IME - USP

MAC0499 - Proposta de trabalho

Detecção de Incidentes de Segurança em Redes por meio de uma Arquitetura para Análise de Grandes Volumes de Dados

Victor Andreas Sprengel
9298002
20 de Abril de 2019

Daniel Macêdo Batista
Orientador

SUMÁRIO

1 PROBLEMA

Temos como aspiração do projeto a vontade de aumentar a segurança de uma rede observando os pacotes que trafegam por seus enlaces. Para isso, iremos abordar o problema do ponto de vista de um CSIRT (*Computer Security Incident Response Team*), isto é, melhorar a forma e velocidade na qual incidentes de rede são detectados.

Em nossa visão, a solução ideal para isso seria monitorar os enlaces apenas no momento exato em que incidentes de segurança estejam sendo iniciados, armazenar apenas os pacotes relevantes em memória volátil e informar ao analista de segurança sobre o possível incidente em um intervalo de tempo que permita ao analista tomar ações que inviabilizam que o incidente de fato tenha sucesso. Em memória não volátil, seria armazenado somente o mínimo necessário de dados que possibilitasse a realização de investigações posteriores e de aprendizado automático.

Dessa solução possivelmente utópica (porém, sendo almejada) este projeto se responsabiliza unicamente pela parte de detecção dos incidentes: utilizando os dados (pacotes de redes) já disponibilizados em uma fila do Kafka, irá usar Spark Streaming para processar esses dados e por fim criar modelos de aprendizado de máquina capazes de detectar um incidente em *near real-time*. Faz parte do escopo, portanto, escolher incidentes de rede para serem estudados, reproduzidos e testados contra o modelo a ser criado.

Um dos pontos principais do projeto será considerar ataques recentes, de modo que modelos capazes de detectá-los eficientemente sejam de utilização prática.

2 MOTIVAÇÃO

A maioria dos sistemas de detecção de intrusões em redes funcionam majoritariamente com regras manualmente criadas ou com detecção de anomalias. Escrever essas regras é extremamente custoso e requer conhecimento profundo do problema ou da área. No caso de detecção de anomalias, acontecem muitos falsos positivos.

Acreditamos que usar aprendizado de máquina aplicado a segurança de informação seja uma evolução lógica para detecção de intrusões, e será mais eficiente em três sentidos:

- Rápida detecção
- Menor armazenamento de pacotes de rede (em memória volátil e não)
- Soluções mais genéricas e/ou fáceis de serem mantidas ou atualizadas

3 OBJETIVOS

O projeto resume-se à realização das seguintes etapas:

3.1 SELEÇÃO DE UM DATASET

Ele deve conter dados realistas e recentes de pacotes de redes de ataques realizados, juntamente com tráfego de "não-ataque".

Boa parte dos datasets utilizados para pesquisa nessa área são muito antigo, vide **KDD-99**, amplamente utilizado em artigos científicos até hoje mas coletado em 1999. Justamente por esse motivo queremos trazer algo mais recente para nosso trabalho.

O cenário ideal é termos os pacotes inteiros e sem modificações, no máximo com anonimização mas sem perder dados dos cabeçalhos dos pacotes. O mais comum, mas também ideal nesse sentido, é que o dataset esteja no formato pcap, um formato utilizado para armazenar pacotes de redes e suportado pelos principais analisadores de tráfego como o `tcpdump` e o `wireshark`.

Caso não encontremos um conjunto de dados próximo ao que estamos procurando será necessário criar um de fato.

3.2 PIPELINE DE DADOS

Como consta na descrição do problema, o fluxo de dados desde a aparição de um pacote de rede até o processamento dele e as conclusões tiradas em cima disso é extremamente importante na hora de determinar a qualidade de alguma solução.

Para tal, escolhemos trabalhar com duas ferramentas altamente testadas e utilizadas mercado a fora para nos auxiliar nessa tarefa: Apache Spark (Streaming) e Apache Kafka. A principal vantagem é o fato escalabilidade, e o quão fácil é de se lidar com uma quantidade imensa de dados usando essa tecnologias - mesmo que isso envolva gastar mais com máquinas. Ambas tecnologias Apache, esperamos que exista uma fácil integração entre as duas, o que também conta para a escolha.

Será necessário estruturar (escrever o código) responsável pelo consumo das filas do Kafka pelo Spark. Como dito, não nos importaremos com a captação para o Kafka, e sim com o processamento após os pacotes já estarem lá disponibilizados. A inserção de dados nas filas do Kafka para o projeto será feita manualmente, utilizando o dataset descrito na etapa anterior.

Para testar a pipeline, a ideia é enviar algum pacote de rede que sozinho configura um ataque à rede e ver se essa detecção (via algo simples como um grep) ocorre *near real time*.

3.3 DETECÇÃO DE ATAQUES

Escolheremos ataques recentes e relevantes para serem detectados usando esse nosso "sistema". O objetivo é usar aprendizado de máquina, e não regras manualmente criadas como é feito hoje em dia. Isso significa que para cada ataque teremos que passar por um processo natural de engenharia de aprendizado de máquina, isto é

- *Data cleansing*
- *Feature Engineering*
- Benchmark de modelos
- *Parameter tuning*
- *Monitoring* (talvez)

No quesito modelos, temos algumas ferramentas interessantes em mãos. Para um começo simples, usar uma regressão logística para determinar se há algum ataque ocorrendo ou não pode formar um baseline para nós. A partir daí podemos partir para modelos de classificação (como uma *Random Forest*) que consiga dizer qual ataque está acontecendo. Outra opção é utilizar métodos de *gradient boosting* (que contém implementações mostrando grande sucesso, inclusive na área, como o **XGBoost**) para uma abordagem mais recente.

Ainda não é claro se iremos incorporar cada ataque de rede a um grande modelo que trabalharemos em cima ou se precisaremos ter um modelo para cada ataque.

Além de utilizar um método simples como baseline, iremos procurar métricas baseline utilizando ferramentas já existentes na área.

3.4 CONCLUIR VANTAGENS E DESVANTAGENS DO MECANISMO PROPOSTO

A nossa intenção é mostrar que há um jeito melhor de se detectar incidentes de redes comparado ao que é feito hoje em dia. Para tal, será necessário reunir todo o processo de desenvolvimento, bem como o desempenho e a facilidade de uso (entrando nesse quesito *expertise* e custo), para concluir as vantagens e desvantagens do que estamos propondo.

Principalmente do ponto de vista de um administrador de rede que não necessariamente terá plenos conhecimentos de ferramentas de análise de dados é que essa análise é importante. Se você precisar de um cientista de dados para utilizar nosso mecanismo, de pouco ele será útil no contexto do problema.

Está no nosso pensamento, se possível e ainda fizer sentido no final do projeto, mostrar o mecanismo e o passo-a-passo de uso para algum(s) administardor(es) de redes e colher feedback.

4 CRONOGRAMA

Nosso planejamento simplificado de trabalho segue abaixo:

Atividade	Período
Encontrar um dataset	Até 10/05/2019
Definir um pipeline de processamento	20/04/2019 - 20/05/2019
"Modelo"para ataque(s) simples	20/05/2019 - 03/06/2019
Criação de modelos para diversos ataques	
Botnet	06/2019
DoS	07/2019
Web Attack (DVWA)	08/2019
DDoS + Portscan	09/2019
Compilar métricas	07/2019 - 14/10/2019
Listar vantagens e desvantagens do método proposto	14/10/2019 - 28/10/2019

Ele vem com alguns *caveats*:

- Não alocamos um tempo específico para a escrita da monografia, que deve ser um trabalho concorrente com as atividades desenvolvidas. Ainda assim, resolvemos acabar o planejamento no final de Outubro para que exista uma certa folga caso ocorra mudança de escopo ou novas ideias.
- Os ataques que tentaremos detectar via modelos de aprendizado de máquina são ataques presentes no dataset, mas não necessariamente teremos sucesso com todos. Talvez sobre tempo de testar outros ataques (ou outras variantes do mesmo ataque) também presentes no dataset.
- O tempo alocado para cada ataque é apenas um guia para que não se perca tempo demais em algum ataque que em específico pode não ser facilmente detectado. Pode ser que algum demore mais, ou que outro demore menos.

5 BÔNUS: DATASET ENCONTRADO

Nós encontramos um dataset chamado **CSE-CIC-IDS2018**, que é um projeto colaborativo entre o **Communications Security Establishment (CSE)** e o **Canadian Institute for Cybersecurity (CIC)**.

Esse conjunto de dados se encaixa no que procuravamos dado que ele contém os pacotes de dados crus (Pcaps), exemplos de ataques mais recentes e relevantes, e como bônus também contém os logs de eventos da máquina (tanto Windows quanto Ubuntu). Ele pode ser encontrado na url **https://www.unb.ca/cic/datasets/ids-2018.html**

Estamos confiantes de que esse dataset será suficiente para nossos objetivos.