**Network Security**

**Professor Gaurav S. Kasbekar**
**Department of Electrical Engineering**
**Indian Institute of Technology, Bombay**

**Week - 10**

**Lecture - 58**

**Firewalls and Intrusion Detection Systems: Part 8**

Hello, recall that in the previous lecture, we discussed different measures such as egress filtering and distributed route filtering. These were measures to prevent distributed denial-of-service attacks. Now we will discuss some techniques that can be used to detect distributed denial-of-service attacks. So, egress filtering and DRF are preventive measures. An alternative approach is to detect the onset of a denial-of-service attack and then take remedial action.

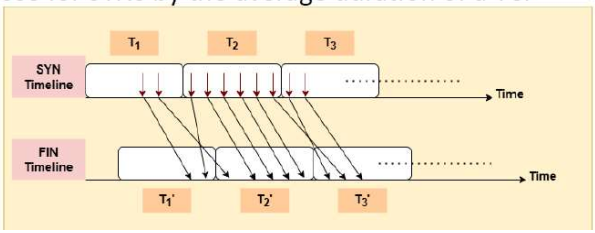
So, that is, we detect denial-of-service attacks. So, recall that a TCP connection is initiated by a three-way handshake in which SYN, SYN-ACK, and ACK packets are sent. Later on, the TCP connection is closed by each side by sending a FIN packet, which is ACKed by the other side. So, for legitimate connections, a server receives SYN packets and FIN packets in pairs. That is, for every SYN packet, it eventually receives a FIN packet.

So, the number of SYN packets and FIN packets should be roughly equal. But in a SYN flood attack, the victim receives a much larger number of SYN packets than FIN packets. So, recall that the attacker just sends a SYN packet and then does not send subsequent packets. So, the connection is only half-open. So, in this case, the FIN packet is not sent corresponding to a particular SYN packet.

So, hence, in a SYN flood attack, the victim receives a much larger number of SYN packets than FIN packets. So, this fact can be used to detect SYN flood attacks by a victim, as we now explain. So, the idea of this scheme is to keep track of the number of SYN packets received and the number of FIN packets received in an interval, and they should be roughly equal. So, if the difference between SYN packets and FIN packets exceeds the threshold, then that is an indicator that there is probably a SYN flood attack ongoing. So, this figure shows two horizontal timelines.

This is the SYN timeline. So, this shows the times of SYN packet arrivals. So, these arrows indicate instances when SYN packets are received by a particular host. And this bottom line shows the corresponding FIN arrivals. So, these arrows show the time instances when FIN packets are received.

- Time is slotted into fixed-length "observation intervals":
  - $T_1, T_2, T_3, \ldots$ on the SYN timeline, during which we record the number of SYN arrivals
  - $T_1', T_2', T_3', \ldots$ on the FIN timeline, during which we record the number of FIN arrivals
- The observation intervals for FINs are shifted to the right relative to those for SYNs by the average duration of a TCP connection




And notice that there is a correspondence between SYN packets and FIN packets. So, for example, first the SYN packet is received for this connection, and then this is the FIN packet for the same connection, and this is a SYN packet for this connection, and this is the FIN packet for that connection, and so on and so forth. Time is sorted into fixed-length "observation intervals", so on the SYN timeline these intervals are $T_1$, $T_2$, $T_3$, and so on, and on the FIN timeline there are corresponding observation intervals $T_1$', $T_2$', $T_3$', and so on and so forth. So, these observation intervals are $T_1$, $T_2$, $T_3$, and so on, on the SYN timeline, during which we record the number of SYN arrivals, and the observation intervals on the FIN timeline are $T_1$', $T_2$', $T_3$', and so on, during which we record the number of FIN arrivals. So, notice that the observation intervals on the FIN timeline are shifted to the right by some amount related to the SYN timeline.

So, for example, this observation interval is shifted by some amount relative to this observation interval. This observation interval is shifted to the right relative to this observation interval, and so on and so forth. So, this shift amount is the average duration of a TCP connection. Hence, if the SYN packet for a particular connection is received in a particular observation interval on the SYN timeline, then the corresponding FIN packet should be received with a high probability in the corresponding observation interval on the FIN timeline. So, hence, the number of SYN packets received in this observation interval on this timeline should be roughly the same as the number of FIN packets received in the observation interval on this FIN timeline.

But the number of SYN packets and FIN packets in a particular observation interval are only roughly equal because the durations of different TCP connections are different in general. Not all TCP connections have the same duration. So, it can happen that if there is a long-lived TCP connection, then the SYN packet is received in this observation interval, but the FIN packet is received much later in some subsequent observation interval. So, hence, we can only roughly say that the number of SYN packets in a particular observation interval on the SYN timeline should be equal to the number of FIN packets in a particular observation interval on the FIN timeline. So, to construct an anomaly detection system, we define the following variables.

- To construct an anomaly detection system, we define the following variables:
  - $S_i$: no. of SYN packet arrivals in $i$'th observation interval
  - $F_i$: no. of FIN packet arrivals in $i$'th observation interval
  - $D_i = \frac{S_i - F_i}{F_i}$
  - $\mathcal{T}$: threshold for detection
- Consider the time series:
  - $D_1, D_2, D_3, \ldots$

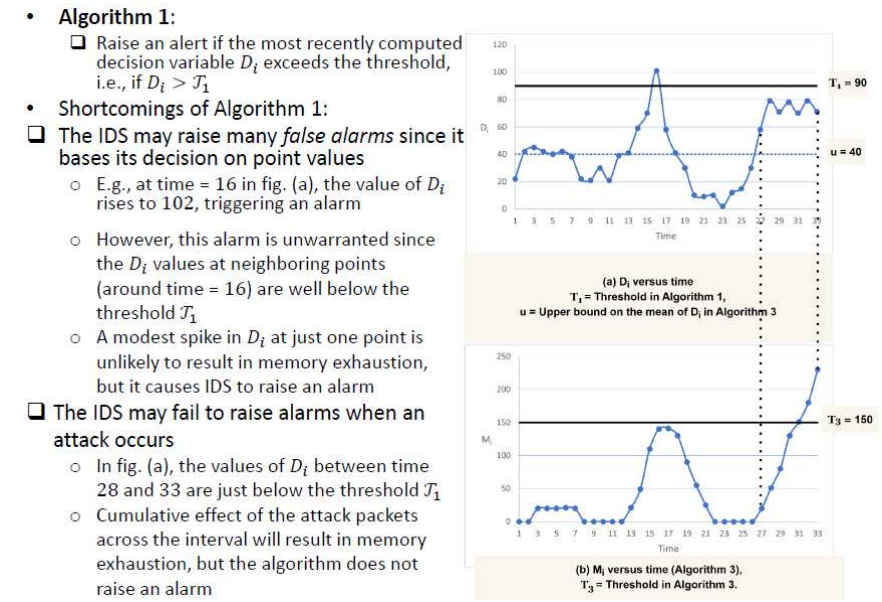
$S_i$ is the number of SYN packet arrivals in the $i^{th}$ observation interval and $F_i$ is the number of FIN packet arrivals in the $i^{th}$ observation interval on the FIN timeline. And $S_i$ is on the SYN timeline. Define $D_i = \frac{S_i - F_i}{F_i}$. So, if the number of SYN packets and the number of FIN packets is the same, then $D_i$ will be 0.

When a SYN flood attack happens, then $S_i$ will be much larger than $F_i$. So, $D_i$ will be a large positive amount. So, that is what we have to detect. If $D_i$ is close to 0, then it is normal operation. But if $D_i$ is a large positive amount, then it is indicative of a SYN flood attack.

Let $\tau$ be the threshold for detection. We designed different algorithms to detect the onset of a SYN flood attack. Consider the following time series: $D_1$, $D_2$, $D_3$, and so on. We'll discuss different algorithms that attempt to detect the onset of a SYN flooding attack by monitoring this time series. If many of these $D_i$s are large, in that case, it is indicative of a SYN flood attack.

So, $D_i$ being large doesn't always mean there is a SYN flood attack because there is some randomness in $S_i$ and $F_i$. So, we know that $S_i$ may not be equal to $F_i$ since the durations of TCP connections are random. But if many consecutive $D_i$s are large, then that indicates with high probability there is a SYN flood attack ongoing. So, that's what we have to design

algorithms for: to detect whether there is indeed a SYN flood attack ongoing. So, in the first algorithm, we raise an alert if the most recently computed decision variable $D_i$ exceeds the threshold, that is, if $D_i$ is greater than some threshold $\tau_1$.




So, this upper figure shows the operation of algorithm 1. $D_i$ is plotted on the y-axis, and the x-axis is time. So, we can see the variation of $D_i$. It is always above 0, and the threshold $\tau_1 = 90$, so that threshold is shown here. In this time interval, at this time instant, $D_i$ exceeds the threshold, 90.

So, in that case, the algorithm raises an alert. So, at this instant, it raises an alert. In this instant, $D_i$s are very high. They are close to the threshold, but they are still below the threshold, so the algorithm does not raise an alert. So, this algorithm has some shortcomings.

The IDS may raise many false alarms because it bases its decision on point values. So, this is an instant when $D_i$ is large. But if we observe the neighboring values of $D_i$ for this instant, the neighboring values are well below the threshold. So, this is probably not the onset of a SYN flood attack because of the randomness in $S_i$ and $F_i$, this $D_i$ crossed the threshold, but the neighboring values are all below, well below the threshold. So, this is a false alarm, where an isolated point has $D_i$ above the threshold, but this algorithm 1 raises an alert at this instant.

So, at time 16, in this figure, the value of $D_i$ rises to 102, that is shown here. This triggers an alarm. But this alarm is unwarranted because the $D_i$ values at neighboring points around 16, they are well below the threshold $\tau_1$. So, this is a false alarm which is raised by algorithm 1. So, hence, it has—algorithm 1 has this shortcoming.

A modest spike in $D_i$ at just one point is unlikely to result in memory exhaustion, but it causes the IDS to raise an alarm. So, at time 16, the $D_i$ value is above the threshold, but since the neighboring points have $D_i$ well below the threshold, it is unlikely to exhaust the memory resources of the host, so, hence, it's not a SYN flood data, so yet it causes the IDS to raise an alarm. And conversely, the IDS may fail to raise alarms when an attack occurs. So, in this example, the value of $D_i$ between time 28 and 33, which is shown here—this is 28 and 33, so these values are just below the threshold $\tau_1$. So, the cumulative effect of the attack packets across this interval, that is, 28 to 33, will result in memory exhaustion, but the algorithm does not raise an alarm.

So, this is a false negative. So, there is actually a SYN flood attack ongoing, but the algorithm fails to detect it. So, this is a false negative. So, does this algorithm one suffers from false positives like at time 16 as well as it suffers from false negatives such as between times 28 and 33. Hence, we need improved algorithms.

So, another algorithm is this. We don't consider point values of $D_i$, but we average out the values of $D_i$. So, algorithm 2 raises an alert if the exponentially smoothed average of the values of $D_i$ exceeds the threshold. So, exponential smoothing is one way of taking averages. It assigns high weights to recent values of $D_i$ and low weights to old values of $D_i$.

- Let $S_i = \alpha D_i + (1 - \alpha) S_{i-1}$, where $\alpha \in (0,1)$, e.g., $\alpha = 0.4$
- An alarm will be raised if $S_i > \mathcal{T}_2$, where $\mathcal{T}_2$ is a threshold
- Value of $\mathcal{T}_2$ set based on empirical data

So, it is calculated in the following manner. Let $S_i = \alpha D_i + (1 - \alpha) S_{i-1}$, where $\alpha \in (0,1)$. For example, $\alpha = 0.4$. So, first $S_1$ is calculated from $D_1$ and $S_0$, then $S_2$ is calculated from $S_1$ and $D_2$, and so on and so forth in this way. So, a simple exercise is to write down the values of $S_2$, $S_3$, $S_4$, and so on, and you can express $S_2$, $S_3$, $S_4$, $S_5$, and so on in terms of $D_1$, $D_2$, $D_3$, $D_4$, $D_5$, and so on and so forth.

So, you can see that Si assigns higher weights to the recent values $D_i$ and low weights to the older values such as $D_{i-1}$, $D_{i-2}$, and so on. So, this is exponential smoothing averaging. An alarm will be raised if $S_i$ is greater than some threshold $\tau_2$. So, in this case, we don't focus on a single value of $D_i$, but we average it and then raise an alarm only if the average value is greater than some threshold $\tau_2$. So, this value of $\tau_2$ is set based on empirical data.

So, there is a trade-off in setting the value of $\tau_2$. If $\tau_2$ is set too low, then it will result in a lot of false positives, and if conversely $\tau_2$ is set too high, then it will result in a lot of false negatives. So, another design parameter is $\alpha$, which is used for computing exponentially smooth average. So, if $\alpha$ is too close to 1, then we can see here that this $D_i$ is assigned a high weight of close to 1. Then it will give disproportionate importance to the most recent value of $D_i$.

But if $\alpha$ is close to 0, then the weights assigned to all values of $D_i$ are even. So, if $\alpha$ is close to 0, then it assigns roughly equal weights to the recent values of $D_i$ and old values of $D_i$. So, this is another parameter that has to be carefully designed. So, this algorithm overcomes the shortcomings of algorithm 1 since it uses averaging. So, it reduces the number of false positives and false negatives compared to algorithm 1.

So, for example, at time 16, this value of $S_i$ will not be very high because the neighboring values of $D_i$ are low. So, Si will be low, and likely it will not cross the threshold $\tau_2$. Hence, an alarm will not be raised at this point 16. Whereas in this interval, 28 to 33, since $D_i$ values are high for several consecutive time instance, so $S_i$ will be high, and likely it will be more than the threshold $\tau_2$. So, there'll be no false negative in this interval.

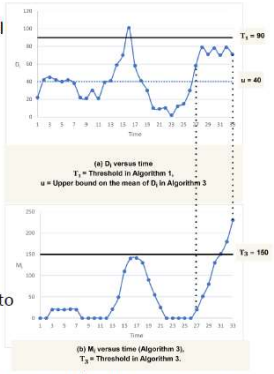
So, the IDS will raise an alert. Let's consider another alternative algorithm, which also overcomes the shortcomings of algorithm 1. So, algorithm 3 is the following. We define a modified cumulative sum of the previous values of $D_i$ and raise an alert if this value exceeds the threshold. So, during normal operation, the number of FINs will balance the number of SYNs, and hence $D_i = \frac{S_i - F_i}{F_i}$ will be close to 0.

Let u be an upper bound on the mean of $D_i$ during normal operations. So, $D_i' = D_i - u$. So, $D_i$' should be less than or equal to 0 in normal operation, but if $D_i$' is greater than 0, then that indicates there is possibly a SYN flood attack. Let $M_i = (M_{i-1} + D_i')^+$ and $M_0 = 0$. So, this plus operator is defined in the following way.

$$x^+ = \begin{cases} x, & if\ x > 0, \\ 0, & Otherwise \end{cases}$$

So, this plus operator prevents $M_i$ from going negative. So, if $M_i$ is going negative, then it is capped to 0. So, $M_i$ is just the sum of the previous iteration value, $M_{i-1}$, and $D_i$', but it is not allowed to go below 0. So, that is done using this plus operator. So, $M_i$ is computed in this way.

- During normal operation, the number of FINs will balance the number of SYNs and hence $D_i = \frac{S_i - F_i}{F_i}$ will be close to 0
- Let $u$ be an upper bound on the mean of $D_i$ during normal operations
- Let $D_i' = D_i - u$
- Let $M_i = (M_{i-1} + D_i')^+$, with $M_0 = 0$,
  - where $x^+ = \begin{cases} x, & \text{if } x > 0, \\ 0, & \text{else.} \end{cases}$
- The IDS sounds an alarm at the end of the $j$'th interval if $M_j > \mathcal{T}_3$, where $\mathcal{T}_3$ is a threshold that is determined empirically
- Fig. (b) shows $M_i$ versus time with $\mathcal{T}_3 = 150$
- Between time 2 and 6, $D_i$ is slightly above $u$, so $M_i$ increases monotonically
- Between time 7 and 12, $D_i$ falls below $u$, so $M_i$ decreases to 0 and remains there until time 12
- Between time 27 and 33, $D_i$ is consistently above $u$, although it is below the threshold $\mathcal{T}_1 = 90$
- This causes $M_i$ to increase and it overshoots the threshold of $\mathcal{T}_3 = 150$; this causes an alarm to be raised




The idea sounds an alarm at the end of the $j^{th}$ interval if $M_j > \tau_3$, where $\tau_3$ is a threshold determined empirically. So, intuitively, if many $D_i$'s consecutively are high, in that case, $M_i$ will cross the threshold $\tau_3$ and vice versa. So, this figure shows $M_i$ versus time with $\tau_3$ = 150. So, this is the threshold 150, and time is on the x-axis and $M_i$ is on the y-axis. The value, u, is 40 in this example. So, $D_i$' is the difference between $D_i$ and u. So, in this case, $D_i$' is negative, and here $D_i$' is slightly positive. Here, $D_i$' is a large positive number, and in this interval again, $D_i$' is positive.

Here, $D_i$' is negative, and so on and so forth. So, this is the corresponding $M_i$ calculated using this equation. So, it is just a cumulative sum, but the sum is not allowed to go below 0. So, between time 2 and 6, $D_i$ is slightly above u, which is in this interval. So, $M_i$ increases monotonically, and we can see the increase here.

So, $M_i$ is increasing in this interval. Between times 7 and 12, $D_i$ falls below u, which is this interval. So, $M_i$ decreases to 0 and remains at 0. So, that is shown in this interval. Between time 27 and 33, that is this interval, $D_i$ is consistently above u. So, we can see that from here, $D_i$ is above u. Hence, $D_i$' is a large positive number, $D_i$ is below the threshold $\tau_1$ = 90 of algorithm 1. Hence, there was a false negative in algorithm 1.

So, this causes $M_i$ to increase. So, since $D_i$ is consecutively above the threshold u, so, that causes $M_i$ to increase, and it overshoots the threshold of $\tau_3$ =150. This causes an alarm to

be raised. So, these are true positives because of the cumulative buildup of SYN attack packets. So, here for a large interval, the $D_i$ values are very large.

So, it is indicative of a SYN flood attack. So, it is likely that actually a SYN flood attack is happening, and hence, $M_i$ crosses the threshold of $\tau_3 = 150$; we can see that here. So, this is $M_i$, and it crosses the threshold of $\tau_3 = 150$ at this point, and the IDS raises an alert. So, here, in this interval, $D_i$ crosses the threshold of algorithm 1 at only one point and here, we can see that $M_i$ is below the threshold of $\tau_3$, so there is no alert raised by the IDS.

So, there is no false positive for algorithm 3 at this point even though there is a false positive for algorithm 1. So, in this way, false positives and false negatives encountered with algorithm 1 are avoided with algorithm 3. So, we can see that algorithm 2 and algorithm 3 are better algorithms than algorithm 1. So, these three are algorithms for DDoS detection. So, in summary, we discussed different methods for detection of SYN flood attacks.

So, these methods use the fact that in normal operations, the number of SYN packets and the number of FIN packets should be roughly the same. So, we designed different algorithms that keep track of the difference between the number of SYN packets and FIN packets, and these algorithms helped us detect distributed denial-of-service attacks. And once these SYN flood attacks are detected, then actions can be initiated to defend against them. So, in that way, these algorithms can be used. So, this concludes our discussion on firewalls and intrusion detection systems.

Thank you.