Protocolos IPFS e IPNS como meio para o controle de botnet: prova de conceito

Bruno Macabeus M. de Aquino[1], **Marcus Vinicius L. de Lima**[1], **João Paolo Cavalcante M. de Oliveira**[1], **Cidcley Teixeira de Souza**[1]

[1]Instituto Federal de Educação, Ciências e Tecnologia do Ceará (IFCE)
Fortaleza, CE – Brasil

bruno.macabeus@gmail.com, marcsvll@acad.ifce.edu.br,
{paolo.oliveira,cidcley}@ifce.edu.br

Abstract. *To make the internet safer, a fundamental step is to avoid the usage of a remotely-controlled infected computer network (Botnet) by a malicious user (Botmaster) who may use it for, among other purposes, DDoS attacks. To fight against this problem, a challenge is the evolution of command and control services (C&C) used by the Botmaster for Botnet management, which are increasingly more sophisticated and hard to detect. A natural evolution for C&C is the usage of new distributed computing protocols. This paper outlines a new proof of concept C&C using two of these protocols, IPFS and IPNS, which would allow the Botmaster to acquire safer and more anonymous communication.*

Resumo. *Para tornar a internet mais segura, um passo fundamental é combater o uso de uma rede de computadores infectados e remotamente controlados (Botnet) por um usuário malicioso (Botmaster), que pode usá-la para, dentre outros fins, ataques DDoS. Para combater esse problema, um desafio é a evolução dos serviços de comando e controle (C&C) usados pelo Botmaster para gerenciar a sua Botnet, cada vez mais sofisticados e difíceis de se detectar. Uma evolução natural no C&C seria o uso de novos protocolos de computação distribuída. Este artigo apresenta uma prova de conceito de C&C com dois desses protocolos, IPFS e IPNS, que possibilitariam ao Botmaster obter uma comunicação mais segura e anônima.*

1. Introdução

Existe *malware* cujo propósito é tornar os computadores das vítimas remotamente controláveis pelo atacante. Uma rede de computadores controlada dessa forma é comumente conhecida como *botnet*, sendo cada máquina infectada chamada de *bot*, e o controlador dessa rede denominado *botmaster*. A rede é acessível para o *botmaster* por meio de um serviço de comunicação conhecido como *Command-and-Control* (C&C). Utilizando-se de sua *botnet*, o *botmaster* adquire um grande poder computacional, do qual pode usufruir para, por exemplo, realizar um ataque do tipo *Distributed Denial-of-Service* (DDoS) [Upadhyaya et al. 2011].

O C&C é a parte fundamental de uma *botnet*. Ele precisa garantir o anonimato do *botmaster* e ser um meio seguro de comunicação, para evitar que o *malware* seja facilmente detectado, além de evitar ataques *Sybil* [Wang et al. 2009]. Diversos protocolos, como IRC,

HTTP e DNS [Dietrich et al. 2011], assim como diferentes topologias, como centralizada e distribuída, são usadas para desenvolver diferentes C&C [Bailey et al. 2009].

Nesta pesquisa foi desenvolvida, como prova de conceito, uma *botnet* cujo C&C usa dois protocolos recentemente propostos: *InterPlanetary File System* (IPFS) e *InterPlanetary NameSpace* (IPNS), verificando-se a viabilidade destes como canal de C&C.

Este artigo está organizado da seguinte maneira. A seção 2.1 trás uma discussão técnica sobre C&C. A seção 2.2 apresenta os protocolos IPFS e IPNS. As seções 3, 3.1 e 3.2 descrevem e discutem os resultados obtidos na implementação de um *bot* usando esses protocolos em seu C&C. Por fim, a seção 4 conclui e descreve trabalhos futuros.

2. Fundamentação Teórica

Nesta seção são explicados os conceitos relacionados a *Command-and-Control* (C&C) e os protocolos *InterPlanetary File System* (IPFS) e *InterPlanetary NameSpace* (IPNS).

2.1. Command-and-Control

Diversos modelos de C&C foram propostos e são usados em *bots*. O mais antigo é o modelo de topologia centralizado, no qual um único ponto é responsável pela troca de mensagens entre os *bots* e o *botmaster*. As maiores vantagens do modelo centralizado são a baixa latência na comunicação e a simplicidade de projeto. No entanto, há a desvantagem do servidor do C&C ser um ponto crítico, uma vez que toda a comunicação é realizada por meio deste. Caso descoberto, todo o sistema pode ser comprometido. Além disso, o monitoramento por terceiros, como pesquisadores e agentes de segurança, torna-se mais fácil [Zeidanloo and Manaf 2009, Bailey et al. 2009].

Devido a essa fragilidade do modelo centralizado, começaram a ser desenvolvidas *botnets* com topologia distribuída, também chamadas de *Peer-to-Peer* (P2P) [Zeidanloo and Manaf 2009]. Com esse modelo, o *botmaster* pode usufruir de um maior grau de anonimato [Upadhyaya et al. 2011] e prover resiliência para a *botnet*, pois não há apenas um único ponto de falha [Zeidanloo and Manaf 2009].

Uma das formas de se detectar um *bot* é através da análise do tráfego de seu C&C. Por conta disso, os atacantes sempre buscam novos protocolos e formas de se comunicar com o C&C, para assim esquivarem-se dos agentes de segurança [Bailey et al. 2009]. Deste modo, é relevante o estudo de novas técnicas para desenvolvimento do C&C, a fim de compreender e mensurar os riscos futuros.

2.2. Protocolos *InterPlanetary File System* (IPFS) e *InterPlanetary NameSpace* (IPNS)

O IPFS é um protocolo *peer-to-peer* para sistemas de arquivos distribuídos [Benet 2014]. A forma mais simples de acessar o conteúdo do IPFS é por meio do *gateway* público[1]. Também é possível executar um *daemon* localmente ou hospedar seu próprio *gateway* para obter acesso ao IPFS.

Na rede do IPFS pode-se adicionar dados, tais como arquivos, os quais são chamados de *objects*. Cada *object* é referenciado por meio de uma *hash* calculada a partir de

[1] `https://gitbook.com/book/flyingzumwalt/decentralized-web-primer`

seu conteúdo. Todo *object* contido no IPFS é imutável, então se for necessário adicionar uma nova versão de um determinado arquivo, deve ser criado um novo *object*, totalmente independente do anterior. Desse modo, uma nova *hash* é obtida para a nova versão do arquivo, mas a anterior também continua acessível.

Tal conceito é chamado de *content-addressing* e tem como objetivo garantir maior integridade, independente de onde ou de quem esteja oferecendo o conteúdo. Também devido a esse conceito, os *links* são permanentes, sempre apontando para o mesmo *object* [Benet 2014]. Porém, essa abordagem torna-se inconveniente caso o dado seja frequentemente atualizado, pois seria necessário obter a *hash* da última versão do *object* a partir de outro canal. Para resolver esse problema, foi desenvolvido o protocolo IPNS, que permite fornecer um redirecionamento para um determinado *object* a partir de uma *peerID* (*hash* da chave pública de um usuário) [Benet 2014]. Deste modo, pode-se referenciar uma única *hash* que sempre aponta para a *hash* do *object* mais novo.

3. Prova de Conceito

Como prova de conceito, desenvolvemos[2] um *bot* que utiliza os protocolos IPFS e IPNS como C&C. Dentre as diferentes formas de acessar os arquivos contidos no IPFS, optamos por usar o *gateway* público, para não precisar instanciar no próprio *malware* um *daemon* para obter acesso ao IPFS, nem executar um *gateway* próprio em outro servidor. O *gateway* público trás algumas limitações, como a impossibilidade de adicionar *objects* a partir do *bot*. No entanto, ele é suficiente para a prova de conceito, pois permite receber dados – neste caso, os comandos emitidos pelo *botmaster* para a sua *botnet*.

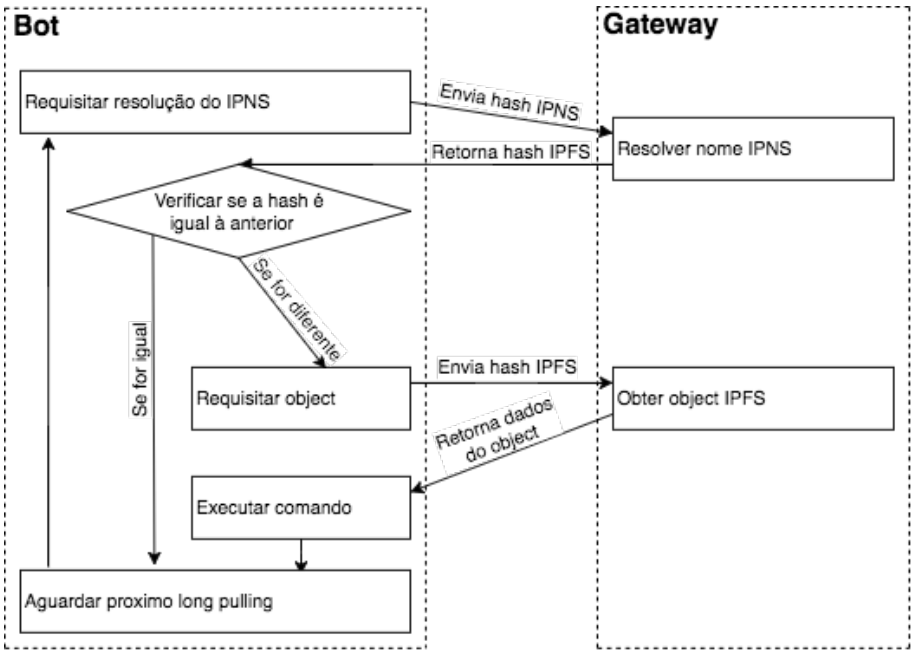

Figura 1. Diagrama do fluxo de execução da prova de conceito.

A Figura 1 apresenta o fluxo de execução do *bot* proposto neste trabalho. Periodicamente (*long polling*), o *bot* requisita uma *hash* endereçada por IPNS ao *gateway* público,

[2] `https://github.com/macabeus/ipfsbot`

que responde enviando a *hash* do *object* apontado. O *bot* verifica, então, se a *hash* recebida é diferente da obtida em requisições anteriores para, se for diferente, requisitar o conteúdo do *object* associado a essa *hash*. Por fim, o *gateway* responde com o conteúdo associado à *hash*, que consiste no comando emitido pelo *botmaster*. Decidimos desenvolver um único tipo de comando, simples mas útil para uma *botnet*: efetuar o *download* do arquivo de uma URL, armazená-lo no diretório de arquivos temporários e executá-lo.

3.1. Resultados

Nos experimentos efetuados, obtivemos sucesso em enviar comandos para a *botnet* através dos protocolos IPFS e IPNS.

Uma das características analisadas foi o tempo de propagação de comandos do *botmaster* até o *bot*. Há dois fatores principais que influenciam nesse tempo: o intervalo de checagem do *long polling* e a resolução do nomes do IPNS. Ao passo que o primeiro fator pode ser configurado pelo *botmaster*, o segundo foge de seu controle. Percebemos que a resolução de nomes do IPNS costuma demandar alguns segundos. Nos testes efetuados, a resolução levou no mínimo 1 segundo e, em média, cerca de 10 segundos. No entanto, algumas vezes a resolução demorava mais que 30 segundos, fato que levou a uma refatoração do *bot* para, nesses casos, desistir da resolução e efetuar uma nova tentativa.

Outro fato observado foi que, após inserir um novo *object* no IPFS e efetuar seu redirecionamento para uma *hash* IPNS, mesmo que o *botmaster* fosse desconectado da rede, os nós do IPFS (inclusive do *gateway* público) continuavam capazes de acessar o endereço IPNS. Esse fato ocorre devido ao sistema de cache, que pode armazenar o conteúdo por até 36 horas[3]. Um *botmaster* poderia aproveitar-se disso para aumentar seu anonimato, desde que republicasse o seu endereço IPNS de tempos em tempos para que o comando emitido continuasse acessível para a *botnet*.

3.2. Discussão

Uma das vantagens do IPNS para o *botmaster* está relacionada ao registro de domínios para o C&C. Os *botmasters* costumam registrar domínios para lidar mais facilmente com migrações ou mudanças no endereço IP de servidores. No entanto, correm o risco de serem descredenciados pela registradora, perdendo completamente o controle de sua *botnet* [Bailey et al. 2009]. Com o IPNS, não existe descredenciamento: o mais próximo seria o mantenedor de um *gateway* adicionar a *hash* IPNS do *botmaster* a uma *blacklist*, entretanto, esta ainda poderia ser acessada a partir de outro *gateway*, ou por meio de um *daemon* local. Além disso, criar várias *peerID* não implica em custo adicional para o *botmaster*, permitindo que este possua diversos endereços IPNS para usar como canal.

O IPNS funciona naturalmente como um mecanismo de assinatura, garantindo que a *botnet* executará somente os comandos legítimos emitidos pelo *botmaster*. Um adversário precisaria gerar uma assinatura válida de posse somente da chave pública (*peerID*) para ser capaz redirecionar o endereço IPNS do *botmaster* para outro *object* no IPFS.

Os protocolos IPFS e IPNS não fornecem, por eles mesmos, um maior ganho de anonimato, uma vez que é possível obter o IP de quem estiver distribuindo ou recebendo determinado *object*. Porém, como esses protocolos são agnósticos à camada de transporte,

[3] `https://github.com/ipfs/faq/issues/154`

é possível usá-los em conjunto com outros protocolos que visam o anonimato, como I2P e Tor[4]. Além disso, o *botmaster* poderia usar alguns artifícios para emitir comandos com o IPFS e manter-se oculto, como usar diversas chaves *peerID*, assim como copiar o *peerID* em diferentes computadores de diferentes localidades, usando-os para adicionar novos *objects* e redirecionar seu endereço IPNS para estes. Desse modo, o IPFS se responsabilizaria por manter e distribuir o arquivo durante um período, sem precisar da manutenção constante da comunicação com o *botmaster* nesse processo.

3.3. Conclusão

Este trabalho apresentou uma prova de conceito de uma *botnet* cujo C&C utiliza os protocolos IPFS e IPNS. Na arquitetura apresentada, obtém-se um tempo de propagação de comandos correspondente ao intervalo do *long polling* somado ao tempo de resolução do IPNS.

Trabalhos futuros poderiam explorar mais diretamente os protocolos IPFS e IPNS. Por exemplo, o *gateway* poderia ser integrado a alguns dos *bots* da *botnet*, dispensando o uso do *gateway* público. Outra opção seria executar o *daemon* dos protocolos em todos os *bots*. Ambas as alternativas permitiriam que os *bots* enviassem o retorno dos comandos por meio dos próprios protocolos IPFS e IPNS.

Devido ao potencial que os protocolos IPFS e IPNS apresentam para o uso em *botnets*, é importante que mais estudos e análises sejam realizadas, permitindo formular métodos de detecção e ampliando a compreensão a respeito dos riscos envolvidos.

Referências

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*, pages 299–304.

Benet, J. (2014). IPFS – content addressed, versioned, P2P file system. http://arxiv.org/abs/1407.3561.

Dietrich, C. J., Rossow, C., Freiling, F. C., Bos, H., v. Steen, M., and Pohlmann, N. (2011). On botnets that use dns for command and control. In *2011 Seventh European Conference on Computer Network Defense*, pages 9–16.

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)*, pages 424–428.

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*, pages 1–8.

Zeidanloo, H. R. and Manaf, A. A. (2009). Botnet command and control mechanisms. In *2009 Second International Conference on Computer and Electrical Engineering*, volume 1, pages 564–568.

[4]`https://discuss.ipfs.io/t/privacy-and-anonymity-in-ipfs-ipns/1068/4`