Processar Single Stream Large Session (Fluxo de Elefante) por Firepower Services

Contents

Introduction

Este documento descreve por que um único fluxo não pode consumir todo o throughput classificado de um dispositivo Cisco Firepower.

Informações de Apoio

O resultado de qualquer site de teste de velocidade de largura de banda ou a saída de qualquer ferramenta de medição de largura de banda (por exemplo, iperf) pode não exibir a classificação de throughput anunciada dos dispositivos Cisco Firepower. Da mesma forma, a transferência de um arquivo muito grande sobre qualquer protocolo de transporte não demonstra a classificação de throughput anunciada de um dispositivo Firepower. Isso ocorre porque o serviço Firepower não usa um único fluxo de rede para determinar seu throughput máximo.

Tráfego de processo por Snort

A tecnologia de detecção subjacente do serviço Firepower é Snort. A implementação do Snort no dispositivo Cisco Firepower é um processo de thread único para processar o tráfego. Um dispositivo é classificado para uma classificação específica com base no throughput total de todos os fluxos que passam pelo dispositivo. Espera-se que os dispositivos sejam implantados em uma rede corporativa, geralmente próxima à borda da fronteira e funcione com milhares de conexões.

Os Firepower Services usam o balanceamento de carga de tráfego para vários processos diferentes de Snort com um processo Snort executado em cada CPU do dispositivo. Idealmente, a carga do sistema equilibra o tráfego uniformemente em todos os processos do Snort. O Snort

precisa ser capaz de fornecer análise contextual adequada para inspeção de firewall de próxima geração (NGFW), sistema de prevenção de intrusão (IPS) e proteção avançada contra malware (AMP). Para garantir que o Snort seja mais eficiente, todo o tráfego de um único fluxo é balanceado de carga para uma instância de snort. Se todo o tráfego de um único fluxo não fosse balanceado para uma única instância de snort, o sistema poderia ser evadido e o tráfego seria derramado de tal forma que uma regra Snort poderia ser menos provável de corresponder ou partes de um arquivo não seriam contíguas para a inspeção de AMP. Portanto, o algoritmo de balanceamento de carga é baseado nas informações de conexão que podem identificar exclusivamente uma determinada conexão.

Algoritmo de 2 tuplas no ASA com Firepower Services e NGIPS Virtual

No Adaptive Security Appliance (ASA) com a plataforma Firepower Service e o virtual Next Generation Intrusion Prevention System (NGIPS), o tráfego é balanceado para Snort com o uso de um algoritmo de 2 tuplas. Os datapopoints para esse algoritmo são:

- IP origem
- IP de Destino

Algoritmo de 3 tuplas na versão de software 5.3 ou inferior nos dispositivos Firepower e FTD

Em todas as versões anteriores (5.3 ou inferior), o tráfego é balanceado para Snort que usa um algoritmo de 3 tuplas. Os datapopoints para esse algoritmo são:

- IP origem
- IP de Destino
- Protocolo IP

Qualquer tráfego com a mesma origem, destino e protocolo IP tem balanceamento de carga para a mesma instância do Snort.

Algoritmo de 5 tuplas na versão de software 5.4, 6.0 e superior nos dispositivos Firepower e FTD

Na versão 5.4, 6.0 ou superior, o tráfego é balanceado para Snort com um algoritmo de 5 tuplas. Os datapopoints considerados são:

- IP origem
- Porta de origem
- IP de Destino
- Porta de Destino
- Protocolo IP

A finalidade de adicionar portas ao algoritmo é equilibrar o tráfego de forma mais uniforme quando há pares de origem e destino específicos que representam grandes partes do tráfego. Além das portas, as portas de origem efêmeras de alta ordem devem ser diferentes por fluxo e devem adicionar entropia adicional de forma mais uniforme que equilibre o tráfego em diferentes instâncias de snort.

Rendimento total

O throughput total de um dispositivo é medido com base no throughput agregado de todas as instâncias de snort que funcionam com todo o seu potencial. As práticas padrão do setor para medir o throughput são para várias conexões HTTP com vários tamanhos de objeto. Por exemplo, a metodologia de teste do NGFW do NSS mede o throughput total do dispositivo com objetos 44k, 21k, 10k, 4.4k e 1.7k. Eles traduzem para um intervalo de tamanhos médios de pacotes de cerca de 1k e bytes para 128 bytes devido aos outros pacotes envolvidos na conexão HTTP.

Você pode estimar a classificação de desempenho de uma instância Snort individual. Obtenha a taxa de transferência do dispositivo e divida-a pelo número de instâncias Snort executadas. Por exemplo, se um dispositivo for classificado em 10 Gbps para IPS com um tamanho médio de pacote de 1 k bytes e esse dispositivo tiver 20 instâncias de Snort, o throughput máximo aproximado para uma única instância seria de 500 Mbps por Snort. Diferentes tipos de tráfego, protocolos de rede, tamanhos dos pacotes, juntamente com diferenças na política de segurança geral, podem afetar o throughput observado do dispositivo.

Resultado do teste da ferramenta de terceiros

Quando você testa com qualquer site de teste de velocidade, ou qualquer ferramenta de medição de largura de banda, como iperf, um grande fluxo de TCP de fluxo único é gerado. Esse tipo de grande fluxo TCP é chamado de Fluxo de Elefante. Um fluxo de elefante é uma única sessão, uma conexão de rede de execução relativamente longa que consome uma grande ou desproporcional quantidade de largura de banda. Esse tipo de fluxo é atribuído a uma instância Snort, portanto, o resultado do teste exibe o throughput de uma única instância de snort, não a classificação de throughput agregado do dispositivo.

Sintomas observados

CPU alta observada

Outro efeito visível dos Fluxos de Elefante pode ser a cpu alta de instância de snort. Isso pode ser visto por meio do "show asp inspect-dp snort" ou da ferramenta shell "top".

```
> show asp inspect-dp snort

SNORT Inspect Instance Status Info

Id  Pid   Cpu-Usage  Conns   Segs/Pkts  Status tot (usr | sys)
--  ----  ---------  -----   --------   ----------

0  48500  30% ( 28%|  1%)  12.4 K      0        READY
1  48474  24% ( 22%|  1%)  12.4 K      0        READY
2  48475  34% ( 33%|  1%)  12.5 K      1        READY
3  48476  29% ( 28%|  0%)  12.4 K      0        READY
4  48477  32% ( 30%|  1%)  12.5 K      0        READY
5  48478  31% ( 29%|  1%)  12.3 K      0        READY
6  48479  29% ( 27%|  1%)  12.3 K      0        READY
7  48480  23% ( 23%|  0%)  12.2 K      0        READY
8  48501  27% ( 26%|  0%)  12.6 K      1        READY
9  48497  28% ( 27%|  0%)  12.6 K      0        READY
10 48482  28% ( 27%|  1%)  12.3 K      0        READY
11 48481  31% ( 30%|  1%)  12.5 K      0        READY
12 48483  36% ( 36%|  1%)  12.6 K      0        READY
13 48484  30% ( 29%|  1%)  12.4 K      0        READY
```

```
14 48485  33% ( 31%|  1%)  12.6 K      0        READY
15 48486  38% ( 37%|  0%)  12.4 K      0        READY
16 48487  31% ( 30%|  1%)  12.4 K      1        READY
17 48488  37% ( 35%|  1%)  12.7 K      0        READY
18 48489  34% ( 33%|  1%)  12.6 K      0        READY
19 48490  27% ( 26%|  1%)  12.7 K      0        READY
20 48491  24% ( 23%|  0%)  12.6 K      0        READY
21 48492  24% ( 23%|  0%)  12.6 K      0        READY
22 48493  28% ( 27%|  1%)  12.4 K      1        READY
23 48494  27% ( 27%|  0%)  12.2 K      0        READY
24 48495  29% ( 28%|  0%)  12.5 K      0        READY
25 48496  30% ( 30%|  0%)  12.4 K      0        READY
26 48498  29% ( 27%|  1%)  12.6 K      0        READY
27 48517  24% ( 23%|  1%)  12.6 K      0        READY
28 48499  22% ( 21%|  0%)  12.3 K      1        READY
29 48518  31% ( 29%|  1%)  12.4 K      2        READY
30 48502  33% ( 32%|  0%)  12.5 K      0        READY

31 48514 80% ( 80%| 0%) 12.7 K 0 READY <<< CPU 31 is much busier than the rest, and will stay
busy for while with elephant flow.

32 48503  49% ( 48%|  0%)  12.4 K      0        READY
33 48507  27% ( 25%|  1%)  12.5 K      0        READY
34 48513  27% ( 25%|  1%)  12.5 K      0        READY
35 48508  32% ( 31%|  1%)  12.4 K      0        READY
36 48512  31% ( 29%|  1%)  12.4 K      0        READY



$ top

PID USER      PR  NI  VIRT  RES  SHR S %CPU %MEM    TIME+  COMMAND

69470 root       1 -19 9088m 1.0g  96m R   80  0.4 135:33.51 snort     <<<< one snort very busy,
rest below 50%

69468 root       1 -19 9089m 1.0g  99m R   49  0.4 116:08.69 snort
69467 root       1 -19 9078m 1.0g  97m S   47  0.4 118:30.02 snort
69492 root       1 -19 9118m 1.1g  97m R   47  0.4 116:40.15 snort
69469 root       1 -19 9083m 1.0g  96m S   39  0.4 117:13.27 snort
69459 root       1 -19 9228m 1.2g  97m R   37  0.5 107:13.00 snort
69473 root       1 -19 9087m 1.0g  96m R   37  0.4 108:48.32 snort
69475 root       1 -19 9076m 1.0g  96m R   37  0.4 109:01.31 snort
69488 root       1 -19 9089m 1.0g  97m R   37  0.4 105:41.73 snort
69474 root       1 -19 9123m 1.1g  96m S   35  0.4 107:29.65 snort
69462 root       1 -19 9065m 1.0g  99m R   34  0.4 103:09.42 snort
69484 root       1 -19 9050m 1.0g  96m S   34  0.4 104:15.79 snort
69457 root       1 -19 9067m 1.0g  96m S   32  0.4 104:12.92 snort
69460 root       1 -19 9085m 1.0g  97m R   32  0.4 104:16.34 snort
```

Com o algoritmo de 5 tuplas descrito acima, um fluxo de vida longa sempre será enviado para a mesma instância de snort. Se houver amplas políticas AVC, IPS, File, etc ativas em snort, a CPU pode ser vista como alta (>80%) em uma instância de snort por algum tempo. Adicionar política SSL aumentará ainda mais o uso da CPU para a natureza computacionalmente cara da descriptografia SSL.

A alta CPU em algumas das muitas CPUs de snort não é causa de alarme crítico. É o comportamento do sistema NGFW ao executar inspeção profunda de pacotes em um fluxo, e isso pode naturalmente usar grandes partes de uma CPU. Como diretriz geral, o NGFW não está em uma situação crítica de privação da CPU até que a maioria das CPUs de snort esteja acima de 95% e permaneça acima de 95% e quedas de pacotes estejam sendo vistas.

As reparações abaixo ajudarão na alta situação da CPU devido aos fluxos de elefante.

Correções

Intelligent Application Bypass (IAB)

A versão 6.0 do software introduz um novo recurso chamado IAB. Quando um Firepower Appliance atinge um limite de desempenho predefinido, o recurso IAB procura fluxos que atendam a critérios específicos para contornar de forma inteligente a pressão nos mecanismos de detecção.

> **Tip**: Mais informações sobre a configuração do IAB podem ser encontradas aqui.

Identifique e confie em grandes fluxos

Fluxos grandes estão frequentemente relacionados ao tráfego de baixo valor de inspeção de alta utilização, por exemplo, backups, replicação de bancos de dados, etc. Muitos desses aplicativos não podem ser beneficiados com a inspeção. Para evitar problemas com fluxos grandes, você pode identificar os fluxos grandes e criar regras de confiança de controle de acesso para eles. Essas regras são capazes de identificar com exclusividade grandes fluxos, permitir que esses fluxos passem sem inspeção e não sejam limitados pelo comportamento de instância de um único snort.

> **Note**: Para identificar grandes fluxos para regras de confiança, entre em contato com o TAC do Cisco Firepower.

Informações Relacionadas

- Controle de acesso usando desvio de aplicativo inteligente
- Suporte Técnico e Documentação - Cisco Systems