Geração Automática de Conhecimento para SDI extraído de OSINTs

Ivo Vacas, Ibéria Medeiros

LaSIGE, Faculdade de Ciências, Universidade de Lisboa, Portugal.
ivo.ricardo1@gmail.com,imedeiros@di.fc.ul.pt

Resumo O cibercrime a organizações tem sido uma constante nos dias de hoje. As organizações para se protegerem destes ataques utilizam mecanismos de defesa, tais como sistemas detetores de intrusões (SDI), no entanto, a eficácia dos SDIs na deteção destes ataques depende do conhecimento que estes contêm sobre as ameaças e da forma como as detetam. O contínuo aparecimento de novos e sofisticados ataques torna obrigatório que os SDIs sejam atualizados constantemente com conhecimento sobre novas ameaças. Este conhecimento pode ser obtido de diversas fontes de inteligência - *Open Source Intelligence* (OSINT) - públicas, que se encontram acessíveis em diversos locais na Internet. Este artigo apresenta uma solução para melhorar uma arquitetura de deteção de intrusões em SDI. A solução propõe um *gerador de regras e blacklists* para SDI, com base em informação OSINT recolhida por uma plataforma de *threat intelligence* bem como a sua integração no SDI de forma automática. Foi realizada uma avaliação experimental da solução em ambiente real, usando 49 fontes de OSINT coletadas pela plataforma *threat intelligence* IntelMQ e o SDI Snort. A arquitetura proposta permitiu detetar ameaças de diversas categorias.

1 Introdução

O cibercrime a organizações tem sido uma constante nos dias de hoje. Ataques a servidores organizacionais, tais como o *Distributed Denial of Service* (DDoS), *spamming* e propagação de *malware*, são cada vez mais frequentes, de tal forma que prevê-se que em 2019 o custo dos seus danos atinja 2 triliões de dólares [9]. Para a concretização destes ataques, geralmente é previamente necessário infetar as máquinas vítimas com *malware* ou *backdoors* para posteriormente os ciber-criminosos remotamente as poderem controlar e realizar os ataques [1]. As máquinas mais apetecíveis de infetar são as com acesso à Internet, pouca monitorização e grande largura de banda [3]. Quando comprometidas são chamadas de *bots* e as redes por elas formadas de *botnets*.

A utilização de *botnets* tem sido um dos recursos pertencentes aos vetores de ataque do cibercrime, em tal escala, que de acordo com o relatório de Akamai, estes ataques aumentaram em 9% do primeiro para o segundo quadrimestre de 2016 e em 129% comparativamente com o ano anterior [33]. Por exemplo, a Mirai *botnet* esteve na base deste aumento. Esta *botnet* é caracterizada pelos seus *bots* serem dispositivos IoT (*Internet of Things*), e em outubro de 2016 foi usada para atacar os servidores da

Dyn (organização que controla servidores de DNS da Internet), onde participaram 100.000 dispositivos IoT que geraram um volume de tráfego na ordem dos 1.2 Tbps, de tal forma que provocou uma interrupção do serviço de Internet em toda a Europa e EUA [4]. Neste ataque poderiam ter participado mais de 500.000 dispositivos IoT, por estarem vulneráveis à Mirai [29].

Face a este problema, as organizações para se protegerem utilizam sistemas detetores de intrusões (SDI) que monitorizam o tráfego da rede, detetando anomalias e gerando alertas destas. O modo primordial de funcionamento dos SDI é baseado no conhecimento do comportamento ou de assinaturas, que quando há um desvio de comportamento ou igualdade de assinatura, o SDI gera um alerta [7]. No entanto, estes sistemas são conhecidos por reportarem falsos positivos (alertas de falsas anomalias) devido a ligeiros desvios comportamentais que possam ocorrer, ao contrário dos falsos negativos (anomalias não detetadas) devido a não possuírem o conhecimento suficiente sobre todas as anomalias e ataques existentes [10].

Apesar da importância dos SDIs na deteção de ameaças, nomeadamente naqueles que utilizam *botnets*, existe a necessidade constante de fornecer conhecimento aos SDIs, pois novos e sofisticados ataques surgem muito rapidamente, deixando os SDIs desatualizados. A obtenção deste conhecimento, por um lado, é considerada privada e segredo de negócio [8], não estando portanto acessível a todos. Por seu turno, este conhecimento pode ser obtido de diversas fontes de inteligência – *Open Source Intelligence* (OSINT) – públicas e acessíveis em diversos locais na Internet. O OSINT contem informação partilhada proveniente de *honeypots* e de eventos reais que causaram algum tipo de dano em dados ou serviços, como o roubo de credenciais por ataques de *Phishing* ou DDoS [7]. Por vezes, esta informação por si só é reduzida e insuficiente para caracterizar ameaças em concreto e ser aplicada na sua detecção, no entanto, se agregada e complementada com outra informação é possível criar IOA (*Indicators of Attack*) [32] que permitem caracterizar com clareza ataques e serem usados na sua deteção.

Este artigo apresenta uma solução para melhorar uma arquitetura de deteção de intrusões em SDI. A solução propõe um *gerador automático de regras e blacklists* para o SDI, com base em IOAs gerados a partir da extração de conhecimento OSINT por uma plataforma de *threat intelligence*, bem como a sua completude e integração no SDI de forma automática. A solução foi implementada e avaliada num ambiente real, na Reitoria da Universidade de Lisboa, usando 49 fontes de OSINT coletadas pela plataforma *threat intelligence* IntelMQ, que após de filtradas e reunida informação necessária e suficiente para criar IOAs, o gerador automaticamente gera regras e *blacklists*, os quais são integrados de forma automática no SDI Snort. A solução proposta permitiu detetar ameaças de diversas categorias.

As contribuições deste trabalho são: 1) uma arquitetura para extração de conhecimento OSINT e criação de IOAs; 2) um gerador de regras e de *blacklists* para SDI a partir de IOAs; 3) uma implementação da solução de extração de OSINT e geração de regras e *blacklists*; 4) uma avaliação experimental da solução em ambiente real.

2 Botnets, ataques e SDIs

Nesta seção é apesentado brevemente como são compostas as *botnets* e ataques com elas realizadas, e as características principais de um sistema detetor de intrusões.

As *botnets* são redes de dispositivos vulneráveis que foram infetados, tais como computadores e dispositivos móveis, e controladas por entidades criminosas (*botmaster*) através de um centro de comando e controlo (C&C) [30]. O uso de *botnets* permite que essas entidades realizem ataques usando recursos que não lhes pertencem, mantendo-se assim anónimas [2]. Um dispositivo infetado (ex., pela instalação de *malware*) e controlado por um *botmaster* é apelidado de *bot*. Os *bots* recebem comandos do *botmaster* para realizarem ações ilícitas, mantendo o anonimato do seu dono.

O C&C das *botnets* é um alvo particularmente apetecível, quer para quem as tenta descobrir e destruir, quer para outros ciber-criminosos que pretendem apropriar-se delas para poderem usa-las em seu proveito. Os donos das *botnets* utilizam mecanismos criptográficos para protegerem a comunicação nas *botnets* e mecanismos de dissimulação para impedirem a sua identificação e desmantelamento [6], evitando assim a sua captura por sistemas de monitorização de redes de computadores, tais como os SDIs. Por exemplo, um mecanismo eficaz de dissimulação de *bots* é o de metamorfose, que permite que estes sejam difíceis de detetar pelos SDIs devido à sua constante mudança comportamental [5].

As *botnets* têm estado na base de diversos tipos de ataques, sendo o *SPAM*, propagação de *malware*, *ransomware*, *phishing*, e DDoS os ataques mais realizados. Existe uma grande dificuldade em eliminar estes ataques devido à constante sofisticação dos métodos de operação dos atacantes e do constante mascaramento das *botnets*.

Os SDI têm contribuído em muito na deteção destes ataques. Conseguem analisar o tráfego gerado nas instituições e reconhecer ameaças através de padrões sobre estas (ex., ataques). Um SDI é tão eficaz quanto o conhecimento que possui das ameaças. Os SDIs detetam intrusões *baseado em rede* e *baseado em host*. Os SDIs baseados em rede são caracterizados por analisarem tráfego de rede, enquanto os SDI baseado em *host* analisam um sistema (ex., computador). Ambas as formas de análise podem ser conseguidas pelos métodos baseados em *assinatura* ou *comportamento* [7].

Os SDI baseados em assinatura utilizam bases de dados de assinaturas de ameaças. Este método analisa os dados que chegam ao SDI, verificando se na base de dados existe uma assinatura que combine com a assinatura dos dados recebidos, denunciando assim uma intrusão. Apesar da trivial implementação de sistemas deste tipo, esta solução apenas permite detetar intrusões conhecidas e que constem na base de dados. Este método, em regra, tem a vantagem de possuir uma taxa de falsos positivos baixa e fornecer informação sobre os alarmes despoletados. Contudo, a taxa de falsos negativos poderá ser elevada devido às intrusões desconhecidas pelo SDI [10] [7].

Os SDI baseados em comportamento é o método mais investigado na área de deteção de intrusões. Tendo como primitiva o total conhecimento do correto funcionamento do sistema em que se insere, este tipo de SDI consegue detetar padrões comportamentais considerados anormais e gerar alarmes relativos a esses desvios comportamentais. Estes tipos de sistemas são complexos e envolvem um estudo extensivo e

constante do sistema a ser analisado. Contrariamente aos SDI's baseados em assinaturas, este método, em regra, tem uma taxa de falsos negativos muito baixa, conseguindo detetar intrusões com comportamentos desconhecidos e divergentes, como por exemplo, exploração de vulnerabilidades. Contudo, esta solução, tende a ter uma taxa de falsos positivos elevada, pois por qualquer desvio do comportamento conhecido é gerado um alerta, no entanto, o comportamento registado pode não ser malicioso.

3 Visão Geral da Arquitetura

A arquitetura da solução proposta é composta por três componentes: *recolha de informação*, *geração de conhecimento* e *deteção de incidentes*. A primeira recolhe e agrega informação de eventos de segurança de *feeds* OSINT, enquanto a segunda, com base nesta informação, gera conhecimento para SDI sob a forma de regras e *blacklists*, e a terceira deteta incidentes pela aplicação do conhecimento gerado.

As três componentes estão representadas na Figura 1. A componente de recolha de informação executa os seguintes passos: o *OSINT colector* recolhe os eventos de segurança dos diversos feeds (fontes) de OSINT. Seguidamente o módulo *Threat information extractor* processa os eventos, sob a forma de IOC (*Indicator of Compromise*, fragmentos de informação forense que podem ser utilizados para identificar potenciais actividades maliciosas num sistema [31] [32]), extraindo a informação relevante sobre ameaças, para seguidamente ser agregada por tipo de ameaça pelo módulo *Threat information aggregator*. A componente de geração de conhecimento, através do módulo *Event generator*, recebe a informação agregada, adiciona-lhe dados relevantes, transformando os IOCs em IOAs (*Indicators of Attack*, contêm informação que permite identificar ataques em execução [31] [32]), e armazena os IOAs gerados. Por fim, o *Rules & blacklist generator* recebe os IOAs e gera regras para o SDI, no formato específico do SDI, e *blacklists* com IPs maliciosos. A componente de deteção de incidentes, o *Rules & blacklist manager* gerencia as novas regras e *blacklists*, vindas da componente geração de conhecimento e do exterior (*Rules* na figura), para de seguida fornecer tal conhecimento ao SDI para este detetar incidentes.

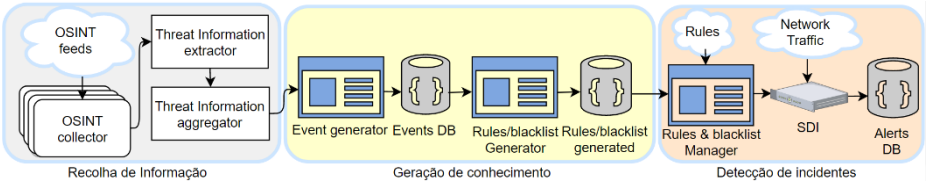

Figura 1. Arquitetura da solução proposta com os três componentes.

4 Implementação

A arquitetura proposta foi implementada usando a plataforma de *threat intelligence IntelMQ* para coletar eventos de 49 fontes OSINT de 8 categorias de eventos de segurança, extrair e agregar informação destes. Os módulos *event generator* e o *Rules &*

blacklists generator foram desenvolvidos por nós para, respetivamente, interagir com o IntelMQ e processar os seus resultados. A implementação foi obtida pelas fases de *gestão do conhecimento* e *deteção de incidentes*, como ilustra a Figura 2. A parte esquerda da figura, gestão do conhecimento, implementa as componentes de recolha de informação e geração de conhecimento da arquitetura proposta, enquanto a parte direita da figura implementa a componente de deteção de incidentes. A fase de gestão do conhecimento é executada diariamente para recolha e processamento de novos eventos de OSINT e geração de conhecimento para SDI, enquanto a fase de detecção de incidentes está em contínua execução. A seção apresenta de seguida os detalhes e funcionamento de cada um dos elementos pertencentes às duas fases.

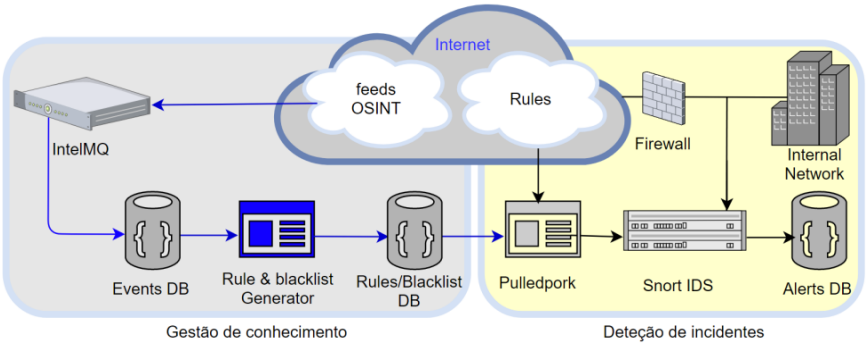

Figura 2. Implementação da arquitetura proposta, com as fases de gestão de conhecimento (esquerda) e deteção de incidentes (direita).

4.1 OSINT feeds

Com o objetivo de reduzir o cibercrime e salvaguardar a integridade intelectual, existem empresas, aficionados, investigadores e instituições (*feeds*) que disponibilizam informação à comunidade cibernética sobre incidentes de segurança (ex., sob a forma de IOC), tais como IPs maliciosos, sites de *phishing*, domínios perigosos, entre outras fontes relevantes. De acordo com a especialidade, alguns exemplos de *feeds* são o Phishtank [16] para *phishing* e CINSScore [17] para reputação de IPs. Esta informação se usada adequadamente e atempadamente pode prevenir a concretização de incidentes perigosos, tais como o roubo de credenciais. A arquitetura implementada foi configurada com 49 *feeds* de OSINT, a partir de 44 repositórios e que foram estratificados em 8 categorias de eventos de segurança. A Tabela 1 apresenta esta divisão pelas categorias de eventos OSINT (colunas 1 e 3), totalizando os 49 *feeds*.

Evento OSINT	**Feeds**	**Evento OSINT**	**Feeds**
Domínio Malicioso	9	IP's que atacam servidores de EMAIL	2
Blacklist IP	16	IP's que atacam serviços de VoIP	4
Phishing ou links perigosos	7	IP's que efetuam scans através de SNMP	1
IP's que atacam servidores FTP	1	IP's que tentam obter acesso remoto	9
Total	**33**	**Total**	**16**

Tabela 1. Categorização dos eventos de OSINT.

4.2 IntelMQ

A ferramenta IntelMQ foi desenhada pelo CERT (*Computer Emergency Response Team*) de alguns países Europeus com o objetivo de facilitar a recolha e tratamento de dados relativo a ameaças e melhorar a resposta a incidentes [11].

No IntelMQ, os *bots* são blocos de código modulares e independentes, que comunicam entre si sob a forma de mensagens em formato JSON, o que permite agilizar a comunicação entre eles. Existem diversos tipos de bots e para diversas funções.

O fluxo de dados do IntelMQ é apresentado na Figura 3. O fluxo inicia-se pelos *coletores*, que são os *bots* responsáveis pela recolha de informação de um dado *feed* OSINT e passagem desta aos *parsers*. Foram configurados 44 coletores OSINT de diversas entidades e respetivos *parsers*. O *parser* analisa a informação e cria eventos a partir desta que serão os *input* dos *experts*. O *deduplicator* remove os eventos duplicados, enquanto o *information expert* acrescenta informação relevante sobre o incidente, por exemplo o *source.asn* que representa o sistema autónomo de origem de um IP (à direita na figura). Os *experts* processam esta informação, obtendo mais informação ou despoletando novas ações, resultando os IOCs e eventos de *blacklist*. O *protocol expert* é um *bot* programado por nós para identificar o protocolo afeto ao IOC, com recurso a expressões regulares (*regex*), e preencher o campo *protocol.application* com esta informação (à direita da figura). Este campo é acrescentado aos IOCs, transformando-os assim em IOAs, e servirá para orientar o módulo *Rules & blacklists generator* na criação de regras e na decisão do tipo de regra a gerar. Desta forma, as regras criadas vão conter mais informação sobre o evento, minimizando assim a tendência de originar falsos positivos. Por fim, o *event writer* recebe os resultados do protocolo expert, formata-os segundo IOAs e armazena os IOAs em *events DB*. O *event writer* é um *output bot* que foi desenvolvido para este efeito.

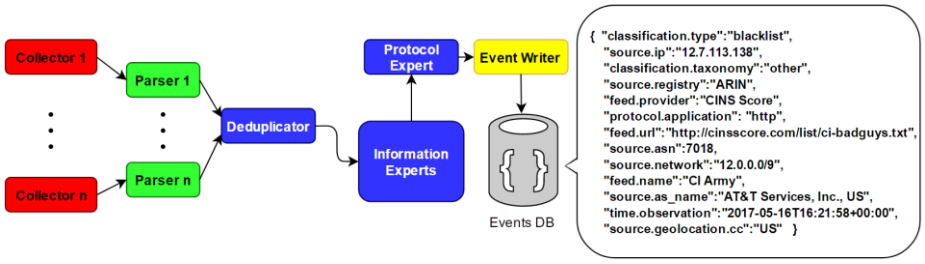

Figura 3. Fluxo de dados do IntelMQ.

4.3 Rules & Blacklists Generator

Este módulo foi desenvolvido para processar os IOAs e os eventos de *blacklist* armazenados em *events DB*, gerando assim as regras e *blacklists* para o SDI, respetivamente. A Figura 4 apresenta este módulo em detalhe. Com base na informação contida no campo *protocol.application* do IOA, o gerador obtém os protocolos e os portos para a regra a criar. De seguida, o gerador cria a regra com a restante informação contida nos IOAs e armazena-a na base de dados de regras (*Rules DB*). A partir dos

eventos de *blacklist*, o gerador extrai o IP e o insere na *blacklist*. O resultado do processamento destes eventos é uma lista negra (*blacklist*) de IPs marcados como maliciosos, a qual é armazenada na base de dados *blacklist* (*Blacklist DB*). Esta lista posteriormente servirá para alimentar o preprocessador de reputação do SDI Snort para verificar se é realizada comunicação para/com IPs maliciosos, a qual pode resultar na contração de vírus e *malware* por parte dos clientes.

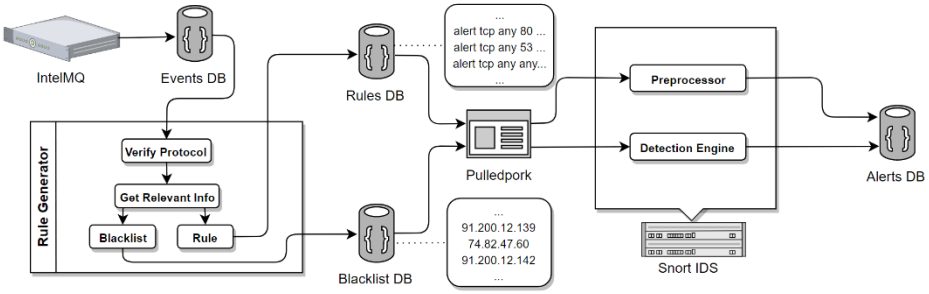

Figura 4. Criação de regras e *blacklists* a partir dos IOAs e eventos de *blacklist*.

4.4 PulledPork e SDI Snort

O PulledPork é um gestor de regras e *blacklists* provenientes de diversas entidades, tanto gratuitas como proprietárias. Como gestor, permite a coexistência de regras de diferentes provedores (ex., *Emerging Threats* [19] e *Talos* [18]), anulando possíveis colisões de regras e IPs destes e das que estão em produção no SDI. Também, gere o fornecimento das regras e IPs ao SDI. Neste sentido, o PulledPork recolhe as regras e as *blacklists* geradas pelo *Rules & blacklists Generator*, evita as possíveis colisões que estas podem originar e reencaminha-as para o Snort.

O Snort analisa todo o tráfego que flui entre a Internet e a intranet da instituição, utilizando estas regras e *blacklists*. As regras contêm padrões sobre ameaças que permitem identificar intrusões no tráfego analisado. As *blacklists* são listas de IPs com má reputação e conhecidos por realizarem diversas ações maliciosas (ex., DDoS). O preprocessador do Snort utiliza estas listas para identificar se houve algum tipo de comunicação por parte destes IPs.

De forma conjunta, o Snort utiliza as regras de alerta geradas pelo *Rule Generato*r no motor de alertas e os IP's *blacklist* no preprocessador de reputação.

5 Avaliação Experimental

A avaliação da solução proposta foi efetuada em ambiente real, na Reitoria da Universidade de Lisboa, na análise do tráfego da rede interna, durante 8 dias. Foram analisadas as redes do *Datacenter*, *Serviços Centrais da Reitoria* (utilizadores) e *Eduroam*. O desempenho da solução foi medido pela contabilização dos alertas advindos das regras e *blacklists* gerados pela solução e confirmação de parte destes por inferência de características conhecidas.

5.1 Set-up e performance da solução

A máquina utilizada na avaliação experimental foi um HP ProLiant DL360 G6 com 2 processadores Intel Xeon CPU X5550 a 2.67GHz, 12Gb de RAM, 165GB de disco rígido e 3 interfaces de rede, sendo uma das interfaces para ligação de fibra ótica a 10Gb/s e as restantes duas de *ethernet* a 1Gb/s. Foi instalado na máquina o sistema operativo *Security Onion*, uma distribuição Linux munida de diversas ferramentas de segurança e análise, entre elas o Snort. O IntelMQ foi instalado numa máquina virtual com 4 Gb de RAM e 17 GB de armazenamento.

De facto, só quando uma solução é colocada em produção é que se consegue validar, seja em configuração, qualidade ou performance, e apurar o quanto é necessário afinar [8]. O set-up da solução não fugiu a esta regra, verificando-se que o *hardware* da solução era insuficiente para aguentar as 3 redes. Perante este constrangimento foi necessário a manutenção constante para possibilitar a recolha dos dados da melhor forma. Contudo, a percentagem de pacotes perdidos não ultrapassou em média os 0.1%, sendo que a pior perda registada foi de 5%, em hora de ponta e tendo o sistema aproximadamente 12.000 regras e 16.500 IPs em funcionamento.

5.2 Registo de incidentes

Durante o período de avaliação da solução foram usados dois conjuntos distintos de regras e *blacklists*. Nos primeiros 3 dias, onde decorreu o teste da solução, foram utilizados os mesmos conjuntos de regras e *blacklists*. Nos restantes 5 dias foram sendo geradas novas regras e *blacklists* todos os dias, e atualizado o Snort. Durante o tempo em que a solução recolheu resultados, foram geradas em média 5.290 regras por dia e 14.590 entradas de *blacklist*.

Foram registados aproximadamente 5.5 milhões de incidentes. A Tabela 2 apresenta este valor estratificado pelas diversas categorias de incidentes. O tráfego gerado por IPs pertencentes a *blacklists* totaliza aproximadamente 70% dos incidentes. Este resultado pode advir do facto do preprocessador não permitir mais nenhum tipo de processamento após haver uma correspondência na *blacklist*.

Os eventos registados pelo processador das regras correspondem aos restantes 30% do total dos incidentes. Destes, a comunicação SSH por um IP considerado malicioso foi a categoria que registou mais alertas. Na categoria da comunicação com domínios maliciosos foi detetada padrões que possivelmente eram referentes a *bots* ou C&C, contudo, dada a larga quantidade de dados não foi humanamente possível confirmar se eram ou não de facto. O mesmo sucedeu com os alertas de IMAP, SMTP e FTP. Alguns dos pedidos SIP OPTIONS foram verificados e provinham maioritariamente de uma *botnet* denominada *Friendly-scanner* [24] que utiliza como recurso a ferramenta *SIPVicious* [25]. O mesmo foi verificado nos pedidos de SIP REGISTER. Os restantes alertas foram gerados por comunicações de IPs maliciosos. Alguns pedidos SNMP foram também confirmados, onde foi possível verificar pedidos realizados com os comandos *get* e *get-next*, cujo objetivo é recolher informação detalhada de equipamentos que poderá ser utilizada em ataques a estes mesmos equipamentos.

Evento	Nº Alertas
Tráfego gerado por um IP em Blacklist	3.779.638
Comunicação SSH por parte de um IP malicioso	1.575.251
Comunicação com Domínio malicioso	45.081
Pedido SIP OPTIONS por IP malicioso	22.126
Comunicação de IP malicioso a servidores WEB	11.770
Comunicação IMAP por parte de um IP malicioso	7.230
Comunicação SMTP por parte de um IP malicioso	4.775
Comunicação FTP por parte de um IP malicioso	1.536
Pedido SIP REGISTER por IP malicioso	1.328
Comunicação Telnet por parte de um IP malicioso	1.003
Comunicação SNMP por parte de um IP malicioso	791
Comunicação SIP por parte de um IP malicioso	369
Total	**5.450.898**

Tabela 2. Registo de incidentes por categoria de evento.

A Tabela 3 mostra o número total de alertas gerados por cada um dos 8 dias. Em média, foram gerados aproximadamente 540.000 alertas nos sete primeiros dias. No último dia houve um aumento significativo de alertas, que possivelmente é justificado com o dia em que surgiu o surto do *ransomware Petya* [28].

Dia 1	Dia 2	Dia 3	Dia 4	Dia 5	Dia 6	Dia 7	Dia 8	Total
410.795	523.005	411.344	505.541	472.309	840.062	605.033	1682.809	**5.450.898**

Tabela 3. Número de alertas por dia.

Relativamente à informação gerada pelas regras de *blacklist*, foram registadas os portos de origem e de destino dos pacotes *blacklist* para melhor compreensão do objetivo da comunicação. A Tabela 4 apresenta estes dados.

Porto de origem	Nº Alertas
62572	126.073
0	125.324
10000	75.689
55763	56.497
52638	44.592
53671	42.513
65535	35.921
58022	35.411
52925	30.241
6000	23.158
Total	**59.5419**

Porto de destino	Nº Alertas
23	568.242
1433	469.893
22	386.229
43526	125.788
0	125.315
5060	108.238
1900	95.962
80	87.001
1443	81.796
443	70.105
Total	**2.118.569**

Tabela 4. Número de eventos *blacklist* registados por porto origem e destino.

Os portos de origem (parte esquerda da tabela) são maioritariamente dinâmicos com a exceção do 6000 e do 0. O porto 6000 é respeitante ao sistema X11 e alguns tipos de *trojans* [26], enquanto o porto 0 é utilizado para recolha de informação, apesar de ser considerado proibido [27]. Analisando os portos de destino (parte direita da tabela) conclui-se que o serviço mais acedido pelos IPs em *blacklist* foi o Telnet (23),

seguidamente dos portos utilizados pelos serviços MySQL (1433) e SSH (22). Curiosamente o quarto porto mais acedido (43526) é um porto utilizado dinamicamente, podendo revelar que uma máquina foi comprometida e está a comunicar com um IP de *blacklist*. Os restantes portos mais acedidos foram os de comunicação VoIP (5060), SSDP (1900), HTTP (80), IES (1443) e HTTPS (443). Estes resultados mostram que os IPs em *blacklist* devem estar bloqueados em qualquer instituição devido à quantidade de acessos indevidos a serviços, como os de SSH e Telnet.

6 Trabalho Relacionado

Zeng *et al.* apresentaram uma arquitetura híbrida (baseada em rede e em host) para a deteção de *botnets* que tinha como principal caraterística a interseção de dados provenientes da rede e dos computadores. Numa primeira fase, a arquitetura, através da análise de tráfego, identifica computadores suspeitos, cujo comportamento seja muito diferente dos restantes. Numa segunda fase, a arquitetura através de análise baseada em host valida se esses computadores são de facto suspeitos. Este método permite uma avaliação mais precisa, pois para além de analisar o comum comportamento de coordenação intrínseco às *botnets*, também realiza uma análise ao seu comportamento como um só [22].

Sperotto *et al*. propuseram um SDI, que por questões de performance e em redes de alta velocidade (ex., 10Gbps), analisa o fluxo do tráfego na rede em vez de analisar o pacote todo. Contudo, o SDI só consegue detetar ataques de negação de serviço, *scans*, *worms* e *botnets*, devido ao facto de somente analisar os cabeçalhos dos pacotes. Contrariamente à solução proposta neste artigo, este tipo de SDI não consegue detetar ataques de *Phishing*, por exemplo, uma vez que é necessário analisar o conteúdo do pacote [23].

Um survey relativo à deteção de *botnets* mostra que, apesar do avanço em alguns métodos de deteção, poucos foram bem-sucedidos e que a técnica mais usada na deteção é a baseada em anomalias. Contudo, continua a existir uma grande dificuldade no desmantelamento de *botnets* porque os *botmasters* continuam a evoluir as técnicas de mascaramento [10]. Aviv *et al.* propõe uma cooperação de diversas entidades para desenvolvimento de mecanismos para deteção de *botnets*, uma vez que estas entidades possuem informação confidencial [8].

Para além do funcionamento *standard* dos SDI, existem SDI específicos para um determinado tipo de ataque e que utilizam técnicas que não as *standard*. O FeatureSmith é um sistema de deteção de *malware* em aplicações android cujas funcionalidades assentam em mineração de dados. Para a aplicação de mineração de dados o sistema coletou dados sobre *malware* em android de 1068 artigos científicos do *google scholar*, onde através do relacionamento semântico com o comportamento de *malware* e do seu mapeamento para funcionalidades o sistema conseguiu uma taxa de deteção de 92.5% de verdadeiros positivos, com uma taxa de 1% de falsos positivos [21]. Ao contrário do FeatureSmith, a solução apresentada baseia-se em deteção de padrões intrusões conhecidos, recolhidos de eventos OSINT e processados para criação de regras e identificação de endereços IP maliciosos, que serão usados pelo SDI.

O MISP é uma plataforma colaborativa de partilha de informação de ameaças de segurança. A sua arquitetura baseia-se em organização e comunidade. Estes dois grupos partilham informação de forma simples e controlada com o objetivo de alertar as entidades pertencentes à organização ou a determinada comunidade [20]. Tal como a solução apresentada, esta plataforma consegue gerar regras para SDI, contudo, quando aplicados num sistema em produção, a sua performance torna a sua utilização impraticável, não tendo portanto a utilidade pretendida.

7 Conclusão

Este artigo apresentou uma arquitetura de melhoramento para deteção de incidentes por um sistema detetor de intrusões (SDI), baseado em conhecimento extraído de eventos de segurança OSINT. A arquitetura é composta por três componentes, que combinados, permitem a extração de conhecimento OSINT para geração de novo conhecimento sob a forma de regras de SDI e *blacklists*, e a deteção de incidentes usando este conhecimento.

A arquitetura foi implementada e avaliada em ambiente real, na Reitoria da Universidade de Lisboa, mostrando-se efetiva na deteção e registo de incidentes.

8 Agradecimentos

Este trabalho foi parcialmente suportado pela EC através do projeto H2020-700692 (DiSIEM) e pelos fundos nacionais através da Fundação para a Ciência e a Tecnologia (FCT) com referência para UID/CEC/00408/2013 (LaSIGE).

9 Bibliografia

1. P. Bächer, T. Holz, M. Kötter, G. Wicherski, 30/11/2016.: “Know your Enemy: Tracking Botnets,” Honeynet, https://www.honeynet.org/book/export/html/50.
2. I. Shakeel, 1/12/2016.: “Evolution in the World of Cyber Crime,”, http://resources.infosecinstitute.com/evolution-in-the-world-of-cyber-crime/#gref.
3. R. Puri, Bots & Botnets: An Overview, SANS Institute, 2003.
4. SearchSecurity, 12/1/2017.: "Details emerging on Dyn DNS DDoS attack, Mirai IoT botnet", http://searchsecurity.techtarget.com/news/450401962/Details-emerging-on-Dyn-DNS-DDoS-attack-Mirai-IoT-botnet.
5. M. Rouse, 5/12/2016.: “Metamorphic and polymorphic malware,”, http://searchsecurity.techtarget.com/definition/metamorphic-and-polymorphic-malware.
6. J. M. Butle, Finding Hidden Threats by Decrypting SSL, SANS Institute, 2013.
7. G. Bruneau, The History and Evolution of Intrusion Detection, SANS Institute, 2001.
8. A. H Haeberlen, A. Aviv: Challenges in Experimenting with Botnet Detection Systems. In: Proceedings of the 4th Conference on Cyber Security Experimentation and Test. CSET’11. pp 6-15 (2011).

9. S. Morgan, 25/06/2017 “Cyber Crime Costs Projected To Reach $2 Trillion by 2019 (2017).”, https://www.forbes.com/sites/stevemorgan/2016/01/17/cyber-crime-costs-projected-to-reach-2-trillion-by-2019/
10. S. Silva, R. Silva, R. Pinto, R. Salles.: Botnets: A Survey. Computer Network 57(2), 378-403 (2012).
11. n0where, 10/01/2017.: “Automate Incident Handling Process: IntelMQ.”, https://n0where.net/automate-incident-handling-process-intelmq/.
12. Shirkdog, 29/05/2017.: “PulledPork.”, https://github.com/shirkdog/pulledpork
13. Suricata, 21/01/2017.: “Suricata.”, https://suricata-ids.org/.
14. Snort, 21/01/2017.: “Snort.”, https://www.snort.org.
15. N. Dietrich. 1/06/2017.: “The Reputation Preprocessor in Snort – Blacklists and White-lists.”, http://sublimerobots.com/2015/12/the-snort-reputation-preprocessor/
16. OpenDNS, 20/06/2017.: “Phishtank.”, https://www.phishtank.com
17. CINScore, 20/06/2017.: “The CINS Score.”, http://cinsscore.com
18. Snort, 21/10/2017.: “Snort.”, https://www.snort.org/advisories/
19. Emerging Threats.net, 20/06/2017.: “Open rulesets.”, https://rules.emergingthreats.net/
20. Wagner, C., Dulaunoy, A., Wagener, G., & Iklody, A. (2016, October). MISP: The Design and Implementation of a Collaborative Threat Intelligence Sharing Platform. In Proceedings of the 2016 ACM on Workshop on Information Sharing and Collaborative Security (pp. 49-56). ACM.
21. Zhu, Z., & Dumitras, T. (2016, October). FeatureSmith: Automatically Engineering Features for Malware Detection by Mining the Security Literature. In Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security (pp. 767-778). ACM.
22. Zeng, Y., Hu, X., & Shin, K. G. (2010, June). Detection of botnets using combined host- and network-level information. In Dependable Systems and Networks (DSN), 2010 IEEE/IFIP International Conference on (pp. 291-300). IEEE.
23. Sperotto, A., Schaffrath, G., Sadre, R., Morariu, C., Pras, A., & Stiller, B. (2010). An overview of ip flow-based intrusion detection. IEEE Communications Surveys and Tutorials, 12(3), 343-356.
24. M. Kezys, 27/06/2017.: “SIP Attack: Friendly-Scanner.”, http://blog.kolmisoft.com/sip-attack-friendly-scanner/
25. S. Gauci, 27/06/2017.: “SIPVicious.”, https://github.com/EnableSecurity/sipvicious
26. Speedguide, 27/06/2017.: “Speedguide”, https://www.speedguide.net/port.php?port=6000
27. B. Mitchell, 28/06/2017.: “Port 0 in TCP and UDP.”, https://www.lifewire.com/port-0-in-tcp-and-udp-818145
28. A. H. Olivia Solon, 28/07/2017.: “'Petya' ransomware attack: what is it and how can it be stopped?”, https://www.theguardian.com/technology/2017/jun/27/petya-ransomware-cyber-attack-who-what-why-how
29. SecurityWeek, 17/01/20017.: “Over 500,000 IoT Devices Vulnerable to Mirai Botnet.”, http://www.securityweek.com/over-500000-iot-devices-vulnerable-mirai-botnet
30. N. Gamer, 1/12/2016.: “The state of botnets in late 2015 and early 2016.”, http://blog.trendmicro.com/the-state-of-botnets-in-late-2015-and-early-2016/.
31. N. Lord, 27/07/2017.: “What are indicators of compromise?,”, https://digitalguardian.com/blog/what-are-indicators-compromise.
32. Intel Security McAfee, 27/07/2017.: “Indicators of Attack (IoA),” Intel Security McAfee, Santa Clara, CA (2014).
33. Akamai, “Akamai's State of Internet / Security,” Akamai, Massachusetts, 2016.