ISS

INTERNET SECURITY SYSTEMS

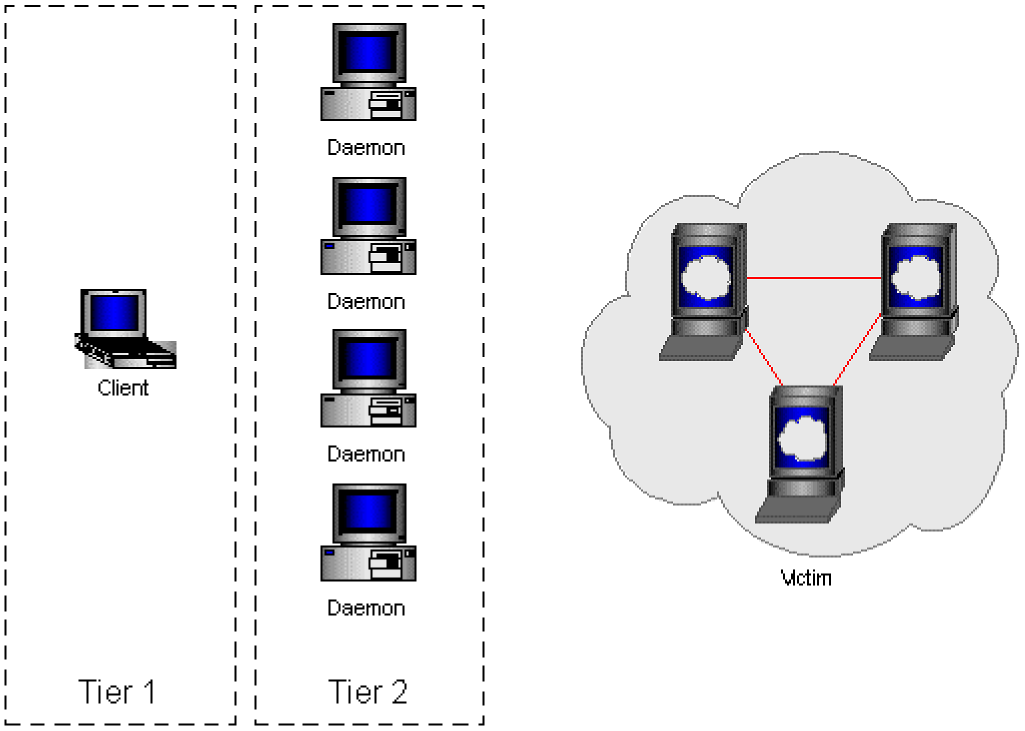
Distributed Denial of Service Attack Tools

Introduction: Distributed Denial of Service Attack Tools

Internet Security Systems (ISS) has identified a number of distributed denial of service tools readily available on the Internet. Some of these attack tools include: TFN, Trin00, TFN2K, and Stacheldraht. These attack tools differ in their capabilities and complexities, but all share the common goal of attempting to overwhelm a victim with an abundant amount of difficult to detect or filter traffic. The evolution of these tools has introduced both encryption and additional tiers to avoid their detection and increase their scalability, as their following descriptions, listed in the order they were discovered, will show:

Tribal Flood Network

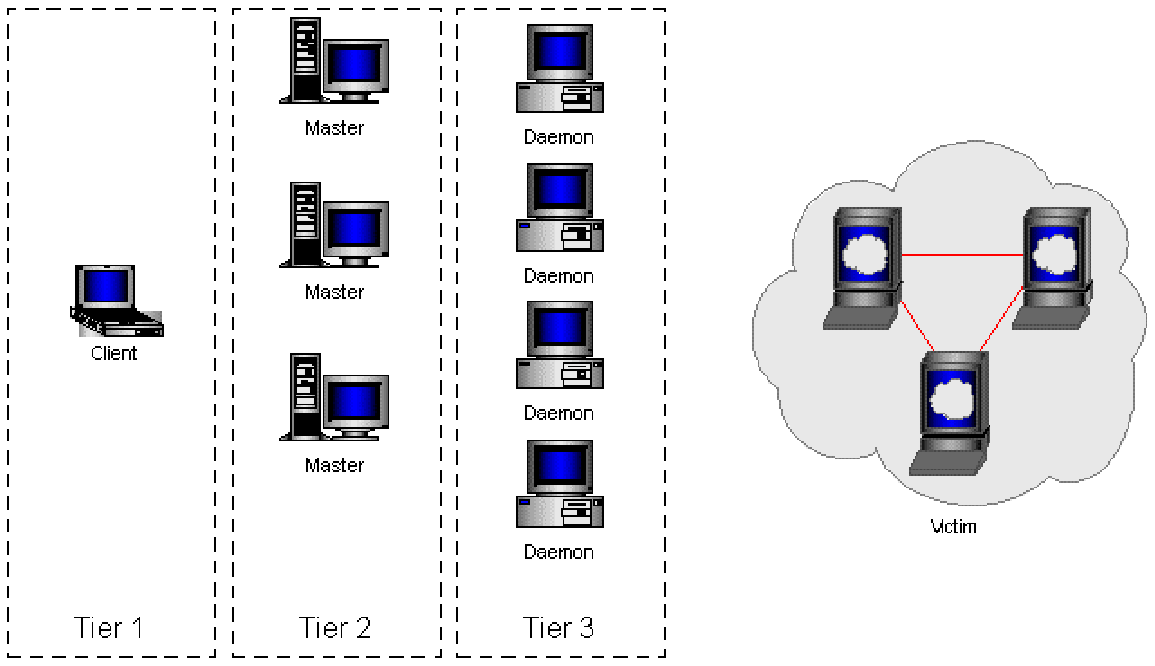
TFN was the first highly visible distributed denial of service attack tool to surface. It is has been nicknamed “Teletubby Flood Network” or “Tribal Flood Network”. TFN exhibits a two tier architecture, involving a client that controls the targeting and options of the attack system, and multiple daemons, which function as listeners for the client’s commands, and perform the actual denial of service attacks, chosen from a variety provided in the tool.



TFN daemon runs as a hidden service on the machines it uses, able to receive commands from the client hidden subliminally in standard network communications/protocols. It also hides the client and daemon’s source in all communications and attacks.

Trin00

Trin00 moved to a three tier architecture, including a client (telnet or netcat), used by the attacker, that sends its commands, including targets, to master servers, which control multiple daemons, knowing their addresses and forwarding commands received from the client.



This additional tier made this tool harder to trace back to the attacker, adding an additional layer to the communication. However, Trin00 did not take advantage of all of TFN's technology to hide itself, communicating using it's own proprietary channels and failing to hide the source of it's attack traffic. Trin00 also was limited to only one form of denial of service attack, unlike TFN, which had a variety.

TFN2K

TFN2K, while not evolving to a three-tier architecture like Trin00, added encryption to its communication between its 2 tiers, client and daemons, making it harder to detect. TFN2K also added a new type of denial of service attack, "Targa3".

Stacheldraht

Stacheldraht took Trin00 and TFN's technology and combined them, hiding the source addresses of it's traffic and adding the variety of denial of service attacks from TFN, while adding the three tier architecture of Trin00. A new version of stacheldraht has recently emerged with additional technology to hide its presence and communications.

Distributed Denial of Service Mitigation

Introduction

This paper discusses various options for dealing with Distributed Denial of Service (DDoS) attacks. Some options are aimed at reducing the effect of an attack, others at detecting the attack, others are aimed at providing forensic information, others discuss how to prevent the attack altogether.

Attack Survival

A DDoS attack involves many hosts sending random data to a target. In most cases, the data is spoofed, typically with random source addresses for each packet. We present an option for packet filtering that uses this feature, along with the robustness of TCP, to filter the flood. The current tools use the same target IP address for the duration of the attack, and we present another option that uses this to avoid the attack.

Moving Target Defense

One method of surviving an attack is to change the IP address of the target system. This causes the remainder of the attack packets to be delivered to the old, now invalid IP address. Depending on whether the routers are flooded, it may be necessary to remove the routes to the old IP address from the Internet (by using BGP or something). In order to maintain connectivity during the IP address change, it will be necessary to update DNS. To perform the IP address change with the minimum amount of downtime to the host system it would be best to have a separate Network Address Translation system, and change the address at the NAT system. This makes the change transparent to the actual target. It might be possible to create an automated system that detects the attack and makes the necessary DNS, BGP, and NAT changes to keep things the target site available.

The method mentioned above can be done differently – instead of changing IP addresses when an attack is detected, the change can occur periodically, every day or every hour, as well as whenever an attack occurs. This forces the attacker to perform frequent DNS requests to determine the current IP address of the target, and these DNS requests can provide useful forensics information, but we'll talk about that later.

Filtering Defenses

Surviving an attack by filtering requires being able to filter the flood packets. There are two ways to do this. One way is with a signature-based packet filter. If we can create signatures for typical flood packets (TCP packets with zero data size for example, or unusually large ICMP packets), and filter out those packets, we can filter the flood packets while allowing "normal" traffic to proceed. This can be done using RealSecure's signature technology and engine, and adding a proxy to it. Obviously, this leads to an arms race between packet generators and signature writers. This technology can also be used to prevent attacks, by filtering out control channels. Since the number of signatures for DDoS is small, it may be possible to run this tool at relatively high throughputs.

Another filtering option is to reject (not pass through the filter) the first IP packet from any IP address. This works with the current generation of attack tools because they all tend to use a flat distribution random number generator to generate spoofed source addresses, and they only use each random address once. This would only work for websites or other TCP-based servers, because TCP is robust enough that if the first packet is rejected, it will send a second request, which this method allows through, along with all subsequent packets. It is also possible that this method will allow normal traffic of UDP and ICMP protocols through, if the protocols implement a retry after the first packet is ignored (i.e. a timeout). The main problem with this approach is that once the method is discovered, hackers will change the tools to work around them (by sending multiple packets from each random source address). There may be a way to respond to this, starting another arms race.

Another possibility is to divert traffic based on IP protocol to different servers or even route it differently. Thus, for a web server it might be possible to route ICMP and UDP traffic bound for the web server somewhere else entirely, or even block it at the router, so that only TCP-based floods will succeed. This at least narrows the scope of attacks that can be made.

Bandwidth Defense

A brute force method of defense for websites and other content providers is to utilize a service like Akamai or Sandpiper that uses large pipes and large distributed networks to provide enough bandwidth to survive an attack.

Rate Filtering

If an attacked site peers with multiple providers, it may be the case that one of the providers is carrying more of the flood traffic than the other. The attacked site may choose to filter access from the provider that is carrying the majority of the traffic, or even terminate their connection with that provider, to reduce the impact of the flood.

Attack Prevention

Preventing the attack in the first place is the ideal situation. Preventing the attacks from utilizing spoofing is also of some use, since it makes it easy to track down the source of the attack, and it also makes it easy to filter traffic from the attacking hosts.

Ingress Filtering

One of the best ways to prevent attacks that utilize spoofing is to implement “ingress” filtering at the point of attack. Ingress filtering prevents spoofed attacks from entering the network by putting rules on point-of-entry routers that restrict source addresses to a known valid range.

Because ingress filtering must be present at each point-of-entry, it must be set for each subnet on each router in the organization. This means it is a lot of work to check each router by hand. There are a couple of ways to check the ingress filtering configuration of an organization.

One way is to provide an easily distributed program that sends spoofed packets to a listener program. If the listener program receives the spoofed packets, it can notify the remote program that the packet was received and also log the network from which it received the spoofed. After the program has been run at each location, the listener can present a status report of the ingress filtering configuration in the organization.

Another option is to integrate with some popular network management platform such as OpenView or Tivoli. These tools may already have stored the filtering rules, and may also be able to push them out to the routers in the organization if they are missing. Our tool can examine the list of rules and determine if any changes need to be made.

A third possibility is to perform automatic ingress filtering by creating a packet filter device which sits on the wire and stores up a list of usual source addresses. When it notices a large number of unusual source addresses, especially if they are all going to the same target address, it can do several things. It can reject the unusual source addressed packets, it can notify the target address, and it can even reject all traffic to the target address until the attack is over. This is also a method for detecting when an attack is happening.

Control Channel Filtering

By filtering out DDoS control messages, we prevent the attacker from causing the attack servers to begin the attack. This prevents the attack altogether. This can be accomplished using a signature-based packet filter mentioned in the Attack Survival section. If we can develop signatures for most control channel packets, we can simply reject them at the control channel packet filter, and they will disappear from the wire.

Active Response

Another possible attack prevention method is especially useful for prevention when control channels are detected and unencrypted (or decrypted). By using credentials sniffed from the control channel, it should be possible to take control of the attack server and shut it down.

Analyze DDoS Tools

We should analyze the different DDoS tools, both for traffic decode signature information, network check information, local host check information, and also looking for buffer overflow vulnerabilities and other ways that the DDoS tool may be vulnerable to being shut down.

Assessment

There are several roles for assessment in preventing distributed denial of service attacks. The DDoS attack happens because of poor network security. By having remote-rootable machines visible to the Internet, the attacker is able to subvert the machines and use them as attack servers, or use them to subvert other machines as attack servers within the organization. These vulnerable machines are the root cause of the DDoS attack, and finding and fixing these remote-rootable machines is one area where assessment can play a role. It may be possible to automatically fix these remote-rootable machines by exploiting the remote root and using it to run a fix script, but there is some risk involved in this, namely that the fix may break things. Writing a remote-root scanning/fixing worm has also been proposed, but this is probably illegal.

Another area is finding attack servers. Network assessment tools can find some of these, and host-based assessment tools can identify others. Finding and removing these attack servers can prevent attacks from taking place.

Attack Forensics

If the attack cannot be prevented or avoided, it is still useful to determine its origin, to enforce accountability on the perpetrator and thereby discourage other attackers.

DNS Logs

The attacker must use DNS to determine the actual IP address of the target before launching the attack. If this is done automatically by the attack tool, the time of the DNS query and the time of the attack might be quite close together, and it may be possible to determine the identify of the attacker's DNS resolver by looking at DNS queries around the time of the start of the attack. It may also be extremely useful to compare DNS logs from different systems that have been attacked – we may see the same small set of hosts making the queries right before the attack. If the same individual or group perpetrates the different attacks, we may be able to locate them using this method.

Control Channel Detection

Detecting large volumes of control channel traffic is a likely indicator that the actual attacker or attack coordinator is close to the detector. Implementing a threshold-based detector that looks for a certain number of control channel packets within a certain time interval may be a good way to provide an early warning of an attack and also provide insight into the network and geographic location of the attacker.

Correlation and Integration

By integrating an attack detector with other tools that can trace spoofed packets, it may be possible to automate the location of the attacker. By correlating data from control channel detectors and flood detectors, it may be possible to determine which control channel caused which flood, or it may be possible to follow spoofed signals from hop to hop, or from attack server to target. For example, identifying the closest attack source hop may serve to minimize the effect of a source IP Range based filtering response.

About Internet Security Systems (ISS)