Botnet Communication Topologies

Understanding the intricacies of botnet Command-and-Control

By Gunter Ollmann, VP of Research, Damballa, Inc.

Introduction

A clear distinction between a bot agent and a common piece of malware lies within a bot's ability to communicate with a Command-and-Control (CnC) infrastructure. CnC allows a bot agent to receive new instructions and malicious capabilities, as dictated by a remote criminal entity. This compromised host then can be used as an unwilling participant in Internet crime as soon as it is linked into a botnet via that same CnC.

The criminals actively controlling botnets must ensure that their CnC infrastructure is sufficiently robust to manage tens-of-thousands of globally scattered bot agents, as well as resist attempts to hijack or shutdown the botnet. Botnet operators have consequently developed a range of technologies and tactics to protect their CnC investment. This paper reviews the tactics commonly employed by botnet operators to maintain control of their botnets and the impact of these tactics on standard network-blocking protection stratagems.

Botnet Topology

Botnets come in all kinds of shapes and sizes. As a result, they employ a range of CnC topologies in response to commercial defenses, legal shutdowns and hijacking attempts. This evolution means that a criminal botnet operator has a number of well-studied CnC topology options to base a new botnet upon – each of which have relative strengths and weaknesses.

Botnet CnC topologies have been optimized to minimize network chatter and system failures, just like commercial-grade technology tasked with remotely managing tens of thousands of hosts. The precise CnC topology selected by a botnet operator often reflects that individual's perceived risk to continued command access and the financial business model of that botnet.

CnC topologies encountered in the wild typically match one of the following types:

- Star
- Multi-server
- Hierarchical
- Random

Star

The Star topology relies upon a single centralized CnC resource to communicate with all bot agents. Each bot agent is issued new instructions directly from the central CnC point.

When a bot agent successfully breaches a victim computer, it is normally preconfigured to "phone home" to this central CnC, whereupon it registers itself as a botnet member and awaits new instructions.

Pros	Cons
Speed of Control The direct communication between the CnC and the bot agent means that instructions (and stolen data) can be transferred rapidly	**Single point of failure** If the central CnC is blocked or otherwise disabled, the botnet is effectively neutered.

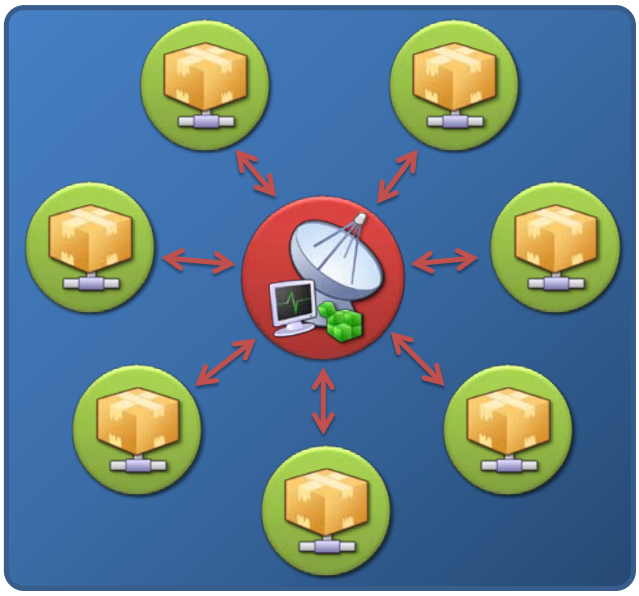

Figure 1: Star CnC topology with direct communications between the central command hub and each bot agent

Multi-Server

Multi-Server CnC topology is a logical extension of the Star topology, in which multiple servers are used to provide CnC instructions to bot agents. These multiple command systems communicate amongst each other as they manage the botnet. Should an individual sever fail or be permanently removed, commands from the remaining servers maintain control of the botnet.

It takes more planning and effort on the part of the botnet's operator to construct a Multi-Server CnC. However the same bot agents can be used for both Star and Multi-Server topologies.

Intelligent distribution of the multiple CnC severs amongst different geographical locations can speed up communications with similarly located bot agents. Likewise, CnC servers simultaneously hosted in multiple countries can make the botnet more resistant to legal shutdown requests.

Pros	Cons
No single point of failure Should any single CnC server be disabled, the botnet operator can still maintain control over all bot agents. **Geographical optimization** Multiple geographically distributed CnC severs can speed up communications between botnet elements.	**Requires advance planning** Additional preparation effort is required to construct a multi-sever CnC infrastructure.

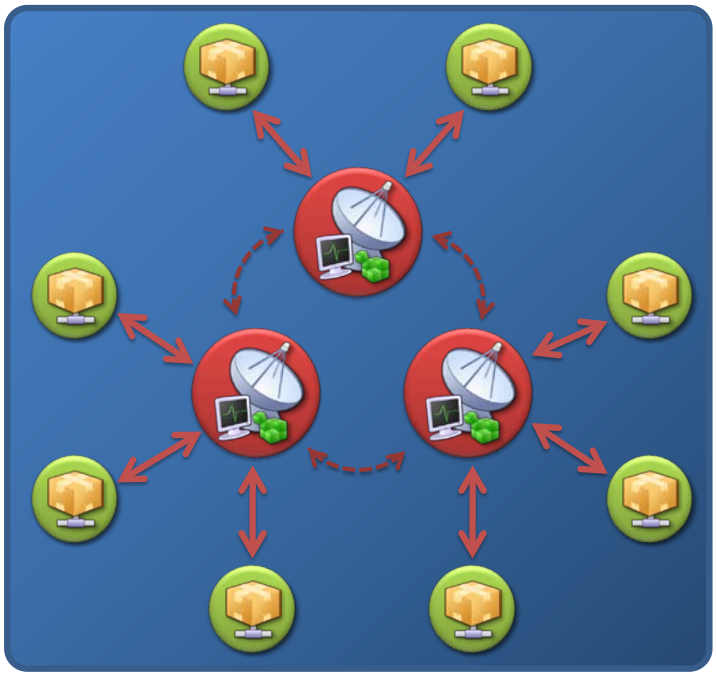

Figure 2: Multi-server CnC topology with direct communications between a distributed cluster of central command servers and each bot agent

Hierarchical

A Hierarchical topology reflects the dynamics of the methods used in the compromise and subsequent propagation of the bot agents themselves. Bot agents have the ability to proxy new CnC instructions to previously propagated progeny agents. However, updated command instructions typically suffer latency issues making it difficult for a botnet operator to use the botnet for real-time activities.

A Hierarchical botnet means that no single bot agent is aware of the location of the entire botnet. This configuration makes it difficult for security researchers to estimate the overall size of the botnet. The hierarchical structure also facilitates carving up larger botnets in to sub-botnets for sale or lease to other botnet operators.

Hierarchical topologies can facilitate a mix of propagation tactics – e.g. an initial drive-by download infection that then initiates worm capabilities once established inside an enterprise network.

Pros	Cons
Botnet awareness Interception or hijacking of bot agents will not enumerate all members of the botnet and is unlikely to reveal the CnC server. **Ease of re-sale** A botnet operator can easily carve off sections of their botnet for lease or resale to other operators.	**Command latency** Because commands must traverse multiple communication branches within the botnet, there can be a high degree of latency with updated instructions being received by bot agents. This delay makes some forms of botnet attack and malicious operation difficult.

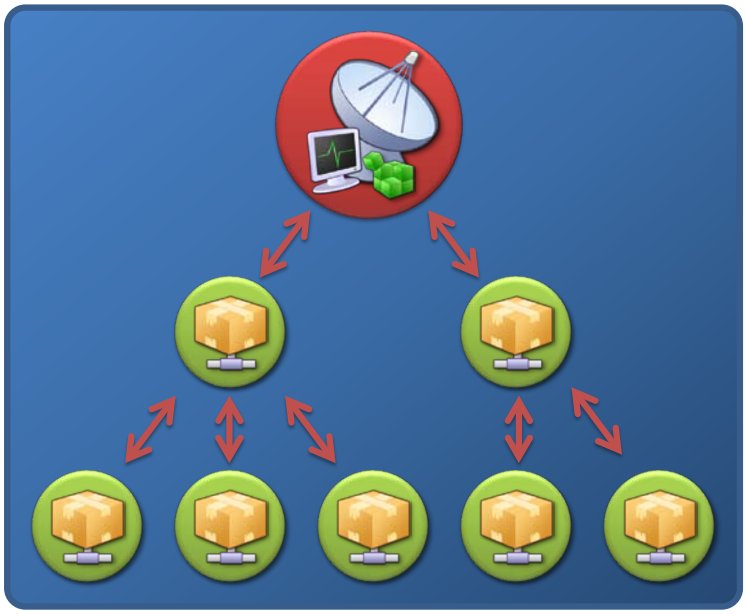

Figure 3: Hierarchical CnC topology with proxied CnC communication

Random

Botnets with a Random topology (i.e., a dynamic master-slave or peer-to-peer relationship) have no centralized CnC infrastructure. Instead, commands are injected in to the botnet via any bot agent. These commands are often "signed" as authoritative, which tells the agent to automatically propagate the commands to all other agents.

Random botnets are highly resilient to shutdown and hijacking because they lack centralized CnC and employ multiple communication paths between bot agents. However, it is often easy to identify members of the botnet by monitoring a single infected host and observing the external hosts it communicates with.

Command latency is a problem for Random topology botnets. However, the multiple communication links between bot agents make latency less of a problem than with Hierarchical topologies.

Pros	Cons
Highly resilient Lack of a centralized CnC infrastructure and the many-to-many communication links between bot agents make it very resilient to shutdown.	**Command latency** The ad hoc nature of links between bot agents make CnC communication unpredictable, which can result in high levels of latency for some clusters of bot agents. **Botnet enumeration** Passive monitoring of communications from a single bot-compromised host can enumerate other members of the botnet.

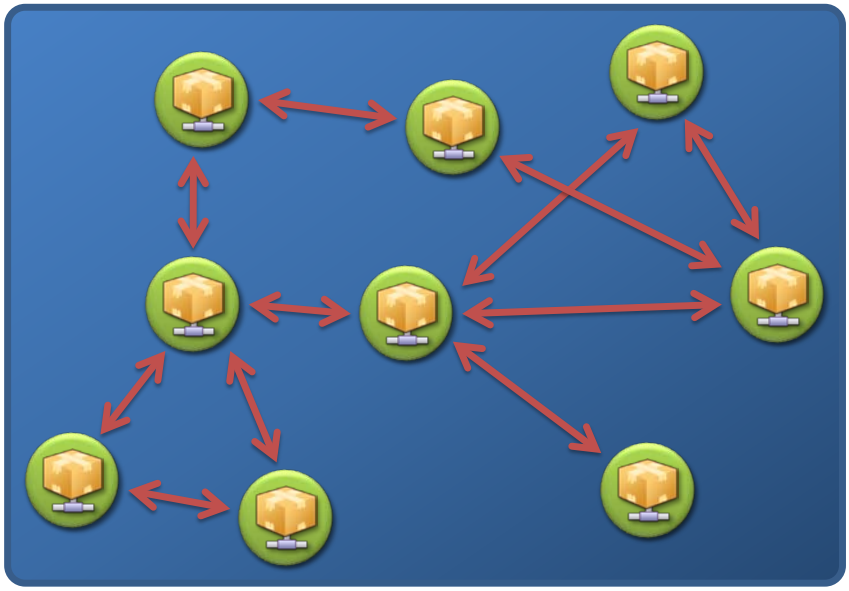

Figure 4: CnC topology with no centralized CnC server infrastructure

Lookup Resilience

The ability for a bot agent to locate CnC infrastructure is a critical requirement for maintaining control of the entire botnet for botnets that rely upon centralized CnC. If the CnC cannot be found, a bot agent will not be able to receive new instructions. While some bot agents may opt to function in an alternative autonomous "zombie" mode – reverting to embedded instructions for propagation and infection – most bot agents will continue to harvest local host information and poll the missing CnC at regularly scheduled times.

Botnet operators use a number of technologies to increase the probability that bot agents will be able to locate the central CnC infrastructure. These tools and techniques also make botnets more resilient to shut-down and hijacking maneuvers.

One key technology that enables CnC location resolution and failover resilience is referred to as "fluxing". Fluxing comes in two major flavors:

- IP Flux
- Domain Flux

Both technologies are used extensively by professional botnet operators.

IP Flux

IP Flux refers to the constant changing of IP address information (e.g. 192.168.1.1) related to a particular, fully-qualified domain name (e.g. mypc.atl.damballa.com). Botnet operators abuse this ability to change IP address information associated with a host name by linking multiple IP addresses with a specific host name and rapidly changing the linked addresses. This rapid changing aspect is more commonly referred to as "fast-flux".

There are two types of fast-flux – "single-flux" and "double-flux".

- ***Single-flux*** is characterized by having multiple (hundreds or even thousands) IP addresses associated with a domain name. These IP addresses are registered and de-registered rapidly – using a combination of round-robin allocation and very short Time-to-live (TTL) values – against a particular DNS Resource Record (i.e., A records).
- ***Double-flux*** is a more advanced evolution of Single-flux. Double-flux not only fluxes the IP addresses associated with the fully-qualified domain name (FQDN), but also fluxes the IP addresses of the DNS servers (e.g., NS records) that are in turn used to lookup the IP addresses of the FQDN.

Domain Flux

Domain flux is effectively the inverse of IP flux and refers to the constant changing and allocation of multiple FQDN's to a single IP address or CnC infrastructure.

Techniques applicable to Domain Flux encompass domain wildcarding and newer domain generation algorithms

- ***Domain Wildcarding*** abuses native DNS functionality to wildcard (e.g., *****) a higher domain such that all FQDN's point to the same IP address. For example, *.damballa.com could encapsulate both mypc.atl.damballa.com and myserver.damballa.com. This technique is most commonly associated with botnets that deliver spam and phishing content – whereby the wildcarded information that appears random (e.g. "asdkjlkwer" of asdkjlkwer.atl.damballa) is

used by the botnet operator to uniquely identify a victim, track success using various delivery techniques, and bypass anti-spam technologies.

- ***Domain Generation Algorithms*** are a more recent addition to bot agents. They create a dynamic list of multiple FQDN's each day, which are then polled by the bot agent as it tries to locate the CnC infrastructure. Since the created domain names are dynamically generated in volume and typically have a life of only a single day, the rapid turnover makes it very difficult to investigate or block every possible domain name.

Blind Proxy Redirection

Both IP Flux and Domain Flux provide advanced levels of redundancy and resilience for the CnC infrastructure of a botnet. However, botnet operators often employ a second layer of abstraction to further increase security and failover – *blind proxy redirection*.

Redirection helps disrupt attempts to trace or shutdown IP Flux service networks. As a result, botnet operators often employ bot agents that proxy both IP/domain lookup requests and CnC traffic. These agents act as redirectors that funnel requests and data to and from other servers under the botnet operator's control. These other servers actually serve the content.

Location Resilience

Most botnets today rely upon DNS as the service for location of CnC infrastructure. Fluxing DNS records provides varying degrees of resilience to shutdown and hijacking that can be best summed up as:

Brittle:	*Single domain*
Less brittle:	*Single flux*
Resilient:	*Double flux*
Very resilient:	*Domain flux*

Conclusion

Understanding the botnet communication topologies that are used by today's criminal operators is a critical component in understanding how to best protect against the overall botnet threat. The topology utilized by the botnet will often dictate the type and degree of actions an enterprise can pursue in either blocking or shutting down a botnet, and the likelihood of success.

Independent of the topology, multiple layers of DNS fluxing and redirection make some botnets highly resilient to shutdown or enumeration. All of these techniques are available to botnet operators. Fortunately, very few botnets employ all of them. As a result, by understanding the nuances of each technique and whether a particular botnet is employing it, enterprise security staff gain critical insight into dealing with the threat.

It is likely an expensive task for a botnet operator to employ all techniques described in this paper – after all, doing so requires a great deal of planning and tuning. That said, some well known botnets do employ all of these techniques successfully, and are generally considered to be stable platforms for the delivery of multiple criminal fraud systems.

While the topology of the botnet CnC greatly influences its resilience to enumeration and eventual shutdown, its architecture may be independent of the location service being used. For example, it may be a centralized HTTP Web server (brittle) or based upon a loose IRC federation model (less brittle). Therefore, locating and shutting down the actual CnC servers (rather than the location services) will effectively cauterize the threat.

The criminals behind botnets are smart and adaptive. It is a safe bet that botnets will increasingly adopt the most advanced permutations of resilient lookup techniques in to the future in order to ensure long-term, stable success. For that reason, Hierarchical or Random topologies will soon replace legacy Star- or Multi-Server based botnets.

Further Reading

http://faculty.cs.tamu.edu/guofei/paper/Dagon_acsac07_botax.pdf

http://us.trendmicro.com/imperia/md/content/us/pdf/threats/securitylibrary/botnettaxonomywhitepapernovember2006.pdf

How Fast-flux Service Networks Work http://www.honeynet.org/node/132

About Damballa, Inc.

Damballa protects businesses from bot-driven targeted attacks used for organized, online crime by using the Internet cloud to identify and isolate threats that evade other technologies. Our unique, global approach monitors the Command-and-Control that coordinates botnet attacks to rapidly identify compromised systems and enable immediate control of malicious activity. Global 1000 corporations, large Internet service providers, OEM partners and government agencies use Damballa's signatureless solutions and industry-leading research to reinforce existing security infrastructure and stop hidden Internet attacks. The result is dramatically improved security both inside and outside the network perimeter. Damballa is privately held and headquartered in Atlanta, Georgia.