Brief Overview and Analysis of the Storm Worm/Trojan.Peacomm Botnet

Conrad Stansbury + Dennis Wang

October 21, 2014

In early 2007 a new botnet appeared using a private network on the nearly defunct P2P protocol called Overnet, which uses the Kademlia DHT, in order to communicate between bots and to distribute commands. Despite many of the servers being taken down due to copyright disputes from the RIAA, the network still existed because it did not require a centralized set of command servers, making it a good candidate for a botnet communication scheme. Despite a number of agressive attacks made and a fair amount of visibility in the security communities and in the public eye, Trojan.Peacomm grew to a size of at least $600,000$ computers by the end of the year, though C&C distribution and certain other characteristics of the botnet probably mean that this is a fairly inaccurate estimate, and other estimates range as high as 50,000,000 computers[1]. The network was reported in mainstream media as being traceable to Russian organized crime, which is consistent with a possible later sale of parts of the botnet, and use for distributed denial of service attacks as well as spam compaigns which helped to infect additional bots through linkbait style social engineering tactics. Given that the actual operators have not been identified though, it is difficult to say precisely what all the intentions were in use of the botnet, except by analyzing the resultant traffic. The botnet was responsible for a large fraction of the global volume of spam at least until 2008–with many of the addresses probably mined from infected bots–and some researchers claimed that the net was responsible for hundreds of gigabits of sustained traffic. At least some of this traffic was directed against spammer-blacklists and e-mail fraud firms, giving the botnet increased efficiency in its attacks and growth.

Interestingly, one of the functions of the network was to agressively defend against the efforts of academic and commercial researchers, using its ability to laucnh DDoS attacks in order to levy retribution against networks launching probes, and to attack the networks of those who published papers describing and analyzing the botnet.

Given the types of attacks launched by the botnet, including DDoS and large waves of spam, a fair amount can be done to protect against the botnet with intelligent firewalls to reject most of the illegitimate traffic. These firewalls can also help to prevent infections because the primary infection vector appears to have been users opening malicious email distributed by the botnet. If a system

in an administered network or several systems became infected or was believed to be infected, BotHunter might be an appropriate tool to track down the infection, as the tool was updated in the months following the release of the botnet to help analyze and counteract it in larger networks. Once the infected systems were identified, they could be cleaned using tools provided through various AV firms, or reimaged or rolled back to an earlier state after important data was backed up. Additionally, the worm uses a somewhat fragile C&C structure that will be described in more depth later, and which was vulnerable to poising pools of resources used by the botnet, and which appears to have been used to some efficacy in attacking the botnet. Ultimately, the techniques used would have to depend on the scale of infection, and the means available to deal with it. First steps of course should focus on ridding systems of the initial infection by identifying them and reinstalling the OS or removing the infection directly if possible. Stopgap measures to reduce the proability of infection like blacklists/firewalls are a next step while OS vendors prepare patches that prevent code execution.

Of course, patches to the vulnerabilities used by the botnet are the ultimate solution in preventing infection. Unfortunately, Trojan.Peacomm contained update mechanisms that allowed existing infections to remain on systems even after updates, but patches would probably be succesful in preventing infection on a computer with no existing installation of the worm. A number of Windows operating system vulnerabilities allowed the worm to execute merely as a result of the user clicking an infected link, but for the most part this was not a common infection vector, with the worm preferring to have users run a website hosted executable, protected from law enforcement efforts by fast-flux techniques as described in class. Organizations of course should also disable most or all P2P traffic in their networks.

While we were unable to find the exact description of the link clicking vulnerability, [2] describes how the worm infects a host once the executable starts running. The first step for the worm is to decrypt the majority of its executable body, and then to save a potentially decrypted copy of itself as spooldr.exe. Different versions might at this point execute conditional code depending on the existence of debuggers and virtual machines on the host, potentially as a means to avoid analysis on sandboxed machines used by research labs. In any case, once the worm has accepted the system, it modifies the driver tcpip.sys, and creates the driver spooldr.sys which is a rootkit giving the worm control of the system (and notifies the worm of which programs are started for potential hijacking) once the driver is added to the list of startup utilities and programs. Other versions of the worm might write the driver wincom.sys, which is added to services.exe. The worm also disables a number of programs upon initial infection, including AV software and earlier versions of the infection, if they exist on the system. Once the initial infection is in place, the worm operates by periodically listening (repetition rate about 10 minutes) for instructions using eDonkey/Overnet. Upon receipt of such a command, the worm rewrites spooldr.ini and sets socket options to allow communication on the necessary ports for its current job. It then proceeds to either send spam, run a denial of service attack, distribute logic to other connected hosts over eDonkey, or pull

new executables over HTTP and update the infection[2].

The written file spooldr.ini also contains a list of the "adjacent" hosts in the Storm worm C&C, which might change as a result of received messages, especially upon initial infection which contained a hard coded list of about 300 addresses to initially check for responses and instruction on how to integrate the new host into the botnet[2].

So far we have not described the P2P structure of the botnet in much detail. It is believed that the Storm worm used a segment P2P structure so that parts of the botnet are disconnected, making it less susceptible to mapping and attack. Once commands were issued to a single computer in the group, they were distributed to all connected computers in each part of the net, eventually reaching all hosts. The designers of the botnet also chose to prevent infected hosts from sharing files from each other, instead transferring only small resource locators such as URLs or Overnet addresses to further locators, minimizing the C&C footprint to avoid detection and make the P2P structure less susceptible to poisoning of individual bots.

The decentralized nature of Trojan.Peacomm, effective social engineering in order to conquer new hosts, frequent binary updates to avoid AV, as well as its use of HTTP for file transfers, and strong enryption protocols to protect the integrity of its traffic using dynamically generated keys all mean that the botnet proved to be fairly resilient to attack[1,2]. However, by the end of 2007 the botnet was substantially decreased in size, in part due to attacks which aimed to poison parts of the P2P network the botnet used to communicate, and also potentially due to the dismantlement and sale of large swaths of the botnet, with hosts changing botnet binaries and communication, which can be deceptive because it paints a picture of a sucessful response to a well designed and operated botnet for its day. Interestingly, a group of German researchers supposedly discovered a vulernability in the C&C software arising due to weak keys on certain critical machines, allowing them to infiltrate the botnet and uninfect machines. However, this tactic was not used legitimately, for legal reasons. In any case, if other attackers found the same vulnerability, it is possible they could have used it to take over portions of the botnet, which might be a reason for its eventual decline as well, and the existence of such a tool may have forced the operators of the network to transition to a different infection vector and/or botnet structure.

References

[1] Sarat, Sandeep, and Andreas Terzis. "Measuring the Storm Worm Network." *HiNRG Technical Report.* Online.

[2] Porras, Philip et al. "A Multi-perspective Analysis of the Storm (Peacomm) Worm" *CSL Technical Note.* Online.