SNS COLLEGE OF TECHNOLOGY
(Autonomous)
COIMBATORE – 35

DEPARTMENT OF COMPUTER SIENCE AND ENGINEERING (UG & PG)

Third Year Computer Science and Engineering, 5th Semester

UNIT I – CYBER SCURITY FUNDAMENTALS

Topic Name : Threat Infrastructure

Botnets

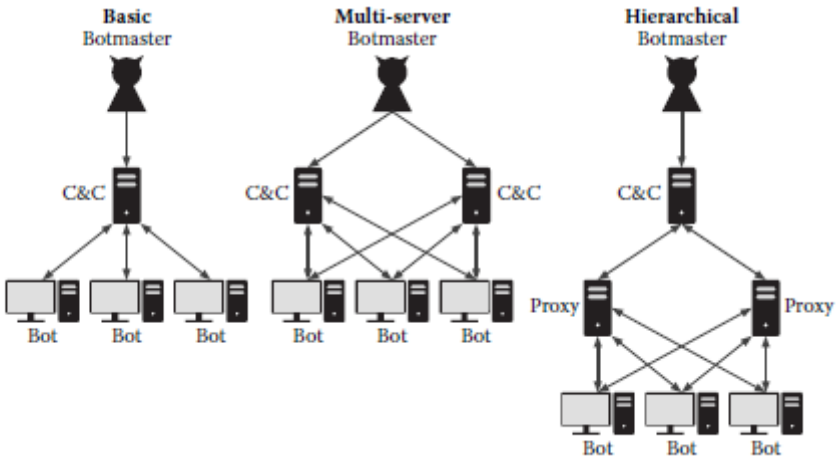
Systems connected to the Internet are at risk of infection from exposure to social-engineering attacks or vulnerability exploitation. Regardless of the infection vector, compromised machines can wait for commands from the attacker, which turns the system into a bot.

A bot is a single node added to a network of other infected systems called a botnet. A botnet is a network of infected systems controlled by an administrator known as a botmaster. A botmaster controls many bots by issuing commands throughout the botnet infrastructure. The ability to run commands on many systems makes botnets practical for malware authors seeking a management solution and provides multiple capabilities.

Botnets would not be capable of performing any activities without communication between the botmaster and bots. The type of communication protocol depends on the network topology of the botnet. While botnets use many different topologies, all botnets fall into two main categories, centralized and decentralized; however, some botnets implement elements from both categories to create a hybrid structure.

A centralized topology receives its name due to the central location of the command-and-control (C&C) server(s). The most basic form of this topology uses a server to C&C all bots within the botnet; however, other more advanced forms of centralized networks exist and fall into two subcategories to describe the differences in infrastructure.

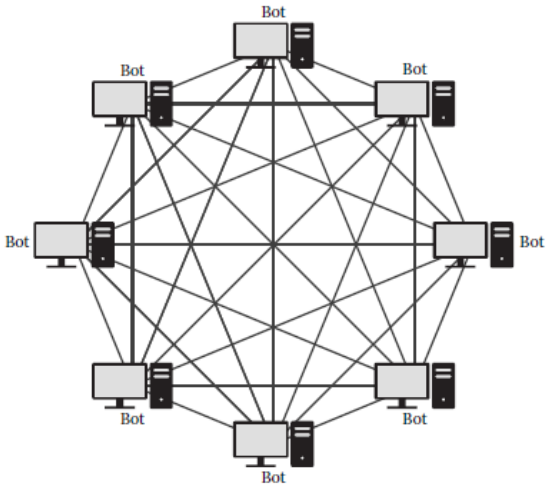
A multiserver builds on the basic centralized botnet topology by using more than one server for C&C. Multiple C&C servers make botnets more reliable and less vulnerable to takedown attempts. This type of topology allows bots to receive commands even if one server is unreachable. If a server goes offline, the botmaster can still communicate with bots through other C&C servers. The As prox botnet was an example of this type of botnet as it issued a configuration file to each bot that included a list of C&C servers.



Centralized botnet infrastructures

Another type of centralized topology uses a hierarchical infrastructure. This type of topology builds on the multiserver technique by using layers of servers to proxy communications between the bots and C&C servers. This setup promotes reliability and longevity within the botnet, as the proxying servers cover the true location of the C&C servers.

The second botnet category, called decentralized , differs dramati-cally from a centralized configuration. A decentralized botnet does not have a particular server or set of servers designated to control bots. These advanced botnets use peer-to-peer (P2P) communications to send commands between bots throughout the botnet. With no centralized location, this type of botnet does not use the same techniques to locate commands as centralized botnets. A bot must locate other peers within the botnet to receive commands by using the P2P protocol's peer discovery mechanisms. This type of botnet is very difficult to dismantle without disinfecting each bot but introduces complexity and latency before all bots receive commands.



Decentralized botnet architecture.

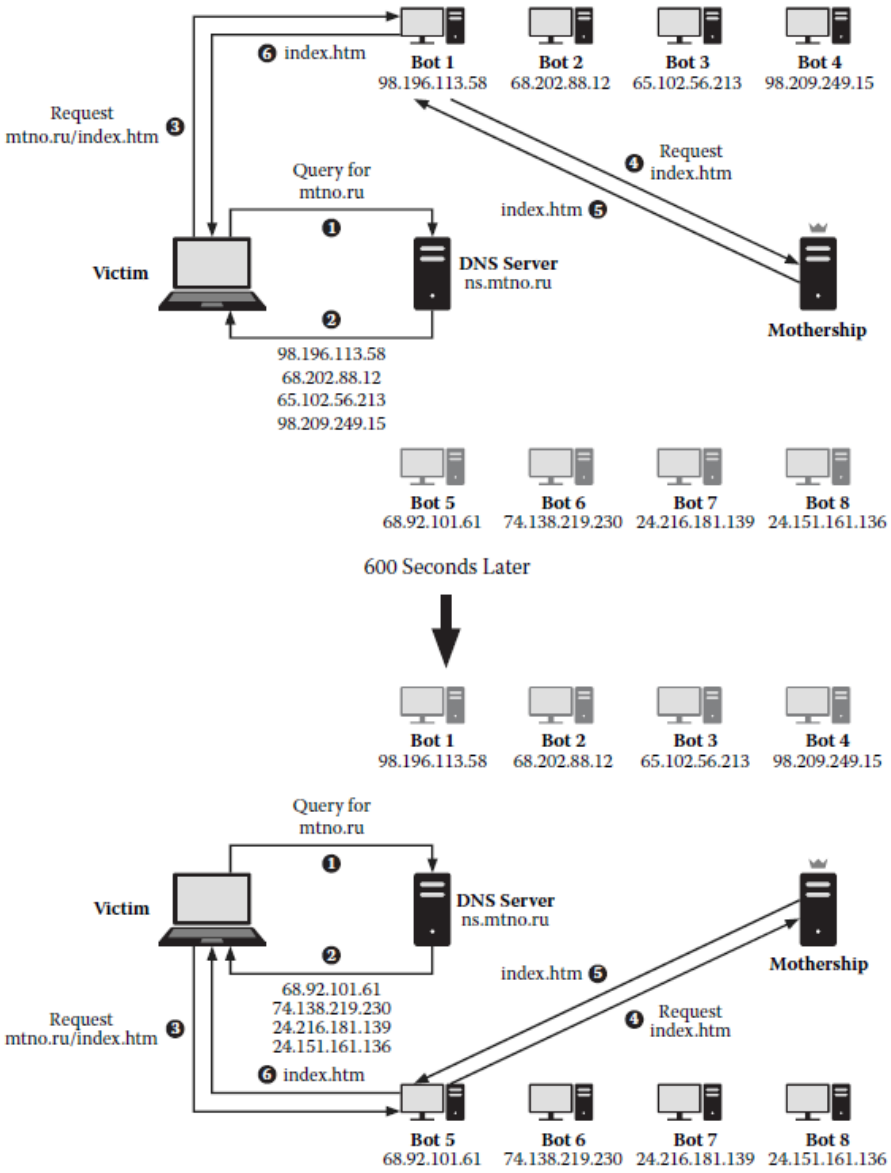
Botnets are very capable at performing malicious deeds for a prolonged period. The most obvious capability available through the control of a large number of infected systems is the capability to carry out distributed denial of service (DDoS) attacks. The botmaster can issue a command to have bots repeatedly send requests to a server or network. With enough bots and enough requests, the botnet can create a denial of service (DoS) condition for the targeted server or network as discussed in the "State of the Hack" article on DoS conditions.

Botmasters can also use their bots to send spam e-mail messages. According to recent research, botnets account for 87.9 percent of all spam sent on the Internet. A botnet can achieve these numbers by having each bot send e-mails at a rapid rate, which can result in millions or billions of e-mails per day. Spam from these systems is difficult to block due to the number of unique systems and the constant addition of new systems.

Fast-Flux

Fast-flux is a technique that creates a resilient and balanced network of compromised computers to carry out malicious actions. Fast flux utilizes DNS to continually update valid domain names with A and NS records that resolve to an ever-changing set of IP addresses. Earning the phrase flux, changing the IP addresses within DNS records allows a domain name to point to different IP addresses at different times. By changing the resolving IP addresses, domains point to the currently active set of infected computers.

Fast-flux domains incorporate a collection of bots into a network of resolvable servers by command-and-control (C&C) servers known as motherships. These motherships have important duties in controlling and maintaining fast-flux by issuing commands to bots and adding and removing bot IP addresses from DNS records. By cycling IP addresses of infected computers in and out of DNS records, the mothership is able to use active bots to host content and services. The IP cycling in DNS records also combats unreachable compromised hosts due to routing issues, firewall filtering, and infection remediation, and insures a high probability of reaching an active bot.



Victim interaction to fast-flux infrastructure

Attributes of a fast-flux domain render it a viable solution to carry out a variety of malicious duties. Phishing campaigns equipped with fast-flux domains remain active for long periods and are difficult to take offline. Exploit toolkits hosted at these domains attempt to exploit vulnerable visitors to install malicious code and ultimately turn the victim into a bot used in the fast-flux system. Malicious code distribution also utilizes fast-flux domains as it allows a centralized location for malicious code downloads without exposure. Spam e-mail also incorporates fast-flux domains to hide mail servers to lengthen the campaign and link to malicious content.

Fast-flux systems are resistant to takedown attempts due to the number of systems involved and the anonymity of the true source of such systems. In a traditional server takedown, an administrator contacts the Internet service provider (ISP) hosting malicious content and provides evidence of abuse. The ISP shuts down the server in response to the report of illegitimate activity. It is impossible to take down a fast-flux system with this traditional process due to the lack of a single ISP to contact. As a result, administrators must contact the domain registrar and provide evidence of malicious content hosted at the domain. The registrar removes access to the fast-flux domain, which stops the current activity, but does not stop the fast-flux operator from registering a new domain name. Adding a new domain name begins the cycle over and allows the fast-flux infrastructure to continue malicious activity.

Fast-flux domains allow actors to carry out malicious deeds anonymously and for relatively long periods. These domains continue to spread malicious code, send spam, host phishing, and exploit victims, and are a danger to any enterprise. Innovative uses of fast-flux continue to change in the wild and require reactive countermeasures from the security community.

Advanced Fast-Flux

The preceding section described fast-flux networks and their general structures, uses, and resiliency to take down. The lack of in-depth detail regarding the types of fast-flux systems requires a second look. The three types of fast-flux existing today are known as single-, double-, and hydra-flux. All three types of fast-flux utilize domain name system (DNS) record updates, occurring on name servers or with domain registrars or both, to conceal the source of malicious activity in attempts to evade detection and takedown. This section describes advanced fast-flux techniques, how they work, and the additional protection each variation provides.

Regardless of type, all fast-flux domains involve a botnet infrastructure that includes one or more C&C servers called motherships and infected computers called bots. The mothership is responsible for managing the DNS infrastructure associated with the domain, controlling bots, and serving malicious content. Managing the domain involves updating the domain registrar and name servers. The registrar receives updates from the mothership in the form of NS records, which point to name servers that answer queries for the domain.

The mothership also updates the configuration file, known as a zone file, on these designated name servers with A records that point to bots that resolve the domain. The zone file on the name server also includes a time-to-live (TTL) value that specifies how many seconds the client caches IP addresses for a domain before querying again. To cycle bot IP addresses into A records and to bypass caching features, fast-flux domains use short TTL values to force clients to frequently query the name server for a new set of A records. Bots designated by A records receive content requests sent to the domain, and act as reverse proxies by sending requests to the mothership and relaying the malicious con-tent hosted on the mothership back to the original requester. These bots provide a layer of protection by obscuring the true source of the malicious content and introduce multiple points for takedown.

Single-, double-, and hydra-flux all use the same techniques to evade detection and takedown, but each addresses weaknesses in its predecessor by adding a layer of complexity. The layers of complex-ity prolong malicious campaigns by obscuring the true source of the activity. Registrar domain takedown procedures are typically drawn out, but provide the most effective solution in stopping malicious activity related to fast-flux. New domains registered for fast-flux reuse the existing infrastructure and restart the takedown process in an endless cycle.