Protecting Public Servers from DDoS Attacks Using Drifting Overlays

Venkata K. Pingali
USC/Information Sciences Institute
Marina del Rey, CA, USA
pingali@isi.edu

Joseph D. Touch
USC/Information Sciences Institute
Marina del Rey, CA, USA
touch@isi.edu

***Abstract*— Drifting overlays enable enterprises a level of control over their own DDoS defenses and routing choices, rather than leaving them at the mercy of their ISPs. Customers connected via a single ISP often lack control over their own network traffic. ISPs cannot allow individual customers to override their routing, nor can they support per-customer DDoS defense. By coupling dynamic partial overlays with traffic 'safe houses', enterprises can play versions of shell games with tunnel endpoints and packets. These drifting overlays allow enterprises control over their own vulnerabilities.**

Keywords-Denial of Service, Overlays

I. INTRODUCTION

Network customers often need some control over how their ISPs handle their traffic. When the ISP provides alternative peering points, customers may want to manage how their own traffic leaves, or where it enters, at a fine granularity to avoid DDoS attacks (for incoming traffic) or for policy or performance (for outgoing traffic). This is normally accomplished by route peering relationships with the customer. Unfortunately, this is coarse grained, expensive and not always an available option; there are ISPs who do not support peering with small customers, and peering support doesn't always propagate beyond the first ISP. Customers connected via a single ISP often lack control over their own network traffic. ISPs neither allow local control of routing, nor can they support per-customer DDoS defenses. ISPs are expected to aggregate control and management, and self-managed routing defeats this.

Drifting overlays enable enterprises a level of control over their own DDoS defenses and routing choices at fine granularity, rather than leaving them at the mercy of their ISPs. The basic architecture starts with "safe houses" - sites on other ISPs, or distributed within an ISP, which are under the control of the customer (Fig. 1). These sites are used solely for traffic redirection. The server is advertised as being reachable only through the safe houses. The customer's primary site tunnels to and from these safe houses, using the LAN IP address, which may be private, of the safe house on the local side of connections. The safe house provides a type of tethered remote network interface, allowing the customer's traffic to appear as if it originates at any of these remote sites. As a result, traffic terminates its ISP-routed path at the safe house, which results in different traffic paths than are possible from the local customer's site alone, without needing to relocate customer resources. By making safe houses and the tunneling path selectable at runtime and dynamically reconfigurable, enterprises can play versions of shell games with tunnels and packets. A type of shell game, developed under the DynaBone project [3] here at ISI, is applied to manage the traffic.

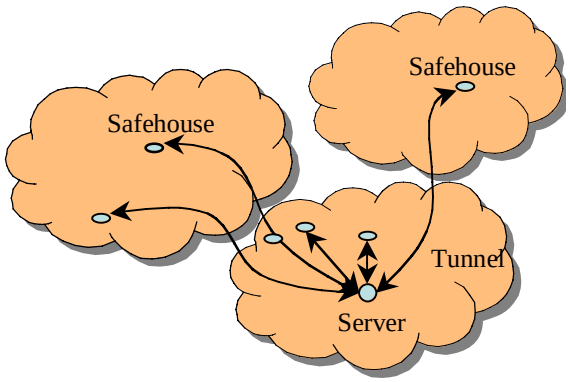


Figure 1. A combination of continously evolving partial IP overlay and hosts (called safehouses) are used to direct traffic to and from public servers.

II. ENTERPRISE ROUTE CONTROL

Enterprises need flexible route control beyond the filtering, rate control and firewalling. The need arises from the desire to better utilize the resources and address the asymmetry in capability of the attacker and the enterprise. *First, the reachability must be controlled*. Almost all the existing DDoS solutions assume the server is globally reachable by default and then address the possibility of attacks. Some sites might prefer the other way round, i.e., a server is globally unreachable by default but reachable through appropriate enabling mechanisms such as tunneling. In case of the reachable by default model, no amount of filtering is sufficient because the attacker has far more capacity to generate traffic than the enterprise can handle and can increasingly mimic flash crowds, i.e., non-attack traffic, accurately. Security through obscurity is not really an option because the attackers' sophistication is growing with time and it takes only one security break for the server address to be known. In case of the unreachable by default mode, even if the enabling mechanisms, such as the safe houses mentioned above, are compromised, they are relatively few in number and under local control. *Second the cost must be paid* ***where*** *necessary.* All clients are not equal, and disruption of traffic to and from certain destinations is acceptable and not from some others. All servers are not equal either. Some need more

protection than others. Further this changes with time. While DDoS mechanisms use some form of marking of packets to determine legitimacy of the traffic, i.e., to treat clients unequally, in most cases they end up wasting key resources on attack traffic because of the cost and quality of the marking schemes. *Third, the cost must be payable **when** necessary*. The solutions must be simple to enough to allow fast deployment when necessary and have non-linear impact to make best use of each incremental resource. Solutions that require cooperation of many hosts/routers in the network, extensive reconfiguration and/or large amounts of state are unlikely to be deployable quickly. Further, a major problem with some existing solutions is that they require the DDoS defense designed for worst-case scenario to be the common case resulting in poor utilization of resources. *Last, the requirement of each enterprise is different.* The solution must be simple and programmable enough to be rapidly customizable to each site. The amount of resources, granularity and strength of the protection must all be programmable.

There is no one solution that meets all the needs above that is simple, scalable and protects all communication. Drifting overlays allows tradeoffs to be made at the enterprise level through a simple framework that provides a degree of control over visibility, reachability and predictability of the paths taken to and from important public servers. The continuous change over time could potentially invalidate the information that the attacker has regarding the location of the safe houses and/or the paths. This has the effect of changing the nature of the attack from one of physical resources to one of information. It is much easier to design and implement defenses for information attacks. Drifting overlays, however, do not completely eliminate the problem of DDoS – especially those that are capable of engaging network ingress and egress routers. In fact, it has the potential of shifting some of the attack targets from application servers to routers or other critical nodes such as the DNS.

III. ARCHITECTURE

The Drifting overlay architecture has three components: (1) hosts, called safe houses, (2) a set of IP tunnels connecting the safe houses to each other and to the server and (3) a tunnel management protocol. The tunnels form a "partial" overlay with significant amount of traffic between overlay nodes and the rest of the Internet. Further this overlay is continuously modified in an open-ended fashion for security, performance and other reasons. In effect, they form a flexible overlay that can be seen to drift over time.

The safe houses could operate as either clients that initiate communication with the server or forwarders that route traffic over the tunnels to and from the server. The public server is advertised as being only reachable through a subset of safe houses that are in client mode, typically through DNS entries. There is no requirement that other safe houses or the server be globally reachable. The server must be reachable from the safe houses through one or more tunnel hops. Finally the forwarders route traffic across tunnels to other safe houses or the server. The safe houses may be shared across deployments, i.e., for multiple servers from the same or different enterprise, using isolation mechanisms such as Clonable stacks [14] and NetFS [13]. They may be available as a commercial service similar to one provided by Akamai [12]. The safe houses that are available for configuration must be discovered at runtime. There are many options for the same including a distributed registry of hosts [15].

The safe houses are connected to the server using standard IP tunnels in an IP overlay. The overlay has a simple topology, typically a tree with a small degree, and link properties are deployment-dependent. They include (1) the reachability and persistence of the tunnel endpoint addresses, (2) routing, (3) Security (IPSec) and (4) QoS (delay, bandwidth). In case of client mode, the hosts run applications as well.

Management software implementing the tunnel management protocol is installed on each of the safe houses, and the server. The tunnel management protocol is specific to each deployment. The protocol selects the safe houses and configures tunnels with the appropriate properties. Management protocols may vary in complexity, reactivity and resultant "strength" of the deployment. Because the number of safe houses is expected to the relatively small, the management algorithms are not required to scale. The protocol could be a client server protocol in which a manager computes all the properties [2][4] and informs the client, or a peer-to-peer protocol in which the topology is being continuously optimized for traffic [19]. A simple management protocol is being built as part of the prototype.

IV. RELATED WORK

The various DDoS solution can be seen as a combination of rate control, resource duplication and a strategy to exploit that redundancy. The resource is typically hosts, paths or information (e.g., addresses). The strategy in most cases is either proactive hard-to-predict resource instance selection or reactive fault tolerance mechanisms. Both can be combined with marking of legitimate traffic and associated rate limiting at nodes.

Host-based solutions use replication and state transfers to reduce the impact of an attack. Examples of simple host-based solutions include server roaming [6], roaming honeypots [7] and MOVE [9]. They tend to be expensive and may require client cooperation. Drifting overlays lets the client decide the level of cooperation and could potentially incorporate server migration mechanisms.

Path-based solutions on the other hand, use alternative paths to reach the server when one or more paths are attacked. Examples of path-based solutions include Secure Overlay Services and its derivatives [9][10][11] and Mayday [8]. Again these approaches are resource intensive and all extra nodes are participating in traffic distribution all the time. The operation of these nodes is determined by the DHT algorithm and not in the local control.

Approaches that exploit information such as IP address [18], and frequency [17], have existed for a long time and in general very effective. However, information-based solutions are not complete by themselves. The attacker could make the server unreachable by attacking a node on path to the server. Drifting overlay uses a combination of path duplication and

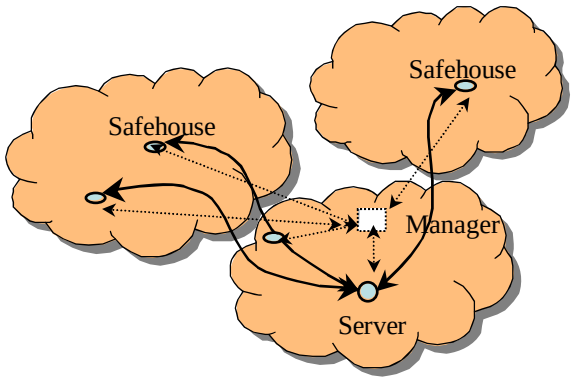
non-uniform reachability of nodes, and randomization. It complements existing rate-limiting approaches and is expected to work with less number of active nodes.

The Drifting Overlay is based on previous work on Dynabone [3] and TetherNet [1]. Dynabone achieves fault tolerance through sophisticated deployment and runtime selection of IP overlays for data transmission. Drifting Overlay uses a single one-layer simple overlay that is modified over time. TetherNet is a special case of a virtual private network (VPN) in which a sub-network is reachable from the rest of the Internet at a chosen point in the global address space. While TetherNet was trying to address the unreachability, Drifting overlays exploits that very unreachability to force traffic along controlled paths and can be considered to be a network of TetherNet links.

The basic solution structure of Firebreak [16] is similar to that of Drifting overlays. Each server is associated with a set of firebreak hosts that are similar to safe houses. The former then tunnels traffic to the server. Anycast addressing is used to route traffic to the firebreak hosts instead of DNS-based solution used here. Anycast as a mechanism has deployment issues on wide area. While Firebreak focuses on the routing of traffic to the firebreak nodes, the emphasis of Drifting overlays is on the tunneling component of the solution. The dynamic reconfiguration of the tunnels is critical for the non-linear scaling of impact as explained earlier. Drifting overlays also differs in terms of non-uniform handling of clients.

Akamai Edge Service [12] provides a similar service and a business model for utilizing the safe houses. The links between the Akamai edge nodes and the servers is application-specific and not controlled by Akamai itself. Further the redirection to the safe houses is controlled by Akamai's DNS-based redirection service. Akamai's service does not eliminate DDoS because the server still has to be globally reachable and DDoS attack clients may not honor the DNS-based redirection. Drifting Overlay are operate at IP-level with fewer number of safe houses. Further, the safe houses may the only way to reach the server i.e., honoring or not honoring redirection has no impact on the server itself.

Figure 2. A Manager coordinates the host and tunnel configuration



V. STATUS

A prototype system is under development on FreeBSD platform. The prototype uses a centralized manager that coordinates the construction of tunnels, routing entries and DNS updates (Fig 2). A control daemon runs on each of the safe houses and receives and executes instructions from the manager. The system uses IPIP tunneling, static routing tables and address randomization. Tunnel addresses are chosen from a large private address space (10.0.0.0/8). The server ensures significant delay in time before reuse of tunnel addresses and consistent timing relationships between the DNS caching time, client end of the configuration and the server end configuration. Early experimentation showed that the header matching in the kernel is a bottleneck when a large number of tunnels are created. Much work remains to be done in terms of topology choices, reconfiguration strategies, characterization of the impact of an attack and deployment issues.

REFERENCES

[1] TetherNet web pages, http://www.isi.edu/tethernet

[2] X-Bone web pages, http://www.isi.edu/xbone

[3] DynaBone web pages, http://www.isi.edu/dynabone

[4] J. Touch, "Dynamic internet overlay deployment and management using the X-Bone", Computer Networks, Jul. 2001, pp. 117-135.

[5] J. Touch, Y. Wang, and L. Eggert, "Virtual Internets," ISI Technical Report ISI-TR-2002-558, July 2002.

[6] S. M. Khattab, C. Sangpachatanaruk, R. Melhem, D Mosse, and T. Znati, "Proactive server roaming for mitigating denial-of-service attacks," Proceedings of International Conference on Information Technology: Research and Education, 2003 (ITRE2003). 11-13 Aug. 2003 Page(s):286 - 290

[7] S. M. Khattab, C. Sangpachatanaruk, D. Moss, R. Melhem, T. Znati. "Roaming honeypots for mitigating service-level denial-of-service attacks," ICDCS, vol. 00, no. , pp. 328-337, 2004.

[8] D. G. Andersen, "Mayday: distributed filtering for internet services," In Proc. USENIX Symposium on Internet Technologies and Systems (USITS), March 2003.

[9] A. Stavrou, A. D. Keromytis, J. Nieh, V. Misra, D. Rubenstein "MOVE: an end-to-end solution to network denial of service," Proceedings of the Internet Society (ISOC) Symposium on Network and Distributed Systems Security (SNDSS). San Diego, CA, February 2005.

[10] A. Keromytis, V. Misra, and D. Rubenstein, "SOS: Secure Overlay Services," In Proceedings of ACM SIGCOMM '02, Pittsburgh, PA, August, 2002.

[11] D. L. Cook, W. G. Morein, A. D. Keromytis, V. Misra, and D. Rubenstein, "WebSOS: protecting web servers from DDoS attacks," In Proceedings of the 11th IEEE International Conference on Networks (ICON), pp. 455 - 460. September/October 2003, Sydney, Australia.

[12] Akamai Edge Computing http://www.akamai.com

[13] J. Train, J. Touch, L. Eggert, Y. Wang, "NetFS: networking through the file system," ISI Technical Report ISI-TR-2003-579.

[14] M. Zec, "Implementing a Clonable Network Stack in the FreeBSD kernel," Proc. USENIX 2003/FREENIX, pp. 137-150.

[15] J. Touch, Y. Wang, V. Pingali, L. Eggert, R. Zhou, G. Finn. "A Global X-Bone for network experiments," Proc. IEEE Tridentcom 2005, Trento Italy, Mar. 2005, pp. 194-203.

[16] P. Francis, "Firebreak: an IP perimeter defense architecture," Webpage http://www.cs.cornell.edu/People/francis/firebreak/

[17] A. Ephremides, J. E. Wieselthier, D. J. Baker, "A design concept for reliable mobile radio networks with frequency hopping signaling," Proceedings of the IEEE , vol.75, no.1pp. 56- 73, Jan. 1987

[18] J. Jones, "Distributed denial of service attacks: defenses, " A Special Publication, Technical Report, Global Integrity, 2000.

[19] N. Fujita, J. Touch, V. Pingali and Y. Wang, "P2P-XBone: a virtual network support for peer-to-peer systems," Technical Report ISI-TR-2005-607, USC/ISI, September 2005.