# Web进程链及SSH操作序列异常检测技术设计文档

## 问题描述

针对Web进程链/SSH异常操作序列，检测上下文关系中的子进程异常。

Web进程链：同一Web服务应用的进程调用关系中，父进程为JAVA，其派生的所有子进程序列。

SSH操作序列：同一台主机系统中，父进程为TTY，其派生的所有子进程序列。（可以当作是管理员的输入）

异常定义：根据历史行为建立基线模型，识别有异于正常行为的异常行为

## 研究意义

如今绝大多数企业及个人用户都选择使用云计算来部署、运营业务。

Web进程链及SSH异常操作检测技术研究将应用在的主机型入侵检测系统，通过分析程序访问的内存、文件系统、日志存储等信息，结合算法能力，精确检测程序异常或恶意入侵行为，以为系统提供主动性的防御机制。

该项技术可通过捕获Linux系统下用户态和内核态通信过程中产生的信息，挖掘更深层次的特征，以进程为粒度进行入侵或异常行为的检测，能够抵御新型攻击手段，特别是隐蔽而又高危的APT攻击，为用户提供安全可靠的云计算环境。

## 已有最新技术描述

实现恶意指令检测的技术，传统一般采用的是基于模式匹配和规则的方法。原理就是基于黑名单的命令匹配，网络管理员构造一些恶意命令集合，通过对比执行的指令与恶意命令集合中的指令，来判断该指令是否是恶意的。此种类型的检测算法依赖于模式和规则的构造，而且对新出现的恶意行为检测能力较差，攻击者可能通过简单地混淆措施，绕过规则检查。

此外，一些异常检测算法也可以应用于异常数据识别。比如基于统计的方法，现在正常数据集上计算出一些统计数据，当测试数据的相关数值与原数据集相差较多（比如正常数据集的长度均值为35， 但是测试命令的长度为150），就将该测试数据识别为恶意指令。基于密度的异常检测方法，数据集中的一些数据点与最近n个邻居的距离，一定程度上反映了该数据点附近的密度，如果某个数据点所在位置密度较小，我们有理由认为该数据点是异常数据。基于聚类的异常检测，对数据进行聚类分析，分析测试数据与聚类中心的距离，如果距离过大，表示该数据是恶意数据。

机器学习算法在分类任务与自然语言处理方面取得的成就可以应用到此问题中。机器学习算法在分类任务中取得了令人惊喜的成绩，T. Minárik, S.Alatalu, S.Biondi 等人在Detection of Malicious Remote Shell Sessions提出了使用基于n-gram的knn算法识别恶意指令的方法[1]。使用n-gram衡量两条shell指令之间的距离，随后使用knn算法进行有监督的学习与预测，模型准确率达到98%。YifanTian，Jiabao Wang等人在CNN-Webshell: Malicious Web Shell Detection with Convolutional Neural Network中提出了使用卷积神经网络识别恶意web shell的方法[2]，将http请求文本当做原始数据，经过word2vec向量化之后，输入卷积神经网络进行有监督的训练与预测，模型准确率达到98%。Danny Hendler，Shay Kels等人在Detecting Malicious PowerShell Commands using Deep Neural Networks中提出了使用深度神经网路识别恶意命令的方法[3]，他们将powershell指令进行base64编码后进行正则化，输入神经网络进行训练与预测，模型准确率在87%左右。

可以看到，当前的机器学习在恶意命令识别的应用模式，主要是：使用自然语言处理的文本向量化方法（n-gram, Tfidf, WordBag, word2vec），将文本转换为向量，进行机器学习算法或神经网络模型训练。

## 拟采取的技术

#### 特征提取

文本向量化的方式有Tfidf，词袋模型，因为tfidf考虑单词顺序信息，相比词袋模型更加精确，所以在初步测试中我们选择TFidf向量化方法。将指令当做文本，使用Tfidf方法提取向量特征。

#### 有监督学习

Knn、svm与dt都是机器学习中常用的分类算法，在数据分类方面具有广泛的应用。神经网络在分类任务中有着不输上述分类算法的能力，全连接神经网络作为神经网络的代表，也参与到我们的初步测试中。在初步测试中我们会使用knn、svm、dt算法、全连接神经网络进行有监督学习测试。

#### 无监督学习

我们使用独立森林和基于聚类的异常检测方法。独立森林算法应用基于一个前提假设，异常样本点在整个数据集中占比小，有明显区别于其他样本的特征。与我们要解决的问题相契合，因为恶意shell指令与正常web进程执行的指令差异较大，且在数据集中恶意指令与web进程产生的指令相比，数量非常少。意味着我们可以使用独立森林算法进行异常检测分析。基于聚类的算法也值得尝试，web进程链执行的进程类型数量有限，使用聚类算法将数据集中的命令分成数量有限的中心，我们测试待测指令与聚类中心的距离，从而判断该指令是否恶意。

## 方案设计

### 验收标准

### 整体方案

### 方案对比

### 项目名词解释

### 具体检测方案

#### 研究内容

#### 5.5.2 模型输入

模型输入

#### 5.5.3 模型输出

#### 5.5.4 数据来源

#### 5.5.5 检测原理

##### 命令语言字典

##### 命令向量化

##### 异常检测算法

##### 机器学习算法

## 已进行的工作

### 数据集

由于恶意数据与正常数据数量相差众多，使用有监督模型训练时，我们仅仅使用了部分正常数据，方法是使用随机函数，从正常数据集中取1000条指令参与模型训练。

### 算法衡量指标

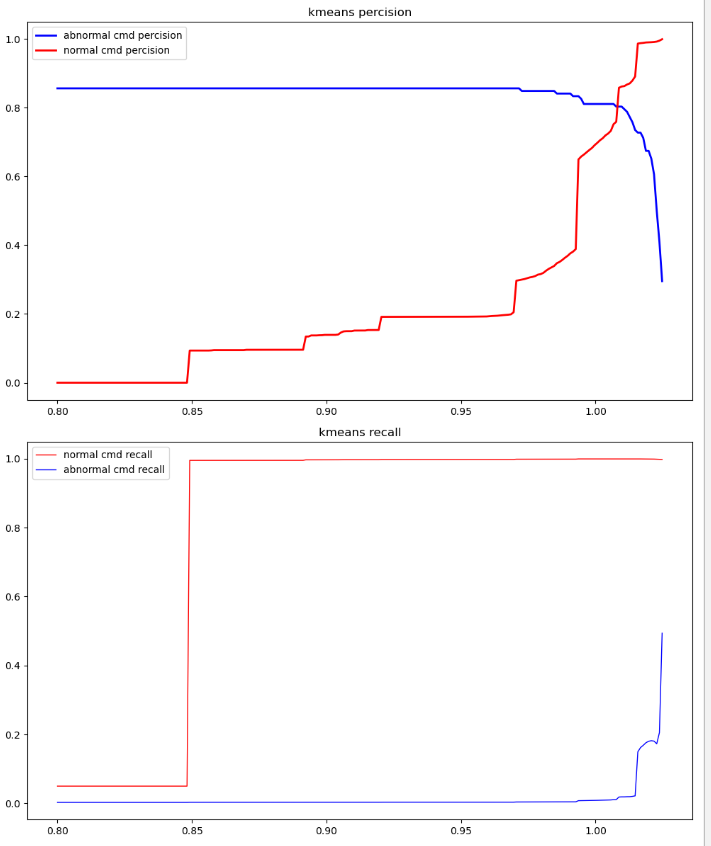
衡量算法我们使用查准率与召回率指标。查准率表征了模型正确识别恶意指令的能力，召回率表征了模型从海量数据中找到恶意指令的能力。

### 无监督学习

#### IForest算法

#### 基于聚类的异常检测

基于聚类的异常，我们使用了kmeans聚类算法，使用单独一个聚类中心，测试的时候计算测试样本向量化后的向量与聚类中心的几何距离，通过修改距离阈值，得到准确率和召回率随阈值变化如图所示



1 kmeans准确率与召回率变化曲线

### 有监督学习

#### Knn

使用sklearn的knn算法实现，使用Tfidf方式将命令文本向量化，在部分数据集上的测试结果如下

1knn算法测试结果

|  |  |  |
| --- | --- | --- |
|  | 查准率 | 召回率 |
| 正常指令 | 96% | 99% |
| 恶意指令 | 89% | 68% |

#### Svm

我们使用sklearn的Svm算法实现，使用Tfidf方式将命令文本向量化，在部分数据集上的测试结果如下

2Svm算法测试结果

|  |  |  |
| --- | --- | --- |
|  | 查准率 | 召回率 |
| 正常指令 | 97% | 97% |
| 恶意指令 | 86% | 84% |

在正常数据集上的准确率较高，而恶意指令的准确率只有86.。并且是在部分数据集上，没有使用所有的数据进行测试。

#### DT

我们使用sklearn的决策树算法的实现，使用Tfidf进行命令向量化，训练模型。在部分数据上的测试结果如下

3决策树算法结果

|  |  |  |
| --- | --- | --- |
|  | 准确率 | 召回率 |
| 正常指令 | 100% | 100% |
| 恶意指令 | 100% | 100% |

可以看到在部分数据集上决策树的分类准确率达到94%以上。由于这是部分数据的测试结果，随后使用该模型对所有数据进行测试，得到结果如下

4决策树算法在所有指令上的预测结果

|  |  |  |
| --- | --- | --- |
|  | 准确率 | 召回率 |
| 正常指令 | 100% | 99% |
| 恶意指令 | 24% | 100% |

正常数据召回率变低，恶意数据召回率变高，并且恶意数据准确率明显下降，表明模型具有较高的误报率。受到随机森林算法的启发，我们也可以采用多颗决策树，最后大家选举最终结果的方式，来进行模型的训练与测试。我们每次从正常数据中取1000条指令，与恶意指令一起进行模型训练，得到若干决策树模型，预测指令是否恶意时，使用这若干个决策树模型同时预测，最后少数服从多数，得到的准确性结果如下

5多颗决策树选举预测结果

|  |  |  |
| --- | --- | --- |
|  | 准确率 | 召回率 |
| 正常数据 | 99% | 100% |
| 恶意数据 | 100% | 38% |

#### 神经网络

### 算法结果对比

## **未来拟采取的方案**

项目的难点主要在特征提取方面，如果可以综合考虑更多的命令执行特征，机器学习算法就可以得到大幅优化。比如：命令涉及到的文件集合，权限集合等信息

基于n-gram的knn，是Detection of Malicious Remote Shell Sessions中提出的一种恶意命令识别的方法，n-gram计算两条指令的距离，使用knn做最终分类，并且考虑了命令上下文。下一步的目标是复现文中的异常检测系统，测试系统性能。

## 8. 参考文献

[1]. Dumont, P., Meier, R., Gugelmann, D., & Lenders, V. (2019, May). Detection of Malicious Remote Shell Sessions. In *2019 11th International Conference on Cyber Conflict (CyCon)* (Vol. 900, pp. 1-20). IEEE.

[2]. Tian, Y., Wang, J., Zhou, Z., & Zhou, S. (2017, December). CNN-webshell: malicious web shell detection with convolutional neural network. In *Proceedings of the 2017 VI International Conference on Network, Communication and Computing* (pp. 75-79).

[3]. Hendler, D., Kels, S., & Rubin, A. (2018, May). Detecting malicious PowerShell commands using deep neural networks. In *Proceedings of the 2018 on Asia Conference on Computer and Communications Security* (pp. 187-197).