RIOREY

ANATOMY OF A DDOS - VALID HTTP GET REQUEST ATTACK

Background

In March and April 2008, a large UK Industrial Equipment vendor's e-commerce web site was under severe DDOS attack. The attack was a well crafted HTTP GET attack that disabled the entire site for several days at a time.

The customer made many unsuccessful attempts to mitigate this DDOS attack. Their first attempt was to move from a small hosting company to a large hosting company. This change made no material difference to the attack despite assurances from the new hosting company. In addition to many technical attempts to mitigate the attack, the customer requested help from the UK police in cybercrime, and was able to shut down two controllers responsible for their attacks.

Unfortunately, shutting down the two botnet controllers only gave them about a week of normal network activities. Soon, the attackers returned with new botnets, and the customer's servers were down once again. This time, RioRey was asked to help mitigate the DDOS attack.

The Attack Technique

The attackers in this case are "smart attackers". They targeted the customer and spent efforts probing the site to find the weaknesses. The method that the attackers settled on was a valid HTTP GET session, targeted at a Microsoft Access Database lock problem. By targeting database locks, and CPU intensive operations, the attackers were able to take down the entire site with relatively small amount of attack traffic.

The attackers launched a continuous set of attacks; usually starting on Tuesday or Wednesday morning, continuing non stop until the close of business on Friday.

The attack pattern we observed started with a slow attack rate using several hundred bots for a certain duration of time. This low rate initial attack fools the netflow baseline, and then gradually more bots are added at about 10 to 30 bots per minute, consisting of a mix of low and high traffic-rate bots.

At a certain point, the attack surged to a peak rate of 10k SYN requests a second, knocking the Access Database out of service. Then the bots settled back to a steady rate of roughly 2k SYN requests a second, keeping the servers busy despite efforts to reboot the system. Using small SYN packets (64 bytes), the attacking SYN traffic was 6 Mbps at its peak and only 1 Mbps on average. The attack mixes well with good traffic, making a lethal portion of the attack invisible to the tools the customer was using before RioRey.

The attacker used a pool of 30k bots, cycling them in and out of the attack. By cycling the low attack rate bots, the attacker made it very difficult to differentiate between good and bad traffic.

An analysis of the bots and their attacking code afterwards revealed that the bots are capable of generating HTTP GET requests, SYN Floods, ICMP Floods, UDP Floods, TCP Session Floods and any combination of the attacks described. The bots used in this attack were primarily located in India, Middle East and Indonesia. A smaller portion of the attack came from the EU and US.

Mitigating The DDOS Attack

The RioRey equipment was installed in the network during a DDOS attack. Within 5 minutes of powering up the RX, we were able to report the attack immediately on rView to the customer.

However, with this carefully crafted attack, the RX Version 4.0.5 installed was indicating that the bad traffic has sufficient resemblance to good traffic and it could be misidentifying the two traffic types. Since the RX's mission is to preserve good traffic, it proceeded to alarm only on the attack traffic without filtering a portion of the attack.

The RioRey response team worked continuously for about 24 hours closely with the client. We were able to capture a signature of the attack and made an adjustment to the RX decision algorithm. This was implemented in one day, then tuned and refined in the following day.

In addition, our team recommended to the customer ways to improve the robustness of their server including migrating off the Access database.

After the 2nd day, the attack was completely mitigated without any operator intervention. This however did not deter the attackers. They continued unsuccessfully probing RioRey defenses for weaknesses and periodically launched attacks in an attempt to break through. Figure 1 below is a chart showing recent attack attempts on the customer's servers. All the attacks were successfully filtered, fully and automatically. Web service has not been interrupted since.

All RX releases 4.1.0 and newer have implemented this adjustment as part of its framework.

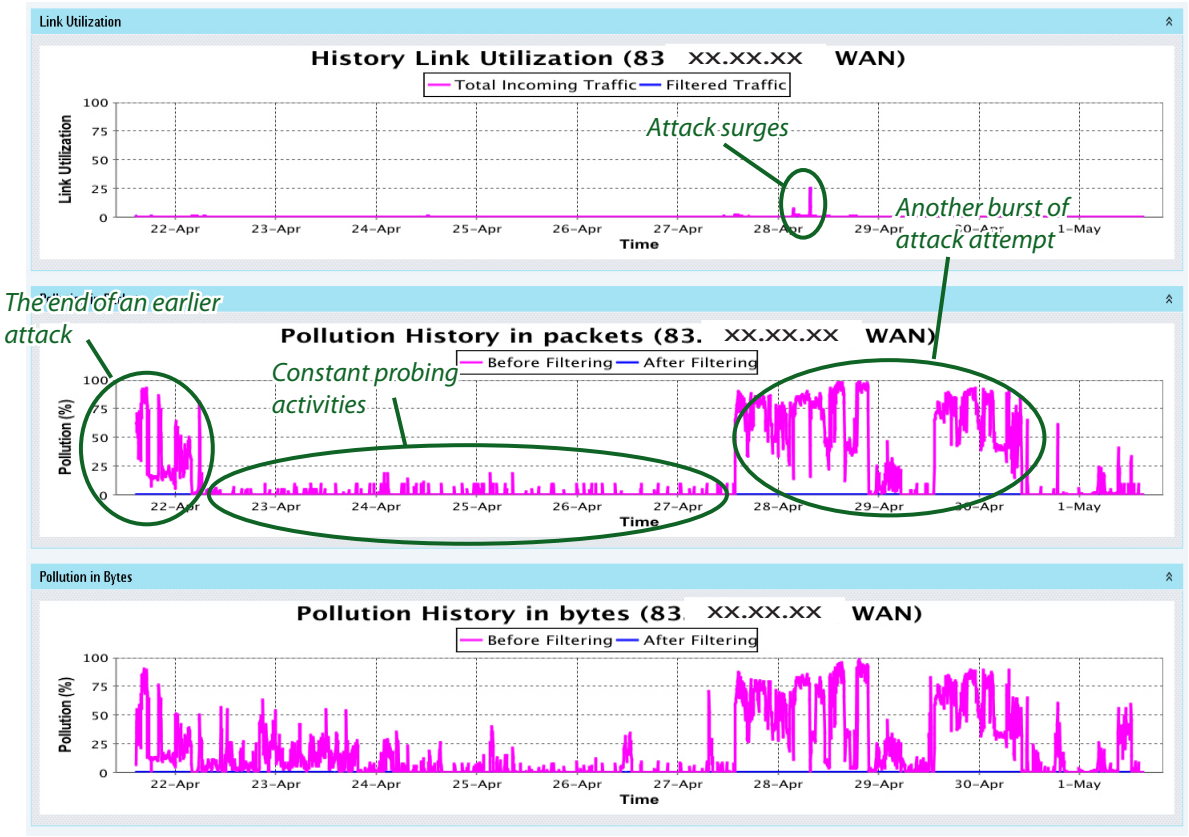

Figure 1. A 10 day snapshot of attack traffic on the Customer's network

The Defense Algorithms

The RioRey RX is unique in the DDOS defense model. Unlike other older generation platforms, RioRey implements not one algorithm, but instead uses a family of algorithms, each monitoring a specific characteristic in the network traffic.

Some of our proprietary algorithms are deployed solely for "single ended attacks" where the bots uses forged IP addresses, flooding the victim with packets, disregarding any response from the victim.

Other algorithms in the RX are deployed for "full handshake attacks" where the bots completes a session handshake properly with the victim, and continuously and relentlessly pursue packet exchange with the victim, until the victim crashes.

In addition, the RX also has a tight syntax checker, verifying and assuring the quality of the packet syntax passing through our filters. Any failures in this process are marked as Port 80 attacks.

In this particular case, the attacker triggered three parts of our defense system:

1. *HTTP attack sensors:* The attack traffic triggered our HTTP sensors when we observed an excessive amount of HTTP GET requests for certain subset of clients targeting a certain subset of host sites.
2. *TCP Session attack sensors:* During peak attack times, some attacking bots were starting so many sessions that they failed our TCP Session detector.
3. *TCP SYN attack sensors:* Also during the peak attack surge, some of the attacking bots were so busy generating SYNs that they were no longer responding to the server's SYN ACK. These busy bots, in effect behaved like a single ended attack and were marked by our TCP SYN detecting engine.

Combining the output of the three sensors, we were able to identify the DDOS attack correctly. The RX was filtering nearly 100% of the attacking traffic, maintaining 100% uptime for our client's web services.

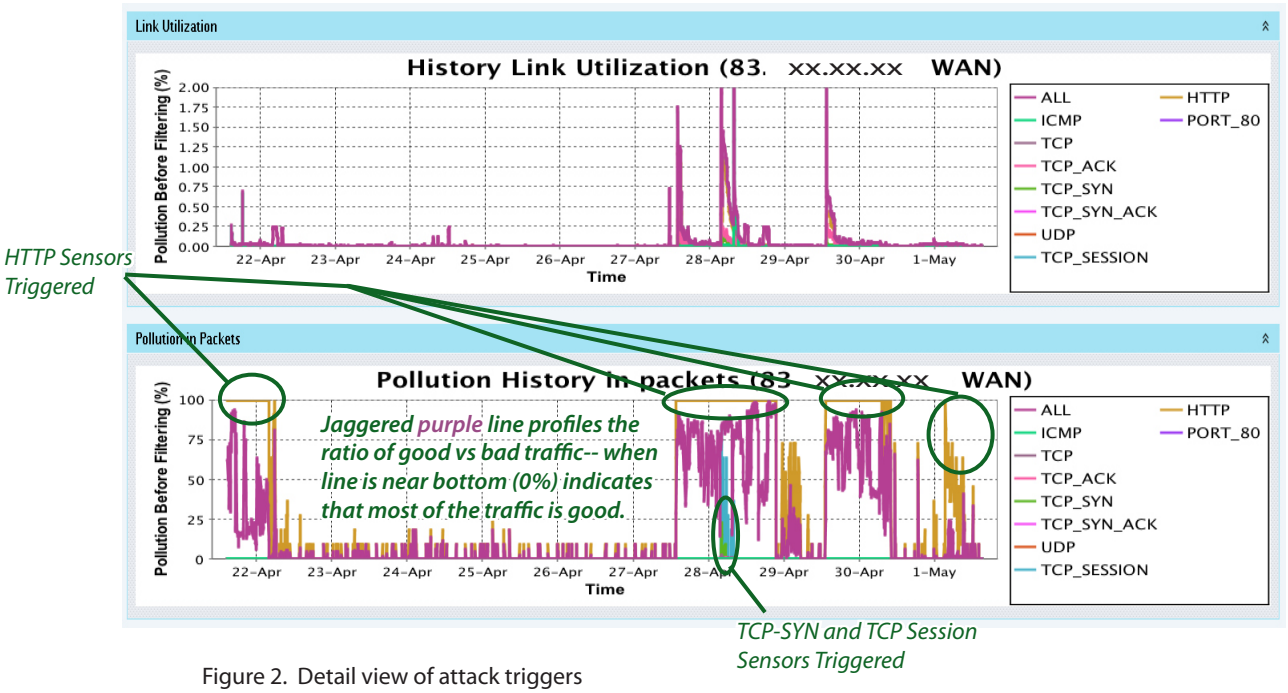

Figure 2. Detail view of attack triggers