Identificação Automática de Servidores C&C e Identificação de Variantes de *Malwares* Bashlite e Mirai

Gabriel Bastos[1] **Artur Marzano**[1] **Osvaldo Fonseca**[1] **Ítalo Cunha**[1]
Elverton Fazzion[2,1] **Marcelo H. P. C. Chaves**[3] **Cristine Hoepers**[3]
Klaus Steding-Jessen[3] **Dorgival Guedes**[1] **Wagner Meira Jr.**[1]

[1]Departamento de Ciência da Computação
Universidade Federal de Minas Gerais

[2]Departamento de Computação
Universidade Federal de São João del-Rei

[3]CERT.br – Centro de Estudos, Resposta e Tratamento de Incidentes de Segurança
NIC.br – Núcleo de Informação e Coordenação do ponto BR

{bastos.gabriel,artur.marzano,osvaldo.morais,cunha,dorgival,meira}@dcc.ufmg.br

{mhp,cristine,jessen}@cert.br fazzion@ufsj.edu.br

Abstract. *The Internet of Things is composed of a large number of devices distributed around the world, and the weak security guarantees of many of these devices has been explored by malicious agents to build botnets. The impact of these botnets may be reduced by enabling security researchers to block access to the Command and Control servers. In this paper, we extend and integrate existing methods and tools into a framework for identifying C&C servers and grouping similar malware. We use static and dynamic analysis in combination with heuristics to infer C&C addresses, and clustering algorithms for grouping binaries by similarity. In our results, the proposed analyses and heuristics extend the identification of C&Cs by mitigating countermeasures implemented by malware developers, while the clustering algorithm is able to group the binaries into meaningful groups, directing the efforts of security researchers.*

Resumo. *A Internet das Coisas é composta de inúmeros dispositivos distribuídos ao redor do mundo. O baixo padrão de segurança por parte desses dispositivos tem sido explorado por agentes maliciosos para construir* botnets. *O impacto dessas* botnets *pode ser reduzido ao facilitar que analistas de segurança bloqueiem o acesso aos servidores de Comando e Controle. Neste artigo, integramos e estendemos ferramentas existentes em um arcabouço para a identificação de servidores C&C e classificação de* malwares *em grupos similares. Utilizamos análise estática e dinâmica em combinação com heurísticas para identificar servidores C&C, e algoritmos de agrupamento para classificar binários. Em nossos resultados, as análises e heurísticas propostas melhoram a identificação de C&Cs, mitigando contramedidas implementadas por desenvolvedores de* malware*, enquanto o algoritmo de agrupamento consegue classificar os binários em grupos significativos, direcionando os esforços de analistas de segurança.*

1. Introdução

A Internet das Coisas (IoT) é uma rede de dispositivos físicos conectados à Internet, como câmeras de segurança e sistemas veiculares. A utilização de dispositivos IoT em diferentes aplicações apresenta oportunidades de desenvolvimento econômico e tecnológico em vários setores da sociedade. Porém, dispositivos IoT possuem baixo custo, o que compromete investimentos em segurança, culminando em vulnerabilidades. Esse problema é agravado pelo fato do *software* ser embarcado, e usuários raramente fazerem atualizações. Na prática, dispositivos IoT introduzem novas possibilidades de abuso à medida que são integrados à rede [Angrishi 2017, Kolias et al. 2017]. Agentes maliciosos exploram vulnerabilidades nestes dispositivos para infectá-los e formar *botnets* IoT. Apesar do poder computacional de cada dispositivo IoT infectado (*bot*) ser pequeno, uma *botnet* IoT pode coordenar milhares de bots na realização de alguma atividade maliciosa, como ataques distribuídos de negação de serviço (DDoS).

Ataques DDoS aumentaram em frequência e intensidade, com serviços sendo atacados diariamente e ataques gerando tráfego na ordem de 1 Tbps [Symantec 2017]. Os prejuízos gerados às empresas atacadas são da ordem de bilhões de dólares, pois esses ataques esgotam recursos como processamento e banda, inclusive de serviços bem provisionados, causando indisponibilidade [Neustar 2017]. O grande número de dispositivos infectados e sua distribuição topológica na Internet permitem que *botnets* IoT realizem ataques de grande volume e de difícil mitigação.

Duas famílias de *malware* IoT, Bashlite e Mirai, ganharam notoriedade após *botnets* executando estes *malwares* realizarem ataques DDoS de 400 Gbps e 1 Tbps, respectivamente [Angrishi 2017, Symantec 2017], com amplo impacto em serviços de grande porte, como o DynDNS. O código fonte de ambos *malwares* estão disponíveis na internet, e isso contribui com o surgimento de um grande número de variantes. Variantes exploram outras vulnerabilidades, incluem novas formas de ataque e utilizam diferentes mecanismos para contornar formas de defesa existentes. O crescente número de variantes desses *malwares* torna dispendioso o processo de análise e engenharia reversa para criação de novas contramedidas, mecanismos de monitoramento e defesa contra ataques. Para lidar com o crescente número de ameaças, analistas de segurança e pesquisadores precisam de ferramentas que automatizem o processo de coleta, análise e engenharia reversa de *malware*.

O servidor C&C é o componente central de uma *botnet*, sendo o responsável por coordenar os *bots* [Antonakakis et al. 2017]. Considerando que seu comprometimento torna a *botnet* inócua, analistas de segurança e operadores de rede dedicam esforços à identificação e bloqueio de servidores C&C. De forma similar, os autores dos *malwares* implementam funcionalidades para dificultar a identificação e o bloqueio de servidores C&C, tornando *botnets* mais furtivas e resilientes. Como exemplo dessas sofisticações, os autores dos *malwares* adicionam ao código funcionalidades para ofuscar a identidade do C&C, e incorporam novas vulnerabilidades encontradas em sistemas.

Neste artigo, estendemos ferramentas existentes para permitir análise automática de *malwares* IoT das famílias Bashlite e Mirai para identificação de servidores C&C. Aprimoramos a ferramenta Detux[1], um *sandbox* para avaliação de *malware*, com meca-

[1] http://detux.org

nismos que melhoram o isolamento entre múltiplas execuções de *malwares*, bem como entre os *malwares* e a Internet. Criamos um módulo que analisa a comunicação de rede realizada pelo *malware* para inferir possíveis servidores C&C, usando quatro heurísticas distintas. Por fim, desenvolvemos clientes que estabelecem conexões com os servidores C&C candidatos e identificam se a comunicação segue o padrão dos protocolos utilizados por *botnets* Bashlite e Mirai.

Propomos também um processo para identificação de variantes de cada família de *malware*. O processo proposto utiliza o Radare2,[2] uma ferramenta para engenharia reversa de binários, para contornar a ausência de metadados nos binários de *malware* coletados e identificar funções. Calculamos uma métrica de distância entre binários utilizando um *hash fuzzy*, que nos permite comparar a similaridade entre funções, e executamos um algoritmo de agrupamento hierárquico para agrupar binários por variante.

Nosso estudo analisa *malwares* IoT coletados por 47 *honeypots* de baixa interatividade distribuídos em 15 estados do Brasil. Apresentamos resultados de análise estática para um conjunto de 25.183 binários coletados entre $1/2017$ e $12/2018$, demonstrando a evolução da utilização de mecanismos anti-análise por variantes do Bashlite e do Mirai. Apresentamos resultados para análise dinâmica de uma amostra de 521 *malwares* coletados entre $27/11/2018$ e $23/12/2018$, onde conseguimos inferir C&Cs candidatos e validá-los para 94% e 59% dos *malwares*, respectivamente. Comparadas a uma abordagem para identificação de servidores C&C baseada apenas em análise estática, nossas heurísticas baseadas em análise dinâmica permitem uma detecção de servidores C&C 31% superior. Por fim, o algoritmo de agrupamento utilizado consegue identificar diferentes versões de uma variante de *malware* Bashlite e Mirai, ao mesmo tempo que isola diferentes variantes, reduzindo o número de binários que analistas de segurança e pesquisadores precisam analisar em 43%.

O arcabouço proposto neste artigo automatiza a identificação de servidores C&C, agilizando o processo de mitigação e combate a *botnets*, e reduz a quantidade de binários que analistas de segurança e pesquisadores precisam analisar para diferentes variantes de *malwares* Bashlite e Mirai. Acreditamos que estas contribuições são mais um passo no combate a estas ameaças.

2. Conjunto de dados

A base de *malwares* utilizada nesse trabalho foi coletada por uma infraestrutura de monitoramento passiva, que recebe e cataloga tentativas de infeção de dispositivos infectados (*bots*). Esta infraestrutura de coleta é composta por 47 *honeypots* de baixa interatividade, que emulam serviços SSH e telnet acessíveis por credenciais conhecidas de dispositivos IoT vulneráveis, comumente abusados por *malwares* das famílias Bashlite e Mirai.

Os *honeypots* capturam as tentativas de autenticação (credenciais utilizadas para login) e a sequência de comandos executados durante a tentativa de infecção. Nenhum comando é executado pelos *honeypots*, que apenas emulam os comandos e enviam as respostas esperadas ao atacante. Isso é possível por que o processo de infecção é automatizado, e utiliza um conjunto de programas conhecidos durante a tentativa de infecção.

Os comandos executados durante tentativas de infecção são armazenados e envia-

[2]https://rada.re

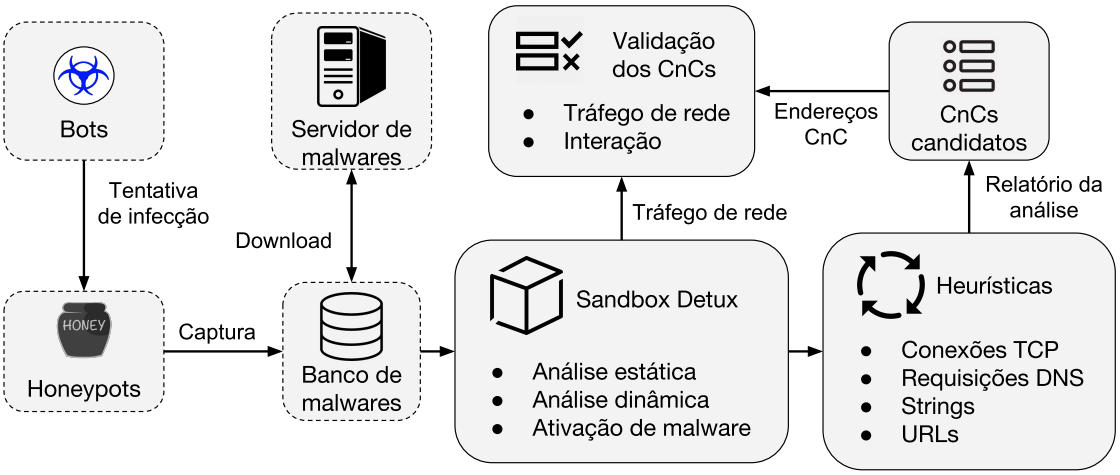

Figura 1. Fluxograma do processo de coleta e análise de *malwares*.

dos para um servidor que identifica tentativas de baixar *malwares* (e.g., chamadas a programas como wget, curl e ftp). As URLs alvo são baixadas dos *servidores de malware*, e os arquivos armazenados em um *banco de malwares* para análise posterior. Nesse trabalho consideramos os *malwares* coletados entre janeiro de 2017 e dezembro de 2018. Ao todo, coletamos 28.004 amostras, sendo 25.183 binários ELF, dos quais 20% foram compilados para processadores MIPS 32 *bits* MSB, arquitetura comumente utilizada em dispositivos IoT.

Neste artigo, estendemos a infraestrutura de monitoramento com módulos para análise automática dos *malwares*, a fim de identificar e validar servidores C&Cs automaticamente (blocos com borda sólida, figura 1). A análise dos *malwares*, desde a identificação e *download* dos servidores de *malware* até a validação dos servidores de C&C, é realizada em até quatro dias. Para trabalhos futuros, estamos trabalhando na infraestrutura para melhorar a integração e reduzir este intervalo para 12 horas. Quanto menor o período de análise, menor o tempo de resposta, e maior a chance de validar um C&C antes que seja desativado ou bloqueado.

3. Análises para identificação de servidores C&C

Para identificar servidores C&C, utilizamos análises estáticas especializadas, que nos permitem detectar e anular contramedidas utilizadas por *malwares* Bahslite e Mirai para mitigar técnicas de análise dinâmica (§3.1). Executamos os *malwares* no *sandbox* Detux para monitorar o comportamento do *malware* num ambiente controlado e realista (§3.2). Propomos e executamos quatro heurísticas para inferir endereços IP que possivelmente hospedam servidores C&C, a partir da observação do comportamento dinâmico do *malware*, particularmente das conexões e tráfego de rede (§3.3). Por fim, executamos ferramentas para validar as inferências e identificar servidores C&C (§3.4).

3.1. Análise estática dos *malwares*

A primeira parte da análise estática consiste na extração de dados dos binários ELF (*Executable and Linkable Format*) realizada pelo Detux. A análise estática contém dados como os cabeçalhos do arquivo e do programa, seções, símbolos de depuração e

sequências de caracteres (*strings*). Utilizamos a tabela de símbolos para identificar os binários compilados com e sem informações de depuração (*stripped* e não-*stripped*). Além disso, armazenamos as sequências de caracteres que correspondem a endereços IP contidas em cada binário. Estas informações coletadas são utilizadas nas próximas etapas da análise.

A segunda parte da análise estática visa contornar mecanismos utilizados pelos atacantes para ludibriar análises dinâmicas. Verificamos a ausência de requisições DNS em diversas execuções de *malwares* Mirai, comportamento atípico para essa família de *malware*, que normalmente identifica seus C&Cs por um nome de domínio. Através de inspeção manual do código fonte, identificamos a presença de um mecanismo de ativação do *malware*, que verifica se o comando de execução do binário (`argv[0]`) é idêntico a um valor pré-definido no código (chave de ativação). Binários com este mecanismo de ativação se comportam da forma esperada (realizando varredura de rede e contactando o servidor C&C) apenas quando a verificação é realizada com sucesso, executando uma rotina alternativa em caso de falha na verificação. A rotina alternativa contacta um servidor C&C falso, impedindo a identificação do C&C real e potencialmente direcionando esforços de mitigação ao alvo incorreto.

Para dificultar a decodificação da chave de ativação, os bytes que a compõe são codificados de forma específica. Sejam $B_1, B_2, \ldots, B_N$ os bytes que representam a chave de ativação, sendo B_N o byte nulo ($0x00$) que finaliza a string. Ao serem codificados no binário, para cada par de bytes consecutivos, a ordem destes é trocada e um byte com valor zero é inserido. Assim, a chave de ativação seria codificada no binário como $B_2|B_1|0x00|B_4|B_3|0x00|\ldots|0x00|B_{N-1}|0x00$.

Além de decodificar a chave de ativação, sua posição no binário pode variar em função de modificações realizadas em diferentes variantes do *malware*. Assim, antes de executar cada um dos binários, procuramos na seção de dados somente leitura (`rodata`) pelo padrão de codificação descrito anteriormente. Consideramos apenas chaves de ativação com pelo menos 3 caracteres ASCII, começando com os caracteres "`./`" (para execução do binário no diretório corrente).[3]

3.2. Análise dinâmica no Detux

O Detux é uma ferramenta que executa o *malware* em uma máquina virtual QEMU (provendo isolamento e um ambiente controlado, *sandbox*) com sistema operacional Debian MIPS, simulando um ambiente típico IoT. A ferramenta monitora e coleta dados sobre a composição e o comportamento do *malware*, capturando todo o tráfego de rede da execução. O Detux gera um relatório incluindo informações como as conexões IP realizadas, as portas utilizadas para cada conexão, e as requisições DNS enviadas.

Para evitar que a execução dos *malwares* gere um impacto negativo na Internet, adotamos um conjunto de medidas protetivas. Limitamos a execução de cada *malware* a 90 segundos, e limitamos o tráfego gerado em 10 kbps, evitando que a iteração do *malware* com a Internet contribua com ataques DDoS. Além disso, bloqueamos conexões às portas 23 e 2323, utilizadas no processo de varredura de dispositivos vulneráveis, de forma a não contribuir com a descoberta e infecção de dispositivos.

[3] Avaliamos definições mais genéricas de chaves de ativação, sem alteração nos resultados. Também observamos que as chaves de ativação em nossa amostra sempre começam com os caracteres "`./`".

Para que a execução de *malwares* no Detux forneça informações válidas, é necessário que o *malware* se comporte da mesma forma que ele se portaria em uma situação real de infecção. Implementamos nosso mecanismo de detecção de chaves de ativação (seção 3.1) como uma fase de pré-processamento no Detux, e estendemos a ferramenta para executar o binário com cada uma das possíveis chaves de ativação identificadas.

Identificamos também que a execução consecutiva de *malwares* no Detux leva a relatórios contaminados. Em particular, a retransmissão de pacotes TCP relativos a uma conexão de uma execução anterior é capturada como tráfego de uma execução posterior. Aplicamos o seguinte método para filtrar pacotes TCP que não são relativos à execução corrente. Durante a execução de cada *malware*, criamos e gerenciamos um banco de conexões com os identificadores de fluxo (*5-tuple*) de cada tentativa de conexão realizada. Identificamos uma tentativa de conexão através de pacotes TCP originados pelo *malware* contendo a flag SYN. Todo pacote TCP *recebido* que *não* possui uma entrada correspondente no banco de conexões da execução atual é ignorado em nossas análises.

3.3. Heurísticas para inferência de servidores C&C

O número de endereços IP observados durante a execução de um *malware* no Detux pode ser muito grande, principalmente devido ao processo de varredura realizado pelos *malwares*. Dessa forma, contactar todos eles na tentativa de encontrar C&Cs pode ser inviável, e além disso, impactar a Internet negativamente ao gerar requisições a dispositivos legítimos. Portanto, propusemos quatro heurísticas para inferir os endereços IP de prováveis servidores C&Cs, antes de iniciar o processo de validação (seção 3.4).

C&C contactados em porta específica `(PE)`. Inferimos como possíveis servidores C&Cs os endereços contactados em uma porta para a qual houve menos de cinco endereços contactados. Essa heurística se baseia no conhecimento de que o *malware* normalmente realiza conexões com dois propósitos: varrer a rede à procura de dispositivos vulneráveis e contatar o servidor C&Cs ou outra infraestrutura da *botnet* [Antonakakis et al. 2017, Marzano et al. 2018]. Como a varredura realiza muitas tentativas de conexão em portas de serviços conhecidos (e.g. SSH), as portas com poucos endereços contactados provavelmente não estão associadas ao processo de varredura.

C&C com endereços IP presentes no *malware* `(IPM)`. Inferimos como possíveis servidores C&Cs os endereços IP que estavam explicitados no *malware* e foram contactados pelo menos uma vez durante sua execução. Dessa forma, a heurística contempla a identificação de C&Cs listados explicitamente no *malware*.

C&C e servidores de *malware* `(SM)`. Inferimos como possíveis servidores C&C os endereços IP e nomes de domínio contidos em comandos utilizados para baixar *malwares* (e.g., wget e curl) encontrados nas cadeias de caracteres do *malware*. Esse é comportamento típico de acesso a um servidor de *malware*. Essa heurística permite inferência de C&Cs hospedados no mesmo endereço que servidores de *malware*.

C&C contactados por nome de domínio `(ND)`. Inferimos como possíveis servidores C&C os nomes de domínio presentes em requisições DNS realizadas pelo *malware* durante sua execução. Algumas *botnets* identificam o servidor C&C por um nome de domínio, permitindo atualizar o endereço IP caso o endereço original seja comprometido.

3.4. Validação e identificação de servidores C&C

Para verificar se os candidatos apontados pelas heurísticas são realmente servidores C&C, utilizamos dois métodos de validação. O primeiro analisa o relatório de captura dos pacotes (incluindo os cabeçalhos e corpo dos pacotes) gerados durante a execução no Detux, e o segundo interage com o servidor C&C candidato em busca de evidências.

No primeiro método, procuramos no corpo dos pacotes por comunicação em codificação ASCII com os C&Cs candidatos. Para cada conexão com codificação ASCII identificada, verificamos se os dados observados fazem parte de um banco de 583 comandos conhecidos, previamente observados na comunicação entre *bots* e C&Cs da família Bashlite [Marzano et al. 2018]. (Esse método de validação se aplica apenas a *malwares* da família Bashlite.)

No segundo método, interagimos com cada um dos candidatos, estabelecendo uma conexão e esperando o envio de comandos por parte do C&C. Esse método de verificação funciona para *malwares* das famílias Bashlite e Mirai (e variantes com comportamento semelhante). Para validar um C&C da família Bashlite, ao interagir com o C&C candidato, verificamos se algum comando do banco de 583 comandos conhecidos do Bashlite é recebido; se esses comandos forem observados, o candidato é validado como um C&C Bashlite. Para validar um C&Cs da família Mirai, interagimos com C&C candidato e enviamos o *heartbeat* padrão Mirai, aguardando a resposta esperada do C&C; em caso positivo, o candidato é validado como um C&C Mirai.

4. Resultados

Nossos resultados mostram a eficiência do método de detecção e validação automática de servidores C&C proposto nesse trabalho. Primeiro, nós avaliamos as análises estáticas em nossa amostra, bem como a evolução dos resultados destas análises ao longo do tempo (seção 4.1). Em seguida, avaliamos nossas análises dinâmicas, incluindo a contribuição de cada uma das heurísticas na identificação de servidores C&C (seção 4.2). Por fim, nós analisamos a taxa de validação de servidores C&C em nossa amostra, discutindo os resultados obtidos (seção 4.3). De forma complementar, caracterizamos os servidores C&C confirmados para evidenciar o comportamento dos atacantes e auxiliar na mitigação de *botnets* IoT (seção 4.4).

4.1. Análise estática dos binários coletados

Executamos nossa análise estática sobre a base completa de binários (25.183 binários, seção 2). Observamos que 22,7% do total de binários não possuem símbolos de depuração, e este número apresenta tendência crescente no período recente (figura 2). Além disso, 18% dos binários são compactados usando o UPX[4] (*Ultimate Packer for Executables*), ferramenta que indiretamente dificulta as análises estáticas. Por fim, 2,7% dos binários utilizam o mecanismo de chave de ativação do nome do executável (`argv[0]`).

[4]https://upx.github.io/

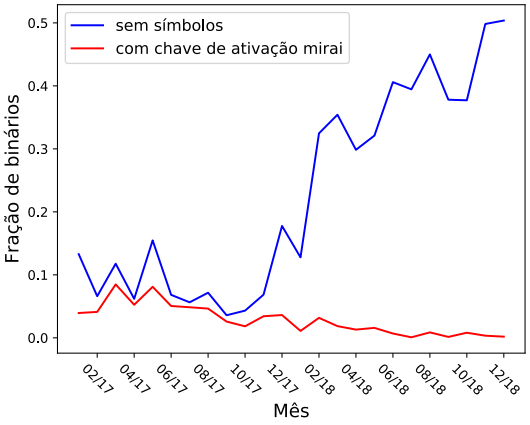

Figura 2. Evolução de propriedades de binários ao longo do tempo.

Malwares	total	521	100%
	com C&Cs candidatos	490	94%
	com C&Cs validados	307	59%
C&Cs	candidatos	483	100%
	confirmados	214	44%
	Bashlite	38	8%
	Mirai	176	36%

Tabela 1. Sumário da identificação de servidores C&C e cobertura de *malwares*.

A figura 2 mostra a fração de binários sem símbolos de depuração e com chaves de ativação Mirai, calculadas por mês do período de coleta. Como não conseguimos precisar a família de *malware* (Bashlite ou Mirai) para todos os binários, mostramos a fração global de binários utilizando o mecanismo de ativação. Além disso, observamos um crescimento consistente na fração de binários sem símbolos, o que mostra o empenho dos agentes maliciosos em dificultar as análises estáticas.

Observamos também uma redução gradativa da fração de binários utilizando o mecanismo de ativação, o que pode estar associado a uma mudança no comportamento dos operadores de *botnets* da família Mirai, ou uma evolução do *malware* (como a substituição do mecanismo de ativação por outros mais elaborados), o que pretendemos investigar em trabalhos futuros. Apesar da diminuição do uso do mecanismo de ativação, notamos que ele ainda é utilizado, sendo importante manter nossa extensão no Detux para prevenir perda de informação. Além disso, considerando que apenas binários Mirai utilizam o mecanismo de ativação, conjecturamos que a fração global mostrada na figura 2 é um limite inferior da fração de binários Mirai que utilizam este mecanismo.

4.2. Análise dinâmica e heurísticas para identificação de servidores C&C

Devido a restrições computacionais, utilizamos um subconjunto dos *malwares* nas análises dinâmicas. Em particular, consideramos 521 *malwares* coletados entre 27/11/2018 e 23/12/2018, sobre os quais a tabela 1 sumariza estatísticas. Em particular, mostramos o número de *malwares* com servidores C&Cs candidatos e validados, bem como o número de servidores C&Cs candidatos e validados.

A tabela 2 sumariza as estatísticas das heurísticas, informando o número de candidatos apontados, quantos foram inferidos exclusivamente por cada heurística, e a quantidade dos que foram validados. Além disso, mostramos a cobertura de cada heurística: de todos os C&Cs validados, quantos foram apontados por aquela heurística.

Observamos que existem candidatos exclusivos apontados por cada umas das heurísticas, mostrando que elas são complementares. Mostramos que a heurística `PE` tem uma cobertura muito boa (99,7%) e é muito importante para aumentar a taxa de validação agregada, uma vez que aponta exclusivamente 117 candidatos. Por outro lado,

Heurística	Candidatos Total	Exclusivos	Validados	Cobertura
Porta específica (PE)	394	117	197 (50%)	99,7%
IP presente no *malware* (IPM)	280	3	137 (49%)	62,5%
Servidor de *malware* (SM)	113	59	63 (56%)	31,1%
Nome de domínio (ND)	27	27	0 (0%)	0,0%
Agregado	483	—	214 (44%)	100%

Tabela 2. Número de candidatos, precisão e cobertura das heurísticas de identificação de servidores C&C.

a heurística IPM tem uma boa cobertura (62,5%), mas apresenta uma sobreposição de candidatos muito grande com as outras heurísticas. Ainda, a heurística SM apresenta a menor porcentagem de falsos positivos (44%), e 52% dos candidatos apontados por ela são exclusivos, evidenciando sua importância na descoberta de servidores C&C. Os domínios candidatos inferidos pela heurística ND não foram correlacionados com os IPs validados, pois reservamos esta tarefa para trabalhos futuros.

4.3. Validação de servidores C&C

Aplicamos nossas duas formas de validação de servidores C&C candidatos: (i) contactamos os servidores candidatos e esperamos receber e identificar o protocolo das famílias Bashlite e Mirai, e (ii) varremos o registro (*dump*) do tráfego de rede por comunicação utilizando estes protocolos. Verificamos que a varredura do registro de tráfego permite a detecção de 53% mais servidores C&C Bashlite quando comparado à tentativa direta de conexão. A principal razão para essa diferença são variantes onde o servidor C&C somente envia mensagem depois de receber uma mensagem de inicialização (*handshake*) do *bot*.

A tabela 1 mostra a fração de C&Cs candidatos validados (44%). A razão para falhas de validação incluem falsos positivos em nossas heurísticas, bloqueio/mitigação (*takedown*) do servidor C&C antes da tentativa de validação e modificações do protocolo que nossos mecanismos de validação não consideram. Em particular, em nosso conjunto de dados o processo de validação pode ser executado até quatro dias após a coleta do *malware*, tempo suficiente para alguns servidores C&C serem bloqueados, prejudicando nossos resultados. Estamos trabalhando para otimizar algumas etapas do nosso arcabouço para limitar o período entre a coleta do *malware*, análise e validação a 12 horas. Como existem diferentes *malwares* que compartilham o mesmo servidor C&C, conseguimos validar servidores C&C de 59% dos *malwares*, indicando que, apesar dos falsos positivos, as heurísticas são efetivas.

4.4. Caracterização dos servidores C&C validados

Entender o comportamento e a infraestrutura utilizada pelas *botnets* IoT é essencial na criação de mecanismos de defesa e mitigação. Nesta seção caracterizamos os servidores C&C validados para identificar aspectos que podem ser utilizados no combate às atividades maliciosas.

Primeiro estudamos a localização dos servidores C&C validados. Para determinar as redes em que cada servidor C&C está hospedado, realizamos o mapeamento dos

endereços IP em seus respectivos Sistemas Autônomos (ASes) utilizando a base do Team Cymru[5], e classificamos os ASes de acordo com o AS-Rank da CAIDA [CAIDA 2018]. A figura 3 mostra a distribuição acumulada dos C&Cs em ASes. Observamos uma concentração muito grande de C&Cs em poucos ASes, com cinco ASes sendo responsáveis por hospedar mais de 70% de todos os C&Cs. Além disso, 86% dos C&Cs estão hospedados em provedores de infraestrutura (*cloud computing*), como indicado pelos círculos na figura 3. Esses resultados mostram que podemos concentrar os esforços de mitigação em poucas redes, e que medidas para penalizar provedores de infraestrutura coniventes com essas práticas podem ser efetivas no combate a *botnets* IoT.

Analisamos também as portas mais utilizadas para contactar os servidores C&C. A figura 4 mostra a distribuição acumulada dos C&Cs em função das portas. Observamos que apenas dez portas cobrem quase 60% dos C&Cs validados. O conhecimento das portas mais frequentemente utilizadas por C&Cs pode ser utilizado para aprimorar métodos de identificação e validação de C&Cs. Algumas heurísticas inferem um endereço IP candidato, mas não indicam exatamente a porta que deve ser utilizada para contactar o C&C. Ao invés de varrer todas as portas, podemos testar apenas aquelas mais prováveis de serem utilizadas; reduzindo a quantidade de requisições realizadas, especialmente em casos de falso positivo das heurísticas de inferência de servidores C&C.

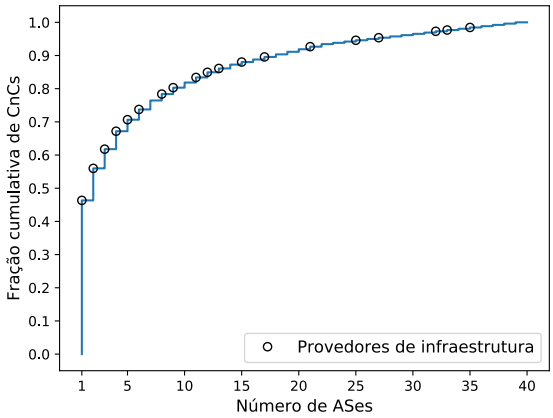

Figura 3. Distribuição dos ASes dos C&Cs.

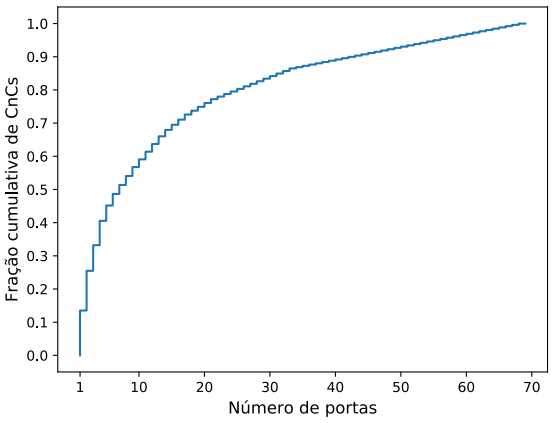

Figura 4. Distribuição das portas utilizadas pelos C&Cs.

5. Análise de similaridade dos binários de *malware*

Analisar grandes conjuntos de binários de *malware* é uma tarefa onerosa. Em particular, analistas de segurança têm de lidar com binários para variadas arquiteturas de *hardware*; binários sem tabela de símbolos, ofuscados, compactados ou cifrados; binários contendo modificações de código tautológicas ou construções introduzidas para desorientar análises. A identificação de binários similares é uma estratégia efetiva para escolher binários representativos no universo de binários e reduzir a quantidade de binários analisados, bem como auxiliar a análise direcionando esforços para as diferenças entre binários. Uma vez identificados binários similares, analistas podem focar em binários representativos e analisar (manual ou computacionalmente) as especificidades das diferenças.

[5]http://www.team-cymru.org/

5.1. Agrupamento de binários de *malware* por similaridade

Nossa estratégia para agrupar binários por similaridade segue o paradigma de divisão e conquista, quantificando a similaridade entre as funções dos binários de *malware* coletados como um indicativo de similaridade das funcionalidades e comportamento dos binários. Para essa análise, utilizamos uma amostra de 250 binários MIPS coletados entre os dias $27/11/2018$ e $23/12/2018$. Um desafio para implementar essa estratégia é que binários de *malware* frequentemente não possuem tabela de símbolos de depuração (binários *stripped*, 87% em nosso conjunto de dados), o que dificulta a análise do código e a identificação de funções.

Para contornar este desafio utilizamos o *radare2*, um arcabouço de código aberto para análise e engenharia reversa de binários, para identificar as funções nos binários. O *radare2* permite, por exemplo, desmontar, depurar, emular fluxos de execução e realizar *patches* em binários para diversas arquiteturas. Utilizamos o *radare2* para extrair endereços, parâmetros e a sequência de instruções de cada função. Descompactamos os binários compactados com o UPX (18%) antes do processo de extração de funções. Identificamos também que alguns binários de *malware* coletados possuem longas sequências de chamadas de funções; para contornar este problema aumentamos o limite de profundidade do grafo de chamada de funções utilizados pelo *radare2* de 64 para 512.

Geramos assinaturas de todas as funções identificadas pelo *radare2* utilizando o *ssdeep* [Kornblum 2006], uma função de *fuzzy hashing* que provê garantias de localidade de referência espacial, ou seja, os valores da função para entradas similares também são similares. Dessa forma, conseguimos calcular assinaturas de funções que nos permitem comparar quão similares são duas funções.

Uma vez definidas as funções e suas assinaturas, podemos calcular a similaridade entre dois binários b_α e b_β como descrito a seguir. Sejam $f_{\alpha,1} \dots f_{\alpha,n_\alpha}$ as funções de b_α e $f_{\beta,1} \dots f_{\beta,n_\beta}$ as funções de b_β, que são os vértices de um grafo bipartido completo $G_{\alpha,\beta}$, onde a aresta e_{ij} conecta $f_{\alpha,i}$ e $f_{\beta,j}$ e o seu peso w_{ij} é igual à similaridade calculada pelo *hash ssdeep* [Kornblum 2006] entre $f_{\alpha,i}$ e $f_{\beta,j}$. Determinamos então o casamento maximal, que escolhe no máximo uma aresta por vértice em $G_{\alpha,\beta}$, enquanto maximiza a soma dos pesos das arestas escolhidas. A similaridade entre um par de binários $S_{\alpha,\beta}$ é a soma dos pesos das arestas que participam do casamento dividido pelo máximo do número de funções nos binários do par (i.e., $\max(n_\alpha, n_\beta)$).[6]

Após calcularmos a similaridade entre todos os pares de binários conforme descrito, construimos uma matriz de distâncias entre todos os pares de binários. Calculamos a distância entre dois binários como o complemento da similaridade entre eles, i.e., $D_{\alpha,\beta} = 1 - S_{\alpha,\beta}$. Utilizamos uma técnica de agrupamento hierárquico baseada em aglomeração para agrupar por similaridade os binários que tiveram servidores C&C validados. Em particular, adotamos um critério de agrupamento considerando os vizinhos mais distantes (*complete-linkage*) [Müllner 2011] sobre a matriz de distâncias.

O algoritmo de agrupamento é guloso e agrupa, a cada iteração, os dois grupos que sejam mais próximos formando um novo grupo. A distância entre dois grupos é definida como a distância máxima de um par de binários, sendo um binário de cada grupo. Por exemplo, a distância entre os grupos $g_1 = (b_\alpha, b_\beta)$ e $g_2 = (b_\gamma)$ é dada por

[6]Dividimos a similaridade por $\max(n_\alpha, n_\beta)$ para considerar diferenças no número de funções n_α e n_β.

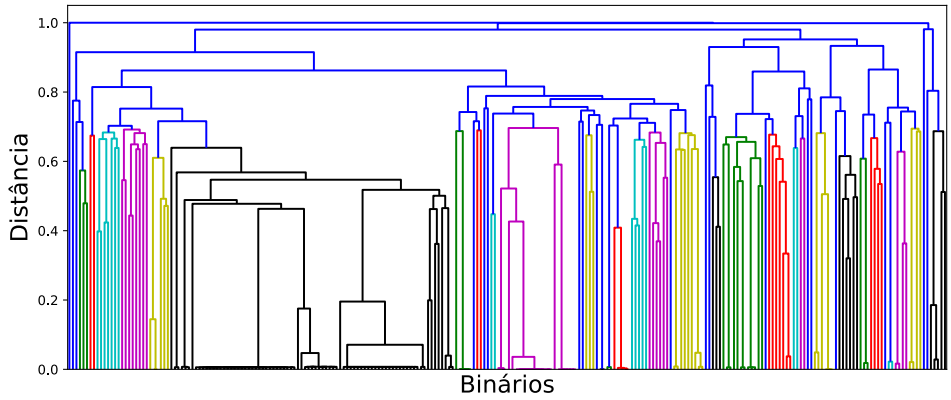

Figura 5. Dendrograma da clusterização hierárquica.

$D_{g_1,g_2} = \max(D_{\alpha,\gamma}, D_{\beta,\gamma})$. A cada passo, a matriz de distâncias é atualizada com a distância entre todos os grupos correntes, e os dois grupos com menor distância são combinados em um único grupo. A cada passo do agrupamento, o novo grupo formado pela combinação dos dois grupos mais próximos é registrado num diagrama de árvore (dendrograma), no qual é possível consultar a hierarquia de combinações e a distância entre cada par de grupos combinados. O algoritmo inicia com grupos formados por cada binário.

5.2. Resultados

A figura 5 mostra o dendrograma resultante do agrupamento de binários com servidores C&C validados em nosso conjunto de dados. O eixo y mostra a distância entre dois grupos de binários D_{g_i,g_j} quando eles são combinados. As longas barras verticais indicam que vários grupos de binários possuem grandes distâncias quando combinados; e as barras verticais curtas indicam binários com pequenas distâncias quando combinados.

Para construir grupos a partir do dendrograma, escolhemos um limiar de dissimilaridade L e executamos os passos do algoritmo, agrupando binários, enquanto a distância dos grupos for menor que L. Em outras palavras, traçamos uma linha horizontal em $y = L$ e os grupos são determinados pelos binários em cada uma das linhas horizontais intersectadas. A figura 6 mostra o número de grupos identificados variando o limiar de dissimilaridade L em passos de 5%.

Nossos resultados mostram uma redução significativa do número de grupos para uma dissimilaridade de até 5%, e redução muito pequena a partir deste ponto. Este resultado indica que vários binários são muito similares (são agrupados mesmo com baixos limiares de dissimilaridade) e que vários binários são muito dissimilares (só são agrupados com altos limiares de dissimilaridade). Este resultado suporta o argumento de que as variantes de binários de *malware* Bashlite e Mirai são dissimilares, mas algumas variantes possuem diversas versões com pequenas modificações. A escolha de um limiar de dissimilaridade de 5%, por exemplo, agrupa 250 binários em 142 grupos (variantes). Por exemplo, o maior grupo é composto por 29 binários com C&Cs Mirai comprovados. Para esse limiar de dissimilaridade, 80% das 142 variantes identificadas possuem apenas um binário, mas 3,5% das variantes possuem pelo menos 5 versões de binários. Neste

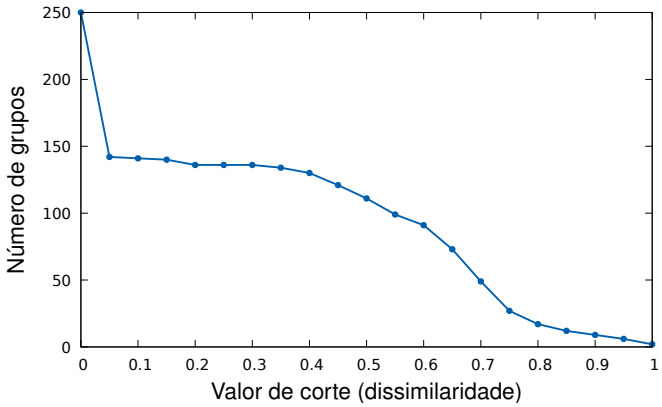

Figura 6. Número de grupos por nível de dissimilaridade

caso, a análise de uma versão de cada variante reduziria em 43% o número de binários que analistas de segurança precisariam estudar.

6. Trabalhos Relacionados

Caracterização de botnets IoT e identificação de servidores C&C. Ataques DDoS são um problema de segurança recorrente agravado com a proliferação de *botnets*. Muitos trabalhos analisam formas de identificar essas *botnets* e propõem mecanismos de defesa contra ataques coordenados e realizados por elas [Silva et al. 2013]. Com o surgimento de *botnets* IoT, houve um aumento muito grande no número de dispositivos infectados e, consequentemente, uma potencialização do volume de tráfego gerado nesses ataques [Kolias et al. 2017]; inviabilizando muitas das contramedidas propostas anteriormente. Assim, estudos recentes caracterizaram o comportamento de *botnets* IoT a fim de entender suas arquiteturas, o processo de infecção e elucidar novos mecanismos de defesa que sejam eficientes contra essa nova ameaça [Kolias et al. 2017, Angrishi 2017, Antonakakis et al. 2017, Marzano et al. 2018].

Mais relacionados ao nosso trabalho são artigos que buscam a identificação de servidores C&C. Alguns trabalhos utilizam análise estática, através de engenharia reversa dos binários, para identificar os servidores C&C [Antonakakis et al. 2017]. Outros trabalhos analisam o tráfego gerado pela execução de *malwares* em um ambiente controlado e tentam identificar, dentre todas as conexões geradas na execução, as conexões com os servidores C&C [Jacob et al. 2011, Zand et al. 2014]. Nosso trabalho utiliza uma abordagem híbrida, que leva em consideração a análise estática e dinâmica na identificação de servidores C&C, e complementa técnicas existentes.

Similaridade entre binários. A abordagem que utilizamos para determinar a similaridade entre dois binários é similar a outros trabalhos na literatura, que definem a similaridade entre fragmentos de diferentes binários [David et al. 2016]. Outra abordagem que pretendemos considerar para identificação de similaridade entre binários é a comparação entre execuções (análise dinâmica) de binários [Egele et al. 2014].

7. Conclusão e trabalhos futuros

Como o controle das *botnets* IoT é centralizado, identificar os servidores C&C pode auxiliar no combate, significativamente reduzindo ameaças de rede. Neste trabalho propusemos um arcabouço para identificação e validação automática de servidores C&C. Nossas técnicas validaram servidores C&C para 59% dos *malwares*. Os resultados mostram que

as heurísticas propostas são complementares. Ainda, nossos resultados corroboram trabalhos anteriores e mostram que os servidores C&C validados estão concentrados em poucos ASes, a maioria deles hospedados em provedores de infraestrutura. Isso pode ajudar na criação de mecanismos de defesa, que podem concentrar os esforços em poucas redes. Por último, desenvolvemos uma estratégia para agrupar binários por similaridade. Nossa estratégia é capaz de agrupar 250 binários com C&C validados em 142 variantes, reduzindo em 43% o número de binários que analistas de segurança precisam analisar.

Como trabalho futuro pretendemos estender as funcionalidades de análise estática e dinâmica do nosso arcabouço integrando novos mecanismos para identificar e desabilitar contramedidas utilizadas pelos *malwares* para combater análises dinâmicas. Pretendemos também estender e aplicar nosso arcabouço a outras famílias de *malware*.

Agradecimentos

Este trabalho foi parcialmente financiado pela FAPEMIG, CNPq, CAPES e NIC.br.

Referências

Angrishi, K. (2017). Turning Internet of Things(IoT) into Internet of Vulnerabilities (IoV): IoT Botnets. *CoRR*, abs/1702.03681.

Antonakakis, M. et al. (2017). Understanding the Mirai Botnet. In *Proc. of USENIX SS*.

CAIDA (2018). CAIDA AS Rank.

David, Y., Partush, N., and Yahav, E. (2016). Statistical Similarity of Binaries. *SIGPLAN Not.*, 51(6).

Egele, M., Woo, M., Chapman, P., and Brumley, D. (2014). Blanket Execution: Dynamic Similarity Testing for Program Binaries and Components. In *Proc. of USENIX SS*.

Jacob, G., Hund, R., Kruegel, C., and Holz, T. (2011). JACKSTRAWS: Picking Command and Control Connections from Bot Traffic. In *Proc. of USENIX SS*.

Kolias, C., Kambourakis, G., Stavrou, A., and Voas, J. (2017). DDoS in the IoT: Mirai and other Botnets. *Computer*, 50(7):80–84.

Kornblum, J. (2006). Identifying Almost Identical Files Using Context Triggered Piecewise Hashing. *Digital Investigation*, 3:91–97.

Marzano, A., Alexander, D., Fazzion, E., Fonseca, O., Cunha, I., Hoepers, C., Steding-Jessen, K., Chaves, M. H. P. C., Guedes, D., and Júnior, W. M. (2018). Monitoramento e Caracterização de Botnets Bashlite em Dispositivos IoT. *Anais do SBRC*, 36.

Müllner, D. (2011). Modern hierarchical, agglomerative clustering algorithms. *CoRR*, abs/1109.2378.

Neustar (2017). Worldwide DDoS Attacks & Cyber Insights Research Report. Online.

Silva, S. S., Silva, R. M., Pinto, R. C., and Salles, R. M. (2013). Botnets: A Survey. *Computer Networks*, 57(2).

Symantec (2017). Internet Security Threat Report, Volume 22. Online.

Tange, O. (2018). *GNU Parallel 2018*. Ole Tange.

Zand, A., Vigna, G., Yan, X., and Kruegel, C. (2014). Extracting Probable Command and Control Signatures for Detecting Botnets. In *Proceedings of the 29th AACMSAC*.