Uma arquitetura para detecção de *botnet*s baseada na análise do tráfego DNS descartado

Luiz Gonzaga Mota Barbosa, Joaquim Celestino Júnior, André Luiz Moura dos Santos

[1]Universidade Estadual do Ceará (UECE)
Fortaleza/CE

Abstract. *Botnets are one of the main threats on the Internet and are able to assist malicious activities. After a host infection, a bot tries to communicate with its command and control servers. At this point, recent* bots *use Domain Generation Algorithms to create a list of candidate domain names for command and control servers. One consequence of this behavior is the increase in negative DNS answers. This traffic, usually discarded or ignored by network administrators, can be used to model a* botnet *behavior. Based on the increase in the number of negative DNS answers and having access to samples collected from a controled environment, this paper discusses an architecture capable of generating detection models to these* bots*. The tests showed an acuracy level above 90% in most test cases.*

Resumo. Botnets *são uma das principais ameaças na Internet, capazes de auxiliar na realização de atividades maliciosas. Ao infectar um hospedeiro, um* bot *tenta comunicar-se com o servidor de comando e controle. Neste ponto,* bots *recentes utilizam Algoritmos de Geração de Domínios para criarem uma lista de nomes de domínio candidatos a servidores de comando e controle. Uma consequência deste comportamtento é o aumento de respostas negativas do protocolo DNS. Esse tráfego, geralmente descartado ou ignorado pelos administradores de rede, pode ser utilizado para modelar o comportamento de uma* botnet*. Com base no aumento no número de respostas negativas do protocolo DNS e acesso a amostras coletadas em um ambiente controlado, este trabalho apresenta uma arquitetura capaz de gerar modelos de detecção para estes* bots*. Os experimentos mostraram um nível de acurácia acima de 90% na maioria dos cenários.*

1. Introdução

Recentemente, o mundo observou a onda de ataques do malware *WannaCry*, o qual utilizava o protocolo DNS para controlar a sua disseminação [Khandelwal 2017]. Caso o domínio requisitado, respondesse as requisições, o ataque seria abortado. Enquanto o domínio não fosse registrado pelo criador do *malware*, haveria um aumento de respostas negativas do protocolo DNS na rede infectada.

Uma classe específica de *malware*, permite que o atacante controle remotamente as máquinas infectadas. O conjunto de máquinas infectadas por um mesmo *malware* desta classe compõe uma *botnet*. Uma das primeiras ações executadas pelo *malware* após uma infecção é tentar localizar seu servidor de comando e controle. Para isso, *bots* mais recentes tem utilizado o protocolo DNS, desfrutando das mesmas vantagens que usuários legítimos.

Uma maneira de contactar seus servidores é carregar o domínio destes em seu código. Porém, para dificultar a detecção a partir de técnicas de engenharia reversa, desenvolvedores de *bots* passaram a utilizar Algoritmos de Geração de Domínios (*Domain Generation Algorithms - DGAs*) com a finalidade de criar uma lista de candidatos a servidores de Comando e Controle (C&C) para o estabelecimento de sua comunicação [Stone-Gross et al. 2009, Bilge et al. 2011, Antonakakis et al. 2012].

A utilização de DGAs ocasiona um aumento na quantidade de respostas negativas do protocolo DNS na rede e esse tráfego é normalmente descartado. Entretanto, esse tráfego pode ser utilizado para modelar a presença de infecções na rede e gerar perfis de comportamento dos *bots* presentes na rede.

Este trabalho apresenta uma arquitetura para detecção de *botnet*s com base na análise so tráfego DNS que seria descartado pela rede, em especial daquelas que utilizam DGAs para a comunicação com seus servidores de C&C.

O restante deste trabalho está organizado da seguinte maneira: a Seção 2 apresenta os trabalhos relacionados; a Seção 3 define o que são *botnets*, suas principais características, seu relacionamento com o protocolo DNS e os aspectos gerais para sua detecção; a Seção 4 apresenta a arquitetura utilizada neste trabalho; a Seção 5 descreve a implementação e experimentos, resultados obtidos e análise dos mesmos; e, por fim, a conclusão na Seção 6.

2. Trabalhos Relacionados

[Holz et al. 2008] discutem uma técnica utilizada por *botnets* chamada Fast-Flux Service Network (FFSN). Esta pode ser resumida como uma versão maliciosa do Round-Robin DNS (RRDNS), na qual são utilizados valores de *Time-To-Live (TTL)* relativamente pequenos quando comparados aos valores utilizados para domínios não-maliciosos. Dessa forma, a lista de endereços IP presentes na resposta, os hospedeiros que fazem parte do ”fluxo”, é atualizada mais rapidamente [Salusky and Danford 2007].

[Stone-Gross et al. 2009] foram os primeiros a discutir outra técnica utilizada por *botnets* para o estabelecimento de sua comunicação: Algoritmos de Geração de Domínios (*Domain Generation Algorithms - DGA*). Com ela, *bots* geram uma lista de nomes de domínio candidatos a servidor de C&C a cada tentativa de comunicação ou dado um intervalo determinado pelo botmaster. A primeira máquina que responder a essas requisições será considerado um servidor de C&C válido.

[Bilge et al. 2011] destacam a importância do protocolo DNS para serviços não-maliciosos e maliciosos e, portanto, sua utilização no processo de detecção de atividades maliciosas. Sua principal hipótese é a de que, ao analisar grandes quantidades de dados do tráfego de uma rede, requisições DNS maliciosas e não-maliciosas exibirão padrões de comportamento diferentes e que poderão ser detectadas automaticamente. Sobre essa diferenciação de padrões na utilização do protocolo DNS, [Antonakakis et al. 2012] destacam que *bots* que utilizam DGAs geram um aumento na quantidade de respostas DNS negativas na rede e este comportamento pode ser utilizado para a geração de modelos de detecção.

Baseado na importância do protocolo DNS na realização de atividades maliciosas e nas técnicas utilizadas por *malwares*, este trabalho concentra sua análise no tráfego DNS

de uma rede monitorada.

Trabalhos como [Schiavoni et al. 2014] e [Erquiaga et al. 2016] buscam identificar as ameaças através da modelagem de comportamentos maliciosos e não-maliciosos. Abordagens como estas quando aplicadas a ambientes muito heterogêneos podem exigir uma carga computacional grande para modelar todos os possíveis comportamentos, maliciosos e não-maliciosos, ali presentes. Portanto, diferente destes trabalhos, aqui é utilizado apenas o tráfego DNS que seria descartado pela rede, diminuindo assim a quantidade de informação a ser processada pelo sistema de detecção de ameaça e, consequentemente, apenas comportamentos tidos como maliciosos serão modelados.

3. Fundamentação Teórica

3.1. Botnets

Botnets são grupos de máquinas comprometidas por *malwares*, conhecidas como bots ou zumbis, controladas remotamente por um indivíduo chamado *botmaster* através de um canal de comunicação de comando e controle (C&C) [Cisco 2007, Antonakakis et al. 2012].

Um mesmo bot pode apresentar características de diferentes tipos de *malware*s tornando as *botnet*s capazes de realizar diversas atividades maliciosas tais como: ataques de negação de serviço, roubo de informações sensíveis, envio massivo de e-mails (spams) entre outras [Gu et al. 2008, Cavallaro et al. 2009, Tiirmaa-Klaar et al. 2013]. Além disso, *bots* podem infectar diferentes tipos de máquinas vulneráveis espalhadas em uma mesma rede ou em redes distintas, como mostrado na Figura 1.

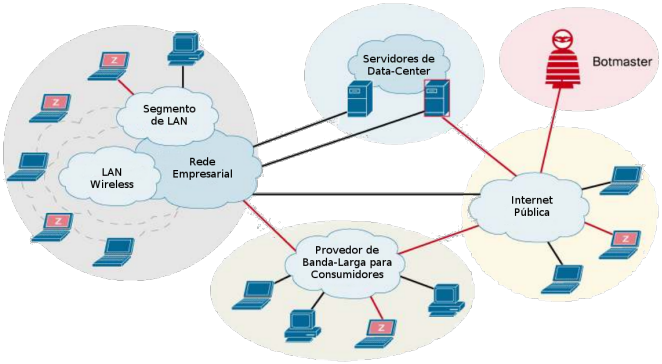


Figura 1. Ilustração de uma Botnet

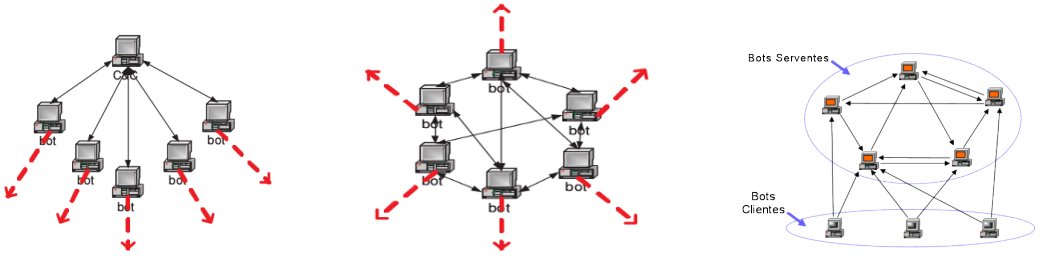
A força de uma *botnet* está em possuir uma rede flexível de computadores que podem ser controlados remotamente [Karim et al. 2014], com isso, a escolha de uma arquitetura de comunicação é uma decisão importante do ponto de vista do botmaster, influenciando em sua capacidade de gerenciamento, resiliência e em alguns aspectos de seu funcionamento [Tiirmaa-Klaar et al. 2013].

3.2. Arquitetura de uma Botnet

A busca por mecanismos que proporcionem uma infraestrutura flexível e confiável para sua *botnet* é uma questão importante enfrentada por desenvolvedores de *botnets*

[Bilge et al. 2011]. Sendo assim, diferentes arquiteturas foram utilizadas a fim de evitar contramedidas de bloqueio, mantendo a operabilidade da *botnet* pelo maior período possível [Tiirmaa-Klaar et al. 2013].

Quanto a sua estrutura de C&C, uma *botnet* pode ser projetada como uma arquitetura centralizada (Figura 2a), descentralizada (Figura 2b) ou híbrida (Figura2c).



(a) Centralizada. (b) P2P (c) Híbrida

Figura 2. Arquiteturas de uma *botnet*

3.3. Botnets e o Protocolo DNS

Devido a sua versatilidade, *bots* não sabem qual ação executar logo após a infecção, portanto, sua primeira ação é tentar estabelecer um canal de C&C com o botmaster a fim de registrar-se na *botnet* e então passar receber comandos, atualizações e trocar dados [Stone-Gross et al. 2009,].

Do ponto de vista do botmaster, é necessária alguma técnica que possibilite o estabelecimento da comunicação entre os membros da *botnet* e que os *bots* continuem invisíveis e portáveis [Choi et al. 2007]. Em resposta a essa necessidade, *bots* passaram a utilizar o protocolo DNS para localizar seus servidores de C&C e estabelecer este canal de comunicação entre estes [Stone-Gross et al. 2009, Bilge et al. 2011, Antonakakis et al. 2012].

O protocolo DNS, além de prover um mapeamento entre nomes de domínio e endereços IP, permite que um determinado hospedeiro possua mais de um nome (*host aliasing*) e que um mesmo domínio seja associado a mais de um endereço IP. Com isso, é possível que um determinado serviço seja distribuído entre servidores diferentes e, em caso de falha de algum destes servidores, outro poderá responder às requisições ao serviço.

Com essas carcaterísticas, a utilização do protocolo DNS para a comunicação entre membros da *botnet* agilizou o processo de migração de *botnet*, mantendo assim sua disponibilidade e a operabilidade ainda que um ou mais de seus servidores de C&C tenha sido descoberto e banido da rede [Choi et al. 2007, Antonakakis et al. 2010].

As principais técnicas utilizadas por *botnet*s envolvento o protocolo DNS são: Fast-Flux Service Networks e Algoritmos de Geração de Domínios, apresentadas em [Holz et al. 2008] e [Stone-Gross et al. 2009], respectivamente.

3.4. Detecção de Botnets

Independente da arquitetura utilizada, *botnets* compartilham uma característica: os *bots* pertencentes a uma mesma *botnet*, tendem a executar um mesmo comportamento, pré-determinado em seu código por seus criadores [Gu et al. 2008].

Quando capturado, um bot pode ser analisado estática e dinamicamente a fim de extrair suas características comportamentais a nível de hospedeiro e/ou de rede [Sikorski and Honig 2012,].

Através da utilização de técnicas de engenharia reversa, são extratídas assinaturas dos exemplares capturados. Porém, nem sempre é viável fazer a engenharia reversa de um exemplar de um bot, uma vez que eles podem ser programados para se modificarem a cada infecção ou ainda, sofrerem mutações como resposta a comandos do botmaster [Stone-Gross et al. 2009, Antonakakis et al. 2012].

Recentemente, o processo de detecção de *botnets* concentrou-se na análise de comportamento dos *bots* nas redes [Cavallaro et al. 2009, Gu et al. 2008, Bilge et al. 2012]. Em uma rede infectada, podem ser observados comportamentos baseados em: protocolo utilizado, quantidade de mensagens trocadas em um determinado período, variações geradas no tráfego da rede, acessos a *sites* marcados como maliciosos, entre outros.

A identificação de comportamentos maliciosos pode ocorrer através: da busca de evidências da infecção de *bots* ou da comunicação entre as máquinas e servidores de C&C, também conhecida como correlação vertical; ou através da correlacão de eventos de rede a fim de identificar hospedeiros envolvidos em atividades maliciosas similares, também conhecida como correlação horizontal.

O processo de análise de comportamento pode ocorrer também seguindo uma abordagem passiva ou ativa [Stone-Gross et al. 2009]. Em uma abordagem passiva, são estudados os efeitos secundários gerados pela atividade das máquinas infectadas. Já em uma abordagem ativa, os pesquisadores de fato buscam infiltrar-se na *botnet* através da utilização de um exemplar do *malware* ou um cliente simulando um bot.

3.5. Modelagem do comportamento malicioso

Uma vez detectada a presença de uma ou mais *botnets* em uma rede, são gerados modelos do comportamento dos *bots* identificados e em seguida esses modelos são aplicados à rede a fim de mitigar a ameaça. Para automatizar o processo de modelagem desses comportamentos, têm-se utilizado técnicas de aprendizagem de máquina [de A Ribeiro et al. 2011, Bilge et al. 2012, Zhao et al. 2013].

Para a modelagem do comportamento, é necessária a identificação de características capazes de representar o comportamento em questão. As características utilizadas podem ser classificadas como:

- **Características do fluxo de rede:** possibilitam uma identificação global dos padrões de comunicação dos *bots* quanto aos hospedeiros que estão se comunicando e o protocolo utilizado por eles;
- **Características de *payload*:** diretamente ligadas ao protocolo utilizado pelos *bots*, permitem um melhor entendimento do comportamento dos *bots*;
- **Características temporais:** utilizadas a fim de identificar padrões temporais na comunicação entre os membros da *botnet*.

Uma *botnet* pode fazer o uso de protocolos diferentes em fases diferentes de seu ciclo de vida e até dependendo de sua arquitetura. Também deve ser levado em consideração que os *bots* podem utilizar técnicas para dificultar sua detecção como: mascaramento de IP, polimorfismo e criptografia.

4. Uma arquitetura para a reciclagem do tráfego DNS

A principal hipótese deste trabalho é que *bots* que utilizam DGAs geram um aumento na quantidade de respostas negativas do protocolo DNS, respostas do tipo NXDOMAIN. Com isso, é possível utilizar este tráfego para identificação de domínios maliciosos bem como quais máquinas estão infectadas em uma rede monitorada.

Com base nessa hipótese, a arquitetura (Figura 3) a seguir realiza uma correlação vertical baseada na análise passiva do comportamento de *bots* quanto ao protocolo DNS, em especial daqueles que utilizam DGAs. Essa análise é realizada sobre o tráfego DNS que seria descartado, ou seja, concentra-se nos pacotes resposta DNS do tipo NXDOMAIN.

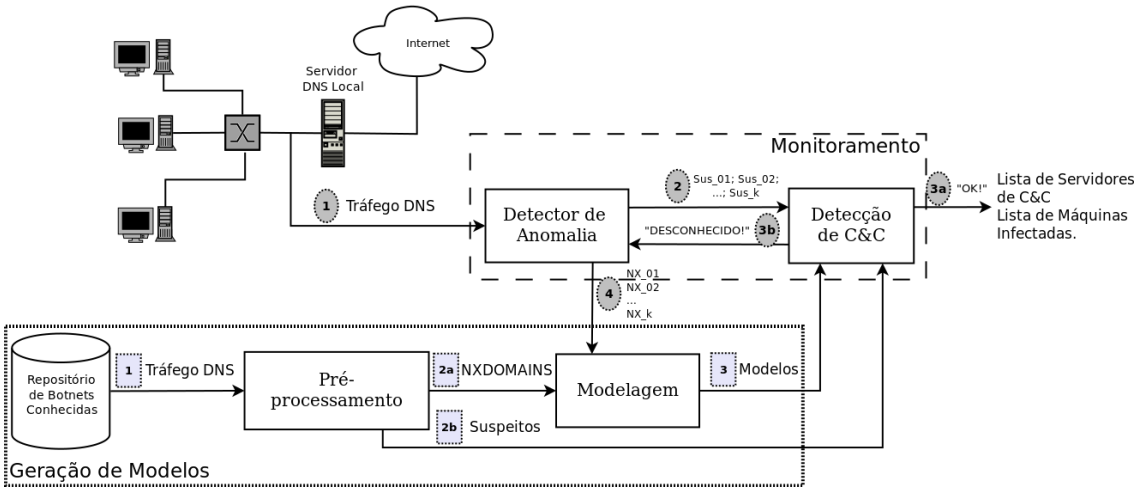


Figura 3. Visão geral e fluxo de informações na arquitetura

Na Figura 3, observa-se que a arquitetura está dividida em duas camadas: Monitoramento – composto por um módulo Detector de Anomalia e um módulo de Detecção de C&C; e Geração de Modelos, composto por uma base de conhecimento de tráfegos maliciosos previamente observados, um módulo de Pré-processamento e um módulo de Modelagem.

Primeiramente, o tráfego DNS é monitorado pelo Detector de Anomalia (Passo 1), este observa a rede monitorada a fim de identificar uma elevação nas taxas de tráfego NXDOMAIN, ao percerber tal comportamento anômalo, os domínios pertencentes a esse tráfego são agrupados e cada agrupamento é devidamente rotulado. As respostas de cada agrupamento são enviadas para a Detecção de C&C (Passo 2). Ao chegarem no módulo de Detecção de C&C, as respostas recebidas podem levar a um dos resultados a seguir:

- Ameaça detectada (”OK!”) – retorna os endereços IP dos servidores de C&C e a lista de máquinas infectadas (Passo 3a);
- Ameaça desconhecida (”DESCONHECIDO”) – padrão de comportamento ainda não modelado. O módulo envia um sinal para o módulo anterior (Passo 3b).

Seguindo o fluxo em caso de uma ameça ainda desconhecida, as respostas NXDOMAIN do agrupamento desconhecido são enviados para a camada de Geração de Modelos para que esta possa gerar um modelo para o novo comportamento (Passo 4).

A camada de Monitoramento é executada na dentro da rede monitorada, dessa forma é possível identificar não somente a existência de infecções, mas também, quais máquinas foram contaminadas a fim de que estas sejam sanadas. A Figura 4 apresenta detalhes do módulo de Detecção de Anomalia e a Figura 5 do módulo de Detecção de C&C.

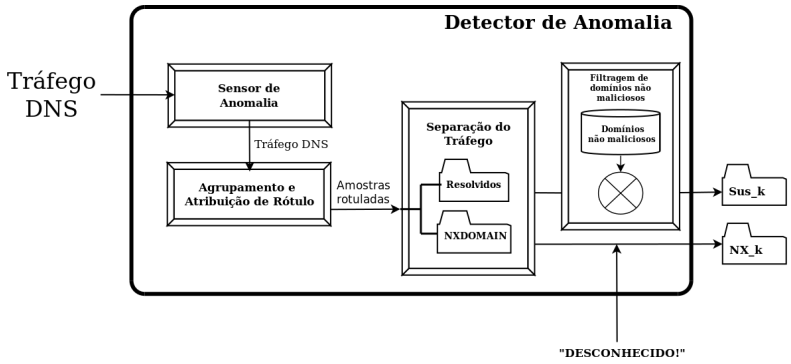


Figura 4. Módulo Detector de Anomalia

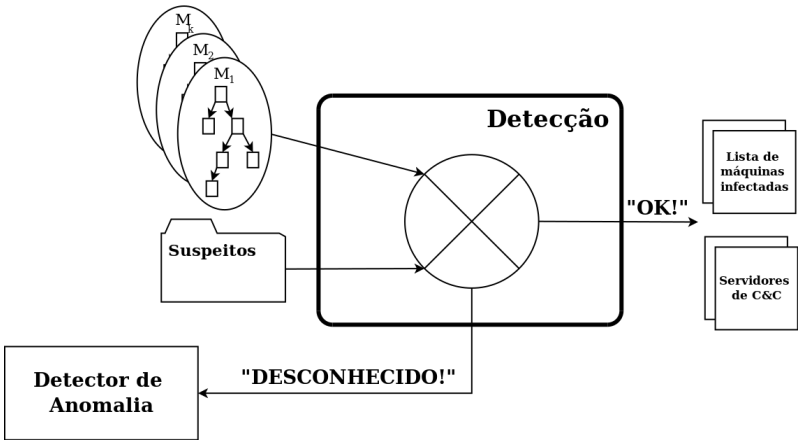


Figura 5. Módulo de Detecção de C&C

A camada de Geração de Modelos pode ser executada em um ambiente diferente do qual a camada anterior se encontra. Recomenda-se que este outro ambiente possua um poder computacional maior em relação ao Monitoramento tal que a geração dos modelos possa ocorrer o mais breve. O tráfego DNS obtido de um repositório de Botnets Conhecidas é enviado para uma fase de Pré-Processamento (Passo 1), onde será separado em respostas NXDOMAIN e respostas Suspeitas. O tráfego NXDOMAINS é enviado para a Modelagem (Passo 2a) e o tráfego Suspeito é enviado para a extração dos endereços IP dos servidores reportados (Passo 2b).

Quando a camada de Geração de Modelos recebe uma nova ameaça (Passo 4 da camada anterior), as requisições NXDOMAIN são enviadas direto para o módulo de Modelagem. Neste Módulo, são extraídas a características dos domínios requisitados e submetidos à técnicas de aprendizado de máquina para a geração dos modelos de detecção que serão enviados ao módulo de Detecção de C&C da camada de Monitoramento (Passo 3 da camada de Geração de Modelos).

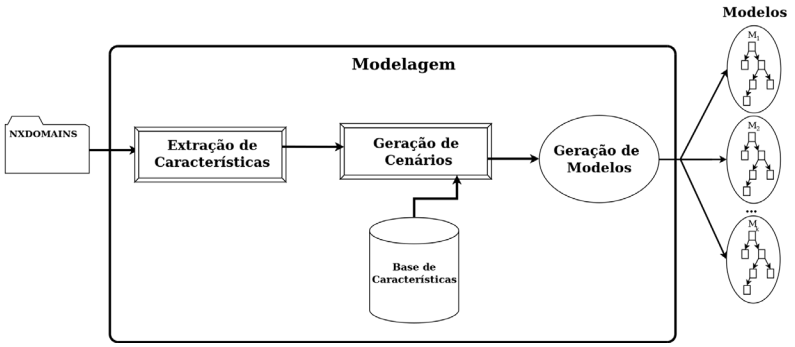


Figura 6. Estrutura interna no módulo Modelagem

A geração dos modelos de decteção leva em consideração o conhecimento observado anteriormente pela arquitetura. Nesta fase, um problema de classificação contendo *M* classes, é transformado em *M* problemas de classificação binária. Isto é alcançado através da Geração de Cenários (Figura 6), executado logo após a extração das características. Cada cenário possui uma série de cruzamentos binários das características de cada amostra repassada (Figura 7). Em seguida, os cenários são enviados para a geração de modelos através do uso de aprendizado de máquina.

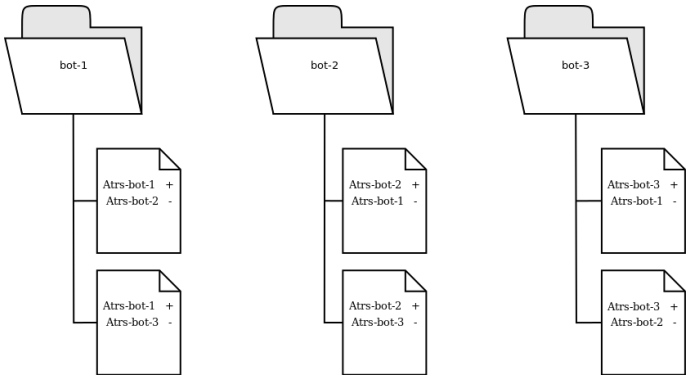


Figura 7. Representação dos cruzamentos binários que compõem cada cenário

5. Implementação e Análise dos Resultados

A implementação da arquitetura está dividida em duas fases, relacionadas as suas duas camadas: Monitoramento e Modelagem. Para este trabalho, a camada de Modelagem encontra-se concluída. Um outro trabalho, em andamento, trará a abordagem de monitoramento.

A base de *botnets* conhecidas foi montada a partir de dados disponibilizados pelo projeto *Malware Capture Facility Project* [Stratosphere 2015]. Este projeto possui um repositório responsável pela obtenção e manutenção de uma base de dados contendo o tráfego malicioso e não-malicioso.

Foram utilizadas 9 amostras de tráfego de rede. Dentre estas estavam 4 famílias de *bots*, onde uma família é composta pelas variantes de um *bot*. Dentre as famílias observadas, 3 utilizavam DGAs e 1 DGA+FFSN para o estabelecimento de comunicação com seus servidores de C&C.

Os domínios gerados por DGAs não necessitam ser algo compreensível por seres humanos, tratam-se, basicamente, de cadeias pseudoaleatórias de caracteres alfanuméricos como: *aeaybikhawkftknjrtxd.org* e *g7ci7ek5ek1di.ad*.

Com base na estrutura dos domínios gerados, os atributos a seguir foram considerados fundamentais para capturar as características básicas dos domínios gerados por um DGA:

- Quantidade de níveis de domínio;
- Tamanho do 2LD;
- Número de caracteres distintos no 2LD;
- Número de dígitos no 2LD;
- Entropia do 2LD;
- Tamanho do 3LD;
- Número de caracteres distintos no 3LD;
- Número de dígitos no 3LD;
- Entropia do 3LD.

A extração das características foi feita através de um script em linguagem Python 2.7 e criação dos cenários através de um script em linguagem Bash.

Os cenários criados foram utilizados para a alimentação da ferramenta Weka [Hall et al. 2009] para a geração dos classificadores. Devido a sua alta versatilidade em problemas de classificação, os algoritmos selecionados para os testes foram: árvores de decisão [Quinlan 1993] e naive-Bayes [John and Langley 1995].

Os experimentos com a ferramenta Weka foram realizados utilizando a técnica de validação cruzada (*cross-validation*) para a geração dos modelos. Nessa técnica, o conjunto de amostras é dividido em k subconjuntos, dos quais k - 1 serão utilizados para treinar o algoritmo e o subconjunto remanescente para a validação. A cada repetição esses subconjuntos sofrem um rodízio no qual aquele que foi utilizado para validação será utilizado para treinamento na próxima iteração. Nos testes realizados nesse trabalho foi utilizado um valor $k = 10$.

O ambiente de execução utilizou um processador Intel(R) Core(TM) i5-2450M CPU @ 2.50GHz, 8 GB de memória RAM (6 GB dedicados à ferramenta WEKA) e sistema operacional ArchLinux (kernel 4.8.13-1).

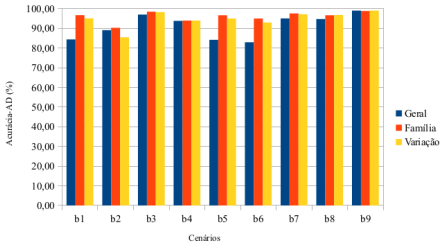
As métricas utilizadas foram:

- **Acurácia:** expressa a proximidade entre o valor obtido no experimento e o valor real;
- **Taxa de Positivos Verdadeiros (TPR):** também conhecida como Recall ou taxa de detecção, ela expressa a quantidade de elementos que foram corretamente detectados;
- **Taxa de Positivos Falsos (FPR):** expressa a quantidade de dados negativos que foram classificados como positivos;
- **Área sob a curva ROC (AUC):** expressa a relação entre TPR e FPR. Pode ser utilizada para expressar a qualidade da classificação (valores abaixo de 0,6 a classificação falhou; entre 0,6 e 0,7 classificação pobre; entre 0,7 e 0,8 classificação razoável; 0,8 e 0,9 classificação boa e; acima de 0,9 classificação excelente).

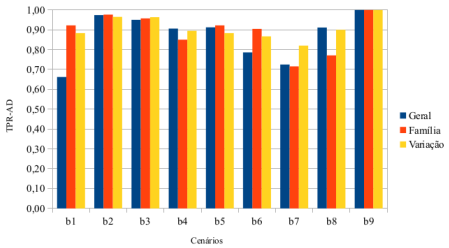
O comportamento dos classificadores foram medidos de acordo com as três visões a seguir:

- **Geral:** expressa o desempenho geral da classificação dentre todas as amostras analisadas, independente de qual família o *bot* pertence;
- **Família:** expressa o desempenho da classificação, levando-se em consideração o comportamento médio obtido com as demais famílias de *bots*;
- **Variação:** expressa o desempenho da classificação, sem considerar os membros da mesma família do cenário.

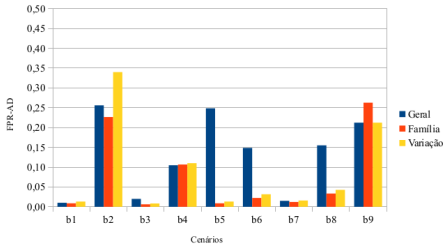
Dentre os algoritmos de aprendizado utilizados, Naive-Bayes e Árvores de Decisão, o algoritmo Naive-Bayes apresentou resultados absolutos abaixo daqueles obtidos com Árvores de Decisão. Entretanto, o comportamento geral da arquitetura para este algoritmo foi similar ao anterior. Sendo assim, serão mostrados apenas os gráficos relacionados ao algoritmo de Árvores de Decisão (Figura 8).



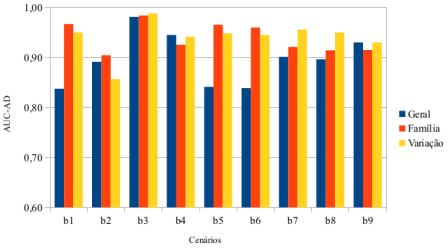
(a) Acurácia.



(b) TPR



(c) FPR



(d) AUC

Figura 8. Resultados para o algoritmo Árvores de Decisão

Como pode ser observado na Figura 8a, as características básicas sugeridas proporcionaram uma acurácia acima de 80% em todos os cenários e independente de considerar a qual família o *bot* pertence. Para a classificação levando em consideração a família dos *bots*, a acurácia passa a níveis acima dos 90% todos os cenários.

Através das Figuras 8b e 8c, observa-se mais detalhadamente os acertos na predição dos cenários. Ainda assim, a predição correta de elementos pertencentes a um dado cenário obteve um valor mínimo de 66%.

De acordo com a Figura 8d, todos os classificadores tiveram um desempenho, no mínimo, bom (entre 0,8 e 0,9) ainda que desconsidere a qual família o *bot* pertence. Quanto à classificação em relação à famílias diferentes, esta é considerada excelente (acima de 0,9) em todos os cenários observados.

5.1. O impacto do parentesco entre *bots*

Organizando as famílias de *bots* das amostras temos: as utilizam que DGAs – (b1-b5-b6), (b2-b3-b4) e (b7-b8); e que utiliza DGA e FFSN – (b9).

Durante os experimentos, para algumas famílias, observou-se uma dificuldade em diferenciar amostras de suas variações.

Como visto nas Tabelas 1a e 1c, a arquitetura apresentou uma dificuldade ao diferenciar os domínios gerados pelas variações destas famílias. Entretanto, com a Tabela 1b nota-se que a arquitetura consegue distinguir entre os elementos da família b2-b3-b4.

b1-b5-b6	Acurácia-AD(%)	TPR-AD	FPR-AD	AUC-AD
b1-b5	51,15	0,00	0,00	0,50
b5-b1	51,15	1,00	1,00	0,50
b1-b6	53,32	0,00	0,00	0,50
b6-b1	53,32	1,00	1,00	0,50
b5-b6	52,27	1,00	0,91	0,54
b6-b5	52,26	0,09	0,00	0,54

(a) Família b1-b5-b6

b2-b3-b4	Acurácia-AD(%)	TPR-AD	FPR-AD	AUC-AD
b2-b3	100,00	1,00	0,00	1,00
b3-b2	100,00	1,00	0,00	1,00
b2-b4	99,81	1,00	0,01	0,99
b4-b2	99,81	0,99	0,00	0,99
b3-b4	86,61	0,82	0,11	0,92
b4-b3	86,61	0,89	0,18	0,92

(b) Família b2-b3-b4

b7-b8	Acurácia-AD(%)	TPR-AD	FPR-AD	AUC-AD
b7-b8	80,04	0,06	0,01	0,52
b8-b7	80,04	0,99	0,94	0,52

(c) Família b7-b8

Table 1. Resultados agrupados por famílias de *bots* para o algoritmo Árvores de Decisão

Esse impacto sobre a predição de elementos da mesma família pode estar relacionado ao grau de semelhança entre suas variações sob o conjunto de características utilizado para a classificação. Ou seja, quando a metodologia erra ao classificar um elemento da classe-A como sendo classe-B, onde A e B representam variações de uma família de *bots*, isso significa que o conjunto de carcaterísticas vigente não possui um elemento que expresse a diferença no comportamento observado para estas variações, neste caso, entre os domínios gerados por estes *bots*.

Outra hipótese para o impacto percebido neste trabalho, é que as diferenças entre as variações não estejam relacionadas ao comportamento dos *bots* para o protocolo DNS. Ou seja, essas variações podem utilizar o mesmo DGA em sua comunicação e suas diferenças correspondem a outros comportamentos do *bot*. A utilização do mesmo DGA

entre variações de um *bot* justifica-se uma vez que todos os membros da família deverão contactar o centro de comando e controle daquela *botnet*.

Porém, nesta metodologia, como todas as classes referem-se a um bot e nenhuma a um tráfego não-malicioso, o impacto gerado está apenas na predição de qual *botnet* foi detectada e não no processo de detecção da infecção em si. Ou seja, um domínio identificado como bot X, independente de pertencer a classe X ou não, continua sendo um domínio envolvido em atividade maliciosa.

6. Conclusão e Trabalhos Futuros

A arquitetura apresentada é capaz de detectar a presença de *botnet*s que utilizam DGA para o estabelecimento de sua comunicação entre os *bots* e servidores de C&C. Destaca-se que, para a geração dos modelos, foram utilizadas apenas as informações contidas no tráfego que seria descartado na rede: as respostas NXDOMAIN do protocolo DNS.

Foi utilizado um conjunto de características consideradas fundamentais em relação aos nomes de domínio gerados pelos *bots*. Essas características apresentaram um nível de detecção razoável durante os experimentos. Entretanto, esse conjunto de atributos mostrou-se insuficiente na predição entre *bots* de uma mesma família, gerando um aumento no número de falsos positivos. Como a metodologia manipula apenas tráfego malicioso, esses falsos alarmes ainda são identificados como domínios gerados por *bots*.

A arquitetura apresenta os seguintes pontos positivos:

- Pelo fato de basear-se no comportamento DNS apresentado por *bots*, a metodologia aqui apresentada não depende da arquitetura da *botnet* (centralizada, P2P ou híbrida);
- Por não inspecionar a carga (*payload*) dos pacotes contendo os comandos e dados manipulados pela *botnet*, limitando-se apenas ao protocolo DNS, essa metodologia é imune à criptografia, polimorfismo e técnicas semelhantes.

A camada de Monitoramento está em processo de desenvolvimento. Durante este processo, estão sendo identificados os critérios de agrupamento de tráfego citados na Seção 4. Outros comportamentos relacionados à presença de *bots* podem ser incorporados à arquitetura através do tratamento de suas características para que estas possam ser utilizadas para gerar os respectivos cenários e modelos de detecção.

Para diminuir o problema que ocorreu na classificação entre variantes de uma mesma família de *bots*, pode ser utilizado um conjunto de características maior que o utilizado nos experimentos. Porém, como os modelos são gerados a partir de tráfego malicioso apenas, um domínio detectado por um destes modelos continua sendo malicioso.

Podem ser utilizadas também técnicas capazes de detectar erros de digitação originadas de usuários legítimos, a partir de erros de digitação por exemplo. Estes, na maioria dos casos, resultarão em respostas NXDOMAIN e, neste caso, representam ruídos na classificação. Por fim, pode-se utilizar técnicas de classificação em tempo real a fim de minimizar o tempo de detecção e resposta.

Referências

Antonakakis, M., Perdisci, R., Dagon, D., Lee, W., and Feamster, N. (2010). Building a dynamic reputation system for dns. In *Proceedings of the 19th USENIX Confer-*

ence on Security, USENIX Security'10, pages 18–18, Berkeley, CA, USA. USENIX Association.

Antonakakis, M., Perdisci, R., Nadji, Y., Vasiloglou, N., Abu-Nimeh, S., Lee, W., and Dagon, D. (2012). From throw-away traffic to bots: Detecting the rise of dga-based malware. In *Proceedings of the 21st USENIX Conference on Security Symposium*, Security'12, pages 24–24, Berkeley, CA, USA. USENIX Association.

Bilge, L., Balzarotti, D., Robertson, W., Kirda, E., and Kruegel, C. (2012). Disclosure: Detecting botnet command and control servers through large-scale netflow analysis. In *Proceedings of the 28th Annual Computer Security Applications Conference*, ACSAC '12, pages 129–138, New York, NY, USA. ACM.

Bilge, L., Kirda, E., Kruegel, C., and Balduzzi, M. (2011). EXPOSURE: Finding malicious domains using passive dns analysis. In *Proceedings of the Network and Distributed System Security Symposium*, NDSS'11. Internet Society.

Cavallaro, L., Kruegel, C., and Vigna, G. (2009). Mining the network behavior of bots. Technical Report Tech. Rep. 2009-12, Department of Computer Science, University of California, Santa Barbara (UCSB), CA, USA.

Choi, H., Lee, H., Lee, H., and Kim, H. (2007). Botnet detection by monitoring group activities in dns traffic. In *Proceedings of the 7th IEEE International Conference on Computer and Information Technology*, CIT '07, pages 715–720, Washington, DC, USA. IEEE Computer Society.

Cisco (2007). Botnets: The new threat landscape. Technical report, Cisco Systems Inc, USA. White Paper.

de A Ribeiro, V., Filho, R., and Maia, J. (2011). Online traffic classification based on sub-flows. In *Integrated Network Management (IM), 2011 IFIP/IEEE International Symposium on*, pages 415–421, Dublin.

Erquiaga, M. J., Catania, C., and GarcÃa, S. (2016). Detecting dga malware traffic through behavioral models. In *2016 IEEE Biennial Congress of Argentina (ARGENCON)*, pages 1–6.

Gu, G., Perdisci, R., Zhang, J., and Lee, W. (2008). Botminer: Clustering analysis of network traffic for protocol- and structure-independent botnet detection. In *Proceedings of the 17th Conference on Security Symposium*, SS'08, pages 139–154, Berkeley, CA, USA. USENIX Association.

Hall, M., Frank, E., Holmes, G., Pfahringer, B., Reutemann, P., and Witten, I. H. (2009). The weka data mining software: An update. *SIGKDD Explor. Newsl.*, 11(1):10–18.

Holz, T., Gorecki, C., Rieck, K., and Freiling, F. C. (2008). Measuring and detecting fast-flux service networks. In *in NDSS. The Internet Society*.

John, G. H. and Langley, P. (1995). Estimating continuous distributions in bayesian classifiers. In *Eleventh Conference on Uncertainty in Artificial Intelligence*, pages 338–345, San Mateo. Morgan Kaufmann.

Karim, A., Salleh, R. S., Shiraz, M., Shah, S. A. A., Awan, I., and Anuar, N. B. (2014). Botnet detection techniques: review, future trends, and issues. *Journal of Zhejiang University-SCIENCE C (Computers & Electronics)*, 15(11):943–483.

Khandelwal, S. (2017). Wannacry kill-switch(ed) its not over! wannacry 2.0 ransomware arrives.

Quinlan, R. (1993). *C4.5: Programs for Machine Learning*. Morgan Kaufmann Publishers, San Mateo, CA.

Salusky, W. and Danford, R. (2007). Know your enemy: FastÂflux service networks.

Schiavoni, S., Maggi, F., Cavallaro, L., and Zanero, S. (2014). *Phoenix: DGA-Based Botnet Tracking and Intelligence*, pages 192–211. Springer International Publishing, Cham.

Sikorski, M. and Honig, A. (2012). *Practical Malware Analysis: The Hands-On Guide to Dissecting Malicious Software*. No Starch Press, San Francisco, 1st edition.

Stone-Gross, B., Cova, M., Cavallaro, L., Gilbert, B., Szydlowski, M., Kemmerer, R., Kruegel, C., and Vigna, G. (2009). Your botnet is my botnet: Analysis of a botnet takeover. In *Proceedings of the 16th ACM Conference on Computer and Communications Security*, pages 635–647. ACM.

Stratosphere, T. (2015). Malware capture facility project.

Tiirmaa-Klaar, H., Gassen, J., Gerhards-Padilla, E., and Martini, P. (2013). *Botnets*. Springer Publishing Company, Incorporated.

Zhao, D., Traore, I., Sayed, B., Lu, W., Saad, S., Ghorbani, A., and Garant, D. (2013). Botnet detection based on traffic behavior analysis and flow intervals. *Comput. Secur.*, 39:2–16.

