IPFS and IPNS protocols as way for controller of botnet: a proof of concept

Abstract. *In order to make the internet safer, a fundamental step is to avoid the use of a remotely-controlled infected computer network (Botnet) by a malicious user (Botmaster) which may use it to, among other purposes, perform DDoS attacks. To deal with this problem, a challenge is the evolution of command and control services (C&C) used by the Botmaster for Botnet management, since they are increasingly more sophisticated and harder to detect. A natural evolution of C&C is the usage of new distributed computing protocols. This paper outlines a proof of concept for a C&C using both protocols, IPFS and IPNS, which allowed the Botmaster to acquire safer and more anonymous communication. Additionally it is presented a brief study of detection of executables using such protocols and techniques.*

1. Introduction

There are malware whose purpose is to make the victim's computers remotely-controlled by an attacker. A computer network controlled in this way is called *botnet*, each computer infected is called *bot*, and the controller of this network is called *botmaster*. This network is available for the botmaster by a communication service called *Command-and-Control* (C&C). Thus the botmaster obtains great computing power, which can be used to perform an attack such as *Distributed Denial-of-Service* (DDoS), for example [Upadhyaya et al. 2011].

The C&C is one of the most fundamental parts of a botnet. It needs to ensure the anonymity of the botmaster and to be a secure communication channel, to prevent the malware from being easily detectable, as well as avoiding sybil attack [Wang et al. 2009]. Many protocols, such as IRC, HTTP and DNS [Dietrich et al. 2011], as well as different topologies, such as centralized and distributed, are used to develop different C&C [Bailey et al. 2009].

In this paper, a botnet was developed as proof of concept whose C&C works with two new protocols: *InterPlanetary File System* (IPFS) and *InterPlanetary NameSpace* (IPNS). Then were checked the viability of these protocols, the level of anonymity offered to the botmaster and the security in communication. Next, an analysis was made on how to detect one botnet that uses these protocols.

This paper is organized in the following way. The section 2.1 is a technical discussion about C&C. The section section 2.2 introduces the protocols IPFS and IPNS. The sections 3, 3.1 and 3.2 discusses the implementation of one bot that uses these protocols in C&C, to proof the concept and describe the results obtained in this experiment. At last, the section 4 concludes and describes future works.

2. Theoretical Foundation

This section explains the concepts about a *Command-and-Control* (C&C) and the protocols *InterPlanetary File System* (IPFS) and *InterPlanetary NameSpace* (IPNS).

2.1. Command-and-Control

Many C&C models were proposed and are used in bots, where the oldest type of topology is the centralized, of which one point is responsible for exchanging messages between the bots and the botmaster. Among the mosts commons protocols to use in C&C server are IRC and HTTP. The major advantages of the centralized model is the low latency in communication and the simplicity of design for development [Zeidanloo and Manaf 2009] [Bailey et al. 2009].

But as a disadvantage, the C&C server is one critical point, since all communication is made by it and once it is detected, it may compromise all the system, moreover this topology facilitates the monitoring by third parties, such as researchers and security agents. In order to reduce the risk of the botmaster to lose its communication with its botnet in case of problem in C&C server, the botmaster can employ multiples C&C servers and configure the bot to communicate with it [Zeidanloo and Manaf 2009] [Bailey et al. 2009].

Due to the fragility of the centralized model, botnets with distributed topology began to be developed, also known as Peer-to-Peer (P2P) [Zeidanloo and Manaf 2009]. With this model, it is possible to offer a better level of anonymity to the botmaster [Upadhyaya et al. 2011] as well as resilience for the botnet, since it doesn't have only one point of failure [Zeidanloo and Manaf 2009]. Some disadvantages of P2P over centralized is a higher latency in communication and a complex design for development [Upadhyaya et al. 2011] [Bailey et al. 2009].

One way to detect a bot is by analysing the network traffic of its C&C. Because of this, the attackers always try to find new protocols and ways to communication with the C&C to escape from the security agents [Bailey et al. 2009]. Thereby it is important to study new techniques to develop the C&C, as well as to understand and measure the future risks.

2.2. Protocols *InterPlanetary File System* (IPFS) and *InterPlanetary NameSpace* (IPNS)

The IPFS is a peer-to-peer protocol for distributed file systems [Benet 2015]. The simplest way to access the content in IPFS is by the public gateway [Zumwalt et al. 2017]. It is also possible to run a daemon locally or host your own gateway to obtain access to the IPFS.

Data, such as files, can be added to IPFS, which are called *objects*. Each object added have a hash depending on its content, and will be available through it. For example, if one text file with the content "ipfs" is added to IPFS, then it will be available with the hash "QmbXBAKDgbhE8HkGuEF4FuQQJej2mxqXtYSMsBPuJDqgjq". All objects contained in IPFS is immutable, then, if it's necessary to add a new version of a file, a new object will be created, completely regardless of the previous. Therefore one new hash is

obtained and the previous one will continue to be available. For example, if the text of the file is changed to "sbseg", is to obtain the hash "QmXXhmvSrjYALb8KZwsfXhyz3ugavbaySNHcp6zAbWQ5yj".

The mechanism described in the previous paragraph is due to the concept of *content-addressing* of IPFS. That is, the data in IPFS are addressed by one hash from its content, it ensure more integrity, regardless of where or who offer the content. Moreover, the links are permanent, always pointing to same object [Zumwalt et al. 2017].

Although the concept of content-addressing brings several benefits, it becomes inconvenient in case it always needs to get the last version of something, because it would need to get the hash of last version of the object in another channel. To solve this problem, it was developed the protocol IPNS. With the peerID, the hash of the user's public key, it is possible to create a redirect to a specific object [Benet 2015].

Both protocols are still being developed, but many experiments were performed and have succeeded, such a chat similar to Slack [Orbitdb 2017] and a package manager [Johnson 2017].

3. Proof of concept

In order to proof the concept of using both the IPFS and IPNS protocols in C&C, it was developed a new bot that uses these protocols. To access the files contained in IPFS, the public gateway was chosen, in a way it is not required to run one daemon in the malware to obtain access to IPFS, or running an own gateway in a server. However the public gateway has some limitations, such as the impossibility to add new contents. Despite this limitation, the public gateway is enough to proof the concept, because it allows to receive data, that is, the commands sent by botmaster to the botnet.

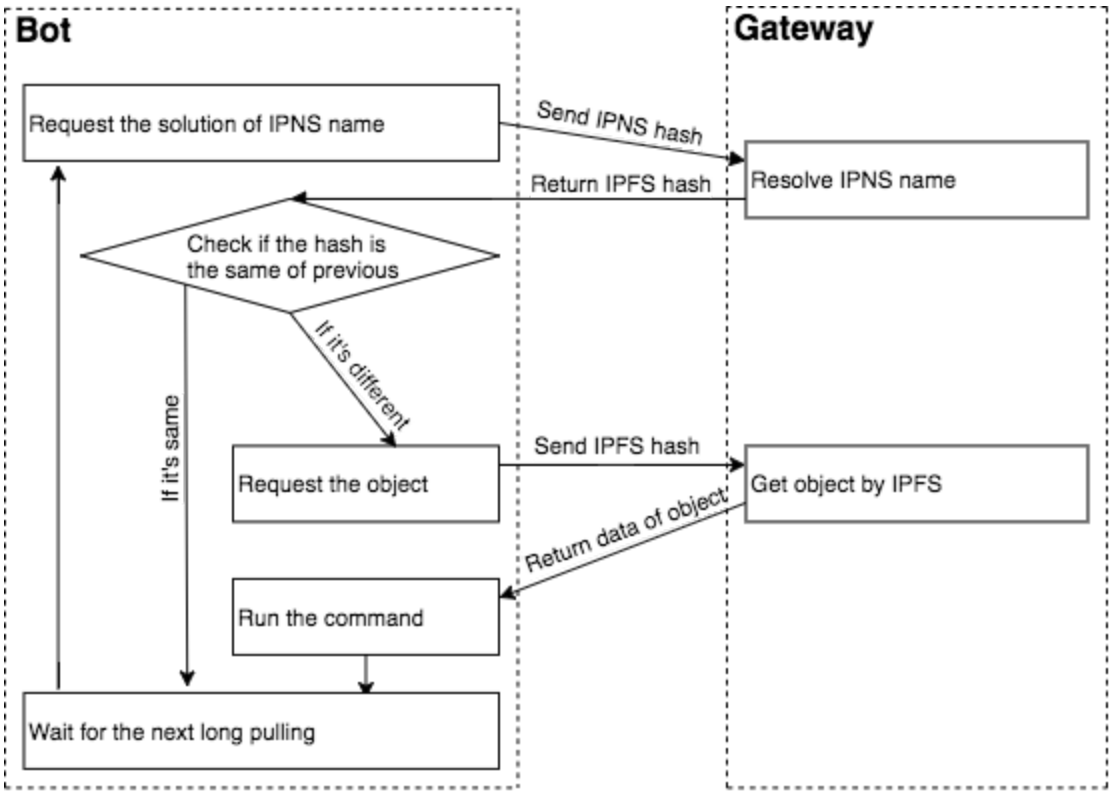

Figure 1. Flow diagram of the execution of the proof of concept.

To validate the hypothesis, it was developed one bot that runs long polling in a specific IPNS hash, whose execution flow is shown in Figure 1. The bot periodically requests to the public gateway the address of the object hash pointed by IPNS, then the gateway sends the hash of the referenced object. The bot then checks if the received hash is different from the previous one in order to, if the difference exists, it requests the content of object associated from hash, that is, the command to run. At the last step, the gateway sends the object pointed by the IPFS hash. The object have the text describing the command to run with the pattern "index command parameter". The field "index" is necessary in case of the botmaster needs that his botnet run twice follow the same command.

Thus, was decided to develop one simple command that the botmaster could run in your botnet: download the file in URL, save it in temp folder and run it.

3.1. Implementation Results Analysis

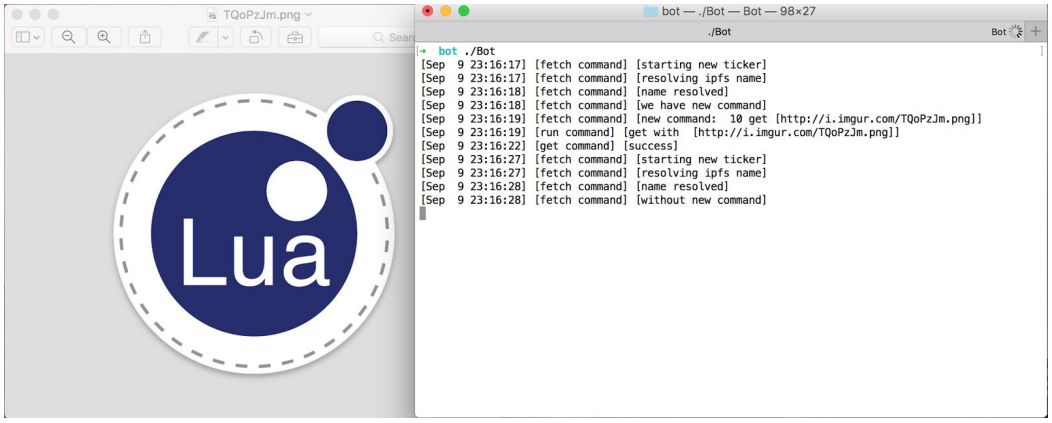

Figure 2. Demonstration of execution of the application.

In performed experiments was possible to send commands to botnet by IPFS and IPNS, as shown in Figure 2. The right side shows the logs of execution, while the left side shows the image that the bot downloaded automatically and opened, using the command sent by the botmaster.

One attribute analysed was the command propagation time from botmaster to the bot. There are two main factor that affect the command propagation time: the interval of checkage of long polling, and the resolution of IPNS name. While the first factor is configurable by the botmaster, the second isn't. Something perceived is that the resolution of IPNS name needs some seconds. According with tests performed, the fastest resolution took 1 second and the average resolution time was 10 seconds. The requests that took 30 seconds or more were abandoned and new request were made to replace them. Currently, the developers of this protocols seek to make this protocol better, aiming to make the resolution of IPNS name faster, consequently facilitating the use of this channel in botnet [Ipfs 2017].

One attribute important in uses of the protocol IPNS is the time that the name is available. When the botmaster add a new object in IPFS and create a redirect of your hash IPNS to the new object, he can shutdown the daemon in your computer. The nodes of IPFS, including the public gateway, will continue to keep the IPNS address because of the cache system but it will be kept for up to 36 hours [Ipfs 2016]. It's useful for maintenance of anonymity of the botmaster but he may need to re-send the IPNS address frequently in order to keep the command available for his botnet.

3.2. Botnet Analyses

One advantage for the botmaster that was observed is the possibility of use IPNS instead of register the domain to uses in C&C, which the botmasters usually uses to facilitate the change of real IP address or server migration. But there is the risk of getting his domain revoked by the domain name register, therefore losing the control of their botnet [Bailey et al. 2009]. With the IPNS, the domain can't be revoked. Moreover, the botmaster can create several peerIDs, this way he will have many differents IPNS addresses to use as channel. In

instead of getting his domain revoked, something similar of it that could happen is the maintainer of a gateway add the botmaster's IPNS hash in a blacklist, however, the IPNS will still be available in others gateway, or through a local daemon, giving the botmaster time to update his IPNS address.

The guaranty that the botnet only run trustworthy commands sent by botmaster is the peerID. In order to another person redirect the botmaster's IPNS to another IPFS hash, it would be needed to copy the botmaster's peerID.

The IPFS and IPNS protocols themselves don't offer greater gain of anonymity, since it is possible to get the IP of who is distributing/receiving some object. But since these protocols are agnostic to the transport layer, is possible to use them with others protocols that aim for a more anonymity gain, like I2P and Tor [Reed 2017]. Moreover, the botmaster can use some tricks to send commands with IPFS and be kept hidden, like using several peerIDs, as well as copying his peerID in differents computers with differents locations, to use it to send new objects and redirect his IPNS address to them. Then, the IPFS will keep this data and share it for a period, without needing the botmaster participation in this process, thus he can gain a higher level of anonymity in the command propagation.

The main approaches used in botnet detection consists in monitoring and analysing the network traffic and using honeypots [Feily et al. 2009]. In this work was used the approaches based in monitoring and analyses the network traffic with the aim to identify information about structure of controller of the botnet and estimate possible countermeasures against botnets that use these protocols.

With the aid of network traffic capturing and analysis tools, it was possible to visualize the requests from the botnet adopted in this work and to identify the format of the commands sent by C&C. This solution was obtained because there have not been adopted measures to hide the command such as the use of polymorphisms or messages encryption.

4. Conclusion

It was possible to develop a proof of concept with a botnet which C&C uses protocols IPFS and IPNS. With it, it was observed a better level of anonymity and security for the botmaster. To fight against botnets that use this protocols, it was described a solution using analyses of the network traffic.

It is still possible to further explore the use of IPFS and IPNS protocols in C&C for bonet, such as not using the public gateway, but running it in one of the bots of botnet, or running the daemon locally on the bots themselves, thus enabling them to share data through IPFS. Furthermore, more studies and analysis are important to better understand the risks and to learn how to detect a botnet with other approaches in the use of IPFS and IPNS protocols.

As it was described, the command propagation time is the time of the next long polling that will be executed plus the time of IPNS resolution. One future work is to compare the command propagation with others botnet topologies and C&C strategies. This is useful for identifying possible signatures in traffic analyses of this botnet methodology.

References

Feily, M., Shahrestani, A., and Ramadass, S. (2009). A survey of botnet and botnet detection. In Emerging Security Information, Systems and Technologies, 2009. SECURWARE '09. Third International Conference on, pages 268 –273.

Bailey, M., Cooke, E., Jahanian, F., Xu, Y. and Karir, M. (2009). A Survey of Botnet Technology and Defenses. In *2009 Cybersecurity Applications & Technology Conference for Homeland Security*. . http://dx.doi.org/10.1109/catch.2009.40.

Benet, J. (2015). IPFS - Content Addressed, Versioned, P2P File System. . https://github.com/ipfs/papers/blob/master/ipfs-cap2pfs/ipfs-p2p-file-system.pdf, [accessed on Aug 6].

Dietrich, C. J., Rossow, C., Freiling, F. C., et al. (2011). On Botnets That Use DNS for Command and Control. In *2011 Seventh European Conference on Computer Network Defense*. . http://dx.doi.org/10.1109/ec2nd.2011.16.

Ipfs (2016). Questions after first learning about IPFS. · Issue #154 · ipfs/faq. https://github.com/ipfs/faq/issues/154, [accessed on Sep 5].

Ipfs (7 jul 2017). namesys/pubsub: pubsub Publisher and Resolver by vyzo · Pull Request #4047 · ipfs/go-ipfs. https://github.com/ipfs/go-ipfs/pull/4047, [accessed on Sep 4].

Johnson, J. (2017). whyrusleeping/gx. https://github.com/whyrusleeping/gx, [accessed on Sep 10].

Orbitdb (2017). orbitdb/orbit. https://github.com/orbitdb/orbit, [accessed on Sep 10].

Reed, J. (9 sep 2017). Privacy and anonymity in IPFS/IPNS. https://discuss.ipfs.io/t/privacy-and-anonymity-in-ipfs-ipns/1068/4, [accessed on Sep 10].

Upadhyaya, A., Jayaswal, D. and Yadav, S. (2011). Botnet: A new network terminology. In *2011 International Conference on Emerging Trends in Networks and Computer Communications (ETNCC)*. . http://dx.doi.org/10.1109/etncc.2011.6255936.

Wang, P., Wu, L., Aslam, B. and Zou, C. C. (2009). A Systematic Study on Peer-to-Peer Botnets. In *2009 Proceedings of 18th International Conference on Computer Communications and Networks*. . http://dx.doi.org/10.1109/icccn.2009.5235360.

Zeidanloo, H. R. and Manaf, A. A. (2009). Botnet Command and Control Mechanisms. In *2009 Second International Conference on Computer and Electrical Engineering*. . http://dx.doi.org/10.1109/iccee.2009.151.

Zumwalt, M., Johnson, J., Benet, J., Gierth, L. and Fisher, L. (2017). *The Decentralized Web Primer*.