Análise Preliminar da Detecção de Ataques Ofuscados e do Uso de Hardware de Baixo Custo em um Sistema para Detecção de Ameaças*

Lucas Seiki Oshiro[1], **Daniel Macêdo Batista**[1]

[1]Departamento de Ciência da Computação – IME
Universidade de São Paulo (USP) – São Paulo, SP, Brasil

lucas.oshiro@usp.br, batista@ime.usp.br

Abstract. *Network traffic transmitted at high rates results in the need for more efficient security mechanisms, since analyzing package by package before taking an action becomes a costly task in terms of processing. One way to solve this problem is with the development and deployment of threat detection systems that use machine learning mechanisms to anticipate attacks. This paper presents the preliminary results obtained in an attempt to improve such a system by analyzing automated attacks that employ obfuscation and by evaluating the performance of a Raspberry Pi unit that can be used as a processing node in the improved system.*

Resumo. *Tráfego de rede transmitido a altas taxas traz como consequência a necessidade de mecanismos de segurança mais eficientes, já que analisar pacote por pacote antes de tomar uma ação torna-se uma tarefa custosa em termos de processamento. Uma forma de resolver esse problema é com o desenvolvimento e implantação de sistemas de detecção de ameaças que utilizem mecanismos de aprendizado de máquina para antecipar os ataques. Este artigo apresenta os resultados preliminares obtidos na tentativa de melhorar um sistema como esse por meio da análise de ataques automatizados que empregam ofuscação e por meio da avaliação de desempenho de uma unidade Raspberry Pi que poderá ser usada como nó de processamento no sistema melhorado.*

1. Introdução

,Com a obtenção de quantidades massivas de dados de tráfego e de aplicações de rede, surge a necessidade de métodos mais "inteligentes" para buscar incidentes de segurança, principalmente porque passa a ser possível encontrar informações novas por meio da correlação das informações e também porque uma análise de força-bruta levaria muito tempo para ser finalizada. Essa necessidade já vem sendo discutida há alguns anos tanto na academia [Brown et al. 2015] quanto na indústria, que já fornece diversos serviços para clientes dos mais diversos tamanhos [Splunk 2019]. Entretanto, as implementações dos métodos não seguem uma arquitetura que seja eficiente para todos os tipos de organizações. De fato, tem-se notado que cada vez mais as arquiteturas precisam ser

*Este artigo resume os resultados parciais do trabalho de conclusão de curso que encontra-se em desenvolvimento por Lucas Seiki Oshiro e que pode ser acompanhado em `https://linux.ime.usp.br/~lucasoshiro/mac0499/`. Último acesso em 22 de Março de 2019.

otimizadas e personalizadas para cada tipo de usuário [Feth 2015]. Além disso, é desejável que o sistema a ser desenvolvido seja capaz de antecipar, ao máximo possível, incidentes em tempo real e que o mesmo não tenha um custo elevado tanto em termos financeiros para adquirir equipamentos, quanto em termos de consumo de energia. Uma forma de antecipar incidentes é com a utilização de mecanismos baseados em aprendizado de máquina. Uma forma de reduzir os custos é com a utilização de clusters baseados em hardware de baixo custo.

No escopo do projeto **GT-BIS – Mecanismos para Análise de Big Data em Segurança da Informação** [GT-BIS 2018], foi desenvolvido um sistema capaz de detectar ataques a partir da análise de grandes volumes de logs de servidores web e de servidores de banco de dados, por meio de técnicas de aprendizado de máquina. No momento é necessário melhorar o sistema com a adição da detecção de novos tipos de incidentes de segurança, como por exemplo aqueles causados por ataques automatizados que usam ofuscação e que burlam mecanismos mais tradicionais de proteção.

Nesse sentido, esse artigo apresenta resultados parciais obtidos com a análise de diferentes ataques contra aplicações web e com a análise de desempenho de uma unidade Raspberry Pi 3 Model B em cenários de uso intensivo de CPU. Como resultado foi possível identificar algumas características dos ataques que podem ser úteis para o treinamento de um modelo de detecção de ameaças. Já os resultados da análise de desempenho da Raspberry Pi mostraram que ela apresenta boa vazão de tráfego de rede e temperatura dentro de limites seguros mesmo quando submetida a altas cargas de processamento.

As próximas seções estão organizadas da seguinte forma. A Seção 2 resume a arquitetura do sistema considerado e os ataques estudados no trabalho. A Seção 3 descreve as ferramentas que foram utilizadas na criação de um ambiente de experimentação para reprodução e detecção dos ataques. A Seção 4 descreve os experimentos realizados e os resultados obtidos. A Seção 5 finaliza o artigo com as conclusões e os próximos passos.

2. Conceitos Básicos

Este artigo apresenta resultados visando melhorar o sistema de detecção de ameaças ilustrado na Figura 1. Os números na figura descrevem os dados que são passados entre cada componente do sistema: **1)** logs de serviços e de sistemas de segurança. **2)** dados normalizados, filtrados e enriquecidos. **3)** fluxos de dados organizados em filas para serem processados. **4)** informações brutas e de dados já processados (p. ex. alertas); comandos de consultas. **5)** dados para serem persistidos ou recuperados. **6)** alertas de ameaças cibernéticas compartilhados pelo sistema central ou parceiros. **7)** dados da interação do administrador com a interface Web. **8)** informações disponíveis nos componentes de Processamento (alertas, dados brutos, dados processados); comandos de consultas.

`Logstash`, `Kafka`, `Spark` e `ElasticSearch`, apresentadas na Figura 1, são ferramentas escaláveis e paralelizáveis de software livre para, respectivamente, filtrar e normalizar dados de logs, implementar um sistema produtor/consumidor para acesso a filas de dados, processar dados empregando por exemplo mecanismos de aprendizado de máquina e armazenar e visualizar dados. Todas essas ferramentas podem ser replicadas em diferentes nós em *clusters* a fim de tornar o funcionamento escalável em função da carga de dados de entrada e é nesse sentido que propomos a utilização de unidades de Raspberry Pi em cada um desses *clusters*.

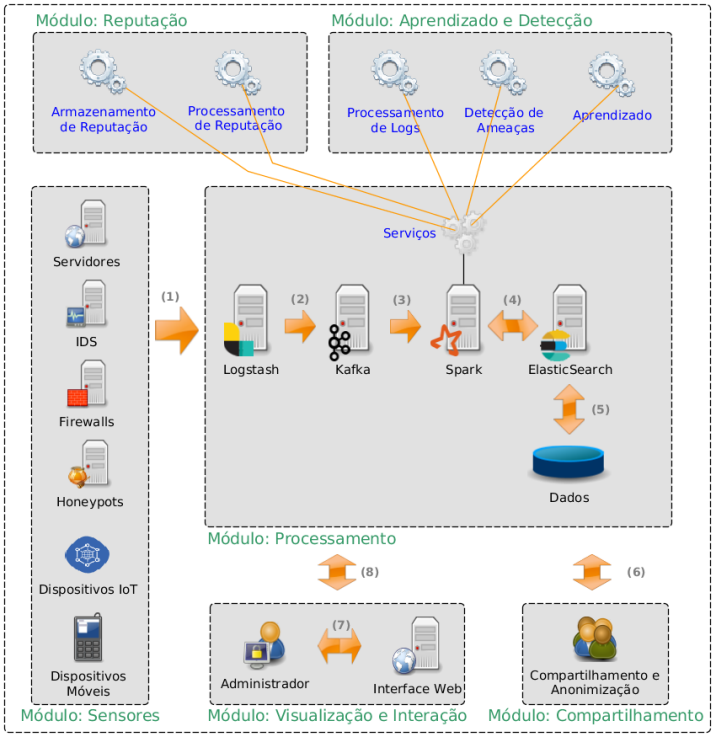

Figura 1. Arquitetura do sistema para detecção de ameaças por meio da análise de logs [GT-BIS 2018]

2.1. Ataques Contra Aplicações Web

Ataques contra redes de computadores ocorrem por conta de vulnerabilidades em diversas camadas da arquitetura Internet. Especificamente no caso de aplicações web, aquelas projetadas para serem acessadas principalmente por meio de um navegador web tendo o protocolo HTTP como base, falhas podem levar à negação do serviço, ao acesso indevido a informações ou mesmo à execução de comandos arbitrários do lado do servidor.

Pelo fato do protocolo HTTP ser um protocolo sem estado, boa parte das aplicações web requer a utilização de algum Sistema Gerenciador de Banco de Dados (SGBD). Tais sistemas costumam ser alvos de atacantes em busca de falhas que possam afetar a infraestrutura de uma aplicação web. Outro alvos para a exploração de falhas é a construção de URLs maliciosas.Alguns exemplos de ataques desses tipos são: **Injeção de SQL** – inserção de dados que, se não tratados, realizam ações indevidas no banco de dados; **XSS Refletido** – inserção de código HTML e JavaScript malicioso em uma URL, que é executado quando um site é aberto por meio daquela URL; **XSS Persistido** – envio de código HTML e JavaScript malicioso, que fica armazenado e é executado no navegador de todos os usuários que acessarem o site.

A distinção de ataques como os listados acima no meio de acessos legítimos pode ser dificultada caso o atacante utilize técnicas de ofuscação facilitadas por ferramentas que automatizam a exploração desses ataques.

2.2. Plataformas de Computação de Baixo Custo

Muitas aplicações para Internet das Coisas [Singh and Kapoor 2017] dependem de placas e sistemas embarcados que funcionem tanto como sensores, obtendo informações

do mundo real, quanto como processadores, analisando os dados capturados e enviando comandos de atuação. A necessidade de implantação desses elementos em larga escala justifica a utilização de hardware de baixo custo. Dentre as várias opções de hardware para esse objetivo destacam-se o Arduino [Arduino 2019] e a Raspberry Pi [Raspberry Pi Foundation 2018]. O Arduino é uma plataforma de código aberto para computação física baseada em entradas e saídas simples [Banzi 2011], enquanto que a Raspberry Pi, embora também seja um dispositivo em placa única, é um computador e possui maior poder de processamento que o Arduino. O grande sucesso na utilização desses dispositivos em aplicações de Internet das Coisas tem levado à sua utilização em outros domínios de aplicação. No caso da Raspberry Pi, vários projetos tem utilizado diversas unidades conectadas via rede criando assim um *cluster* de baixo custo [Pahl et al. 2016]. No Brasil, uma unidade Arduino UNO pode ser adquirida por cerca de R$50,00, enquanto uma unidade Raspberry Pi 3 Model B pode ser adquirida por cerca de R$200,00.

Antes de considerar a utilização dessa plataforma como nó de processamento em algum dos componentes de um sistema de detecção de ataques como o ilustrado na Figura 1 é importante analisar o seu desempenho em situações de uso intensivo de CPU.

3. Ambiente de Experimentação

Dois conjuntos de experimentos foram realizados. O primeiro conjunto teve por objetivo a análise dos logs gerados nos diversos servidores envolvidos na execução de uma aplicação web em situações de ataques. O segundo conjunto teve por objetivo avaliar uma unidade Raspberry Pi em situações de uso intensivo da CPU.

3.1. Simulação de Ataques

Optou-se por utilizar uma aplicação web já existente, a *Damn Vulnerable Web Application* (DVWA)[1]. A DVWA é uma aplicação Web disponibilizada para profissionais de segurança testarem suas ferramentas e habilidades. Cada página disponibilizada na DVWA possui uma vulnerabilidade.

A utilização da DVWA tinha por objetivo garantir que o servidor web instalado teria uma aplicação com as vulnerabilidades esperadas, de modo que o sucesso, ou insucesso, dos ataques pudesse ser de fato avaliado a partir dos logs do servidor e do comportamento da aplicação. A DVWA versão 1.10 foi instalada em uma máquina virtual (MV) gerenciada pelo VirtualBox versão 5.2. As configurações de hardware da MV eram: 4GB de memória RAM, 10GB de armazenamento, usando apenas 1 núcleo do processador. As configurações de hardware da máquina física utilizada eram: 8GB de memória RAM, 200GB de disco rígido e processador Intel Core i5, com 2 núcleos e 4 *hyperthreads* a 2,7GHz. As configurações de software da MV eram: Ubuntu 16.04 LTS, servidor web Apache 2.0, SGBD MySQL 14.14 e PHP 7.0. As configurações de software da máquina física eram: sistema operacional Manjaro Linux 17.1 Hakoila, com kernel Linux 4.14 LTS. O MySQL precisou ser configurado para produzir um arquivo de log com os acessos feitos a ele. Essa configuração não vinha por padrão no software.

Os ataques realizados contra a DVWA foram realizados em uma MV gerenciada no VirtualBox na mesma versão e na mesma máquina física. As configurações de hardware da MV eram: 3GB de memória RAM, 20GB de armazenamento, usando apenas

[1] `http://www.dvwa.co.uk/`. Último acesso em 22 de Março de 2019.

1 núcleo do processador. O sistema operacional usado nessa máquina foi o Kali Linux versão 4.15. O Kali Linux é uma distribuição Linux baseada no Debian, contendo software especializado para a execução de testes de penetração e auditoria de segurança. As ferramentas utilizadas para a realização dos ataques foram o SQLMap versão 1.2.4 para automatizar os ataques de injeção de SQL, o XSSer versão 1.7 para automatizar os ataques de *Cross-Site Scripting* (XSS) refletido e persistido e o Burp Suite versão 1.7.33 para interceptar as informações transmitidas pelo navegador utilizado nos experimentos. A interconexão entre as MVs foi feita no modo rede exclusiva de hospedeiro no VirtualBox.

3.2. Análise de Desempenho da Raspberry Pi

A Raspberry Pi utilizada foi do modelo Raspberry Pi 3 Model B. Seu processador é um Broadcom BCM2837 64bit, com 4 núcleos a 1,2GHz, e sem dissipador de calor; sua memória RAM tem 1GB; o sistema operacional utilizado foi o Raspbian 2018-06-27. Para auxiliar as análises, foi utilizada a mesma máquina hospedeira das MVs usadas para os ataques. As máquinas estavam conectadas fisicamente à Raspberry Pi via Fast Ethernet com um roteador Linksys WRT54G.

4. Resultados

4.1. Injeção de SQL

Os ataques de injeção de SQL foram feitos conforme o fluxo de trabalho da Figura 2. Ao término das ações foram obtidos os nomes de todos os usuários da base de dados e os *hashes* de suas senhas. Através de força bruta baseada em dicionário com o software SQLMap, foi possível descobrir todas as senhas.

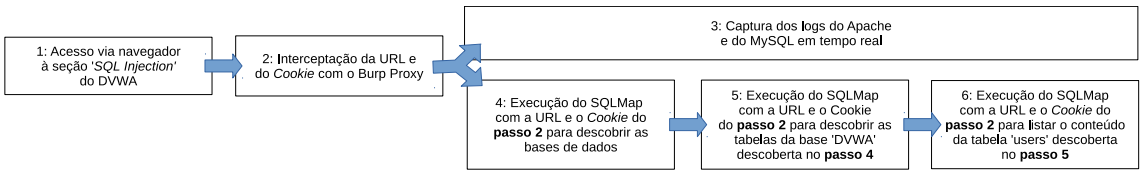

Figura 2. Passo-a-passo da realização dos ataques de injeção de SQL

Com a coleta dos logs realizada no passo 3 foi possível observar as seguintes características no log do Apache: URLs de acesso com grande quantidade (em geral, mais de 5) de caracteres em notação hexadecimal que são representados pelo caracter `%` seguido de um número hexadecimal; Grande uso de palavras reservadas da linguagem SQL como `SELECT`, `AND`, `OR`, `NOT`, `WHERE`, `BOOLEAN`; Intervalo de tempo entre acessos de um mesmo endereço IP muito curto (foram mais de 30 acessos por segundo); Embora seja algo fácil de modificar, o agente de usuário de todos os acessos registrados foi `sqlmap`.

No log do MySQL foi observada uma grande frequência de comparações do tipo "numero1=numero2" na cláusula `WHERE` das consultas, sendo ambos os números escritos explicitamente. Essas comparações vêm acompanhadas dos booleanos `AND` e `OR`.

Considerando a criação de um modelo baseado nas características listadas acima, a presença de apenas uma delas não significa a certeza de um ataque. No caso de caracteres hexadecimais por exemplo, isso pode ocorrer caso o caracter sendo passado na URL não possa ser representado na notação ASCII. Já no caso de vários acessos de um mesmo

IP, isso pode ocorrer por conta da utilização de NAT. Considerar apenas uma das características pode levar a uma alta taxa de falsos positivos, entretanto, como eles estiveram sempre presentes nos ataques realizados pelo SQLmap, eles representam potenciais candidatos de *features* a serem analisadas por algoritmos de aprendizado de máquina. Claro que o comportamento normal dos logs deve ser estudado para avaliar o peso dado a cada uma das *features* reduzindo a chance de falsos positivos e de falsos negativos.

4.2. XSS Refletido

Os ataques de XSS refletido foram feitos conforme o fluxo de trabalho da Figura 3. Ao término das ações foi possível acessar com sucesso a DVWA por meio de URLs contendo código HTML malicioso inserido pelos ataques.

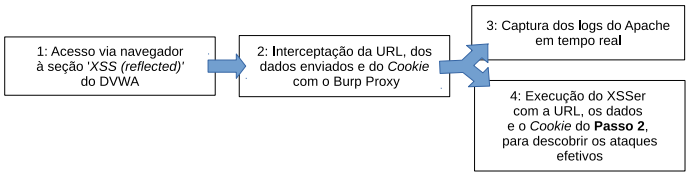

Figura 3. Passo-a-passo da realização dos ataques de XSS Refletido

Com a coleta dos logs realizada no passo 3 foi possível observar as seguintes características no log do Apache: URLs de acesso com código HTML inserido; Intervalo de tempo entre acessos de um mesmo endereço IP muito curto (foram cerca de 300 acessos por segundo); Embora seja algo fácil de modificar, o agente de usuário de todos os acessos registrados foi `Googlebot/2.1 (+http://www.google.com/bot.html)`.

Vale observar a similaridade com algumas das características presentes durante os ataques de Injeção de SQL. Os mesmos comentários anteriores sobre a consideração dessas características para a criação de um modelo continuam pertinentes.

4.3. XSS Persistido

Os ataques de XSS persistido foram feitos de forma similar ao fluxo de trabalho da Figura 3, com a diferença de que a seção acessada na DVWA foi a *XSS (store)* (Passo 1) e que os logs do MySQL também foram monitorados (Passo 3). Apesar do XSSer não identificar nenhum ataque como bem sucedido, a página da DVWA referente ao XSS persistido sempre redirecionou para uma página diferente, efeito desse ataque.

Diferente dos experimentos anteriores, o servidor web precisou ser configurado para registrar nos logs os dados enviados pelo método `POST` do HTTP. Esse procedimento, porém, não é indicado uma vez que podem ser fornecidos dados muito grandes através desse método. Além disso, a quantidade de entradas no *log* é muito grande. No experimento foram executados 558 envios de formulários, e a partir deles foram geradas 46497 entradas no *log*, sendo que a maior parte possui informações pouco relevantes.

Com a coleta dos logs realizada no passo 3 foi possível observar as mesmas características observadas durante os experimentos de XSS refletido no log do Apache, com a diferença de que as *tags* HTML estavam incompletas. Como os dados fornecidos são armazenados no banco de dados, as inserções no banco de dados observadas no log do MySQL também contém essas *tags*.

4.4. Análise de Desempenho da Raspberry Pi

A fim de avaliar de forma preliminar a eficiência da Raspberry Pi como nó de um cluster para análises em busca de ameaças de segurança, ela teve seu processador estressado pela ferramenta `stress` com diferentes números de processos "pesados". Cada um desses processos é chamado de *worker*. Avaliou-se vazão da rede e temperatura da placa. A primeira métrica foi avaliada para verificar se a unidade era capaz de receber dados mesmo se estivesse com utilização elevada da CPU, simulando uma situação em que dados de logs para análise estejam constantemente sendo enviados em paralelo à análise desses dados. A ferramenta `iperf` foi utilizada para esse fim, enviando dados com o protocolo UDP e com o protocolo TCP. A segunda métrica foi avaliada para verificar se a placa era capaz de operar em situações extremas mesmo em um local que não tivesse uma infraestrutura preparada para resfriamento de dispositivos computacionais de alto desempenho, algo que aumentaria o custo da solução. A ferramenta `vcgenmod` foi utilizada para esse fim.

A Figura 4 apresenta a vazão média retornada pelo `iperf` em função de diferentes quantidades de `workers`. A dispersão dos dados foi omitida do gráfico por ter ficado baixa em todos os experimentos (o maior desvio padrão foi 0,37Mbps). Pelos resultados obtidos, o uso da CPU não afetou a capacidade da placa em receber dados via rede.

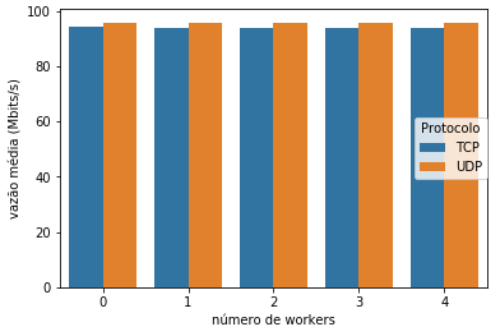

Figura 4. Vazão de rede da Raspberry Pi com CPU estressada

A Figura 5 apresenta a temperatura retornada pelo `vcgenmod` em função do tempo. Antes da execução do `stress`, foram amostrados 40 valores de temperatura por 20 segundos; durante a execução do `stress`, foram amostrados 1200 valores de temperatura durante 600 segundos; após a execução do `stress`, foram amostrados 2400 valores de temperatura durante 1200 segundos. Desta forma, foi possível verificar o tempo que o processador levou para esquentar e esfriar. A placa continuou em pleno funcionamento durante todos os experimentos, suportando a carga de trabalho imposta.

5. Conclusões e Próximos Passos

Este artigo apresentou os resultados parciais de um trabalho de conclusão de curso que encontra-se em andamento e que tem por objetivo detectar ataques ofuscados utilizando hardware de baixo custo. Os resultados mostraram diversas características observadas durante os ataques que podem ser utilizadas para a criação de modelos de aprendizado de máquina e a boa eficiência de uma unidade Raspberry Pi 3 Model B tanto em termos de vazão de rede quanto em termos de temperatura em situações de uso intensivo da CPU. Os próximos passos são a criação de modelos baseados nas características observadas e a implantação de unidades Raspberry Pi no sistema baseado nesses modelos.

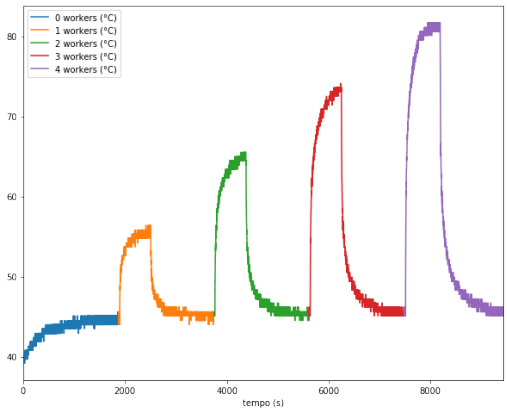

Figura 5. Temperatura da Raspberry Pi com CPU estressada

Agradecimentos

Esta pesquisa é parte do INCT da Internet do Futuro para Cidades Inteligentes financiado pelo CNPq proc. 465446/2014-0, Coordenação de Aperfeiçoamento de Pessoal de Nível Superior – Brasil (CAPES) – Código de Financiamento 001, FAPESP proc. 14/50937-1, e FAPESP proc. 15/24485-9.

Referências

Arduino (2019). Arduino - Home. `https://www.arduino.cc/`. Último acesso em 22 de Março de 2019.

Banzi, M. (2011). *Getting Started with Arduino*. Make: projects. O'Reilly Media.

Brown, S., Gommers, J., and Serrano, O. (2015). From Cyber Security Information Sharing to Threat Management. In *Proceedings of the 2Nd ACM WISCS'15*, pages 43–49.

Feth, D. (2015). User-centric Security: Optimization of the Security-usability Trade-off. In *Proceedings of the 10th ESEC/FSE 2015*, pages 1034–1037.

GT-BIS (2018). GT-BIS – Mecanismos para Análise de Big Data em Segurança da Informação. `http://gtbis.ime.usp.br/`. Último acesso em 22 de Março de 2019.

Pahl, C., Helmer, S., Miori, L., Sanin, J., and Lee, B. (2016). A Container-Based Edge Cloud PaaS Architecture Based on Raspberry Pi Clusters. In *4th IEEE FiCloudW*, pages 117–124.

Raspberry Pi Foundation (2018). Raspberry Pi – Teach, Learn, and Make with Raspberry Pi. `https://www.raspberrypi.org/`. Último acesso em 22 de Março de 2019.

Singh, K. J. and Kapoor, D. S. (2017). Create Your Own Internet of Things: A survey of IoT Platforms. *IEEE Consumer Electronics Magazine*, 6(2):57–68.

Splunk (2019). SIEM, AIOps, Application Management, Log Management, Machine Learning, and Compliance — Splunk. `https://www.splunk.com/en_us`. Último acesso em 22 de Março de 2019.