HTTP Responses)** 选项已启用，还会在响应 set-cookie 数据中搜索内容匹配；请注意，系统将消息正文中包含的 cookie 看作正文内容。

若要仅对 cookie 进行内容匹配搜索，必须启用 HTTP 检查预处理器的**检查 HTTP Cookie (Inspect HTTP Cookies)** 选项；否则，规则引擎将搜索包括 cookie 在内的整个报头。

请注意以下提示：

- 不能将此选项与 pcre 关键字 HTTP 原始 cookie (K) 选项结合使用来搜索相同的内容。
- Cookie: 和 Set-Cookie: 报头名称、标题行中的前导空格以及终止标题行的 CRLF 将作为报头的一部分而非 cookie 的一部分进行检查。

HTTP Client Body

选择此选项将会在 HTTP 客户端请求消息正文中搜索内容匹配。

请注意，要使此选项起作用，必须为 HTTP 检查预处理器的**HTTP 客户端正文提取深度 (HTTP Client Body Extraction Depth)** 选项指定一个 0 到 65535 之间的值。

HTTP Status Code

选择此选项将会在 HTTP 响应的三位数状态代码中搜索内容匹配。

要使此选项能够返回匹配，必须启用 HTTP 检查预处理器的**检查 HTTP 响应 (Inspect HTTP Responses)** 选项。

HTTP Status Message

选择此选项将会在 HTTP 响应中状态代码随附的文字描述中搜索内容匹配。

要使此选项能够返回匹配，必须启用 HTTP 检查预处理器的**检查 HTTP 响应 (Inspect HTTP Responses)** 选项。

相关主题

[pcre 修饰符选项](#)，第 49 页

[服务器级别 HTTP 规范化选项](#)

概述: content 关键字快速模式匹配程序



注释 配置 `protected_content` 关键字时, 这些选项不可用。

快速模式匹配程序快速确定在将数据包传递到规则引擎之前要对哪些规则进行评估。这项初步工作可大大减少用于数据包评估的规则数量, 从而提高性能。

默认情况下, 快速模式匹配程序会在数据包内搜索规则中指定的最长内容; 这样可最大程度地消除不必要的规则评估。以如下规则片段为例:

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

几乎所有 HTTP 客户端请求都包含内容 `GET`, 但是很少会包含内容 `/exploit.cgi`。使用 `GET` 作为快速模式内容将会导致规则引擎在大多数情况下评估此规则, 但极少会产生匹配。但是, 对于大多数客户端 `GET` 请求, 将不会使用 `/exploit.cgi` 对其进行评估, 从而提高性能。

规则引擎仅在快速模式匹配程序检测到指定内容时根据规则评估数据包。例如, 如果某个规则中的三个 `content` 关键字分别指定内容 `short`、`longer` 和 `longest`, 快速模式匹配程序将使用内容 `longest`, 并且仅在规则引擎在负载中找到 `longest` 的情况下对该规则进行评估。

content 关键字快速模式匹配程序参数

Use Fast Pattern Matcher

使用此选项可指定较短的搜索模式以供快速模式匹配程序使用。理想情况下, 指定的模式在数据包中被找到的可能性低于最长模式, 因此, 因此能够更具体地识别所针对的漏洞。

在同一个 `content` 关键字中选择使用快速模式匹配程序 (**Use Fast Pattern Matcher**) 和其他选项时, 请注意以下限制:

- 只能为每个规则指定一次使用快速模式匹配程序 (**Use Fast Pattern Matcher**)。
- 如果同时选择使用快速模式匹配程序 (**Use Fast Pattern Matcher**) 和不匹配 (**Not**), 将不能使用距离 (**Distance**)、范围内 (**Within**)、偏移 (**Offset**) 和深度 (**Depth**)。
- 不能同时选择“快速模式匹配程序” (**Use Fast Pattern Matcher**) 和以下任何 HTTP 字段选项:

HTTP 原始 URI (HTTP Raw URI)、**HTTP 原始报头 (HTTP Raw Header)**、**HTTP 原始 Cookie (HTTP Raw Cookie)**、**HTTP Cookie**、**HTTP 方法 (HTTP Method)**、**HTTP 状态消息 (HTTP Status Message)** 或 **HTTP 状态代码 (HTTP Status Code)**

但是, 可以在也使用快速模式匹配程序搜索以下其中一个规范化字段的 `content` 关键字中包含上述选项:

HTTP URI、**HTTP 报头 (HTTP Header)** 或 **HTTP 客户端正文 (HTTP Client Body)**

例如, 如果选择 **HTTP Cookie**、**HTTP 报头 (HTTP Header)** 和使用快速模式匹配程序 (**Use Fast Pattern Matcher**), 规则引擎将会在 HTTP cookie 和 HTTP 报头中搜索内容, 但快速模式匹配程序仅适用于 HTTP 报头, 而不适用于 HTTP cookie。

请注意，不能在同一个 `content` 关键字中同时使用原始 HTTP 字段选项（**HTTP 原始 URI [HTTP Raw URI]**、**HTTP 原始报头 [HTTP Raw Header]** 或 **HTTP 原始 Cookie [HTTP Raw Cookie]**）及其对应的规范化选项（分别是 **HTTP URI**、**HTTP 报头 [HTTP Header]** 或 **HTTP Cookie**）。

如果将受限选项和不受限选项结合使用，快速模式匹配程序将仅搜索您指定的不受限字段，以测试是否要将数据包传递到规则引擎以完成评估（包括受限字段的评估）。

- 或者，如果选择使用快速模式匹配程序 (**Use Fast Pattern Matcher**)，还可以选择仅快速模式匹配程序 (**Fast Pattern Matcher Only**) 或快速模式匹配程序偏移和长度 (**Fast Pattern Matcher Offset and Length**) 选项，但不能同时选择这两个选项。
- 检测 Base64 数据时，不能使用快速模式匹配程序。

Fast Pattern Matcher Only

通过此选项，您可以将 `content` 关键字仅用作快速模式匹配程序选项，而不用作规则选项。如果无需规则引擎评估指定的内容，可以使用此选项来节省资源。例如，假设规则仅要求内容 `12345` 位于负载中的任何位置。如果快速模式匹配程序检测到该模式，可根据规则中的其他关键字对数据包进行评估。规则引擎无需重新评估数据包来确定其是否包含模式 `12345`。

如果规则包含其他与指定内容相关的状况，无需使用此选项。例如，如果另一个规则条件尝试确定 `abcd` 是否出现在 `1234` 之前，将无需使用此项选项搜索内容 `1234`。在这种情况下，规则引擎无法确定相对位置，因为选择仅快速模式匹配程序 (**Fast Pattern Matcher Only**) 将会指示规则引擎不搜索指定内容。

使用此选项时请注意以下情况：

- 指定的内容与位置无关，也就是说，该内容可出现在负载中的任何位置；因此，不能使用位置选项（**距离 [Distance]**、**范围内 [Within]**、**偏移 [Offset]**、**深度 [Depth]** 或 **快速模式匹配程序偏移和长度 [Fast Pattern Matcher Offset and Length]**）。
- 不能将此选项与**不匹配 (Not)** 结合使用。
- 不能将此选项与**快速模式匹配程序偏移和长度 (Fast Pattern Matcher Offset and Length)** 结合使用。
- 指定的内容将被视为不区分大小写，因为所有模式均以不区分大小写的方式插入到快速模式匹配程序中；系统会自动处理这种情况，因此您无需在选择此选项时选择不区分大小写 (**Case Insensitive**)。
- 不可在使用**仅快速模式匹配程序**选项的 `content` 关键字后紧接着使用以下关键字（这些关键字设置相对于当前搜索位置的搜索位置）：

- `isdataat`
- `pcre`
- `content`（在选择**距离 [Distance]** 或**范围内 [Within]** 的情况下）
- `content`（在选择**HTTP URI** 的情况下）
- `asn1`

- `byte_jump`
- `byte_test`
- `byte_math`
- `byte_extract`
- `base64_decode`

Fast Pattern Matcher Offset and Length

使用快速模式匹配程序偏移和长度 (**Fast Pattern Matcher Offset and Length**) 选项可指定要搜索的部分内容。如果模式很长，且只需模式的一部分即足以识别出可能是匹配的规则，使用此选项可减少内存消耗。如果快速模式匹配程序选择了某个规则，将会根据该规则评估整个模式。

可以确定快速模式匹配程序要使用的部分，方法是，使用以下语法以字节为单位指定搜索的开始位置（偏移）以及搜索内容的深入度（长度）：

```
offset,length
```

例如，对于以下内容：

```
1234567
```

如果如下指定偏移量和长度字节数：

```
1,5
```

快速模式匹配程序仅搜索内容 23456。

请注意，不能将此选项与 **Fast Pattern Matcher Only** 结合使用。

相关主题

[概述：HTTP content 和 protected_content 关键字参数](#)，第 29 页

[base64_decode 和 base64_data 关键字](#)，第 113 页

replace 关键字

可以在内联部署中使用 `replace` 关键字替换指定内容或替换思科 SSL 设备检测到的 SSL 流量中的内容。

要使用 `replace` 关键字，请构建一个使用 `content` 关键字来查找特定字符串的自定义标准文本规则。然后使用 `replace` 关键字指定一个字符串，以替换该内容。替代值和内容值必须是相同长度的字符串。



注释 不能使用 `replace` 关键字替换 `protected_content` 关键字中的散列内容。

或者，可以用引号将替代字符串引起来，以便向后兼容以前的 Firepower 系统 软件版本。如果不加引号，替代字符串将被自动添加到规则，以使规则在语法上正确。要将前引号或后引号纳入为替代文本的一部分，必须使用反斜杠对引号进行转义，如以下示例所示：

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

每个规则可包含多个 `replace` 关键字，但只能包含一个 `content` 关键字。仅替代规则发现的内容中的第一个实例。

下面介绍 `replace` 关键字的使用示例：

- 如果系统检测到传入数据包包含漏洞，您可以使用一个无害字符串来替换该恶意字符串。有时，这种方法比单纯地丢弃违规数据包更有效。在某些攻击场景中，攻击者只需重新发送被丢弃的数据包，直至该数据包绕过网络防御或对网络造成泛洪攻击。通过用一个字符串替换另一个字符串，而不是丢弃数据包，可以哄骗攻击者认为已发动针对不易受攻击的目标的攻击。
- 如果您担心侦测攻击试图了解您是否正在运行易受攻击的 Web 服务器版本（示例），可以检测传出的数据包，并将横幅替换为自己的文本。



注释 请确保在要其中使用替换规则的内联入侵规则中将规则状态设置为“生成事件” (Generate Events)；如果将规则设置为“丢弃并生成事件” (Drop and Generate Events)，将会导致数据包被丢弃，进而造成无法替换内容。

在替换字符串的过程中，系统会自动更新数据包校验和，以便目标主机可准确收到该数据包。

请注意，不能将 `replace` 关键字与 HTTP 请求消息的 `content` 关键字选项结合使用。

相关主题

[content 和 protected_content 关键字](#)，第 24 页

[概述：HTTP content 和 protected_content 关键字参数](#)，第 29 页

byte_jump 关键字

`byte_jump` 关键字首先计算指定字节段中定义的字节数，然后在数据包中跳过该数量的字节 - 可以从指定字节段的末尾向前跳，也可以从数据包负载起点或末尾向前跳，还可以从与上一次内容匹配有关的点向前跳，具体取决于指定的选项。这对于具有如下特点的数据包很有用：数据包中的特定字节段说明数据包内变量数据包含的字节数。

下表介绍了 `byte_jump` 关键字所需的参数。

表 10: 所需的 `byte_jump` 参数

参数	说明
Bytes	<p>要从数据包采集的字节数量。</p> <p>如果不与 DCE/RPC 一起使用，则允许的值为 0 - 10，具有以下限制：</p> <ul style="list-style-type: none"> • 如果与 From End 参数一起使用，则字节数可以是 0。如果 Bytes 为 0，则提取的值为 0。 • 如果指定除 1、2 或 4 以外的字节数，则必须指定数字类型（十六进制、八进制或十进制。） <p>如果与 DCE/RPC 一起使用，则允许的值为 1、2 和 4。</p>
Offset	<p>从负载开头到开始进行处理之间的字节数。Offset 计数器从字节 0 开始计数，因此，应该如下计算 Offset 值：用从数据包负载起点或上一次成功内容匹配起向前跳所需的字节数减去 1。</p> <p>可指定 -65535 到 65535 字节。</p> <p>您也可以使用现有 <code>byte_extract</code> 变量或 <code>byte_math</code> 结果指定此参数的值。</p>

下表介绍了可用于定义系统如何解释您为必需参数指定的值的选项。

表 11: 其他可选 `byte_jump` 参数

参数	说明
Relative	使偏移量相对于上一次成功内容匹配中找到的上一个模式。
调整	将转换的字节数四舍五入为下一个 32 位边界。
乘数	<p>指明规则引擎应将其与从数据包获取的 <code>byte_jump</code> 值相乘的值，以获得最终的 <code>byte_jump</code> 值。</p> <p>也就是说，规则引擎跳过一个与您通过 <code>Multiplier</code> 参数指定的整数相乘的字节数，而不是跳过指定字节段中定义的字节数。</p>
Post Jump Offset	<p>应用其他 <code>byte_jump</code> 参数后要向前跳或向后跳的字节数（-65535 到 65535）。选择正值将会向前跳，选择负值将会向后跳。将此字段留空或输入 0 将会禁用此字段。</p> <p>请注意，选择 DCE/RPC 参数时，一些 <code>byte_jump</code> 参数不适用。</p>
From Beginning	指明规则引擎应从数据包负载起点开始跳过负载中指定的字节数，而不是从数据包的当前位置开始跳。
From End	跳转将从缓冲区最后一个字节后面的字节开始。

参数	说明
Bitmask	将使用 AND 运算符的指定十六进制位掩码应用于从 Bytes 参数提取的字节数。 位掩码可以是 1 到 4 个字节。 结果将按与掩码中的末尾零的数量相等的位数向右位移。

只能指定 **DCE/RPC**、**Endian** 或 **Number Type**。

如果要定义 `byte_jump` 关键字如何计算字节数，则可以选择下表中介绍的参数。如果未选择字节排序参数，规则引擎使用大端字节顺序。

表 12: 字节排序 `byte_jump` 参数

参数	说明
大端字节	按大端字节顺序处理数据（大端字节顺序是默认的网络字节顺序）。
小端字节	按小端字节顺序处理数据
DCE/RPC	为 DCE/RPC 预处理器处理的流量指定 <code>byte_jump</code> 关键字。 由 DCE/RPC 预处理器确定大端字节顺序或小端字节顺序， Number Type 和 Endian 参数不适用。 如果启用此参数，还可以将 <code>byte_jump</code> 与其他特定 DCE/RPC 关键字结合使用。

使用下表所列的其中一个参数来定义系统如何查看数据包中的字符串。

表 13: 数字类型参数

参数	说明
Hexadecimal String	使用十六进制格式表示转换的字符串数据。
Decimal String	使用十进制格式表示转换的字符串数据。
Octal String	使用八进制格式表示转换的字符串数据。

例如，如果为 `byte_jump` 设置的值如下：

- Bytes = 4
- Offset = 12
- Relative 已启用
- Align 已启用

规则引擎将会计算自上一次成功内容匹配后显示的 13 个字节当中 4 个字节中描述的数量，并在数据包中向前跳过该数量的字节。例如，如果特定数据包中计算出的 4 个字节是 00 00 00 1F，规则引擎

会将其转换为 31。由于指定了 `Align`（指示引擎移到下一个 32 位边界），因此，规则引擎将在数据包中向前跳过 32 个字节。

或者，如果为 `byte_jump` 设置的值如下：

- Bytes = 4
- Offset = 12
- From Beginning 已启用
- Multiplier = 2

规则引擎将会计算在数据包起点后显示的 13 个字节当中 4 个字节中描述的数值。然后，引擎会将该数值乘以 2，以获得将要跳过的字节总数。例如，如果特定数据包中计算出的 4 个字节是 00 00 00 1F，规则引擎会将其转换为 31，然后再乘以 2 得到 62。由于启用了 `From Beginning`，因此，规则引擎会跳过数据包中的前 63 个字节。

相关主题

[byte_extract 关键字](#)，第 41 页

[DCE/RPC 关键字](#)，第 73 页

byte_test 关键字

`byte_test` 关键字根据 `Value` 参数及其运算符测试指定的字节段。

下表介绍了 `byte_test` 关键字所需的参数。

表 14: 所需的 `byte_test` 参数

参数	说明
Bytes	<p>从数据包进行计算的字节数。</p> <p>如果不与 DCE/RPC 配合使用，则允许的值为 1 到 10。但是，如果指定除 1、2 或 4 以外的字节数，则必须指定数字类型（十六进制、八进制或十进制。）</p> <p>如果与 DCE/RPC 一起使用，则允许的值为 1、2 和 4。</p>
值	<p>要测试的值，包括其运算符。</p> <p>支持的运算符：<、>、=、!、&、^、!>、!<、!=、!& 或 !^。</p> <p>例如，如果指定 !1024，<code>byte_test</code> 将会转换该指定数字，且如果该数字不等于 1024，则会生成事件（如果其他所有关键字参数都匹配）。</p> <p>请注意，! 和 != 等效。</p> <p>您也可以使用现有 <code>byte_extract</code> 变量或 <code>byte_math</code> 结果指定此参数的值。</p>

参数	说明
Offset	从负载开头到开始进行处理之间的字节数。偏移量计数器从字节0开始计数，因此，应该如下计算 offset 值：用从数据包负载起点或上一次成功内容匹配起向前计算所需的字节数减去 1。 可以使用现有 <code>byte_extract</code> 变量或 <code>byte_math</code> 结果指定此参数的值。

可以用下表中所述的参数进一步定义系统如何使用 `byte_test` 参数。

表 15: 其他可选 `byte_test` 参数

参数	说明
Bitmask	将使用 AND 运算符的指定十六进制位掩码应用于从 Bytes 参数提取的字节数。 位掩码可以是 1 到 4 个字节。 结果将按与掩码中的末尾零的数量相等的位数向右位移。
Relative	使偏移量相对于上一次成功模式匹配。

只能指定 **DCE/RPC**、字节存储次序 (**Endian**) 或数字类型 (**Number Type**)。

要定义 `byte_test` 关键字如何计算其测试的字节，请从下表中选择参数。如果未选择字节排序参数，规则引擎使用大端字节顺序。

表 16: 字节排序 `byte_test` 参数

参数	说明
大端字节	按大端字节顺序处理数据（大端字节顺序是默认的网络字节顺序）。
小端字节	按小端字节顺序处理数据
DCE/RPC	指定 DCE/RPC 预处理器处理的流量的 <code>byte_test</code> 关键字。 由 DCE/RPC 预处理器确定大端字节顺序或小端字节顺序， Number Type 和 Endian 参数不适用。 如果启用此参数，还可以将 <code>byte_test</code> 与其他特定 DCE/RPC 关键字结合使用。

可以使用下表所列的其中一个参数来定义系统如何在数据包中查看字符串。

表 17: 数字类型 `byte-test` 参数

参数	说明
Hexadecimal String	使用十六进制格式表示转换的字符串数据。
Decimal String	使用十进制格式表示转换的字符串数据。

参数	说明
Octal String	使用八进制格式表示转换的字符串数据。

例如，如果如下指定 `byte_test` 的值：

- Bytes = 4
- Operator and Value > 128
- Offset = 8
- Relative 已启用

规则引擎计算距离（相对于）上一个成功的内容匹配项 9 个字节的 4 个字节中描述的数值，如果计算的数值大于 128 字节，则会触发规则。

相关主题

[byte_extract 关键字](#)，第 41 页

[DCE/RPC 关键字](#)，第 73 页

byte_extract 关键字

可以使用 `byte_extract` 关键字将数据包中指定数量的字节读取到某个变量中。然后，可以在同一规则中使用该变量作为某些其他检测关键字中特定参数的值。

此参数很有用，例如，可用于从其中的特定字节段描述数据包数据所包含的字节数的数据包提取数据大小。例如，特定字节段可能指出后续数据是由 4 个字节组成；您可以提取 4 个字节的数据大小来作为变量值。

可以使用 `byte_extract` 在一条规则中同时创建多达两个独立变量。可以任意多次地重新定义 `byte_extract` 变量；如果输入变量名称相同但变量定义不同的新的 `byte_extract` 关键字，它将覆盖该变量的上一个定义。

下表介绍 `byte_extract` 关键字所需的参数。

表 18: 所需的 `byte_extract` 参数

参数	说明
要提取的字节数	要从数据包采集的字节数量。 如果指定除 1、2 或 4 以外的字节数，则必须指定数字类型（十六进制、八进制或十进制。）

参数	说明
偏移	<p>从负载开头到开始提取数据之间的字节数。可指定 -65535 到 65535 字节。偏移量计数器从字节 0 开始计数，因此，计算偏移量值时，应该用向前计算所需的字节数减去 1。例如，指定 7 将会从 8 字节开始向前计算。规则引擎会从数据包负载起点开始向前计算；如果还指定了 Relative，规则引擎则会从上一次内容成功匹配后开始向前计算。请注意，如果还指定了 Relative，则只能指定负数。</p> <p>可以使用现有 <code>byte_math</code> 结果来指定此参数的值。</p>
变量名称	用于其他检测关键字的参数中的变量名称。可以指定字母数字字符串，但必须以字母开头。

要进一步定义系统如何查找要提取的数据，可以使用下表中所述的参数。

表 19: 其他可选的 `byte_extract` 参数

参数	说明
乘数	从数据包提取的值的乘数。可指定 0 到 65535 之间的任意数字。如果未指定乘数，系统将使用默认值 1。
调整	将提取的数值四舍五入为最接近的 2 字节或 4 字节边界。如果也选择了乘数，系统会在进行舍入之前应用该乘数。
相对	使偏移相对于上一次内容成功匹配的结尾而不是负载起点。
位掩码	<p>使用 AND 运算符将指定的十六进制位掩码应用于从 Bytes to Extract 参数提取的字节。</p> <p>位掩码可以是 1 到 4 个字节。</p> <p>结果将按与掩码中的末尾零的数量相等的位数向右位移。</p>

只能指定 **DCE/RPC**、**Endian** 或 **Number Type**。

要定义 `byte_extract` 关键字如何计算其测试的字节数，可以从下表中选择参数。如果未选择字节排序参数，规则引擎使用大端字节顺序。

表 20: 字节排序 `byte_extract` 参数

参数	说明
大端字节	按大端字节顺序处理数据（大端字节顺序是默认的网络字节顺序）。
小端字节	按小端字节顺序处理数据

参数	说明
DCE/RPC	<p>为 DCE/RPC 预处理器处理的流量指定 <code>byte_extract</code> 关键字。</p> <p>由 DCE/RPC 预处理器确定大端字节顺序或小端字节顺序，数字类型 (Number Type) 和 字节存储次序 (Endian) 参数不适用。</p> <p>如果启用此参数，还可以将 <code>byte_extract</code> 与其他特定 DCE/RPC 关键字结合使用。</p>

可以指定数字类型来将数据读取为 ASCII 字符串。要定义系统如何在数据包中查看字符串，可选择下表中所述的其中一个参数。

表 21: 数字类型 `byte_extract` 参数

参数	说明
Hexadecimal String	以十六进制格式读取提取的字符串数据。
Decimal String	以十进制格式读取提取的字符串数据。
Octal String	以八进制格式读取提取的字符串数据。

例如，如果为 `byte_extract` 指定如下值：

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

那么，规则引擎会将距离（相对于）上一次内容成功匹配 9 字节的四个字节中描述的数字读取到名为 `var` 的变量中（然后，您可以在规则中将该数字指定为某些关键字参数的值）。

下表列出了可以在其中指定 `byte_extract` 关键字中定义的变量的关键字参数。

表 22: 接受 `byte_extract` 变量的参数

关键字	参数
content	Depth、Offset、Distance、Within
byte_jump	Offset
byte_test	Offset、Value
byte_math	RValue、Offset
isdataat	Offset

相关主题

[DCE/RPC 预处理器](#)[DCE/RPC 关键字](#)，第 73 页[基本 content 和 protected_content 关键字参数](#)，第 26 页[byte_jump 关键字](#)，第 36 页[byte_test 关键字](#)，第 39 页[数据包特征](#)，第 96 页

byte_math 关键字

byte_math 关键字对提取的值和指定值或现有变量执行数学运算，并将结果存储在生成的新变量中。然后，可以使用生成的变量作为其他关键字中的参数。

您可以在规则中使用多个 byte_math 关键字执行多个 byte_math 运算。

下表介绍 byte_math 关键字所需的参数。

表 23: 所需的 byte_math 参数

参数	说明
Bytes	<p>从数据包进行计算的字节数。</p> <p>如果不与 DCE/RPC 配合使用，则允许的值为 1 到 10:</p> <ul style="list-style-type: none"> 当运算符为 +、-、* 或 / 时，Bytes 可以为 1 到 10。 当运算符为 << 或 >> 时，Bytes 可以为 1 到 4。 如果指定除 1、2 或 4 以外的字节数，则必须指定数字类型（十六进制、八进制或十进制。） <p>如果与 DCE/RPC 一起使用，则允许的值为 1、2 和 4。</p>
Offset	<p>从负载开头到开始进行处理之间的字节数。offset 计数器在字节 0 处启动，因此请按如下计算 offset 值：将要从数据包负载的开头或（如果指定 Relative）从上次成功内容匹配处向前跳转的字节数减 1。</p> <p>可指定 -65535 到 65535 字节。</p> <p>您还可以在此处指定 byte_extract 变量。</p>
Operator	+、-、*、/、<< 或 >>
RValue	运算符后面的值。这可以是 byte_extract 传递的无符号证书或变量。

参数	说明
Result Variable	<p>byte_math 计算的结果将存储到的变量的名称。可以使用此变量作为其他关键字中的参数。</p> <p>此值存储为无符号整数。</p> <p>此变量名称：</p> <ul style="list-style-type: none"> • 必须使用字母数字字符 • 不得以数字开头 • 可能包含 Microsoft 文件名/变量名称约定支持的特殊字符 • 不能完全由特殊字符组成

下表介绍了可用于定义系统如何解释您为必需参数指定的值的选项。

表 24: 其他可选 *byte_math* 参数

参数	说明
Relative	使偏移相对于在上次成功内容匹配中找到的最后一个模式而不是负载开头。
Bitmask	<p>将使用 AND 运算符的指定十六进制位掩码应用于从 Bytes 参数提取的字节数。</p> <p>位掩码可以是 1 到 4 个字节。</p> <p>结果将按与掩码中的末尾零的数量相等的位数向右位移。</p>

只能指定 **DCE/RPC**、**Endian** 或 **Number Type**。

如果要定义 *byte_math* 关键字如何计算字节数，则可以选择下表中介绍的参数。如果未选择字节排序参数，规则引擎使用大端字节顺序。

表 25: 字节排序 *byte_math* 参数

参数	说明
大端字节	按大端字节顺序处理数据（大端字节顺序是默认的网络字节顺序）。
小端字节	按小端字节顺序处理数据
DCE/RPC	<p>为 DCE/RPC 预处理器处理的流量指定 <i>byte_math</i> 关键字。</p> <p>由 DCE/RPC 预处理器确定大端字节顺序或小端字节顺序，Number Type 和 Endian 参数不适用。</p> <p>当启用此参数时，还可以将 <i>byte_math</i> 与其他特定 DCE/RPC 关键字结合使用。</p>

使用下表所列的其中一个参数来定义系统如何查看数据包中的字符串。

表 26: 数字类型参数

参数	说明
Hexadecimal String	表示十六进制格式的字符串数据。
Decimal String	表示十进制格式的字符串数据。
Octal String	表示八进制格式的字符串数据。

例如, 如果为 `byte_math` 设置的值如下:

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

规则引擎提取数据包中前两个字节中描述的数值, 并将其乘以 RValue (使用现有变量 `height`) 来创建新变量 `area`。

表 27: 接受 `byte_math` 变量的参数

关键字	参数
<code>byte_jump</code>	Offset
<code>byte_test</code>	Offset、Value
<code>byte_extract</code>	Offset
<code>isdataat</code>	Offset

概述: pcre 关键字

`pcre` 关键字使您可以使用兼容 Perl 的正则表达式 (PCRE) 为指定的内容检查数据包负载。使用 PCRE 可避免编写以匹配相同内容的细微变化为目的的多个规则。

搜索可以多种方式显示的内容时, 正则表达式很有用。内容可能有不同的属性; 在尝试从数据包负载中查找内容时, 您会需要考虑其属性。

请注意, 入侵规则使用的正则表达式语法是完整正则表达式库的一个子集, 并且该库中所用命令的语法在某些方面存在不同之处。使用入侵规则编辑器添加 `pcre` 关键字时, 请按以下格式输入完整值:

```
!/pcre/ ismxAEGRBUIPHDMCKSY
```

其中:

- ! 是可选求反（如果要匹配与正则表达式不匹配的模式，请使用此求反）。
- /pcre/ 是兼容 Perl 的正则表达式。
- ismxAEGRBUIPHDMCKSY 是修饰符选项的任意组合。

另请注意，在 PCRE 中使用下表所列字符在数据包负载中搜索特定内容时，必须对这些字符进行转义，以使规则引擎能正确地解译这些字符。

表 28: 转义的 PCRE 字符

必须转义的字符...	使用反斜线...	或使用十六进制代码...
#（散列标记）	\#	\x23
；（分号）	\;	\x3B
（竖线）	\	\x7C
:（冒号）	\:	\x3A

您还可以使用 `m?regex?`，其中 `?` 是除 `/` 以外的分隔符。如果需要在正则表达式中匹配一个正斜杠，但不想用反斜杠来进行转义，可能需要使用此分隔符。例如，可以使用 `m?regex?`

`ismxAEGRBUIPHDMCKSY`，其中 `regex` 是兼容 Perl 的正则表达式，`ismxAEGRBUIPHDMCKSY` 是修饰符选项的任意组合。



提示 或者，可以用引号将兼容 Perl 的正则表达式引起来，例如，`pcre_expression` 或 “`pcre_expression`”。这一做法适合习惯使用旧版本的有经验的用户（旧版本要求必须用引号将正则表达式引起来）。在保存规则后显示该规则时，入侵规则编辑器不显示引号。

pcre 语法

`pcre` 关键字接受兼容 Perl 的正则表达式 (PCRE) 标准语法。以下各节介绍这种语法。



提示 尽管本节介绍可用于 PCRE 的基本语法，但您可能想要参阅专门关于 Perl 和 PCRE 的网上参考资料或书籍，以获取更多高级信息。

元字符

元字符是在正则表达式中具有特殊含义的原义字符。在正则表达式中使用元字符时，必须通过在元字符前添加一个反斜杠来对其进行“转义”。

下表举例说明可用于 PCRE 的元字符。

表 29: PCRE 元字符

元字符	Description	示例
.	匹配除换行符以外的任何字符。如果将 <code>s</code> 用作修饰选项，还将匹配换行符。	<code>abc.</code> 匹配 <code>abcd</code> 、 <code>abc1</code> 、 <code>abc#</code> 等等。
*	匹配字符或表达式的零次或多次出现次数。	<code>abc*</code> 匹配 <code>abc</code> 、 <code>abcc</code> 、 <code>abccc</code> 、 <code>abccccc</code> 等等。
?	匹配字符或表达式的零次或一次出现次数。	<code>abc?</code> 匹配 <code>abc</code> 。
+	匹配字符或表达式的一次或多次出现次数。	<code>abc+</code> 匹配 <code>abc</code> 、 <code>abcc</code> 、 <code>abccc</code> 、 <code>abccccc</code> 等等。
()	组表达。	<code>(abc)+</code> 匹配 <code>abc</code> 、 <code>abcabc</code> 、 <code>abcabcabc</code> 等等。
{}	为字符或表达式指定匹配项数限制。如果要设置下限和上限，请用逗号将下限和上限隔开。	<code>a{4,6}</code> 匹配 <code>aaaa</code> 、 <code>aaaaa</code> 或 <code>aaaaaa</code> 。 <code>(ab){2}</code> 匹配 <code>abab</code> 。
[]	允许定义字符类，并匹配字符集中包含的任意字符或字符组合。	<code>[abc123]</code> 匹配 <code>a</code> 、 <code>b</code> 或 <code>c</code> 等等。
^	匹配字符串开头的内容。如果在字符类中使用，也可用于否定。	<code>^in</code> 匹配 <code>info</code> 中的“in”，但不匹配 <code>bin</code> 中的“in”。 <code>[^a]</code> 匹配不包含 <code>a</code> 的任何内容。
\$	匹配字符串结尾的内容。	<code>ce\$</code> 匹配 <code>announce</code> 中的“ce”，但不匹配 <code>cent</code> 中的“ce”。
	指示 OR 表达式。	<code>(MAILTO HELP)</code> 匹配 <code>MAILTO</code> 或 <code>HELP</code> 。
\	元字符可用作实际字符，还可用于指定预定义的字符类。	<code>\.</code> 匹配句号， <code>*</code> 匹配星号， <code>\\</code> 匹配反斜线，依此类推。 <code>\d</code> 匹配数字字符， <code>\w</code> 匹配字母数字字符，依此类推。

字符类

字符类包括字母字符、数字字符、字母数字字符和空白字符。可以用方括号创建自己的字符类，也可以使用预定义类作为不同字符类型的快捷方式。如果不与其他限定符配合使用，一个字符类通常匹配一个数字或字符。

下表举例说明 PCRE 接受的预定义字符类。

表 30: PCRE 字符类

字符类	说明	字符类定义
<code>\d</code>	匹配数字字符（“数字”）。	<code>[0-9]</code>
<code>\D</code>	对应不是数字字符的任何字符。	<code>[^0-9]</code>
<code>\w</code>	匹配字母数字字符（“单词”）。	<code>[a-zA-Z0-9_]</code>

字符类	说明	字符类定义
\W	匹配不是字母数字字符的任何字符。	[^a-zA-Z0-9_]
\s	匹配空白字符，包括空格、回车符、制表符、换行符和换页符。	[\r\t\n\f]
\S	匹配不是空白字符的任何字符。	[^\r\t\n\f]

pcre 修饰符选项

指定 `pcre` 关键字值中的正则表达式语法后，可以使用修饰选项。这些修饰符执行特定于 Perl、PCRE 和 Snort 的处理功能。修饰符始终按以下格式显示在 PCRE 值的末尾：

```
/pcre/ismxAEGRBUIPHDMCKSY
```

其中，`ismxAEGRBUPHMC` 可以包括下表中的任何修饰选项。



提示 或者，可以用引号将正则表达式和任何修饰选项引起来，例如 `"/pcre/ismxAEGRBUIPHDMCKSY"`。这一做法适合习惯使用旧版本的有经验的用户（旧版本要求必须用引号将正则表达式引起来）。在保存规则后显示该规则时，入侵规则编辑器不显示引号。

下表介绍了可用于执行 Perl 处理功能的选项。

表 31: Perl 相关的后正则表达式选项

选项	说明
i	使正则表达式不区分大小写。
s	点字符 (.) 匹配除换行符和 \n 字符以外的所有字符。可使用 "s" 选项覆盖此选项，这样点字符将匹配所有字符（包括换行符）。
m	默认情况下，一个字符串被视为单行字符，^ 和 \$ 分别匹配特定字符串的开头和结尾。如果使用 "m" 代替选项，^ 和 \$ 将匹配紧接在缓冲区内所有换行符之前或之后的内容，以及位于缓冲区开头或结尾的内容。
x	忽略可能在这一模式中出现的空格数据字符，除非其为转义字符（前面加有反斜线）或包含在字符类中。

下表介绍了可用于正则表达式之后的 PCRE 修饰符。

表 32: PCRE 相关的后正则表达式选项

选项	说明
A	模式必须在字符串开头进行匹配（与在正则表达式中使用 ^ 具有相同的效果）。

选项	说明
E	将 <code>\$</code> 设置为只在目标字符串结尾进行匹配。（如果最后一个字符是换行符，即使没有 <code>E</code> ， <code>\$</code> 也会匹配紧接在该字符之前的内容，但不会匹配任何其他换行符之前的内容）。
G	默认情况下， <code>*</code> 、 <code>+</code> 和 <code>?</code> 是“贪婪”的，这意味着，如果找到两个或更多匹配项，将会选择最长的匹配项。使用 <code>G</code> 字符可使这些字符在后面的无问号字符(?)的情况下总是选择第一个匹配项。例如，在使用 <code>G</code> 修饰符的构造中， <code>*?+?</code> 和 <code>??</code> 将是贪婪字符，而 <code>*</code> 、 <code>+</code> 或 <code>?</code> 在不附带问号的情况下不是贪婪字符。

下表介绍了可用于正则表达式后的 `Snort` 特定修饰符。

表 33: 特定于 `Snort` 的后正则表达式修饰符

选项	说明
R	相对于规则引擎上一次找到的匹配项的结尾搜索匹配的内容。
B	在未被预处理器解码的数据中搜索内容（此选项类似于使用带有 <code>content</code> 或 <code>protected_content</code> 关键字的 <code>Raw Data</code> 参数）。
U	在已由 HTTP 检查预处理器解码的规范化 HTTP 请求消息的 URI 中搜索内容。请注意，不能将此选项与 <code>content</code> 或 <code>protected_content</code> 关键字的 HTTP URI 选项结合使用来搜索相同的内容。 请注意，管道化 HTTP 请求数据包包含多个 URI。包含 U 选项的 PCRE 表达式使规则引擎仅在管道化 HTTP 请求数据包的第一个 URI 中搜索内容匹配。要搜索数据包中的所有 URI，请使用已选择 HTTP URI 的 <code>content</code> 或 <code>protected_content</code> 关键字（可随附或不随附使用 U 选项的 PCRE 表达式）。
I	在已由 HTTP 检查预处理器解码的原始 HTTP 请求消息的 URI 中搜索内容。请注意，不能将此选项与 <code>content</code> 或 <code>protected_content</code> 关键字 HTTP 原始 URI (HTTP Raw URI) 选项结合使用来搜索相同的内容
P	在已由 HTTP 检查预处理器解码的规范化 HTTP 请求消息的正文中搜索内容。
H	在已由 HTTP 检查预处理器解码的 HTTP 请求或响应消息的报头（不包括 <code>cookie</code> ）中搜索内容。请注意，不能将此选项与 <code>content</code> 或 <code>protected_content</code> 关键字 HTTP Header 选项结合使用来搜索相同的内容。
D	在已由 HTTP 检查预处理器解码的原始 HTTP 请求或响应消息的报头（不包括 <code>cookie</code> ）中搜索内容。请注意，不能将此选项与 <code>content</code> 或 <code>protected_content</code> 关键字 HTTP Raw Header 选项结合使用来搜索相同的内容。
M	在已由 HTTP 检查预处理器解码的规范化 HTTP 请求消息的方法字段中搜索内容；该方法字段确定要对 URI 中识别出的资源执行的操作（例如， <code>GET</code> 、 <code>PUT</code> 、 <code>CONNECT</code> 等）。

选项	说明
C	<p>如果 HTTP 检查预处理器的检查 HTTP Cookie (Inspect HTTP Cookies) 选项已启用，将会在 HTTP 请求报头的任何 cookie 中搜索规范化内容；如果该预处理器的检查 HTTP 响应 (Inspect HTTP Responses) 选项已启用，还会在 HTTP 响应报头的任何 set-cookie 中搜索规范化内容。如果未启用检查 HTTP Cookie (Inspect HTTP Cookies) 选项，将会搜索包括 cookie 或 set-cookie 数据在内的整个报头。</p> <p>请注意以下提示：</p> <ul style="list-style-type: none"> • 消息正文中包含的 cookie 将被视为正文内容。 • 不能将此选项与 <code>content</code> 或 <code>protected_content</code> 关键字 HTTP Cookie 选项结合使用来搜索相同的内容。 • <code>Cookie:</code> 和 <code>Set-Cookie:</code> 报头名称、标题行中的前导空格以及终止标题行的 CRLF 将作为报头的一部分而非 cookie 的一部分进行检查。
K	<p>如果 HTTP 检查预处理器的检查 HTTP Cookie (Inspect HTTP Cookies) 选项已启用，将会在 HTTP 请求报头的任何 cookie 中搜索原始内容；如果该预处理器的检查 HTTP 响应 (Inspect HTTP Responses) 选项已启用，还会在 HTTP 响应报头的任何 set-cookie 中搜索原始内容。如果未启用检查 HTTP Cookie (Inspect HTTP Cookies) 选项，将会搜索包括 cookie 或 set-cookie 数据在内的整个报头。</p> <p>请注意以下提示：</p> <ul style="list-style-type: none"> • 消息正文中包含的 cookie 将被视为正文内容。 • 不能将此选项与 <code>content</code> 或 <code>protected_content</code> 关键字 HTTP Raw Cookie 选项结合使用来搜索相同的内容。 • <code>Cookie:</code> 和 <code>Set-Cookie:</code> 报头名称、标题行中的前导空格以及终止标题行的 CRLF 将作为报头的一部分而非 cookie 的一部分进行检查。
S	搜索 HTTP 响应中的三位数状态代码。
Y	搜索 HTTP 响应中状态代码随附的文字描述。



注释 请勿将 U 选项与 R 选项结合使用，否则可能会导致性能问题。此外，请勿将 U 选项与任何其他 HTTP 内容选项（I、P、H、D、M、C、K、S 或 Y）结合使用。

相关主题

[概述：HTTP content 和 protected_content 关键字参数](#)，第 29 页

pcre 示例关键字值

以下示例显示可为 `pcre` 输入的值，并说明每个示例将会匹配的内容。

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

此示例搜索 `feedback` 的数据包负载，`feedback` 后面紧跟着零个或一个数字字符，再紧跟着 `.cgi`，且仅在 URI 数据中进行搜索。

此示例将匹配：

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

此示例不匹配：

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

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

此示例在字符串开头搜索 `ez` 的数据包负载，`ez` 后面跟有一个包含 3 到 5 个字母的单词，该单词后面跟着 `.cgi`。此搜索不区分大小写，且仅搜索 URI 数据。

此示例将匹配：

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

此示例不匹配：

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

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

此示例在 URI 数据中搜索后面跟有 `file` 或 `seek` 的 `mail` 的数据包负载。

此示例将匹配：

- mailfile.cgi
- mailseek.cgi

此示例不匹配:

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

此示例跟在任意数量字符后面的 HTTP 请求中为制表符或换行符搜索 URI 内容的数据包负载。此示例使用 `m?regex?` 以避免在表达式中使用 `http:\\/\`。请注意，冒号前面有一个反斜线。

此示例将匹配:

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

此示例不匹配:

- ftp://ftp.example.com?scriptvar=&othervar=\n...\
- http://www.example.com?scriptvar=|/bin/sh -i|
- **m?http\\x3a\\x2f\\x2f.*=|.|.*\|+?sU**

此示例为带有任意数量字符（包括换行符）的 URL 搜索数据包负载，后面跟有一个等号以及包含任意数量字符或空格的竖线。此示例使用 `m?regex?` 以避免在表达式中使用 `http:\\/\`。

此示例将匹配:

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

此示例不匹配:

- 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**

此示例为任何 MAC 地址搜索数据包负载。请注意，此示例使用反斜杠对冒号进行转义。

metadata 关键字

您可以使用 `metadata` 关键字向规则中添加自己的描述性信息。您还可以使用具有 `service` 参数的 `metadata` 关键字识别网络流量中的应用和端口。可以使用所添加的信息通过适合需求的方式组织或识别规则，并且可以搜索规则中所添加的信息和有关 `service` 参数的信息。

系统根据参数格式验证元数据：

key value

其中，*key* 和 *value* 提供以空格分隔的组合描述。这是 Talos 情报小组 用于向思科提供的规则添加元数据的格式。

也可以使用其他格式：

key = value

例如，通过按如下方式使用类别和子类别，可以使用 *key value* 格式按作者和日期识别规则：

```
author SnortGuru_20050406
```

可以在规则中使用多个 `metadata` 关键字。还可以使用逗号分隔单个 `metadata` 关键字中的多个 *key value* 参数，如以下示例所示：

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

并非只能使用 *key value* 或 *key=value* 格式；但是，应了解根据这些格式进行验证产生的局限性。

要避免的受限字符

请注意以下字符限制：

- 请勿使用分号 (;) 或冒号 (:)。
- 系统将逗号解释为多个 *key value* 或 *key=value* 参数的分隔符。例如：

key value, key value, key value

- 系统将等于 (=) 字符或空格字符解释为 *key* 和 *value* 之间的分隔符。例如：

key value

key=value

允许使用所有其他字。

要避免的保留元数据

请避免在 `metadata` 关键字中使用以下单词作为单个参数或作为 *key value* 参数中的 *key*；这些单词保留供 Talos 使用：

```
application  
engine  
impact_flag
```

```
os
policy
rule-type
rule-flushing
soid
```



注释 如需有关将受限元数据添加到可能不具有预期作用的本地规则方面的帮助，请联系支持部门。

影响级别 1

可以在 `metadata` 关键字中使用以下保留的 *key value* 参数：

```
impact_flag red
```

此 *key value* 参数针对导入的本地规则或使用入侵规则编辑器创建的自定义规则将影响标志设置为红色（级别 1）。

请注意，当 Talos 在思科提供的规则中包含 `impact_flag red` 参数时，Talos 已确定触发该规则的数据包指示源主机或目标主机可能受病毒、特洛伊木马或其他恶意软件的损害。

服务元数据

系统检测在网络中的主机上运行的应用，并将应用协议信息插入网络流量中；无论如何配置发现策略，它都会执行此操作。您可以在 TCP 或 UDP 规则中使用 `metadata` 关键字 `service` 参数来匹配网络流量中的应用协议和端口。您可以在某个规则中组合一个或多个 `service` 应用参数与单个端口参数。

服务应用

可以使用带 `service` 的 `metadata` 关键字作为 *key*，并使用应用作为 *value*，以匹配具备已识别应用协议的数据包。例如，`metadata` 关键字中的以下 *key value* 参数会将规则与 HTTP 流量关联：

```
service http
```

可以识别多个以逗号分隔的应用。例如：

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



注意 如[配置自适应配置文件](#)中所述，必须为入侵规则启用（其默认状态）自适应分析，才能使用服务元数据。

下表介绍与 `service` 关键字配合使用的最常见应用值。



注释 如需有关识别表中未列出的应用方面的帮助，请联系支持部门。

表 34: service 值

值	说明
cvs	当前版本系统
dcerpc	分布式计算环境/远程过程调用系统
dns	域名系统
finger	Finger 用户信息协议
ftp	文件传输协议
ftp-data	文件传输协议（数据通道）
http	超文本传输协议
imap	互联网消息访问协议
isakmp	互联网安全关联和密钥管理协议
mysql	我的结构化查询语言
netbios-dgm	NetBIOS 数据报服务
netbios-ns	NetBIOS 名称服务
netbios-ssn	NETBIOS 会话服务
nntp	网络新闻传输协议
oracle	Oracle 网络服务
外壳	操作系统外壳
pop2	邮局协议第 2 版
pop3	邮局协议第 3 版
smtp	简单邮件传输协议
snmp	简单网络管理协议
ssh	安全外壳网络协议
sunrpc	Sun 远程过程调用协议
telnet	Telnet 网络协议
tftp	简单文件传输协议
x11	X Window 系统

服务端口

可以使用带 `service` 的 `metadata` 关键字作为 `key`，并使用指定端口参数作为 `value`，以定义规则如何与应用结合来匹配端口。

可以指定下表中的任何一个端口值，每条规则一个值。

表 35: `service` 端口值

值	说明
<code>else-ports</code> 或 <code>unknown</code>	<p>如果符合以下任一条件，则系统将应用规则：</p> <ul style="list-style-type: none"> • 数据包应用已知，且匹配规则应用。 • 数据包应用未知，且数据包端口匹配规则端口。 <p>当 <code>service</code> 指定一个不含端口修饰符的应用协议时，<code>else-ports</code> 和 <code>unknown</code> 将产生系统使用的默认行为。</p>
<code>and-ports</code>	<p>如果数据包应用已知且匹配规则应用，则系统应用规则，并且数据包端口匹配规则报头中的端口。您无法在未指定应用的规则中使用 <code>and-ports</code>。</p>
<code>or-ports</code>	<p>如果符合以下任何条件，则系统将应用规则：</p> <ul style="list-style-type: none"> • 数据包应用已知，且匹配规则应用。 • 数据包应用未知，且数据包端口匹配规则端口。 • 数据包应用不匹配规则应用，且数据包端口匹配规则端口。 • 规则未指定应用，且数据包端口匹配规则端口。

请注意以下提示：

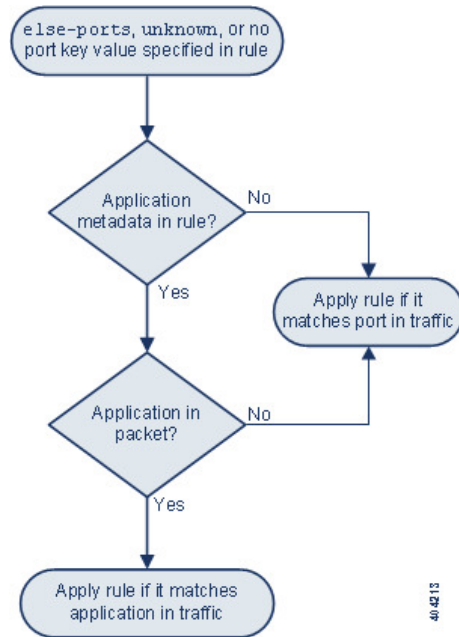
- 必须包含带 `service and-ports` 参数的 `service` 应用参数。
- 如果某个规则指定上表中的多个值，则系统将应用该规则中最后出现的一个值。
- 端口和应用参数可以为任何顺序。

除了 `and-ports` 值，可以包含带或不带一个或多个 `service` 应用参数的 `service` 端口参数。例如：
`service or-ports, service http, service smtp`

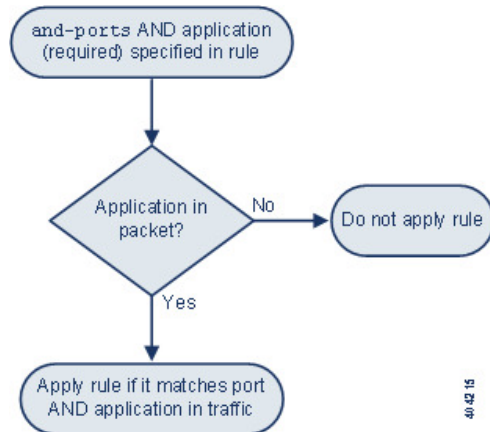
流量中的应用和端口

下图展示了入侵规则支持的应用和端口组合，以及将这些规则限制应用到数据包数据的结果。

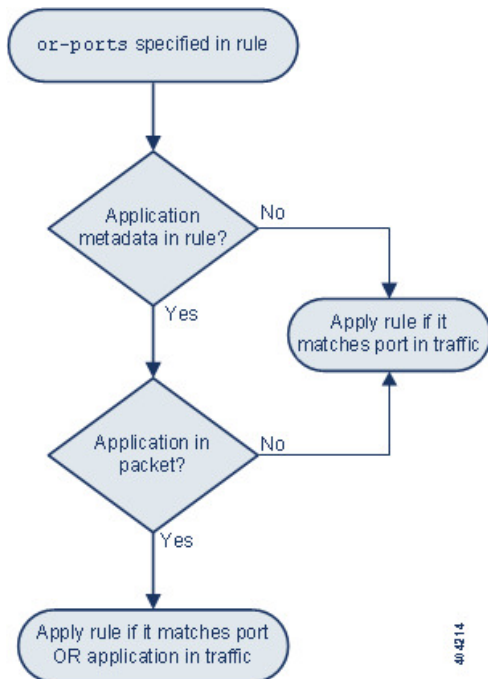
主机应用协议的其他源/目标端口:



主机应用协议和源/目标端口:



主机应用协议或源/目标端口:



示例匹配

以下规则示例使用带 `service` 参数的 `metadata` 关键字，与其匹配和不匹配的数据示例一同显示：

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

示例匹配	不匹配示例
<ul style="list-style-type: none"> • TCP 端口 80 上的 HTTP 流量 • TCP 端口 8080 上的 HTTP 流量 • TCP 端口 80 上的 SMTP 流量 • TCP 端口 8080 上的 SMTP 流量 	<ul style="list-style-type: none"> • 端口 80 或 8080 的 POP3 流量 • 端口 80 或 8080 上的未知应用流量 • 端口 9999 上的 HTTP 流量

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

示例匹配	不匹配示例
<ul style="list-style-type: none"> • 任意端口上的 HTTP 流量 • 端口 80 上的 SMTP 流量 • 端口 8080 上的 SMTP 流量 • 端口 80 和 8080 上的未知应用流量 	<ul style="list-style-type: none"> • 除 80 或 8080 外的端口上的非 HTTP 和非 SMTP 流量

- 以下任何一规则：

- 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;)

示例匹配	不匹配示例
<ul style="list-style-type: none"> • 任意端口上的 HTTP 流量 • 如果数据包应用未知，则为端口80 • 如果数据包应用未知，则为端口8080 	<ul style="list-style-type: none"> • 端口 80 或 8080 的 SMTP 流量 • 端口 80 或 8080 的 POP3 流量

元数据搜索准则

要搜索使用 `metadata` 关键字的规则，请在规则搜索页面上选择 `metadata` 关键字，或者键入元数据的任何部分。例如，可以键入：

- `search`，以显示在其中对 `key` 使用了 `search` 的所有规则。
- `search http`，以显示在其中对 `key` 使用了 `search` 并对 `value` 使用了 `http` 的所有规则。
- `author snortguru`，以显示在其中对 `key` 使用了 `author` 并对 `value` 使用了 `SnortGuru` 的所有规则。
- `author s`，以显示在其中为 `key` 使用了 `author` 并对 `value` 使用了任何词条（例如 `SnortGuru`、`SnortUser1` 或 `SnortUser2`）的所有规则。



提示 如果同时搜索 `key` 和 `value`，应在搜索中使用与规则的 `key value` 参数中使用的相同连接运算符（等号 [=] 或空格字符）；搜索将返回不同的结果，具体取决于 `key` 后面跟的是等号 (=) 还是空格字符。

请注意，无论使用何种格式添加元数据，系统都会将元数据搜索词解释为 `key value` 或 `key=value` 参数的全部或一部分。例如，以下是没有遵循 `key value` 或 `key=value` 格式的有效元数据：

```
ab cd ef gh
```

但是，系统会将此示例中的每个空格解释为 `key` 和 `value` 之间的分隔符。因此，对于并列和单个术语，可以使用以下任何搜索成功查找到包含元数据示例的规则：

```
cd ef
ef gh
ef
```

但是，使用以下搜索不会找到该规则（系统会将其解释为单个 `key value` 参数）：

ab ef

相关主题

[搜索规则](#)，第 19 页

IP 报头值

可以使用关键字来识别数据包 IP 报头中可能存在的攻击或安全策略违规。

fragbits

`fragbits` 关键字检查 IP 报头中的分片和保留位。可以检查每个数据包的 Reserved 位、More Fragments 位和 Don't Fragment 位的任意组合。

表 36: *Fragbits* 参数值

参数	说明
R	“保留”位
M	“更多分片”位
D	“不分片”位

为进一步改进使用 `fragbits` 关键字的规则，可以在规则的参数值后指定下表中所述的任何运算符。

表 37: *Fragbit* 运算符

Operator	说明
加号 (+)	数据包必须匹配所有指定的位。
星号 (*)	数据包可以匹配任何指定的位。
感叹号 (!)	如果未设置任何指定的位，数据包将符合条件。

例如，要生成有关设置了“保留”(Reserved)位（还可能设置了任何其他位）的数据包的事件，请使用 `R+` 作为 `fragbits` 值。

id

`id` 关键字根据您在此关键字的参数中指定的值测试 IP 报头分片标识字段。某些拒绝服务工具和扫描工具将此字段设置为容易检测的特定数字。例如，在 SID 630（检测 Synscan 端口扫描）中，`id` 设置为 39426，这是在扫描仪传输的数据包中用作 ID 号的静态值。



注释 `id` 参数值必须为数字。

ipopts

使用 `IPopts` 关键字可在数据包中搜索指定的 IP 报头选项。下表列出了可用的参数值。

表 38: *IPoption* 参数

参数	说明
rr	记录路由
eol	列表结束
nop	无操作
ts	时间戳
秒	IP 安全选项
lsrr	松散源路由
ssrr	严格源路由
satid	数据流标识符

分析师最经常监视严格和松散源路由，因为这两个选项可能指出欺骗性源 IP 地址。

ip_proto

使用 `ip_proto` 关键字可识别使用指定为关键字值的 IP 协议的数据包。可以为 IP 协议指定 0 到 255 之间的数字。可以将这些协议号与以下运算符结合使用：`<`、`>` 或 `!`。例如，要检查使用非 ICMP 的任何协议的流量，请使用 `!1` 作为 `ip_proto` 关键字的值。也可以在一个规则中多次使用 `ip_proto` 关键字；但请注意，规则引擎会将此关键字的多个实例解释为具有布尔 AND 关系。例如，如果创建一个包含 `ip_proto:!3; ip_proto:!6` 的规则，该规则将忽略使用 GGP 协议和 TCP 协议的流量。

tos

有些网络使用服务类型 (ToS) 值设置在网络上传输的数据包的优先级。使用 `tos` 关键字可根据指定为该关键字的参数的值测试数据包的 IP 报头 ToS 值。对于其 ToS 已设置为指定值且符合规则中规定的其他条件的数据包，使用 `tos` 关键字的规则将会触发。



注释 `tos` 参数值必须为数字。

ToS 字段已在 IP 报头协议中弃用，取而代之的是“差分服务代码点 (DSCP)” (Differentiated Services Code Point [DSCP]) 字段。

ttl

数据包的生存时间 (ttl) 值指明数据包在被丢弃之前可以跳多少次。可以使用 `ttl` 关键字根据指定为关键字参数的值或值范围测试数据包的 IP 报头 ttl 值。将 `ttl` 关键字参数设置为较小的值（例如 0 或

1) 可能会有帮助，因为小的生存时间值有时表示跟踪路由或入侵逃避行为。（但请注意，此关键字的相应值取决于受管设备的位置和网络拓扑。）使用以下语法：

- 将 TTL 值设置为 0 到 255 之间的整数。也可以该值前面加上一个等号 (=)（例如，可以指定 5 或 =5）。
- 使用连字符 (-) 指定 TTL 值的范围（例如，0-2 指定 0 到 2 之间的所有值，-5 指定 0 到 5 之间的所有值，5- 指定 5 到 255 之间的所有值）。
- 使用大于号 (>) 指定 TTL 值大于一个特定值（例如，>3 指定大于 3 的所有值）。
- 使用大于或等于号 (>=) 指定 TTL 值大于或等于一个特定值（例如，>=3 指定大于或等于 3 的所有值）。
- 使用小于号 (<) 指定 TTL 值小于一个特定值（例如，<3 指定小于 3 的所有值）。
- 使用小于或等于号 (<=) 指定 TTL 值小于或等于一个特定值（例如，<=3 指定小于或等于 3 的所有值）。

ICMP 报头值

Firepower 系统支持可用于识别 ICMP 数据包报头中的攻击和安全策略违规的关键字。但请注意，存在的预定义规则检测大多数 ICMP 类型和代码。可考虑启用现有规则或者根据现有规则创建本地规则；如果您从头开始构建 ICMP 规则，可能会更快找到符合您需求的规则。

icmp_id 和 icmp_seq

ICMP 识别号和序列号有助于将 ICMP 响应与 ICMP 请求关联起来。在正常流量中，这些值动态地分配给数据包。有些隐蔽通道和分布式拒绝服务 (DDoS) 程序使用静态 ICMP ID 和序列值。使用以下关键字可识别具有静态值的 ICMP 数据包。

关键字	Definition
icmp_id	检查 ICMP 回应请求或应答数据包的 ICMP ID 号。应使用对应于 ICMP ID 号的数值作为 icmp_id 关键字的参数。
icmp_seq	icmp_seq 关键字检查 ICMP 回应请求或应答数据包的 ICMP 序列。应使用对应于 ICMP 序列号的数值作为 icmp_seq 关键字的参数。

itype

使用 itype 关键字可查找具有特定 ICMP 消息类型值的数据包。您可以指定有效的 ICMP 类型值或无效的 ICMP 类型值来测试不同类型的流量。例如，攻击者可以将 ICMP 类型值设置为超出范围，从而导致拒绝服务和泛洪攻击。

可以使用小于号 (<) 和大于号 (>) 指定 itype 参数值的范围。

例如：

- <35

- >36
- 3<>55

icode

ICMP 消息有时包含代码值，用于在目标不可达的情况下提供有关详细信息。

您可以使用 `icode` 关键字来识别具有特定 ICMP 代码值的数据包。可以指定有效的 ICMP 代码值或无效的 ICMP 代码值来测试不同类型的流量。

可以使用小于号 (<) 和大于号 (>) 指定 `icode` 参数值的范围。

例如：

- 要查找小于 35 的值，请指定 <35。
- 要查找大于 36 的值，请指定 >36。
- 要查找 3 到 55 之间的值，请指定 3<>55。



提示 可以同时使用 `icode` 和 `itype` 关键字来识别与这两者都匹配的流量。例如，要识别包含“ICMP 目标不可达” (ICMP Destination Unreachable) 代码类型和“ICMP 端口不可达” (ICMP Port Unreachable) 代码类型的 ICMP 流量，请指定 3 作为 `itype` 关键字的值（用于“目标不可达” [Destination Unreachable] 类型），并指定 3 作为 `icode` 关键字的值（用于“端口不可达” [Port Unreachable] 类型）。

TCP 报头值和数据流大小

Firepower 系统支持专门用于使用数据包 TCP 报头和 TCP 数据流大小来识别尝试的攻击的关键字。

ack

可以使用 `ack` 关键字将某个值与数据包的 TCP 确认号进行比较。如果数据包的 TCP 确认号与为 `ack` 关键字指定的值相匹配，则会触发规则。

`ack` 参数值必须为数字。

标志

可以使用 `flags` 关键字指定 TCP 标志的任意组合，如果在已检查的数据包中设置此关键字，将触发规则。



注释 在使用 `A+` 作为 `flags` 的值的一般情况下，应转为使用具有 `established` 值的 `flow` 关键字。通常，如果使用标志以确保标志的所有组合均已检测到，应使用具有 `stateless` 值的 `flow` 关键字。

可以检查或忽略下表中所述的 `flag` 关键字的值。

表 39: `flag` 参数

参数	TCP 标志
Ack	确认数据。
Psh	数据应该在此数据包中发送。
Syn	新的连接。
Urg	包含紧急数据的数据包。
Fin	关闭的连接。
Rst	中止的连接。
CWR	ECN 堵塞窗口已减少。这以前是 R1 参数，仍支持向后兼容。
ECE	ECN 响应。这以前是 R2 参数，仍支持向后兼容。

使用 `flags` 关键字时，可以使用运算符来指示系统如何匹配多个标志。下表介绍了这些运算符。

表 40: 与 `flags` 配合使用的运算符

Operator	Description	示例
all	数据包必须包含所有指定的标志。	选择 <code>Urg</code> 和 <code>all</code> 可规定数据包必须包含紧急标志，且可以包含任何其他标志。
any	数据包可包含任何指定的标志。	选择 <code>Ack</code> 、 <code>Psh</code> 和 <code>any</code> 可规定必须设置 <code>Ack</code> 和/或 <code>Psh</code> 标志才能触发规则，且也可以对数据包设置其他标志。
not	数据包不得包含指定的标志集。	选择 <code>Urg</code> 和 <code>not</code> 可规定不对会触发此规则的数据包设置紧急标志。

流

可以使用 `flow` 关键字选择由规则根据会话特征进行的检查的数据包。`flow` 关键字允许您指定规则应用的流量的方向，从而将规则应用于客户端流量或服务器流量。要指定 `flow` 关键字如何检查数据包，可以设置要分析的流量的方向、已检查的数据包的状态以及这些数据包是否是重建数据流的一部分。

数据包状态检测发生在规则处理之后。如果要使某个 TCP 规则忽略无状态流量（尚未建立会话情景的流量），必须将 `flow` 关键字添加到该规则，并为该关键字选择 **Established** 参数。如果要使某个 UDP 规则忽略无状态流量，必须将 `flow` 关键字添加到该规则，并选择 **Established** 参数和/或方向参数。这样，TCP 或 UDP 规则就会执行数据包状态检查。

如果添加方向参数，规则引擎将只检查具有已建立状态且流向与指定方向匹配的数据包。例如，如果将具有 `established` 参数和 `From Client` 参数的 `flow` 关键字添加到某个规则，且该规则会在检测到 TCP 或 UDP 连接的情况下触发，那么规则引擎将只检查从特定客户端发送的数据包。



提示 为了获得最佳性能，应始终在 TCP 规则或 UDP 会话规则中包含 `flow` 关键字。

下表介绍了可为 `flow` 关键字指定的数据流相关参数：

表 41: 状态相关的 `flow` 参数

参数	说明
<code>Established</code>	在已建立连接的情况下触发。
<code>Stateless</code>	无论数据流处理器的状态如何，都会触发。

下表介绍了可为 `flow` 关键字指定的方向选项：

表 42: `flow` 方向参数

参数	说明
<code>To Client</code>	服务器响应时触发。
<code>To Server</code>	客户端响应时触发。
<code>From Client</code>	客户端响应时触发。
<code>From Server</code>	服务器响应时触发。

请注意，`From Server` 和 `To Client` 执行相同的功能，`To Server` 和 `From Client` 执行相同的功能。这些选项是为了是规则具有上下文和可读性。例如，如果要创建用于检测从服务器向客户端发起的木马攻击的规则，应使用 `From Server`。但是，如果要创建用于检测从客户端向服务器发出的木马攻击的规则，应使用 `From Client`。

下表介绍了可为 `flow` 关键字指定的数据流相关参数：

表 43: 数据流相关的 `flow` 参数

参数	说明
<code>忽略流流量</code>	重建流数据包时不触发。
<code>仅流流量</code>	仅在重建流数据包时触发。

例如，可以使用 `To Server`, `Established`, `Only Stream Traffic` 作为 `flow` 关键字的值，这样将会检测在建立的会话中从客户端流向服务器并且由数据流预处理器重组的流量。

seq

使用 `seq` 关键字可指定静态序列号值。序列号与指定参数相匹配的数据包将会触发包含此关键字的规则。虽然此关键字很少使用，但它有助于识别使用生成的具有静态序列号的数据包的攻击和网络扫描。

window

可以使用 `window` 关键字指定感兴趣的 TCP 窗口大小。包含此关键字的规则在遇到具有指定 TCP 窗口大小的数据包时，都会触发。虽然此关键字很少使用，但它有助于识别使用生成的具有静态 TCP 窗口大小的数据包的攻击和网络扫描。

stream_size

可以将 `stream_size` 关键字与数据流预处理器配合使用，以确定 TCP 数据流的大小（以字节为单位），具体格式如下：

```
direction,operator,bytes
```

其中，`bytes` 是字节数。必须以逗号 (,) 分隔参数中的每个选项。

下表介绍了可为 `stream_size` 关键字指定的不区分大小写的方向选项：

表 44: `stream_size` 关键字定向参数

参数	说明
<code>client</code>	当来自客户端的数据流与指定数据流大小相匹配时触发。
<code>server</code>	当来自服务器的数据流与指定数据流大小相匹配时触发。
<code>both</code>	当来自客户端和服务器的流量都与指定数据流大小相匹配时触发。 例如，如果来自客户端的流量大于 200 字节，且来自服务器的流量也大于 200 字节，参数 <code>both, >, 200</code> 将会触发。
<code>either</code>	当来自客户端或服务器流量与指定数据流大小相匹配时触发（无论哪一种情况先发生）。 例如，如果来自客户端的流量大于 200 字节，或来自服务器的流量大于 200 字节，参数 <code>either, >, 200</code> 将会触发。

下表介绍了可与 `stream_size` 关键字配合使用的运算符：

表 45: `stream_size` 关键字参数运算符

Operator	说明
<code>=</code>	等于
<code>!=</code>	不等于
<code>></code>	大于

Operator	说明
<	小于
>=	大于或等于
<=	小于或等于

例如，可以使用 `client, >=, 5001216` 作为 `stream_size` 关键字的参数，以检测从客户端发往服务器的且大于或等于 5001216 字节的 TCP 数据流。

stream_reassembly 关键字

如果单个连接上的已检测流量与规则条件相匹配，则可以使用 `stream_reassemble` 关键字为该连接启用或禁用 TCP 流重组。或者，可以在一个规则中多次使用此关键字。

可使用以下语法启用或禁用数据流重组：

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

下表介绍可与 `stream_reassemble` 关键字配合使用的可选参数。

表 46: `stream_reassemble` 可选参数

参数	说明
<code>noalert</code>	无论规则中是否指定任何其他检测选项，都不生成事件。
<code>fastpath</code>	当有匹配时，忽略连接流量的其余部分。

例如，以下规则禁用 TCP 客户端数据流重组，而且不针对在 HTTP 响应中检测到 200 OK 状态代码的连接生成事件：

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

SSL 关键字

可以使用 SSL 规则关键字调用安全套接字层 (SSL) 预处理器，并从加密会话中的数据包提取有关 SSL 版本和会话状态的信息。

客户机和服务器进行通信以使用 SSL 或安全传输层 (TLS) 建立加密会话时，它们之间会交换握手消息。虽然在会话中传输的数据是加密的，但握手消息没有加密。

SSL 预处理器从特定握手字段提取状态和版本信息。握手中的两个字段分别指明用于加密会话的 SSL 或 TLS 版本以及握手的阶段。

ssl_state

ssl_state 关键字可用于匹配加密会话的状态信息。要同时检查所用的两个或更多 SSL 版本，请在规则中使用多个 ssl_version 关键字。

如果规则使用 ssl_state 关键字，规则引擎将调用 SSL 预处理器来检查流量的 SSL 状态信息。

例如，要检测是否有攻击者试图通过发送具有超长长度和过量数据的 clientHello 消息来造成服务器缓冲区溢出，可以使用带有 client_hello 参数的 ssl_state 关键字，然后检查异常大的数据包。

可使用逗号分隔列表为 SSL 状态指定多个参数。如果列出多个参数，系统将使用 OR 运算符对这些参数进行评估。例如，如果指定 client_hello 和 server_hello 作为参数，系统将会根据带有 client_hello 或 server_hello 的流量对规则进行评估。

还可以否定任何参数；例如：

```
!client_hello, !unknown
```

为确保连接已达到状态集中的每种状态，应使用具有 ssl_state 规则选项的多个规则。ssl_state 关键字将以下标识符作为参数：

表 47: ssl_state 参数

参数	目的
client_hello	当客户端请求加密会话时，匹配消息类型为 ClientHello 的握手消息。
server_hello	当服务器响应客户端的加密会话请求时，匹配消息类型为 ServerHello 的握手消息。
client_keyx	当客户端向服务器发出密钥以确认收到来自服务器的密钥时，匹配消息类型为 ClientKeyExchange 的握手消息。
server_keyx	当客户端向服务器发出密钥以确认收到来自服务器的密钥时，匹配消息类型为 ServerKeyExchange 的握手消息。
unknown	匹配任何握手消息类型。

ssl_version

ssl_version 关键字可用于匹配加密会话的版本信息。如果规则使用 ssl_version 关键字，规则引擎将调用 SSL 预处理器来检查流量的 SSL 版本信息。

例如，如果知道 SSL 2 版本中存在缓冲区溢出漏洞，可以使用带有 sslv2 参数的 ssl_version 关键字来识别使用该 SSL 版本的流量。

可使用逗号分隔列表为 SSL 版本指定多个参数。如果列出多个参数，系统将使用 OR 运算符对这些参数进行评估。例如，如果要识别任何未使用 SSLv2 的加密流量，可以向规则添加 ssl_version:ssl_v3,tls1.0,tls1.1,tls1.2。这样，规则将会评估任何使用 SSL 3 版本、TLS 1.0 版本、TLS 1.1 版本或 TLS 1.2 版本的流量。

ssl_version 关键字将以下 SSL/TLS 版本标识符作为参数：

表 48: `ssl_version` 参数

参数	目的
<code>sslv2</code>	匹配使用安全套接字层 (SSL) 2 版本编码的流量。
<code>sslv3</code>	匹配使用安全套接字层 (SSL) 3 版本编码的流量。
<code>tlsv1.0</code>	匹配使用传输层安全 (TLS) 1.0 版本编码的流量。
<code>tlsv1.1</code>	匹配使用传输层安全 (TLS) 1.1 版本编码的流量。
<code>tlsv1.2</code>	匹配使用传输层安全 (TLS) 1.2 版本编码的流量。

appid 关键字

可以使用 `appid` 关键字识别数据包中的应用协议、客户端应用或 Web 应用。例如，可以针对据知易受特定漏洞影响的特定应用。

在入侵规则的 `appid` 关键字中，点击 **配置 AppID (Configure AppID)** 以选择一个或多个要检测的应用。

浏览可用应用

首次开始构建条件时，**可用应用 (Available Applications)** 列表不受限制，并且显示系统检测的每个应用（每页 100 个）：

- 要翻页浏览应用，请点击列表下方的箭头。
- 要打开弹出窗口，显示有关应用特性的摘要信息以及可点选的互联网搜索链接，请点击应用旁边的 **信息** (i)。

使用应用过滤器

为帮助查找要匹配的应用，您可以通过以下方式限制 **Available Applications** 列表：

- 要搜索应用，请点击列表上方的 **Search by name** 提示，然后键入名称。列表会在您键入内容时进行更新，以显示匹配的应用。
- 要通过应用过滤器来限制应用，请使用 **应用过滤器 (Application Filters)** 列表。**Available Applications** 列表在您应用过滤器时进行更新。为方便起见，系统使用 **解锁图标** 来标记系统只能在解密流量中识别（在加密或未加密流量中无法识别）的应用。



注释 如果您在 **Application Filters** 列表中选择一个或多个过滤器，并在这种状态下搜索 **Available Applications** 列表，系统会使用 AND 运算将您的选择与搜索过滤出的 **Available Applications** 列表进行组合。

选择应用

要选择单个应用，请选择该应用并点击**添加到规则 (Add to Rule)**。要选择当前受限制视图中的所有应用，请右键点击并选择**全选 (Select All)**。

应用层协议值

虽然预处理器执行应用层协议值的大部分检查和规范化工作，但您仍可以使用各种预处理器选项来检查应用层值。

RPC 关键字

`rpc` 关键字识别 TCP 或 UDP 数据包中的开放网络计算远程过程调用 (ONC RPC) 服务。这使您可以检测尝试识别主机上 RPC 程序的行为。入侵者可以使用 RPC 端口映射程序来确定网络上是否运行着可以利用的任何 RPC 服务。他们还可能尝试访问不使用端口映射程序运行 RPC 的其他端口。下表列出了 `rpc` 关键字接受的参数。

表 49: `rpc` 关键字参数

参数	说明
<code>application</code>	RPC 应用编号
<code>procedure</code>	调用的 RPC 过程
<code>version</code>	RPC 版本

要为 `rpc` 关键字指定参数，请使用以下语法：

```
application,procedure,version
```

其中，`application` 是 RPC 应用编号，`procedure` 是 RPC 过程编号，`version` 是 RPC 版本号。必须指定 `rpc` 关键字的所有参数 - 如果无法指定其中一个参数，请将其替换为星号 (*)。

例如，要搜索具有任意程序或版本的 RPC 端口映射程序（以数字 100000 表示的 RPC 应用），可使用 `100000,*,*` 作为参数。

ASN.1 关键字

`asn1` 关键字使您可以解码整个或部分数据包，以查找各种恶意编码。

下表介绍了 `asn1` 关键字的参数。

表 50: `asn.1` 关键字参数

参数	说明
无效位串编码	检测可远程攻击的无效位串编码。

参数	说明
双溢出	检测大于标准缓冲区的双 ASCII 编码。这是 Microsoft Windows 中的一个已知漏洞，但目前不知道哪些服务可能会被利用。
超长长度	检测长度大于提供的参数的 ASN.1 类型。例如，如果将 Oversize Length 设置为 500，任何大于 500 的 ASN.1 类型都会触发规则。
绝对偏移	设置从数据包负载起点算起的绝对偏移量。（请记住，偏移量计数器从字节 0 开始计算。）例如，如果要解码 SNMP 数据包，请将 Absolute Offset 设置为 0，但不设置 Relative Offset。Absolute Offset 可以是正数或负数。
相对偏移	从上一次成功内容匹配、pcre 或 byte_jump 算起的相对偏移量。要解码紧接在内容“foo”后的 ASN.1 序列，请将 Relative Offset 设置为 0，但不设置 Absolute Offset。Relative Offset 可以是正数或负数。（请记住，偏移量计数器从字节 0 开始计算。）

例如，Microsoft ASN.1 库中存在一个会造成缓冲区溢出的已知漏洞，使得攻击者能够利用包含特制的身份验证数据包的条件。当系统解码 ASN.1 数据时，数据包中的攻击代码可以在具有系统级别权限的主机上执行，或可能导致 DoS 条件。以下规则使用 `asn1` 关键字检测试图利用此漏洞的行为：

```
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;)
```

当有 TCP 流量从 `$EXTERNAL_NET` 变量中定义的使用任何端口的任何 IP 地址流向 `$HOME_NET` 变量中定义的使用端口 445 的任何 IP 地址，上述规则将会生成事件。此外，它仅对与服务器之间建立的 TCP 连接执行规则。然后，该规则在特定位置对特定内容进行测试。最后，该规则使用 `asn1` 关键字检测位串编码和双 ASCII 编码，以及确定自上一次成功内容匹配结束以来从 55 字节起算超过 100 字节的 `asn.1` 类型长度。（请记住，`offset` 计数器从字节 0 开始计算。）

urilen 关键字

可以将 `urilen` 关键字和 HTTP 检查预处理器结合使用，以检查 HTTP 流量中特定长度、小于最大长度、大于最小长度或在指定范围内的 URI。

在 HTTP 检查预处理器对数据包进行规范化和检查后，规则引擎将根据规则评估数据包，并确定 URI 是否与 `urilen` 关键字指定的长度条件相匹配。可以使用此关键字来检测试图利用 URI 长度漏洞的攻击，例如，创建缓冲区溢出，以使攻击者可以在具有系统级别权限的主机上形成 DoS 条件或执行代码。

在规则中使用 `urilen` 关键字时，请注意：

- 实际上，`urilen` 关键字总是与 `flow:established` 关键字以及一个或多个其他关键字结合使用。
- 规则协议始终是 TCP。
- 目标端口始终是 HTTP 端口。

可以使用十进制字节数、小于号 (<) 和大于号 (>) 指定 URI 长度。

例如：

- 指定 5 将会检测长度为 5 字节的 URI。
- 指定 < 5 (用一个空格字符隔开) 将会检测长度小于 5 字节的 URI。
- 指定 > 5 (用一个空格字符隔开) 将会检测长度大于 5 字节的 URI。
- 指定 3 <> 5 (<> 前后各有一个空格字符) 将会检测长度为 3 到 5 字节的 URI。

例如，Novell 服务器的监控和诊断实用程序 iMonitor 2.4 版中存在一个已知漏洞，该漏洞来自 eDirectory 8.8 版。包含过长 URI 的一个数据包造成缓冲区溢出，使得攻击者能够利用包含特制数据包的条件，该数据包可以在具有系统级别权限的主机上执行或可能导致 DoS 条件。以下规则使用 `urilen` 关键字检测试图利用此漏洞的行为：

```
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;)
```

当有 TCP 流量从 `$EXTERNAL_NET` 变量中定义的使用任何端口的任何 IP 地址流向 `$HOME_NET` 变量中定义的使用 `$HTTP_PORTS` 变量中定义的端口的任何 IP 地址，上述规则将会生成事件。此外，仅针对与服务器之间建立的 TCP 连接根据该规则评估数据包。该规则使用 `urilen` 关键字检测长度超过 8192 字节的任何 URI。最后，该规则在 URI 中搜索不区分大小写的特定内容 `/nds/`。

相关主题

[入侵规则报头协议](#)，第 4 页

[入侵规则报头源和目标端口](#)，第 8 页

[预定义默认变量](#)

DCE/RPC 关键字

下表中描述的三个 DCE/RPC 关键字可用于监控 DCE/RPC 会话流量的漏洞。当系统处理带有这些关键字的规则时，会调用 DCE/RPC 预处理器。

表 51: DCE/RPC 关键字

使用.....	使用方式...	要检测的内容...
<code>dce_iface</code>	独立	识别特定 DCE/RPC 服务的数据包
<code>dce_opnum</code>	在前面加上 <code>dce_iface</code>	识别特定 DCE/RPC 服务操作的数据包
<code>dce_stub_data</code>	在前面加上 <code>dce_iface</code> 和 <code>dce_opnum</code>	定义特定操作请求或响应的存根数据

请注意，在上表中，应始终在 `dce_opnum` 前面加上 `dce_iface`，并应始终在 `dce_stub_data` 前面加上 `dce_iface` 和 `dce_opnum`。

也可以将这些 DCE/RPC 关键字与其他规则关键字结合连用。请注意，对于 DCE/RPC 规则，应使用选择了 **DCE/RPC** 参数的 `byte_jump`、`byte_test` 和 `byte_extract` 关键字。

思科建议在包含 DCE/RPC 关键字的规则中至少包含一个 `content` 关键字，以确保规则引擎使用快速模式匹配程序，从而加快处理速度和提高性能。请注意，如果规则包含至少一个 `content` 关键字，无论您是否启用 `content` 关键字的 **Use Fast Pattern Matcher** 参数，规则引擎都会使用快速模式匹配程序。

在以下情况下，可以将 DCE/RPC 版本及相邻报头信息用作匹配的内容：

- 规则不包括其他 `content` 关键字
- 规则包含另一个 `content` 关键字，但 DCE/RPC 版本及相邻信息代表比其他内容更独特的模式
例如，DCE/RPC 版本及相邻信息更有可能比单个字节的内容更加独特。

应使用以下其中一个版本及相邻信息内容匹配来终止限定规则：

- 对于面向连接的 DCE/RPC 规则，使用内容 `|05 00 00|`（用于 05 主要版本、00 次要版本和请求 PDU [协议数据单元] 类型 00）。
- 对于无连接的 DCE/RPC 规则，使用内容 `|04 00|`（用于 04 版本和请求 PDU 类型 00）。

在这两种情况下，都应将版本及相邻信息的 `content` 关键字放在规则末尾，以调用快速模式匹配程序而不重复 DCE/RPC 预处理器已完成的处理。请注意：将 `content` 关键字放在规则末尾这种做法适用于被用作调用快速模式匹配程序手段的版本内容，对于规则中的其他内容匹配无需这样做。

相关主题

[DCE/RPC 预处理器](#)

[content 和 protected_content 关键字](#)，第 24 页

[content 关键字快速模式匹配程序参数](#)，第 33 页

[概述：byte_jump 和 byte_test 关键字](#)

[byte_extract 关键字](#)，第 41 页

dce_iface

可以使用 `dce_iface` 关键字识别特定 DCE/RPC 服务。

或者，还可以将 `dce_iface` 与 `dce_opnum` 和 `dce_stub_data` 关键字结合使用，以进一步限制要检查的 DCE/RPC 流量。

固定的 16 字节通用唯一标识符 (UUID) 用于识别分配给每个 DCE/RPC 服务的应用接口。例如，UUID `4b324fc8-670-01d3-1278-5a47bf6ee188` 识别 DCE/RPC `lanmanserver` 服务（又称为 `srvsvc` 服务），该服务提供大量用于共享对等网络打印机、文件和 SMB 命名管道的管理功能。DCE/RPC 预处理器使用 UUID 及相关报头值来跟踪 DCE/RPC 会话。

接口 UUID 是由 5 个十六进制字符串（字符串之间用连字符分隔）组成：

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

可以通过输入整个 UUID（包括连字符）来指定接口，如以下用于 `netlogon` 接口的 UUID 中所示：

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

请注意，必须以大端字节顺序指定 UUID 中的前三个字符串。尽管发布的接口列表和协议分析工具通常以正确的字节顺序显示 UUID，但您可能需要在输入前重新排列 UUID 字节顺序。考虑以下所示的信使服务 UUID，在原始 ASCII 文本中，该 UUID 的前三个字符串有时可能会以小端字节顺序显示：

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

可以为 `dce_iface` 关键字指定这个相同的 UUID，方法是先插入连字符，然后以大端字节顺序放置前三个字符串，如下所示：

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

尽管一个 DCE/RPC 会话可能包含发向多个接口的请求，但在一个规则中只能包含一个 `dce_iface` 关键字。可创建其他规则来检测其他接口。

DCE/RPC 应用接口也有接口版本号。或者，可以指定带有运算符的接口版本，用该操作符指明版本是等于、不等于、小于还是大于指定值。

除了 TCP 分段或 IP 分片外，还可以对面向连接和无连接的 DCE/RPC 进行分片。通常，将任何 DCE/RPC 分片（第一个除外）与指定接口相关联没有任何作用，而且这样做可能导致大量误报。但是，为了提高灵活性，可以根据指定接口对所有分片进行评估。

下表总结了 `dce_iface` 关键字参数。

表 52: `dce_iface` 参数

参数	说明
Interface UUID	UUID（包括连字符），用于识别要在 DCE/RPC 流量中检测的特定服务的应用接口。与指定接口相关的任何请求将匹配接口 UUID。
Version	或者，可以选择应用接口版本号 0 到 65535 和一个操作符，以指明是否检测大于 (>)、小于 (<)、等于 (=) 或不等于 (!) 指定值的版本。
All Fragments	或者，可以选择匹配与 DCE/RPC 分片相关的所有接口和（如有指定）接口版本。默认情况下，此参数被禁用，表示关键字仅在第一个分片或整个未分片数据包与指定接口相关时才进行匹配。请注意，启用此参数可能会导致误报。

dce_opnum 关键字

可以将 `dce_opnum` 关键字和 DCE/RPC 预处理器结合使用，以检测识别 DCE/RPC 服务提供的一个或多个特定操作的数据包。

客户端功能调用请求特定服务函数（这些函数在 DCE/RPC 规范中称为操作）。操作编号 (opnum) 用于识别 DCE/RPC 报头中的特定操作。漏洞可能会针对特定操作。

例如，UUID 12345678-1234-abcd-ef00-01234567cffb 识别用于 netlogon 服务的接口；该服务提供几十个不同的操作，其中之一是操作 6，NetrServerPasswordSet 操作。

应该在 `dce_opnum` 关键字前面加上 `dce_iface` 关键字，以识别操作的服务。

dce_stub_data 关键字

可以为特定操作指定一个 0 到 65535 之间的十进制值，可以指定一系列由连字符分隔的操作，或者指定逗号分隔的操作和范围列表，其中的操作和范围可按任何顺序排列。

以下任何示例都将指定有效的 netlogon 操作编号：

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

dce_stub_data 关键字

可以将 `dce_stub_data` 关键字与 DCE/RPC 预处理器结合使用，以指定无论任何其他规则选项如何，规则引擎都应从存根数据的开头开始检查。紧跟在 `dce_stub_data` 关键字后面的数据包负载规则选项相对于存根数据缓冲区适用。

DCE/RPC 存根数据提供客户端程序调用和 DCE/RPC 运行时系统之间的接口，这种机制可提供对于 DCE/RPC 至关重要的例程和服务。DCE/RPC 漏洞在 DCE/RPC 数据包的存根数据部分中识别出。由于存根数据与特定的操作或函数调用相关，因此，应始终在 `dce_stub_data` 前面加上 `dce_iface` 和 `dce_opnum`，以识别相关的服务和操作。

`dce_stub_data` 关键字没有参数。

SIP 关键字

有四个 SIP 关键字可用于监控 SIP 会话流量的漏洞。

请注意，SIP 协议容易受到拒绝服务 (DoS) 攻击。基于速率的攻击防御可能对解决这类攻击的规则有利。

sip_header 关键字

可以使用 `sip_header` 关键字从提取的 SIP 请求或响应报头开头开始检查，并将检查限制为仅针对报头字段。

`sip_header` 关键字没有参数。

以下示例规则分片指向 SIP 报头并匹配 CSeq 报头字段：

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

相关主题

[动态入侵规则状态](#)

[基于速率的攻击防御](#)

sip_body 关键字

可以使用 `sip_body` 关键字在提取的 SIP 请求或响应消息正文开头开始检查，并将检查限制为仅针对消息正文。

`sip_body` 关键字没有参数。

以下示例规则分片指向 SIP 消息正文，并匹配所提取 SDP 数据的 c（连接信息）字段中的特定 IP 地址：

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

请注意，规则不仅限于搜索 SDP 内容。SIP 预处理器将提取整个消息正文并使其可供规则引擎使用。

sip_method 关键字

每个 SIP 请求中的 *method* 字段用于识别请求的目的。可以使用 sip_method 关键字测试特定方法的 SIP 请求。使用逗号隔开多种方法。

可以指定以下当前定义的任何 SIP 方法：

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

方法不区分大小写。可以使用逗号分隔多个方法。

由于可能在将来定义新的 SIP 方法，因此也可以指定自定义方法（即，当前未定义的方法）。RFC 2616 中定义了接受的字段值，该规范允许除控制字符和分隔符（例如 =、(和 }）以外的所有字符。有关被排除分隔符的完整列表，请参阅 RFC 2616。如果系统在流量中遇到指定的自定义方法，它将检查数据包报头，但不检查消息。

系统最多支持 32 种方法，包括 21 种当前定义的方法和 11 种其他方法。系统将忽略您可能配置的任何未定义的方法。请注意，总共有 32 种方法，包括使用**检查方法 (Methods to Check)** SIP 预处理器选项指定的方法。

如果使用否定形式，只能指定一种方法。例如：

```
!invite
```

但请注意，规则中的多个 sip_method 关键字与 **AND** 运算相关联。例如，为测试除 invite 和 cancel 以外的所有提取的方法，将会使用两个否定形式的 sip_method 关键字：

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

思科建议在包含 sip_method 关键字的规则中至少包含一个 content 关键字，以确保规则引擎使用快速模式匹配程序，从而加快处理速度和提高性能。请注意，如果规则包含至少一个 content 关键字，无论是否启用 content 关键字的**使用快速模式匹配程序 (Use Fast Pattern Matcher)** 参数，规则引擎都会使用快速模式匹配程序。

相关主题

[SIP 预处理器选项](#)

[content 和 protected_content 关键字](#)，第 24 页

[content 关键字快速模式匹配程序参数](#)，第 33 页

sip_stat_code 关键字

每个 SIP 响应中的三位数状态代码指明请求操作的结果。您可以使用 sip_stat_code 关键字测试特定状态代码的 SIP 响应。

可以指定一个一位响应型数字（1 到 9）、一个特定的三位数（100 到 999）或者包含这两项的任意组合的逗号分隔列表。如果列表中的任何一个数字与 SIP 响应中的代码相匹配，则列表匹配。

下表介绍了可指定的 SIP 状态代码值。

表 53: sip_stat_code 值

要检测的内容...	需指定的内容...	示例	会检测的内容...
特定状态代码	三位数状态代码	189	189
任何以指定一位数开始的三位数代码	一位数	1	1xx; 即, 100、101、102 等
值列表	以逗号分隔的特定代码与一位数的组合	222, 3	222 以及 300、301、302 等

另请注意，规则引擎不使用快速模式匹配程序搜索用 sip_stat_code 关键字指定的值，无论规则是否包含 content 关键字。

GTP 关键字

有三个 GSRP 隧道协议 (GTP) 关键字可用于检查 GTP 命令通道的 GTP 版本、消息类型和信息元素。GTP 关键字不可与其他入侵规则关键字（例如 content 或 byte_jump）结合使用。必须在使用 gtp_info 或 gtp_type 关键字的每个规则中使用 gtp_version 关键字。

gtp_version 关键字

可以使用 gtp_version 关键字检查 GTP 控制信息以确定 GTP 版本为 0、1 还是 2。

由于不同的 GTP 版本定义不同的信息类型和信息元素，因此，使用 gtp_type 或 gtp_info 关键字时必须同时使用 gtp_version。可以将值指定为 0、1 或 2。

gtp_type 关键字

每条 GTP 消息由一种消息类型标识，消息类型由一个数值和一个字符串组成。可以使用 gtp_type 关键字检查特定 GTP 消息类型的流量。由于不同的 GTP 版本定义不同的信息类型和信息元素，因此在使用 gtp_type 或 gtp_info 关键字时必须同时使用 gtp_version。

可以为消息类型指定定义的十进制值，可以指定定义的字符串，或者指定包含这两项的任意组合的逗号分隔列表，如下示例所示：

```
10, 11, echo_request
```

系统使用 OR 操作来匹配列出的每个值或字符串。值和字符串的列出顺序并不重要。列表中的任何一个值或字符串均与此关键字匹配。如果尝试保存包含无法识别的字符串或超出范围的值的规则，将会出现错误消息。

请注意，下表中不同的 GTP 版本有时会对同一种消息类型使用不同的值。例如，sgsn_context_request 这一消息类型在 GTPv0 和 GTPv1 中值是 50，但在 GTPv2 中值是 130。

gtp_type 关键字匹配不同的值，具体取决于数据包中的版本号。在上述示例中，在 GTPv0 或 GTPv1 数据包中，此关键字匹配消息类型值 50，在 GTPv2 数据包中，则匹配值 130。如果数据包中的消息类型值不是在数据包中指定的版本的已知值，此关键字不会匹配数据包。

如果为消息类型指定一个整数，则当关键字中的消息类型与 GTP 数据包中的该值匹配时，关键字将会匹配，无论数据包中指定的版本如何。

下表列出了系统识别出的为每种 GTP 消息类型定义的值和字符串。

表 54: GTP 消息类型

值	版本 0	版本 1	版本 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
4	node_alive_request	node_alive_request	不适用
5	node_alive_response	node_alive_response	不适用
6	redirection_request	redirection_request	不适用
7	redirection_response	redirection_response	不适用
16	create_pdp_context_request	create_pdp_context_request	不适用
17	create_pdp_context_response	create_pdp_context_response	不适用
18	update_pdp_context_request	update_pdp_context_request	不适用
19	update_pdp_context_response	update_pdp_context_response	不适用
20	delete_pdp_context_request	delete_pdp_context_request	不适用
21	delete_pdp_context_response	delete_pdp_context_response	不适用
22	create_aa_pdp_context_request	init_pdp_context_activation_request	不适用
23	create_aa_pdp_context_response	init_pdp_context_activation_response	不适用
24	delete_aa_pdp_context_request	不适用	不适用
25	delete_aa_pdp_context_response	不适用	不适用
26	error_indication	error_indication	不适用
27	pdu_notification_request	pdu_notification_request	不适用
28	pdu_notification_response	pdu_notification_response	不适用

值	版本 0	版本 1	版本 2
29	pdu_notification_reject_request	pdu_notification_reject_request	不适用
30	pdu_notification_reject_response	pdu_notification_reject_response	不适用
31	不适用	supported_ext_header_notification	不适用
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
38	不适用	不适用	change_notification_request
39	不适用	不适用	change_notification_response
48	identification_request	identification_request	不适用
49	identification_response	identification_response	不适用
50	sgsn_context_request	sgsn_context_request	不适用
51	sgsn_context_response	sgsn_context_response	不适用
52	sgsn_context_ack	sgsn_context_ack	不适用
53	不适用	forward_relocation_request	不适用
54	不适用	forward_relocation_response	不适用
55	不适用	forward_relocation_complete	不适用
56	不适用	relocation_cancel_request	不适用
57	不适用	relocation_cancel_response	不适用
58	不适用	forward_sms_context	不适用
59	不适用	forward_relocation_complete_ack	不适用
60	不适用	forward_sms_context_ack	不适用
64	不适用	不适用	modify_bearer_command
65	不适用	不适用	modify_bearer_failure_indication

值	版本 0	版本 1	版本 2
66	不适用	不适用	delete_bearer_command
67	不适用	不适用	delete_bearer_failure_indication
68	不适用	不适用	bearer_resource_command
69	不适用	不适用	bearer_resource_failure_indication
70	不适用	ran_info_relay	downlink_failure_indication
71	不适用	不适用	trace_session_activation
72	不适用	不适用	trace_session_deactivation
73	不适用	不适用	stop_paging_indication
95	不适用	不适用	create_bearer_request
96	不适用	mbms_notification_request	create_bearer_response
97	不适用	mbms_notification_response	update_bearer_request
98	不适用	mbms_notification_reject_request	update_bearer_response
99	不适用	mbms_notification_reject_response	delete_bearer_request
100	不适用	create_mbms_context_request	delete_bearer_response
101	不适用	create_mbms_context_response	delete_pdn_request
102	不适用	update_mbms_context_request	delete_pdn_response
103	不适用	update_mbms_context_response	不适用
104	不适用	delete_mbms_context_request	不适用
105	不适用	delete_mbms_context_response	不适用
112	不适用	mbms_register_request	不适用
113	不适用	mbms_register_response	不适用
114	不适用	mbms_deregister_request	不适用
115	不适用	mbms_deregister_response	不适用
116	不适用	mbms_session_start_request	不适用
117	不适用	mbms_session_start_response	不适用
118	不适用	mbms_session_stop_request	不适用

值	版本 0	版本 1	版本 2
119	不适用	mbms_session_stop_response	不适用
120	不适用	mbms_session_update_request	不适用
121	不适用	mbms_session_update_response	不适用
128	不适用	ms_info_change_request	identification_request
129	不适用	ms_info_change_response	identification_response
130	不适用	不适用	sgsn_context_request
131	不适用	不适用	sgsn_context_response
132 个	不适用	不适用	sgsn_context_ack
133	不适用	不适用	forward_relocation_request
134	不适用	不适用	forward_relocation_response
135	不适用	不适用	forward_relocation_complete
136	不适用	不适用	forward_relocation_complete_ack
137	不适用	不适用	forward_access
138	不适用	不适用	forward_access_ack
139	不适用	不适用	relocation_cancel_request
140	不适用	不适用	relocation_cancel_response
141	不适用	不适用	configuration_transfer_tunnel
149	不适用	不适用	detach
150	不适用	不适用	detach_ack
151	不适用	不适用	cs_paging
152	不适用	不适用	ran_info_relay
153	不适用	不适用	alert_mme
154 种	不适用	不适用	alert_mme_ack
155	不适用	不适用	ue_activity
156	不适用	不适用	ue_activity_ack
160	不适用	不适用	create_forward_tunnel_request

值	版本 0	版本 1	版本 2
161	不适用	不适用	create_forward_tunnel_response
162	不适用	不适用	suspend
163	不适用	不适用	suspend_ack
164	不适用	不适用	继续执行
165	不适用	不适用	resume_ack
166	不适用	不适用	create_indirect_forward_tunnel_request
167	不适用	不适用	create_indirect_forward_tunnel_response
168	不适用	不适用	delete_indirect_forward_tunnel_request
169	不适用	不适用	delete_indirect_forward_tunnel_response
170	不适用	不适用	_access_bearer_request
171	不适用	不适用	release_access_bearer_response
176	不适用	不适用	downlink_data
177	不适用	不适用	downlink_data_ack
179	不适用	不适用	pgw_restart
180 个	不适用	不适用	pgw_restart_ack
200	不适用	不适用	update_pdn_request
201	不适用	不适用	update_pdn_response
211	不适用	不适用	modify_access_bearer_request
212	不适用	不适用	modify_access_bearer_response
231	不适用	不适用	mbms_session_start_request
232	不适用	不适用	mbms_session_start_response
233	不适用	不适用	mbms_session_update_request
234	不适用	不适用	mbms_session_update_response
235	不适用	不适用	mbms_session_stop_request
236	不适用	不适用	mbms_session_stop_response
240	data_record_transfer_request	data_record_transfer_request	不适用

值	版本 0	版本 1	版本 2
241	data_record_transfer_response	data_record_transfer_response	不适用
254	不适用	end_marker	不适用
255	pdu	pdu	不适用

gtp_info 关键字

一条 GTP 消息可以包含多个信息元素，其中的每一个元素均由已定义的一个数值和一个字符串来识别。可以使用 `gtp_info` 关键字在指定的信息元素开头开始检查，并将检查限于指定的信息元素。由于不同的 GTP 版本定义不同的消息类型和信息元素，因此在使用此关键字时还必须使用 `gtp_version`。

可以为信息元素指定已定义的十进制值或字符串。可以指定一个值或字符串，也可以在一个规则中使用多个 `gtp_info` 关键字来检查多个信息元素。

如果一条消息包含相同类型的多个信息元素，系统将会全部检查这些元素来进行匹配。如果信息元素按无效顺序出现，将仅检查最后一个实例。

请注意，不同的 GTP 版本有时对同一个信息元素使用不同的值。例如，`cause` 这个信息元素在 GTPv0 和 GTPv1 中值是 1，但在 GTPv2 中值是 2。

`gtp_info` 关键字匹配不同的值，具体取决于数据包中的版本号。在上述示例中，在 GTPv0 或 GTPv1 数据包中，此关键字匹配信息元素值 1，在 GTPv2 数据包中，则匹配值 2。如果数据包中的信息元素值不是在数据包中指定的版本的已知值，此关键字不会匹配数据包。

如果为信息元素指定一个整数，则当关键字中的消息类型与 GTP 数据包中的该值匹配时，关键字将会匹配，无论数据包中指定的版本如何。

下表列出了系统识别出的每个 GTP 信息元素的值和字符串。

表 55: GTP 信息元素

值	版本 0	版本 1	版本 2
1	cause	cause	imsi
2	imsi	imsi	cause
3	rai	rai	recovery
4	tlli	tlli	不适用
5	p_tmsi	p_tmsi	不适用
6	qos	不适用	不适用
8	recording_required	recording_required	不适用
9	authentication	authentication	不适用

值	版本 0	版本 1	版本 2
11	map_cause	map_cause	不适用
12	p_tmsi_sig	p_tmsi_sig	不适用
13	ms_validated	ms_validated	不适用
14	recovery	recovery	不适用
15	selection_mode	selection_mode	不适用
16	flow_label_data_1	teid_1	不适用
17	flow_label_signalling	teid_control	不适用
18	flow_label_data_2	teid_2	不适用
19	ms_unreachable	teardown_ind	不适用
20	不适用	nsapi	不适用
21	不适用	ranap	不适用
22	不适用	rab_context	不适用
23	不适用	radio_priority_sms	不适用
24	不适用	radio_priority	不适用
25	不适用	packet_flow_id	不适用
26	不适用	charging_char	不适用
27	不适用	trace_ref	不适用
28	不适用	trace_type	不适用
29	不适用	ms_unreachable	不适用
71	不适用	不适用	apn
72	不适用	不适用	ambr
73	不适用	不适用	ebi
74	不适用	不适用	ip_addr
75	不适用	不适用	mei
76	不适用	不适用	msisdn
77	不适用	不适用	indication

值	版本 0	版本 1	版本 2
78	不适用	不适用	pco
79	不适用	不适用	paa
80	不适用	不适用	bearer_qos
80	不适用	不适用	flow_qos
82	不适用	不适用	rat_type
83	不适用	不适用	serving_network
84	不适用	不适用	bearer_tft
85	不适用	不适用	tad
86	不适用	不适用	uli
87	不适用	不适用	f_teid
88	不适用	不适用	tmsi
89	不适用	不适用	cn_id
90	不适用	不适用	s103pdf
91	不适用	不适用	s1udf
92	不适用	不适用	delay_value
93	不适用	不适用	bearer_context
94	不适用	不适用	charging_id
95	不适用	不适用	charging_char
96	不适用	不适用	trace_info
97	不适用	不适用	bearer_flag
99	不适用	不适用	pdn_type
100	不适用	不适用	pti
101	不适用	不适用	drx_parameter
103	不适用	不适用	gsm_key_tri
104	不适用	不适用	umts_key_cipher_quin
105	不适用	不适用	gsm_key_cipher_quin

值	版本 0	版本 1	版本 2
106	不适用	不适用	umts_key_quin
107	不适用	不适用	eps_quad
108	不适用	不适用	umts_key_quad_quin
109	不适用	不适用	pdn_connection
110	不适用	不适用	pdn_number
111	不适用	不适用	p_tmsi
112	不适用	不适用	p_tmsi_sig
113	不适用	不适用	hop_counter
114	不适用	不适用	ue_time_zone
115	不适用	不适用	trace_ref
116	不适用	不适用	complete_request_msg
117	不适用	不适用	guti
118	不适用	不适用	f_container
119	不适用	不适用	f_cause
120	不适用	不适用	plmn_id
121	不适用	不适用	target_id
123	不适用	不适用	packet_flow_id
124	不适用	不适用	rab_ctxt
125	不适用	不适用	src_rnc_pdcph
126	不适用	不适用	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	不适用
131	apn	apn	change_report_action
132 个	protocol_config	protocol_config	fq_csid

值	版本 0	版本 1	版本 2
133	gsn	gsn	信道
134	msisdn	msisdn	emlpp_pri
135	不适用	qos	node_type
136	不适用	authentication_qu	fqdn
137	不适用	tft	ti
138	不适用	target_id	mbms_session_duration
139	不适用	utran_trans	mbms_service_area
140	不适用	rab_setup	mbms_session_id
141	不适用	ext_header	mbms_flow_id
142	不适用	trigger_id	mbms_ip_multicast
143	不适用	omc_id	mbms_distribution_ack
144 个	不适用	ran_trans	rfsp_index
145	不适用	pdp_context_pri	uci
146	不适用	addi_rab_setup	csg_info
147	不适用	sgsn_number	csg_id
148	不适用	common_flag	cmi
149	不适用	apn_restriction	service_indicator
150	不适用	radio_priority_lcs	detach_type
151	不适用	rat_type	ldn
152	不适用	user_loc_info	node_feature
153	不适用	ms_time_zone	mbms_time_to_transfer
154 种	不适用	imei_sv	throttling
155	不适用	camel	ARP
156	不适用	mbms_ue_context	epc_timer
157	不适用	tmp_mobile_group_id	signalling_priority_indication
158	不适用	rim_routing_addr	tmgi

值	版本 0	版本 1	版本 2
159	不适用	mbms_config	mm_srvcc
160	不适用	mbms_service_area	flags_srvcc
161	不适用	src_rnc_pdcch	nمبر
162	不适用	addi_trace_info	不适用
163	不适用	hop_counter	不适用
164	不适用	plmn_id	不适用
165	不适用	mbms_session_id	不适用
166	不适用	mbms_2g3g_indicator	不适用
167	不适用	enhanced_nsapi	不适用
168	不适用	mbms_session_duration	不适用
169	不适用	addi_mbms_trace_info	不适用
170	不适用	mbms_session_repetition_num	不适用
171	不适用	mbms_time_to_data	不适用
173	不适用	bss	不适用
174	不适用	cell_id	不适用
175	不适用	pdu_num	不适用
177	不适用	mbms_bearer_capab	不适用
178	不适用	rim_routing_disc	不适用
179	不适用	list_pfc	不适用
180 个	不适用	ps_xid	不适用
181	不适用	ms_info_change_report	不适用
182	不适用	direct_tunnel_flags	不适用
183	不适用	correlation_id	不适用
184	不适用	bearer_control_mode	不适用
185	不适用	mbms_flow_id	不适用
186	不适用	mbms_ip_multicast	不适用

值	版本 0	版本 1	版本 2
187	不适用	mbms_distribution_ack	不适用
188	不适用	reliable_inter_rat_handover	不适用
189	不适用	rfsp_index	不适用
190	不适用	fqdn	不适用
191	不适用	evolved_allocation1	不适用
192 个	不适用	evolved_allocation2	不适用
193	不适用	extended_flags	不适用
194	不适用	uci	不适用
195	不适用	csg_info	不适用
196	不适用	csg_id	不适用
197	不适用	cmi	不适用
198	不适用	apn_ambr	不适用
199	不适用	ue_network	不适用
200	不适用	ue_ambr	不适用
201	不适用	apn_ambr_nsapi	不适用
202	不适用	ggsn_backoff_timer	不适用
203	不适用	signalling_priority_indication	不适用
204	不适用	signalling_priority_indication_nsapi	不适用
205	不适用	high_bitrate	不适用
206	不适用	max_mbr	不适用
251	charging_gateway_addr	charging_gateway_addr	不适用
255	private_extension	private_extension	private_extension

SCADA 关键字

规则引擎使用 Modbus、DNP3、CIP 和 S7Commplus 规则来访问某些协议字段。

Modbus 关键字

可以单独使用 Modbus 关键字，也可以将它与其他关键字（例如 `content` 和 `byte_jump` 关键字）结合使用。

modbus_data

可以使用 `modbus_data` 关键字指向 Modbus 请求或响应中 Data 字段的开头。

modbus_func

可以使用 `modbus_func` 关键字来匹配 Modbus 应用层请求或响应报头中的“函数代码” (Function Code) 字段。可以为 Modbus 函数代码指定一个已定义的十进制值或一个已定义的字符串。

下表列出了系统识别出的为 Modbus 函数代码定义的值和字符串。

表 56: Modbus 函数代码

值	字符串
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

值	字符串
43	encapsulated_interface_transport

modbus_unit

可以使用 `modbus_unit` 关键字来匹配 Modbus 请求或响应报头中的 Unit ID 字段。

DNP3 关键字

可以单独使用 DNP3 关键字，也可以将它与其他关键字（例如 `content` 和 `byte_jump` 关键字）结合使用。

dnp3_data

可以使用 `dnp3_data` 关键字指向重组 DNP3 应用层分片的开头。

DNP3 预处理器将链路层帧重组到应用层分片中。`dnp3_data` 关键字指向每个应用层分片的开头；其他规则选项可匹配分片中的重组数据，而无需每 16 个字节分隔数据并添加校验和。

dnp3_func

可以使用 `dnp3_func` 关键字来匹配 DNP3 应用层请求或响应报头中的“函数代码” (Function Code) 字段。可以为 DNP3 函数代码指定一个已定义的十进制值或一个已定义的字符串。

下表列出了系统识别出的为 DNP3 函数代码定义的值和字符串。

表 57: DNP3 函数代码

值	字符串
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

值	字符串
12	freeze_at_time_nr
13	cold_restart
14	warm_restart
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

可以使用 `dnp3_ind` 关键字来匹配 DNP3 应用层响应报头中“内部指示” (Internal Indications) 字段中的标志。

可以为单个已知标志或以逗号分隔的标志列表指定字符串，如以下示例所示：

```
class_1_events, class_2_events
```

如果指定多个标志，此关键字将会匹配列表中的任何标志。要检测标志组合，可在一个规则中多次使用 `dnf3_ind` 关键字。

以下列表提供了系统识别出的用于已定义的 DNP3 内部指示标志的字符串语法。

```
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
```

dnf3_obj

可以使用 `dnf3_obj` 关键字来匹配请求或响应中的 DNP3 对象报头。

DNP3 数据由一系列不同类型的 DNP3 对象组成，例如模拟输入、二进制输入，等等。每种类型均以组进行识别，例如模拟输入组、二进制输入组等，每个组可由一个十进制值进行识别。每个组中的对象均以对象变体进一步识别，例如 16 位整数、32 位整数、短浮点等，每个这些变体均指定对象的数据格式。每种类型的对象变体也可以十进制值进行识别。

可以通过为对象报头组类型和对象变体类型分别指定一个十进制数值来识别对象报头。这两种类型的组合可定义特定类型的 DNP3 对象。

CIP 和 ENIP 关键字

可以单独使用以下关键字或组合来创建自定义入侵规则，这些规则根据 CIP 预处理器检测的 CIP 和 ENIP 流量识别攻击。对于可配置的关键字，指定在允许的范围内的单个整数。有关详细信息，请参阅 [CIP 预处理器](#)。

表 58:

此关键字...	匹配...	范围
<code>cip_attribute</code>	CIP 消息中的“对象类/实例属性”字段。指定单个定义的整数值。	0 - 65535
<code>cip_class</code>	CIP 消息中的“对象类”字段。指定单个定义的整数值。	0 - 65535

此关键字...	匹配...	范围
<code>cip_conn_path_class</code>	连接路径中的“对象类”。指定单个整数值。	0 - 65535
<code>cip_instance</code>	CIP 消息中的“实例 ID”字段。指定单个整数值。	0 - 4284927295
<code>cip_req</code>	服务请求消息。	不适用
<code>cip_rsp</code>	服务响应消息。	不适用
<code>cip_service</code>	CIP 服务请求消息中的“服务”字段中。指定单个整数值。	0 - 127
<code>cip_status</code>	CIP 服务响应消息中的“状态”字段中。指定单个整数值。	0 - 255
<code>enip_command</code>	EthNet/IP 报头中的命令代码。指定单个整数值。	0 - 65535
<code>enip_req</code>	EthNet/IP 请求消息。	不适用
<code>enip_rsp</code>	EthNet/IP 响应消息。	不适用

S7Commplus 关键字

可以单独使用 S7Commplus 关键字或组合来创建自定义入侵规则，这些规则根据 S7Commplus 预处理器检测的流量识别攻击。对于可配置的关键字，指定在允许的范围内的单个已知值或单个整数。有关详细信息，请参阅[S7Commplus 预处理器](#)。

请注意以下提示：

- 同一规则中的多个 S7commplus 关键字都使用 AND 运算。
- 在同一规则中使用多个 `s7commplus_func` 或 `s7commplus_opcode` 关键字会否定该规则，并且从不会匹配流量。要使用这些关键字搜索多个值，请创建多个规则。

s7commplus_content

在 S7Commplus 入侵规则中使用 `content` 或 `protected_content` 关键字之前，请使用 `s7commplus_content` 关键字将光标定位到数据包负载的开头。有关详细信息，请参阅[content](#) 和 [protected_content](#) 关键字，第 24 页。

s7commplus_func

使用 `s7commplus_func` 关键字匹配 S7Commplus 报头中的以下值之一：

- `explore`
- `createobject`

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

请注意，数字表达式允许使用其他值。

s7complus_opcode

使用 `s7complus_opcode` 关键字匹配 S7Complus 报头中的以下值之一：

- 请求
- response
- 通知
- response2
- 0x0 至 0xFF

请注意，数字表达式允许使用其他值。

数据包特征

可以编写只针对具有特定特征的数据包生成事件的规则。

dsize

`dsize` 关键字测试数据包负载大小。使用此关键字时，可以用大于号和小于号（< 和 >）指定值的范围。可以使用以下语法来指定范围：

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

例如，要表示大于 400 字节的数据包大小，请使用 `>400` 作为 `dtype` 值。要表示小于 500 字节的数据包大小，请使用 `<500`。要指定规则针对介于 400 到 500 字节（包括 400 和 500 字节）的任何数据包触发，请使用 `400<>500`。



注意 `dsiz` 关键字测试未经任何预处理器解码的数据包。

isdataat

`isdataat` 关键字指示规则引擎验证数据是否驻留在负载中的特定位置。

下表列出了可与 `isdataat` 关键字配合使用的参数。

表 59: `isdataat` 参数

参数	类型	说明
偏移	必要	负载中的特定位置。例如，要测试显示在数据包中字节 50 处的数据，需要指定 50 作为偏移量值。! 修饰符对 <code>isdataat</code> 测试的结果进行求反；如果负载中不存在一定的数据量，则会发出警报。 您也可以使用现有 <code>byte_extract</code> 变量或 <code>byte_math</code> 结果指定此参数的值。
相对	可选	使位置相对于上一次成功内容匹配。指定相对位置时请注意，计数器从字节 0 开始计算，因此，应该如下计算相对位置：用从上一次成功内容匹配起向前计算所需的字节数减去 1。例如，要指定数据必须显示在上一次成功内容匹配后的第九个字节处，需要将相对偏移量指定为 8。
原始数据	可选	指定数据在由任何 Firepower 系统预处理器进行解码或应用层规范化之前位于原始数据包负载中。如果上一次内容匹配出现在原始数据包数据中，可以将此参数与 Relative 结合使用。

例如，在查找内容 `foo` 的规则搜索中，如果如下指定 `isdataat` 的值：

- `Offset = !10`
- `Relative = enabled`

那么，如果规则引擎在负载结束前未能在 `foo` 之后检测到 10 字节，系统将会发出警报。

sameip

`sameip` 关键字测试数据包的源 IP 地址和目标 IP 地址是否相同。此关键字没有参数。

fragoffset

`fragoffset` 关键字测试分片数据包的偏移量。由于某些漏洞（例如，WinNuke 拒绝服务攻击）使用手动生成的具有特定偏移量的数据包分片，因此，此关键字很有用。

例如，要测试分片数据包的偏移量是否为 31337 字节，应指定 31337 作为 `fragoffset` 的值。

为 `fragoffset` 关键字指定参数时，可以使用以下运算符。

表 60: fragoffset 关键字参数运算符

Operator	说明
!	not
>	大于
<	小于

请注意，不能将非 (!) 运算符与 < 或 > 结合使用。

CVS

cv_s 关键字测试并发版本系统 (CVS) 流量中是否存在格式不正确的 CVS 条目。攻击者可以使用格式不正确的条目来强制堆溢出，并且在 CVS 服务器上执行恶意代码。此关键字可用于识别针对两种已知 CVS 漏洞的攻击：CVE-2004-0396 (CVS1.11.x 至 1.11.15，以及 CVS1.12.x 至 1.12.7) 和 CVS-2004-0414 (CVS1.12.x 至 1.12.8，以及 CVS1.11.x 至 1.11.16)。cv_s 关键字检查格式正确的记录，如果检测到格式不正确的条目，将会发出警报。

规则应包含 CVS 运行所在的端口。此外，应将任何可能出现流量的端口添加到 TCP 策略的数据流重组端口列表，以便为 CVS 会话维护状态。TCP 端口 2401 (pserver) 和 514 (rsh) 包含在出现数据流重组的客户端端口列表中。但请注意，如果服务器作为 xinetd 服务器 (即，pserver) 运行，它可以在任何 TCP 端口上运行。应将任何非标准端口添加到数据流重组 **Client Ports** 列表中。

相关主题

[byte_extract 关键字](#)，第 41 页

[TCP 数据流预处理选项](#)

活动响应关键字

resp 和 react 关键字提供了两种不同的活动响应发起方法。如果数据包触发规则，包含任意一个关键字的入侵规则将发起单一活动响应。活动响应关键字可以发起活动响应，以在响应触发的 TCP 规则时关闭 TCP 连接，或者在响应触发的 UDP 规则时关闭 UDP 会话。请参阅[入侵丢弃规则中的活动响应](#)。出于一些原因，活动响应并不用于取代防火墙；这些原因包括攻击者可能已选择忽略或绕过活动响应。

内联支持活动响应，包括路由部署或透明部署。例如，在内联部署中对 react 关键字作出响应时，系统可以为连接的两端将 TCP 重置 (RST) 数据包直接插入到流量中 (正常情况下，这样应该会关闭连接)。主动响应不支持或不适合被动部署。

由于活动响应可以回送，因此，系统不允许 TCP 重置发起 TCP 重置；这样可防止活动响应出现无尽的顺序。此外，为了符合标准做法，系统也不允许 ICMP 不可达数据包发起 ICMP 不可达数据包。

可以配置 TCP 数据流预处理器，使它在入侵规则触发了活动响应后检测 TCP 连接的其他流量。如果预处理器检测到其他流量，它会将指定最大数量的其他活动响应发送到连接或会话的两端。请参阅[高级传输/网络预处理器选项](#)中的最大活动响应数和最小响应秒数。

相关主题

[入侵丢弃规则中的活动响应](#)

resp 关键字

可以使用 `resp` 关键字来主动响应 TCP 连接或 UDP 会话，具体取决于在规则报头中指定的是 TCP 还是 UDP 协议。

使用关键字参数可指定数据包方向，以及指定是使用 TCP 重置 (RST) 数据包还是 ICMP 不可达数据包作为活动响应。

可以使用任何 TCP 重置或 ICMP 不可达参数来关闭 TCP 连接。只能使用 ICMP 不可达参数来关闭 UDP 会话。

此外，不同的 TCP 重置参数使得可以将数据包源和/或目标作为活动响应的目标。所有 ICMP 不可达参数都将数据包源作为目标，并且允许指定是使用 ICMP 网络、主机还是端口的不可达数据包，还是同时使用这三者的不可达数据包。

下表列出可与 `resp` 关键字结合使用以明确指定希望 Firepower 系统在规则触发时采取的操作的参数。

表 61: `resp` 参数

参数	说明
<code>reset_source</code>	将 TCP 重置数据包引至发送触发规则的数据包的终端。此外，可以指定 <code>rst_snd</code> （为了获得向后兼容性，仍支持使用此参数）。
<code>reset_dest</code>	将 TCP 重置数据包引至触发规则的数据包的预期目标终端。此外，可以指定 <code>rst_rcv</code> （为了获得向后兼容性，仍支持使用此参数）。
<code>reset_both</code>	将 TCP 重置数据包引至发送终端和接收终端。此外，可以指定 <code>rst_all</code> （为了获得向后兼容性，仍支持使用此参数）。
<code>icmp_net</code>	将 ICMP 网络不可达消息引至发送方。
<code>icmp_host</code>	将 ICMP 主机不可达消息引至发送方。
<code>icmp_port</code>	将 ICMP 端口不可达消息引至发送方。此参数用于终止 UDP 流量。
<code>icmp_all</code>	将以下 ICMP 消息引至发送方： <ul style="list-style-type: none"> • 网络不可达消息 • 主机无法到达 • 端口无法到达

例如，要将规则配置为会在规则触发时重置连接的两端，可使用 `reset_both` 作为 `resp` 关键字的值。

可以使用逗号分隔列表指定多个参数，如下所示：

```
argument,argument,argument
```

react 关键字

如果数据包触发规则，您可以使用 `react` 关键字将默认 HTML 页面发送到 TCP 连接客户端；发送 HTML 页面后，系统将使用 TCP 重置数据包来发起对连接两端的活动响应。`react` 关键字不会对 UDP 流量触发活动响应。

或者，可以指定以下参数：

`msg`

如果数据包触发使用 `msg` 参数的 `react` 规则，HTML 页面将包含规则事件消息。

如果未指定 `msg` 参数，HTML 页面将包含以下消息：

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



注释 由于活动响应可以回送，因此，请确保 HTML 响应页面不会触发 `react` 规则；否则，可能会导致活动响应出现无穷尽的顺序。思科建议在生产环境中激活 `react` 规则之前先对其进行广泛测试。

相关主题

[规则剖析](#)，第 2 页

detection_filter 关键字

可以使用 `detection_filter` 关键字来防止某个规则生成事件，除非在指定时间内有指定数量的数据包触发该规则。这样可防止规则过早生成事件。例如，在几秒钟内登录失败两三次可能是预期行为，但在同一时间内出现大量登录尝试可能表示存在蛮力攻击。

`detection_filter` 关键字需要使用参数来定义系统是否跟踪源或目标 IP 地址、满足检测条件多少次后才会触发事件以及持续计数多长时间。

可使用以下语法延迟事件触发：

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

`track` 参数指定在计算符合规则检测条件的数据包数量时，是否使用数据包的源或目标 IP 地址。可选择下表中所述的参数值来指定系统如何跟踪事件实例。

表 62: `detection_filter` 跟踪参数

参数	说明
<code>by_src</code>	按源 IP 地址计算检测条件。
<code>by_dst</code>	按目标 IP 地址计算检测条件。

`count` 参数指定要使某个规则生成事件，在指定时间内必须有多少数据包为指定 IP 地址触发该规则。

`seconds` 参数指定要使某个规则生成事件，必须在多少秒内有指定数量的数据包触发该规则。

假设某个规则在数据包中搜索内容 `foo`，并将以下参数与 `detection_filter` 关键字配合使用：

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

在此示例中，规则在 20 秒内从来自给定 IP 地址的 10 个数据包中检测到 `foo` 后才会生成事件。如果系统在头 20 秒内仅检测到有 7 个数据包包含 `foo`，将不会生成事件。但是，如果在头 20 秒内 `foo` 出现 40 次，规则将会生成 30 个事件，并在 20 秒后再次进行计数。

比较 `threshold` 和 `detection_filter` 关键字

`detection_filter` 关键字取代已被弃用的 `threshold` 关键字。但是，为了获得向后兼容性，仍支持使用 `threshold` 关键字，其作用与您在入侵策略中设置的阈值相同。

`detection_filter` 关键字是一种检测功能，适合在数据包触发规则前使用。在达到指定的数据包数量之前，规则不会针对触发检测到的数据包生成事件；在内联部署中，如果规则设置为丢弃数据包，在达到指定的数据包数量之前，规则不会丢弃数据包。相反，规则会针对会触发规则且在达到指定数据包数量后出现的数据包生成事件；在内联部署中，如果规则设置为丢弃数据包，规则将会丢弃数据包。

阈值是一种事件通知功能，不会造成检测操作。此功能适合在数据包触发事件后使用。在内联部署中，被设置为丢弃数据包的规则将会丢弃触发其本身的所有数据包，无论规则阈值如何。

请注意，可以在入侵策略中使用使用 `detection_filter` 关键字与入侵事件阈值、入侵事件抑制和基于速率的攻击防御等功能的任意组合。另请注意，如果启用某个导入的本地规则，而该规则将弃用的 `threshold` 关键字与某个入侵策略中的入侵事件阈值功能结合起来使用，策略验证将会失败。

相关主题

[入侵事件阈值](#)

[入侵策略抑制配置](#)

[从规则页面设置动态规则状态](#)

tag 关键字

使用 `tag` 关键字可指示系统记录主机或会话的其他流量。使用 `tag` 关键字指定要捕获的流量的类型和数量时，可使用以下语法：

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

以下三个表介绍了其他可用参数。

有两种标记类型可供选择。下表介绍了这两种标记类型。请注意，如果您在入侵规则中仅配置规则报头选项，会话标记参数类型会使系统像记录来自不同会话的数据包一样来记录来自同一个会话的数据包。要将来自同一会话的数据包分组在一起，请在同一入侵规则中配置一个或多个规则选项（例如，`flag` 关键字或 `content` 关键字）。

表 63: 标记参数

参数	说明
session	记录触发规则的会话中的数据包包。
host	记录来自发送触发规则的数据包的主机的数据包。可以添加方向修饰符，以仅记录来自主机 (src) 或发送到主机 (dst) 的流量。

要指明想要记录的流量数量，请使用以下参数：

表 64: 计数参数

参数	说明
count	您想在规则触发后记录的数据包数量或秒数。 此度量单位用指标参数指定（该参数跟在计数参数后面）。

选择下表中所述的其中一个指标，以指明是要按时间还是流量数量进行记录。



注意 高带宽网络可以每秒查看成千上万个数据包，而且对大量数据包进行标记可能会严重影响性能，因此，请务必根据网络环境调整设置。

表 65: 日志记录指标参数

参数	说明
数据包	在规则触发后记录计数指定的数量的数据包。
seconds	在规则触发后在计数指定的秒数内记录流量。

例如，如果带有以下 tag 关键字值的规则触发：

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

将会记录在接下来的 30 秒内从客户端传输到主机的所有数据包。

flowbits 关键字

可以使用 flowbits 关键字为会话分配状态名称。通过根据之前命名的状态分析会话中的后续数据包，系统可以检测在一个会话中跨越多个数据包的攻击，并发出有关警报。

flowbits 状态名称是用户定义的标签，将被分配给会话特定部分中的数据包。可以根据数据包内容给数据包分配状态名称标签，以帮助将恶意数据包和那些您不想对其发出警报的数据包区分开。最多可以为每个受管设备定义 1024 个状态名称。例如，如果要对您知道仅在成功登录后才会出现的恶意数据包发出警报，可以使用 flowbits 关键字过滤掉构成初始登录尝试的数据包，这样就能够重点关注恶意数据包。要这样做，首先要创建一个会给具有状态为 logged_in 的已建立登录的会话中的

所有数据包分配标签的规则，然后创建另一个包含 `flowbits` 的规则，用以检查具有您在第一个规则中设置的状态的数据包，并且只对这些数据包采取操作。

可选的组名称用于向状态组添加状态名称。一个状态名称可以属于若干个组。未与组关联的状态并不相互排斥，因此，触发和设置未与组关联的状态的规则不会影响其他同时设置的状态。

flowbits 关键字选项

下表介绍了可用于 `flowbits` 关键字的运算符、状态和组的各种组合。请注意，状态名称可以包含字母数字字符、句号 (.)、下划线 (_) 和破折号 (-)。

表 66: `flowbits` 选项

Operator	状态选项	组	说明
<code>set</code>	<code>state_name</code>	可选	为数据包设置某个指定状态。如果定义了某个组，则在该指定的组中设置该状态。
<code>set</code>	<code>state_name&state_name</code>	可选	为数据包设置多个指定状态。如果定义了某个组，则在该指定的组中设置这些状态。
<code>setx</code>	<code>state_name</code>	强制	为数据包在指定组中设置某个指定状态，并取消设置该组中的所有其他状态。
<code>setx</code>	<code>state_name&state_name</code>	强制	为数据包在指定组中设置多个指定状态，并取消设置该组中的所有其他状态。
<code>unset</code>	<code>state_name</code>	没有组	为数据包取消设置某个指定状态。
<code>unset</code>	<code>state_name&state_name</code>	没有组	为数据包取消设置多个指定状态。
<code>unset</code>	<code>all</code>	强制	取消设置指定组中的所有状态。
<code>toggle</code>	<code>state_name</code>	没有组	取消设置某个指定状态（如果已设置），以及设置某个指定状态（如果未设置）。
<code>toggle</code>	<code>state_name&state_name</code>	没有组	取消设置多个指定状态（如果已设置），以及设置多个指定状态（如果未设置）。
<code>toggle</code>	<code>all</code>	强制	取消设置指定组中已设置的所有状态，以及设置指定组中未设置的所有状态。
<code>isset</code>	<code>state_name</code>	没有组	确定是否已在数据包中设置了某个指定状态。
<code>isset</code>	<code>state_name&state_name</code>	没有组	确定是否已在数据包中设置了多个指定状态。
<code>isset</code>	<code>state_name state_name</code>	没有组	确定是否已在数据包中设置了任何指定状态。

Operator	状态选项	组	说明
isset	any	强制	确定是否已在指定组中设置了任何状态。
isset	all	强制	确定是否已在指定组中设置了所有状态。
isnotset	state_name	没有组	确定是否未在数据包中设置某个指定状态。
isnotset	state_name&state_name	没有组	确定是否未在数据包中设置多个指定状态。
isnotset	state_name state_name	没有组	确定是否未在数据包中设置任何指定状态。
isnotset	any	强制	确定是否未在数据包中设置任何状态。
isnotset	all	强制	确定是否未在数据包中设置所有状态。
reset	(无状态)	可选	为所有数据包取消设置所有状态。取消设置某个组中的所有状态（如果已指定该组）。
noalert	(无状态)	没有组	可将此运算符与任何其他运算符结合使用，以抑制事件生成。

flowbits 关键字使用准则

使用 flowbits 关键字时，请注意：

- 使用 setx 运算符时，指定的状态只能属于指定的组，而不能属于任何其他组。
- 可以多次定义 setx 运算符，每次用一个实例指定不同的状态和同一个组。
- 如果使用 setx 运算符并指定了某个组，则不能对该指定的组使用 set、toggle 或 unset 运算符。
- isset 和 isnotset 运算符会对指定状态进行评定，无论该状态是否在组中。
- 在保存入侵策略、重新应用入侵策略以及应用访问控制策略期间（不管访问控制策略是引用一个入侵策略还是多个入侵策略），如果启用包含未指定组的 isset 或 isnotset 运算符的规则，并且至少有一个影响对应状态名称 flowbits 分配的规则（set、setx、unset、toggle）未启用，那么系统将启用影响对应状态名称 flowbits 分配的所有规则。
- 在保存入侵策略、重新应用入侵策略以及应用访问控制策略期间（不管访问控制策略是引用一个入侵策略还是多个入侵策略），如果启用包含已指定组的 isset 或 isnotset 运算符的规则，系统还将启用影响 flowbits 分配（set、setx、unset、toggle）和定义对应组名称的所有规则。

flowbits 关键字示例

本节提供三个使用 flowbits 关键字的示例。

flowbits 关键字示例：使用 state_name 的配置

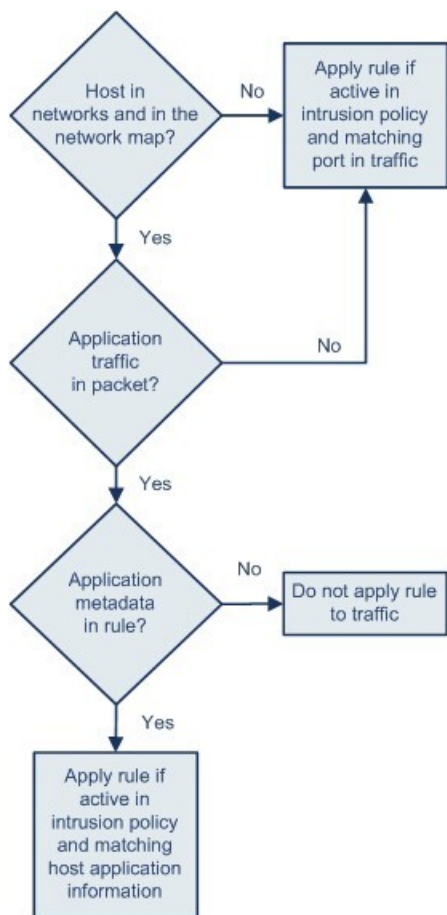
这是使用 state_name 的 flowbits 配置的示例。

以 CVE ID 2000-0284 中所述的 IMAP 漏洞为例。该漏洞存在于 IMAP 的实现中，尤其是在 LIST、LSUB、RENAME、FIND 和 COPY 命令中。但是，要想利用该漏洞，攻击者必须登录到 IMAP 服务器。由于来自 IMAP 服务器的登录确认及紧随着而来的漏洞必定存在于不同的数据包中，因此，难以构建非基于流量的规则来捕获该漏洞。通过使用 flowbits 关键字，可以构建一系列规则来跟踪用户是否登录 IMAP 服务器；如果已登录，则在检测到其中一项攻击时生成事件。如果用户未登录，则攻击不能利用该漏洞，且不会生成事件。

以下两个规则分片说明了此示例。第一个规则分片查找来自 IMAP 服务器的 IMAP 登录确认：

```
alert tcp any 143 -> any any (msg:"IMAP login"; content:"OK  
LOGIN"; flowbits:set,logged_in; flowbits:noalert;)
```

下图展示前述规则分片中 flowbits 关键字的影响：



371863

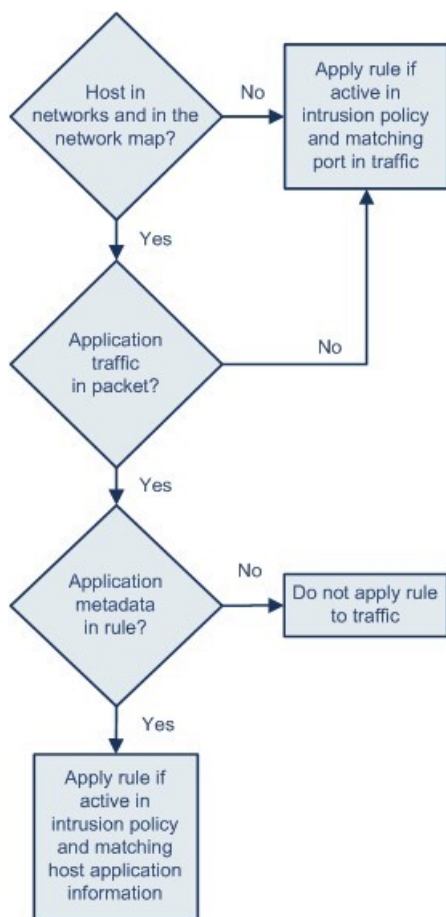
请注意，flowbits:set 设置 logged_in 状态，flowbits:noalert 则抑制警报，因为 IMAP 服务器上可能会出现许多无恶意的登录会话。

flowbits 关键字示例：导致误报事件的配置

以下规则分片查找 LIST 字符串，但不生成事件，除非由于会话中某个之前的数据包而设置了 logged_in 状态：

```
alert tcp any any -> any 143 (msg:"IMAP LIST";
content:"LIST"; flowbits:isset,logged_in;)
```

下图展示前述规则分片中 flowbits 关键字的影响：



在这种情况下，如果之前的数据包已促使包含第一个分片的规则触发，则包含第二个分片的规则将会触发并生成事件。

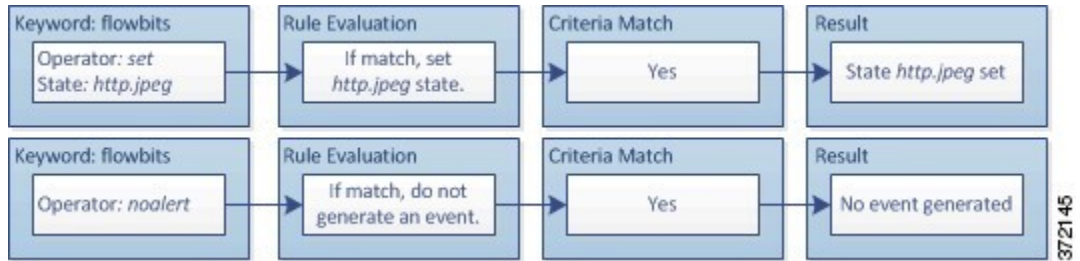
flowbits 关键字示例：导致误报事件的配置

在一个组中包含在不同规则中设置的不同状态名称可防止误报事件；如果后续数据包中的内容与状态不再有效的规则相匹配，就会出现误报事件。以下示例说明不在一个组中包含多个状态名称如何会导致误报。

假设以下三个规则分片在一个会话中按所示的顺序触发：

```
(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;)
```

下图展示前述规则分片中 `flowbits` 关键字的影响：

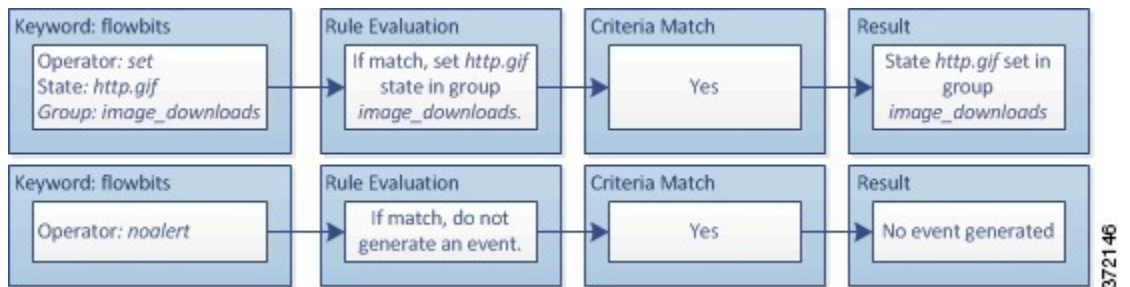


第一个规则分片中的 `content` 和 `pcrc` 关键字与 JPEG 文件下载相匹配，`flowbits:set,http.jpeg` 设置 `http.jpeg` flowbits 状态，`flowbits:noalert` 阻止规则生成事件。系统不会生成事件，因为该规则的目的是检测文件下载并设置 `flowbits` 状态；为此，一个或多个伴随规则可以测试状态名称和恶意内容，如果检测到恶意内容，则会生成事件。

以下规则分片检测在上述 JPEG 文件下载之后发生的 GIF 文件下载：

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

下图展示前述规则分片中 `flowbits` 关键字的影响：



第二个规则中的 `content` 和 `pcrc` 关键字与 GIF 文件下载相匹配，`flowbits:set,http.jpg` 设置 `http.jpg` flowbit 状态，`flowbits:noalert` 阻止规则生成事件。请注意，仍会设置由第一个规则分片设置的 `http.jpeg` 状态，即使不再需要使用它；这是因为如果检测到后续 GIF 下载，JPEG 下载必须已终止。

第三个规则分片伴随第一个规则分片出现：

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

下图展示前述规则分片中 `flowbits` 关键字的影响：



在第三个规则分片中，`flowbits:isset,http.jpeg` 确定是否已设置现在不相关的 `http.jpeg` 状态，`content` 和 `pcrc` 则匹配在 JPEG 文件中是恶意的但在 GIF 文件中并非恶意的内容。第三个规则分片会针对 JPEG 文件中不存在漏洞生成误报事件。

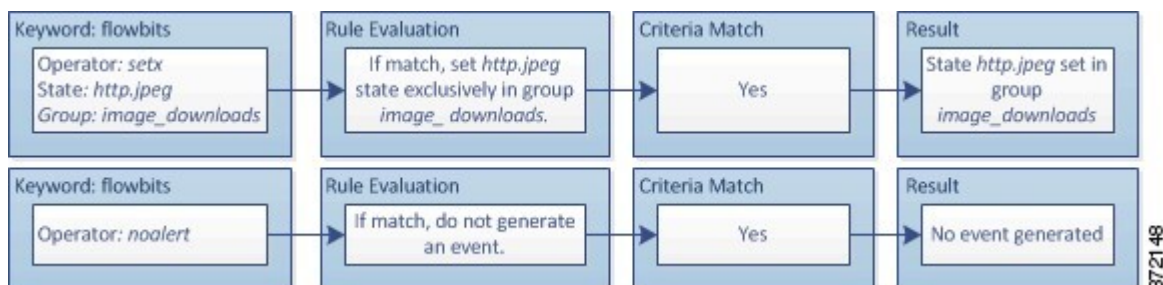
flowbits 关键字示例：防止误报事件的配置

以下示例说明在一个组中包含多个状态名称并使用 `setx` 运算符如何能防止误报。

以下规则分片与上一个规则分片示例大致相同，不同之处是，以下示例的前两个规则将两个不同的状态名称包含在同一个状态组中。

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

下图展示前述规则分片中 `flowbits` 关键字的影响：

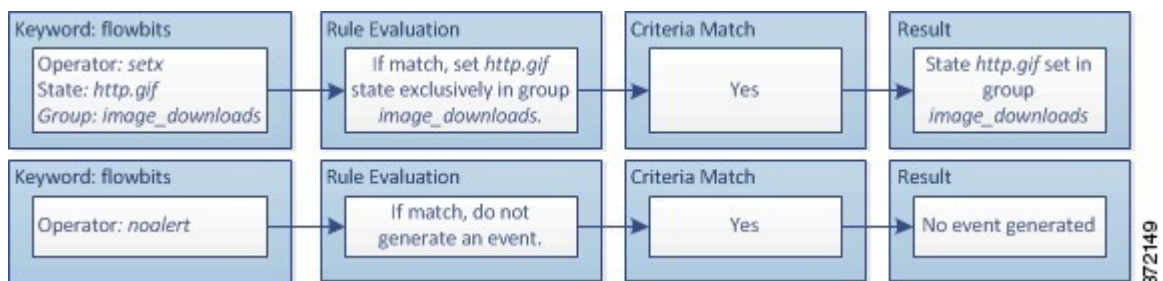


如果第一个规则分片检测到 JPEG 文件下载，`flowbits:setx,http.jpeg,image_downloads` 关键字会将 `flowbits` 状态设置为 `http.jpeg`，并将该状态包含在 `image_downloads` 组中。

然后，下一个规则会后续 GIF 文件下载：

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

下图展示前述规则分片中 `flowbits` 关键字的影响：

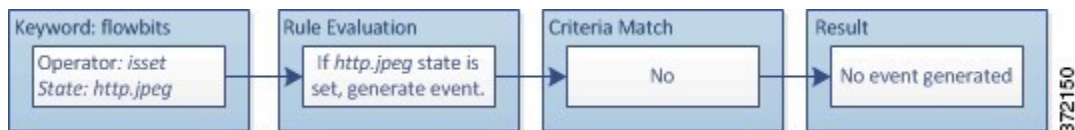


如果第二个规则分片与 GIF 下载相匹配，`flowbits:setx,http.jpg,image_downloads` 关键字将会设置 `http.jpg` `flowbits` 状态，并取消设置组中的另一个状态 `http.jpeg`。

第三个规则分片不会导致误报：

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

下图展示前述规则分片中 flowbits 关键字的影响：



由于 flowbits:isset,http.jpeg 为假，因此，规则引擎会停止处理规则，且不会生成事件，从而避免误报（即使 GIF 文件中的内容与 JPEG 文件的漏洞内容相匹配）。

http_encode 关键字

您可以使用 http_encode 关键字在未经规范化的 HTTP 请求或响应中生成关于编码类型的事件 - 是在 HTTP URI 中，在 HTTP 报头的非 cookie 数据中，在 HTTP 请求报头的 cookie 中，或者在 HTTP 响应的 set-cookie 数据中。

必须配置 HTTP 检查预处理器以检查 HTTP 响应和 HTTP cookie，从而使用 http_encode 关键字返回规则的匹配项。

此外，您必须在 HTTP 检查预处理器配置中为每个特定编码类型启用解码和警报选项，以使入侵规则中的 http_encode 关键字可以触发关于该编码类型的事件。

下表介绍了此选项可在 HTTP URI、报头、cookie 和 set-cookie 中为其生成事件的编码类型。

表 67: HTTP_encode 编码类型

编码类型	说明
utf8	如果此编码类型已启用，并可供 HTTP 检查预处理器进行解码，将会在指定位置检测 UTF-8 编码。
double_encode	如果此编码类型已启用，并可供 HTTP 检查预处理器进行解码，将会在指定位置检测双编码。
non_ascii	当检测到非 ASCII 字符但未启用检测到的编码类型时，在指定位置检测非 ASCII 字符。
uencode	如果此编码类型已启用，并可供 HTTP 检查预处理器进行解码，将会在指定位置检测 Microsoft %u 编码。
bare_byte	如果此编码类型已启用，并可供 HTTP 检查预处理器进行解码，将会在指定位置检测裸字节编码。

相关主题

[HTTP 检查预处理器](#)

[服务器级别 HTTP 规范化选项](#)

http_encode 关键字语法

编码位置

指定是要在 HTTP URI、报头还是 cookie（包括 set-cookie）中搜索指定的编码类型。

编码类型

使用以下格式之一指定一个或多个编码类型：

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

其中，encode_type 是以下其中一项：

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

请注意，不能同时使用求反 (!) 和 OR (|) 运算符。

http_encode 关键字示例：使用两个 http_encode 关键字搜索两种编码

以下示例使用同一规则中的两个 http_encode 关键字在 HTTP URI 中搜索 UTF-8 AND Microsoft IIS %u 编码：

第一个，http_encode 关键字：

- **Encoding Location:** HTTP URI
- **Encoding Type:** utf8

然后，另一个 http_encode 关键字：

- **编码位置 (Encoding Location):** HTTP URI
- **编码类型 (Encoding Type):** uencode

概述：file_type 和 file_group 关键字

file_type 和 file_group 关键字允许根据文件的类型和版本检测通过 FTP、HTTP、SMTP、IMAP、POP3 和 NetBIOS-ssn (SMB) 传输的文件。请勿在单个入侵规则中使用多个 file_type 或 file_group 关键字。



提示 更新漏洞数据库 (VDB) 可以使入侵规则编辑器获得最新的文件类型、版本和组。



注释 系统不会自动启用预处理器以适应 `file_type` 和 `file_group` 关键字。

如果希望流量的生成事件并在内联部署中丢弃攻击性数据包与您的 `file_type` 或 `file_group` 关键字相匹配，则必须启用特定的预处理器。

表 68: `file_type` 和 `file_group` 入侵事件生成

协议	所需的预处理器或预处理器选项
FTP	FTP/Telnet 预处理器和规范化 TCP 负载内联规范化预处理器选项
HTTP	在 HTTP 流量中生成入侵事件的 HTTP 检查预处理器
SMTP	在 HTTP 流量中生成入侵事件的 SMTP 预处理器
IMAP	IMAP 预处理器
POP3	POP 预处理器
Netbios-ssn (SMB)	DCE/RPC 预处理器和 SMB 文件检查 (SMB File Inspection) DCE/RPC 预处理器选项

相关主题

[FTP/Telnet 解码器](#)
[内联规范化预处理器](#)
[HTTP 检查预处理器](#)
[SMTP 预处理器](#)
[IMAP 预处理器](#)
[POP 预处理器](#)
[DCE/RPC 预处理器](#)

file_type 和 file_group 关键字

file_type

使用 `file_type` 关键字可指定在流量中检测到的文件的类型和版本。文件类型参数（例如 **JPEG** 和 **PDF**）用于识别要在流量中查找的文件格式。



注释 请勿在同一入侵规则中将 `file_type` 关键字与另一个 `file_type` 或 `file_group` 关键字配合使用。

系统默认选择 **Any Version**，但某些文件类型允许选择版本选项（例如 PDF 版本 **1.7**）来确定要在流量中查找的特定文件类型版本。

file_group

使用 `file_group` 关键字可选择思科定义的、包含在流量中找到的类似文件类型（例如多媒体或音频）的组。文件组还包含思科为组中的每种文件类型定义的版本。



注释 请勿在同一入侵规则中将 `file_group` 关键字与另一个 `file_group` 或 `file_type` 关键字配合使用。

file_data 关键字

`file_data` 关键字提供一个指针，该指针作为可用于其他关键字（例如 `content`、`byte_jump`、`byte_test` 和 `pcre`）的位置参数参考。检测到的流量确定 `file_data` 关键字指向的数据类型。您可以使用 `file_data` 关键字来指向以下负载类型的开头：

- HTTP 响应正文

要检查 HTTP 响应数据包，必须启用 HTTP 检查预处理器，还必须将该预处理器配置为会检查 HTTP 响应。如果 HTTP 检查预处理器检测到 HTTP 响应正文数据，`file_data` 关键字将会进行匹配。

- 未压缩的 gzip 文件数据

要检查 HTTP 响应正文中未压缩的 gzip 文件，必须启用 HTTP 检查预处理器，还必须将该预处理器配置为会检查 HTTP 响应以及会解压缩 HTTP 响应正文中的 gzip 压缩文件。有关详细信息，请参阅检测 HTTP 响应和检测压缩数据服务器级别 HTTP 规范化选项。如果 HTTP 检查预处理器在 HTTP 响应正文中检测到未压缩的 gzip 数据，`file_data` 关键字将会进行匹配。

- 规范化的 JavaScript

要检查规范化的 JavaScript 数据，必须启用 HTTP 检查预处理器，还必须将该预处理器配置为检查 HTTP 响应。如果 HTTP 检查预处理器在响应主体数据中检测到 JavaScript，`file_data` 关键字将会进行匹配。

- SMTP 负载

要检查 SMTP 负载，必须启用 SMTP 预处理器。如果 SMTP 预处理器检测到 SMTP 数据，`file_data` 关键字将会进行匹配。

- SMTP、POP 或 IMAP 流量中的编码电子邮件附件

要检查 SMTP、POP 或 IMAP 流量中的邮件附件，必须分别启用 SMTP、POP 或 IMAP 预处理器或者启用它们的任意组合。然后，必须确保将已启用的每个预处理器配置为会对您想要解码的每种附件编码类型进行解码。可以为每个预处理器配置的附件解码选项是：**Base64 解码深度 (Base64 Decoding Depth)**、**7 位/8 位/二进制解码深度 (7-Bit/8-Bit/Binary Decoding Depth)**、**Quoted-Printable 解码深度 (Quoted-Printable Decoding Depth)** 和 **UUnix-to-Unix 解码深度 (Unix-to-Unix Decoding Depth)**。

可以在一个规则中使用多个 `file_data` 关键字。

相关主题

- [HTTP 检查预处理器](#)
- [服务器级别 HTTP 规范化选项](#)
- [SMTP 预处理器](#)
- [IMAP 预处理器](#)

pkt_data 关键字

pkt_data 关键字提供一个指针，该指针作为可用于其他关键字（例如 content、byte_jump、byte_test 和 pcre）的位置参数参考。

如果检测到规范化的 FTP、telnet 或 SMTP 流量，pkt_data 关键字将指向规范化数据包负载的开头。如果检测到其他流量，pkt_data 关键字将指向原始 TCP 或 UDP 负载的开头。

必须启用以下规范化选项，系统才会对相应流量进行规范化以供入侵规则进行检测：

- 启用 FTP 和 Telnet 预处理器的检测 **FTP 命令内的 Telnet 转义代码 (Detect Telnet Escape Codes within FTP Commands)** 选项可规范化 FTP 流量以进行检查。
- 启用 FTP 和 Telnet 预处理器的规范化 (**Normalize**) Telnet 选项可规范化 Telnet 流量以进行检查。
- 启用 SMTP 预处理器的规范化 (**Normalize**) 选项可规范化 SMTP 流量以进行检查。

可以在一个规则中使用多个 pkt_data 关键字。

相关主题

- [客户端级别 FTP 选项](#)
- [Telnet 选项](#)
- [SMTP 预处理器选项](#)

base64_decode 和 base64_data 关键字

可以结合使用 base64_decode 和 base64_data 关键字，以指示规则引擎将指定数据作为 Base64 数据进行解码和检查。这可能很有用，例如，对于检查 Base64 编码 HTTP 身份验证请求报头，以及对于检查 HTTP PUT 和 POST 请求中的 Base64 编码数据。

这两个关键字对于编码和检查 HTTP 请求中的 Base64 数据尤其有用。但是，也可以将这两个关键字与像 HTTP 一样使用空格和制表符的任何协议（例如 SMTP）结合使用，以将长的报头行展开为跨越多行。如果协议中不存在这样的行展开（即为“折叠”），检查将在后面不跟有空格或制表符的任何回车符或换行符处结束。

base64_decode

base64_decode 关键字指示规则引擎将数据包数据解码为 Base64 数据。使用可选参数可指定要解码的字节数量以及在数据中的哪个位置开始解码。

可以在一个规则中使用 base64_decode 关键字一次；此关键字必须位于至少一个 base64_data 关键字实例前面。

解码 Base64 数据之前，规则引擎会将跨越多行的已折叠的长报头展开。当规则引擎遇到以下任何情况时，解码将会结束：

- 报头行结尾
- 要解码的指定字节数
- 数据包结尾

下表介绍了可与 `base64_decode` 关键字配合使用的参数。

表 69: 可选 `base64_decode` 参数

参数	说明
Bytes	指定要解码的字节数。如果未指定，解码将持续到报头行结尾或数据包负载结尾（以先到者为准）。可以指定非零的正值。
Offset	确定相对于数据包负载开头的偏移量，如果还指定了 Relative ，则确定相对于当前检查位置的偏移量。可以指定非零的正值。
Relative	指定相对于当前检查位置的检查。

base64_data

`base64_data` 关键字提供用于检查使用 `base64_decode` 关键字进行解码的 Base64 数据的参考。

`base64_data` 关键字将检查设置在解码的 Base64 数据开头开始。或者，可以随后使用可用于其他关键字的位置参数（例如 `content` 或 `byte_test`）进一步指定要检查的位置。

使用 `base64_decode` 关键字之后，必须至少使用一次 `base64_data` 关键字；或者，可以多次使用 `base64_data` 以返回到解码的 Base64 数据的开头。

检查 Base64 数据时，请注意：

- 不能使用快速模式匹配程序。
- 如果在规则中使用干预性 HTTP 内容参数中断 Base64 检查，则必须在该规则中插入另一个 `base64_data` 关键字，然后再进一步检查 Base64 数据。

相关主题

[概述：HTTP content 和 protected_content 关键字参数](#)，第 29 页

[content 关键字快速模式匹配程序参数](#)，第 33 页

当地语言翻译版本说明

思科可能会在某些地方提供本内容的当地语言翻译版本。请注意，翻译版本仅供参考，如有任何不一致之处，以本内容的英文版本为准。