# 基于统计的攻击流量类型识别-调研

## 涉及公式

标准差公式：$\sigma = \sqrt{\frac{\sum_{i=1}^{n}(x-\overline{x})^2}{n}}$

CUSUM公式：$S_{H_{n+1}} = max(0, S_{H_n} + Z_n - w); Z_n = \frac{X_n - \overline{x}}{\sigma_X}$

其中 $\sigma x$为标准差，$\overline{x}$为算术平均数，$S_{H_n}$为当前时刻的累积和。

## SynFlood 分析

SynFlood 是一种阻断式服务攻击，攻击者发送大量伪造的SYN包，使服务器打开并维护大量半连接，从而造成无法为正常请求服务的一种攻击类型。

### Syn数据包统计分析

在正常情况下，流入的SYN包与流出的FIN包成1:1的关系，通过对流量最大100个IP地址的数据包类型的统计信息进行分析，发现 `InSYN:OutFIN` 包的比例主要出现 `2:1` 和 `1:1` 两种情况， `标准差` 普遍在0.5以下（不统计每秒小于500个SYN包的情况）

怀疑`2:1`的情况为错误统计了清洗回注的流量。

### 使用CUSUM算法对SynFlood进行识别

**确认持续输入的变量:** `TcpInSyns/TcpOutFins`

```
// InSynsDivOutFins Check SynFlood
func (i *Item) InSynsDivOutFins() float64 {
  if i.Metrics.TcpInSyns.Max < 500 { // 小于指定数量的syn包，不计算
    return float64(0)
  }
  if i.Metrics.TcpOutFins.Max == 0 {
    return float64(i.Metrics.TcpInSyns.Max)
  }
  return float64(i.Metrics.TcpInSyns.Max) / float64(i.Metrics.TcpOutFins.Max)
}
```

**确认CUSUM模型参数:**

```
// 算术平均数:  2.2
// 标准差:    0.5
// weight:   4
p.synFlood = CUSUM.NewCUSUM(2.2, 0.5, 4)
```

**确认CUSUM模型警戒线：**

当累积和达到一定的数值时，则触发告警；此值越小则检查越灵敏，误报的可能性增大；越大则检查越迟钝，漏报的可能性增大。这里简单使用100进行检查。

```
p.synFlood.Add(it.InSynsDivOutFins())
if rh, _ := p.synFlood.Result(); rh > 100 {
  p.synFlood.Reset()
}
```

**结果确认：**

==> 发现IP地址为 `106.39.168.3` 在timestamp为 `1633898649` 时发生SynFlood攻击，持续到 `1633898659` 止。

```
MEAN:  ./data/106.39.168.3.txt  0.426843  42.899501
-> CUSUM SynFlood:  ./data/106.39.168.3.txt  1633898649  2037.200000   5114   5
-> CUSUM SynFlood:  ./data/106.39.168.3.txt  1633898650  9893.600000   4951   1
-> CUSUM SynFlood:  ./data/106.39.168.3.txt  1633898651  10701.600000  5355   0
-> CUSUM SynFlood:  ./data/106.39.168.3.txt  1633898652  5217.600000   5226   2
-> CUSUM SynFlood:  ./data/106.39.168.3.txt  1633898653  5168.600000   5177   2
-> CUSUM SynFlood:  ./data/106.39.168.3.txt  1633898654  10577.600000  5293   0
-> CUSUM SynFlood:  ./data/106.39.168.3.txt  1633898655  2058.400000   5167   5
-> CUSUM SynFlood:  ./data/106.39.168.3.txt  1633898656  5152.600000   5161   2
-> CUSUM SynFlood:  ./data/106.39.168.3.txt  1633898657  10461.600000  5235   0
-> CUSUM SynFlood:  ./data/106.39.168.3.txt  1633898658  10405.600000  5207   0
-> CUSUM SynFlood:  ./data/106.39.168.3.txt  1633898659  1991.600000   1000   1
```

与原始数据进行比对，发现此刻Syn包有明显增多的趋势，持续时间为11秒，检查结果符合预期。

| | A | N | O | P | Q | R |
|---|---|---|---|---|---|---|
| 1 | Timestamp | TcpInSyns | TcpOutSyn | TcpInAcks | TcpOutAck | TcpInSynA |
| 74647 | 1633898645 | 7 | 0 | 57982 | 54141 | 0 |
| 74648 | 1633898646 | 6 | 0 | 57612 | 54597 | 0 |
| 74649 | 1633898647 | 4 | 0 | 58879 | 55769 | 0 |
| 74650 | 1633898648 | 2 | 0 | 57116 | 53823 | 0 |
| 74651 | 1633898649 | 5114 | 0 | 57661 | 54154 | 0 |
| 74652 | 1633898650 | 4951 | 0 | 59476 | 56583 | 0 |
| 74653 | 1633898651 | 5355 | 0 | 56378 | 54243 | 0 |
| 74654 | 1633898652 | 5226 | 0 | 56448 | 54384 | 0 |
| 74655 | 1633898653 | 5177 | 0 | 54899 | 52925 | 0 |
| 74656 | 1633898654 | 5293 | 0 | 56749 | 54927 | 0 |
| 74657 | 1633898655 | 5167 | 0 | 54914 | 53081 | 0 |
| 74658 | 1633898656 | 5161 | 0 | 56193 | 54396 | 0 |
| 74659 | 1633898657 | 5235 | 0 | 57885 | 55420 | 0 |
| 74660 | 1633898658 | 5207 | 0 | 55302 | 54216 | 0 |
| 74661 | 1633898659 | 1000 | 0 | 54948 | 54001 | 0 |
| 74662 | 1633898660 | 6 | 0 | 56281 | 55089 | 0 |
| 74663 | 1633898661 | 3 | 0 | 57033 | 55000 | 0 |
| 74664 | 1633898662 | 8 | 0 | 55976 | 53305 | 0 |
| 74665 | 1633898663 | 1 | 0 | 55245 | 52418 | 0 |

==> 发现IP地址为 `106.39.164.60` 在timestamp为 `1633899704` 时发生SynFlood攻击，持续到 `1633899776` 止。

```
MEAN:   ./data/106.39.164.60.txt    0.533345    22.788553
-> CUSUM SynFlood:   ./data/106.39.164.60.txt   1633899704  582.600000     591   2
-> CUSUM SynFlood:   ./data/106.39.164.60.txt   1633899705  1317.600000    663   0
...... 这里是省略号
-> CUSUM SynFlood:   ./data/106.39.164.60.txt   1633899773  1175.600000    592   0
-> CUSUM SynFlood:   ./data/106.39.164.60.txt   1633899774  2937.600000   1473   0
-> CUSUM SynFlood:   ./data/106.39.164.60.txt   1633899775  2471.600000   1240   0
-> CUSUM SynFlood:   ./data/106.39.164.60.txt   1633899776  3697.600000   1853   0
```

与原始数据进行比对，发现此刻Syn包有明显增多的趋势，持续时间为11秒，检查结果符合预期。

| | A | N | O | P | Q | R |
|---|---|---|---|---|---|---|
| 1 | Timestamp | TcpInSyns | TcpOutSyn | TcpInAcks | TcpOutAck | TcpInS |
| 75696 | 1633899694 | 1 | 0 | 7836 | 5209 | |
| 75697 | 1633899695 | 0 | 0 | 12648 | 11911 | |
| 75698 | 1633899696 | 1 | 0 | 2703 | 2966 | |
| 75699 | 1633899697 | 0 | 0 | 248 | 120 | |
| 75700 | 1633899698 | 0 | 0 | 4483 | 3734 | |
| 75701 | 1633899699 | 3 | 0 | 387 | 272 | |
| 75702 | 1633899700 | 0 | 0 | 22574 | 16876 | |
| 75703 | 1633899701 | 0 | 0 | 163 | 86 | |
| 75704 | 1633899702 | 0 | 0 | 1037 | 770 | |
| 75705 | 1633899703 | 266 | 0 | 166 | 85 | |
| 75706 | 1633899704 | 591 | 0 | 160 | 85 | |
| 75707 | 1633899705 | 663 | 0 | 13801 | 12132 | |
| 75708 | 1633899706 | 696 | 0 | 9188 | 8447 | |
| 75709 | 1633899707 | 846 | 0 | 172 | 96 | |
| 75710 | 1633899708 | 835 | 0 | 209 | 121 | |
| 75711 | 1633899709 | 1013 | 0 | 9989 | 10604 | |
| 75712 | 1633899710 | 853 | 0 | 17968 | 11959 | |
| 75713 | 1633899711 | 891 | 0 | 13531 | 15057 | |
| 75714 | 1633899712 | 972 | 0 | 14633 | 20026 | |
| 75715 | 1633899713 | 1000 | 0 | 22119 | 22846 | |
| 75716 | 1633899714 | 979 | 0 | 188 | 94 | |
| 75717 | 1633899715 | 1004 | 0 | 11052 | 8959 | |
| 75718 | 1633899716 | 1012 | 0 | 34476 | 37533 | |
| 75719 | 1633899717 | 1072 | 0 | 15354 | 17551 | |
| 75720 | 1633899718 | 879 | 0 | 714 | 462 | |
| 75721 | 1633899719 | 1029 | 0 | 7677 | 7392 | |
| 75722 | 1633899720 | 946 | 0 | 13107 | 9258 | |
| 75723 | 1633899721 | 411 | 0 | 14534 | 14900 | |
| 75724 | 1633899722 | 140 | 0 | 5958 | 6909 | |
| 75725 | 1633899723 | 180 | 0 | 2691 | 1879 | |
| 75768 | 1633899766 | 1033 | 0 | 16176 | 18578 | |
| 75769 | 1633899767 | 799 | 0 | 18997 | 18754 | |
| 75770 | 1633899768 | 441 | 0 | 35345 | 38130 | |
| 75771 | 1633899769 | 499 | 0 | 14225 | 14598 | |
| 75772 | 1633899770 | 569 | 0 | 12816 | 8731 | |
| 75773 | 1633899771 | 429 | 0 | 215 | 118 | |
| 75774 | 1633899772 | 482 | 0 | 13933 | 16704 | |
| 75775 | 1633899773 | 592 | 0 | 25074 | 27407 | |
| 75776 | 1633899774 | 1473 | 0 | 30213 | 34615 | |
| 75777 | 1633899775 | 1240 | 0 | 30975 | 39118 | |
| 75778 | 1633899776 | 1853 | 0 | 23097 | 27152 | |
| 75779 | 1633899777 | 0 | 0 | 35838 | 41437 | |
| 75780 | 1633899778 | 0 | 0 | 50318 | 51450 | |
| 75781 | 1633899779 | 5 | 0 | 43166 | 48349 | |
| 75782 | 1633899780 | 4 | 0 | 32912 | 31648 | |
| 75783 | 1633899781 | 6 | 0 | 198 | 103 | |
| 75784 | 1633899782 | 8 | 0 | 7827 | 7694 | |
| 75785 | 1633899783 | 0 | 0 | 10467 | 11882 | |
| 75786 | 1633899784 | 1 | 0 | 13870 | 13917 | |
| 75787 | 1633899785 | 2 | 0 | 15512 | 20185 | |
| 75788 | 1633899786 | 1 | 0 | 12225 | 14823 | |

| | | | | | |
|---|---|---|---|---|---|
| 75789 | 1633899787 | 1 | 0 | 14008 | 15395 |
| 75790 | 1633899788 | 0 | 0 | 49748 | 51618 |
| 75791 | 1633899789 | 7 | 0 | 58406 | 63744 |
| 75792 | 1633899790 | 8 | 0 | 66573 | 68623 |
| 75793 | 1633899791 | 3 | 0 | 24999 | 29797 |
| 75794 | 1633899792 | 5 | 0 | 17342 | 19122 |
| 75795 | 1633899793 | 6 | 0 | 15811 | 17221 |

# FinFlood 分析

## Fin数据包统计分析

在正常情况下，流入的Fin包与流出的FIN包成1:1的关系，通过对流量最大100个IP地址的数据包类型的统计信息进行分析，发现 `InFin:OutFin` 包的比例比较符合 `1:1` ， `标准差` 普遍在0.5以下（不统计每秒小于500个FIN包的情况）

## 使用CUSUM算法对FinFlood进行识别

**确认持续输入的变量:** `TcpInFins/TcpOutFins`

```
// InFinsDivOutFins Check FinFlood
func (i *Item) InFinsDivOutFins() float64 {
  if i.Metrics.TcpInFins.Max < 500 {
    return float64(0)
  }
  if i.Metrics.TcpOutFins.Max == 0 {
    return float64(i.Metrics.TcpInFins.Max)
  }
  return float64(i.Metrics.TcpInFins.Max) / float64(i.Metrics.TcpOutFins.Max)
}
```

**确认CUSUM模型参数:**

```
// 算术平均数:  1.1
// 标准差:    0.5
// weight:   2
p.finFlood = CUSUM.NewCUSUM(1.1, 0.5, 2)
```

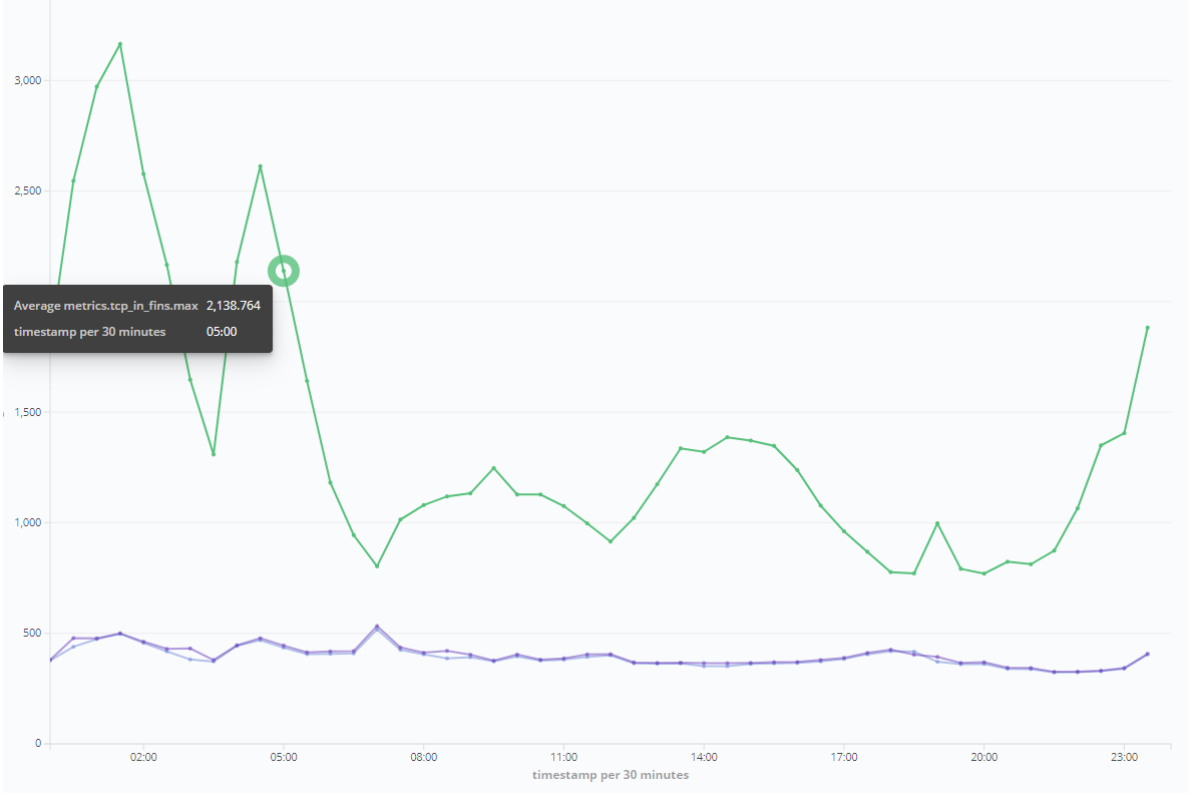
**确认CUSUM模型警戒线:**

当累积和达到一定的数值时，则触发告警；此值越小则检查越灵敏，误报的可能性增大；越大则检查越迟钝，漏报的可能性增大。这里简单使用100进行检查。

```
p.finFlood.Add(it.InFinsDivOutFins())
if rh, _ := p.finFlood.Result(); rh > 100 {
  p.finFlood.Reset()
}
```

**结果确认：**

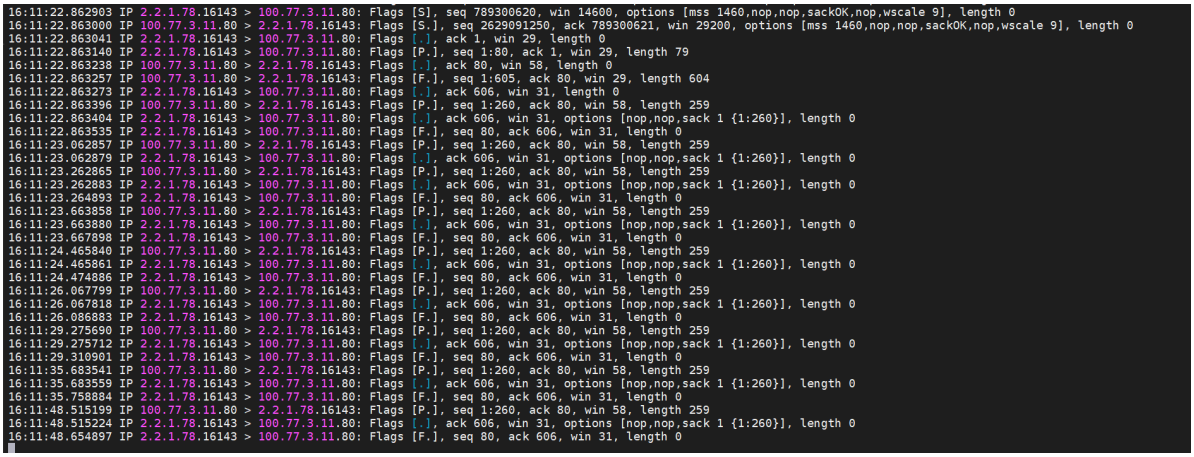
==> 发现IP地址为 `114.67.98.218` 在全天的流入的Fin包都超过警戒值，查看全天Fin的比例关系如图：




通过查询运营后台数据，发现该IP地址处于未备案域名拦截状态：

| 公网IP | 攻击状态 | 防护状态 |
| --- | --- | --- |
| 114.67.98.218 | 域名未备案<br>清洗 | 域名未备案防护中<br>清洗防护中 |

由于未备案域名会触发一个Fin包，用来终结用户的请求，客户端会重复发3个或10个Fin包来与服务器确认连接关闭：




**除了未备案域名拦截的情况，未出现其他可能是FinFlood的情况**

# 结论

基于统计的数据流量计算累积和的方法，经过简单测试证明其是有效的。

遗留问题：

1. `InSyn:OutFin`的比例存在`2:1`的情况，需要排查是否对清洗流量进行了累加所致
2. 未备案域名的阻断会造成客户端重传Fin包的情况发生，有可能会对FinFlood造成误报