REAL-TIME ANOMOLY DETECTION IN PHYSICAL ACCESS CONTROL SYSTEM(PACS) USING PREDICTIVE MODELING

STUDENT NAME:	Vijayaram Boopalan
COURSE NAME:	Doctor of Philosophy (Computer Science Engineering)
DEPARTMENT:	Department of Computer Science and Engineering
SUPERVISOR:	YTD
DATE OF SUBMISSION:	**15 Nov 2020**

CONTENTS

ABSTRACT **3**

INTRODUCTION **3**

PACS 3

ACCESS POINT 3

PIV Credential *3*

Card Reader / Bio Metric Reader *4*

Control Panel *4*

Access control server *4*

Credential Holder Repository *4*

Auxiliary Systems *4*

PROBLEM STATEMENT **4**

OVERVIEW 4

RESEARCH QUESTION/HYPOTHESIS 5

OBJECTIVES AND AIMS **5**

OVERALL OBJECTIVE 5

SPECIFIC AIMS 5

RESEARCH DESIGN AND METHODS **6**

OVERVIEW 6

REFERENCES **7**

ABSTRACT

In recent years machine learning based anomaly detection is gaining popularity in Intrusion detection system. Once up a time physical security is of utmost important as 1st line (SL-1) of defence to protect the asserts like Buildings, Industries, even in providing National Security protecting the peoples. As the Internet era started security solutions based on hardware and software had become the ubiquitous 2nd line(SL-2) of defence using Physical Access Control Systems(PACS) consisting of components like Access point, Personal Identity Verification(PIV) credentials, credential reader and keypad, biometric reader, control panel, access control server, credential holder data repository, auxiliary systems(fire alarm systems, surveillance systems, smoke detectors, evacuation systems). With the advancements of Artificial Intelligence and Machine Learning, Systems are developed to be more intelligent to detect anomalies based on the historical data. In the paper we will analysis different anomaly prediction algorithms and propose a predictive modelling solution using Clustering Based Local Outlier Factor(CBLOF) and Histogram Based Outlier Score(HBOS) with the help of Python library(PyOD-Python Outlier Detection) to detect real time anomaly based on the PACS access events and audit logs.

INTRODUCTION

PACS

Physical Access Control Systems(PACS) is a system provided as a solution in combination with hardware products like access controller panels, card readers, biometric readers, siren alarms, cameras, smoke detectors, etc., and software which controls the management of credentials, rules, access control servers, etc., Some of the common components in every PACS are

Access Point

Access point is the first interaction point of the PACS system where end-user interacts. Some of the examples of access point are gates with access card reader / biometric reader mounted, turnstiles with access card reader / biometric reader mounted, doors with access card reader / biometric reader mounted.

PIV Credential

Personal Identity Verification (PIV) Credential to either physically or logically access the PACS. This credentials are used to validate the person and act (allow / deny) access to the physical location based on the rules set in the PACS.

Card Reader / Bio Metric Reader

Card Reader / Bio Metric Reader are physical devices used to read data from PIV credential and sends the data to access controller panel to authenticate the PIV credential and take decision to allow/deny based on the rules set in the PACS. Access Card / Badge Number, Biometric Fingerprint, Facial Image and IRIS scan are some of the PIV credentials used.

Control Panel

It's the heart of the Access control system to receive the reader data and verifies its authenticity against the PIV credential repository or rules set to allow/deny access to the location. This decision signals the reader, door / gates to open door / remain closed with green/red LED indicators in the reader.

Access control server

Access control server is a set of software for managing PIV credentials and configuring access and privileges for the end-users (Card / badge Holders).

Credential Holder Repository

It's the repository used by PACS software to store the PIV credentials and access privileges. Control Panel uses this data to grant/deny the access to the end-users.

Auxiliary Systems

Auxiliary systems are those systems which integrates with PACS to provide additional level of security features like Fire Alarm System, Elevator System, Evacuation System, Surveillance system, etc, These systems / products when combined with PACS will provide robust security for the building/premises providing utmost safety and comfort for the people.

PROBLEM STATEMENT

Overview

Most of the PACS behaves based on the predefined set of the rules and access privileges configured. There are times threats arise out of insider authorized persons and it becomes difficult to identify those insider threat as they hold valid PIV credentials and are authorised to enter or access those restricted locations. So manual processing or investigating any insider threat security

incident becomes tedious, time consuming and more over breaks the dignity of the officials by putting dignified employees under the investigation umbrella. So, the need to intelligently identify the insider treat has become inevitable now-a-days as part of every PACS security solutions. Most of the security solutions providers lacks on this feature due to the unavailability of the direct solution to detect those incidents.

Research Question/Hypothesis

Many Anomaly detection algorithms are available and used for various other purposes, but to predict or identify the insider intrusion based on PACS events data in real-time is of less use due to its drawback of high false-positive detection result. So a detailed analysis of the PACS system data analysis and transformation is required, along with combined anomaly detection algorithms with relevant hyper-parameters are of utmost import to get real-time detection/prediction solution with less false-positive detection rate.

OBJECTIVES AND AIMS

Overall Objective

A Predictive analytics model / system should be designed and put in place to automatically detect those insider threats or anomaly based on the access control event data and audit log pattern using machine learning training and prediction data model.

Specific Aims

Through this research a more robust real-time prediction model is proposed based on Clustering Based Local Outlier Factor (CBLOF) and Histogram Based Outlier Score (HBOS) with the help of Python library (PyOD-Python Outlier Detection) for anomaly detection based on the historical and live access control event and audit data.

RESEARCH DESIGN AND METHODS

Overview

This research is to propose a robust real-time prediction model based on Clustering Based Local Outlier Factor (CBLOF) and Histogram Based Outlier Score (HBOS) with the help of Python library (PyOD-Python Outlier Detection) for anomaly detection based on the historical and live access control event and audit data.

Please find below the code snippet for real-time prediction model. Prediction model design and method evaluation is in progress…

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
from pyod.models.lof import LOF
from pyod.models.mcd import MCD
from pyod.models.auto_encoder import AutoEncoder
from pyod.models.lscp import LSCP
from pyod.models.cblof import CBLOF
from pyod.models.hbos import HBOS
import timeit
import swifter
import pickle

data = pd.read_csv('evlog2.csv',header = 0,names = ['BADGENO','LOGDEVID','REC_DAT'])
data = data[15000000-3000000:]
starttime = timeit.default_timer()
class rtad():
    def __init__(self):
        i =1
    def trigtrain(self,data):
        def dateformat(df):
            df['date'] = 0
            df['hour'] = 0
            df['date'] = df.swifter.apply(lambda row : row['REC_DAT'].split()[0],axis = 1)
            df['hour'] = df.swifter.apply(lambda row : float(row['REC_DAT'].split()[1].split(':')[0])+float(row['REC_DAT'].split()[1].split(':')[1
            ])/60,axis=1)
        dateformat(data)
        starttime = timeit.default_timer()
        data['mean'] = 0
        data['std'] = 0
        data['acc_mean'] = 0
        data['acc_std'] = 0
```

REFERENCES

ADAPTIVE LOAD BALANCING BASED ON CONTEXT AND SITUATIONAL AWARENESS OF TARGET

STUDENT NAME:	Vijayaram Boopalan
COURSE NAME:	Doctor of Philosophy (Computer Science Engineering)
DEPARTMENT:	Department of Computer Science and Engineering
SUPERVISOR:	YTD
DATE OF SUBMISSION:	**11 Nov 2020**

CONTENTS

ABSTRACT **3**

INTRODUCTION **3**

L7 LOAD BALANCER 3

LOAD BALANCER ALGORITHMS 3

Round Robin *4*

Weighted Round Robin *4*

Least Connection *4*

Weighted Least Connection *4*

Resource Based (Adaptive) *5*

Resource Based (SDN Adaptive) *5*

Fixed Weighting *5*

Weighted Response Time *5*

Source IP Hash *5*

URL Hash *6*

PROBLEM STATEMENT **6**

OVERVIEW 6

RESEARCH QUESTION/HYPOTHESIS 6

OBJECTIVES AND AIMS **7**

OVERALL OBJECTIVE 7

SPECIFIC AIMS 7

RESEARCH DESIGN AND METHODS **7**

OVERVIEW 7

REFERENCES **8**

ABSTRACT

Load Balancing application or services in a high scale cloud or an enterprise infrastructure in inevitable to achieve reliability and scalability of the system. Be it a physical load balancer or virtual load balancer or a cloud load balancer, its primary job is to route client traffic to all servers configured. There more complex algorithms involved in different types of load balancers to decide on routing traffic to the server for processing the client requests. Most of the algorithms does not consider the characteristic and context of the different services running in the participating servers. So, it is extremely important to know the current situation context and characteristic of different services before the traffic is routed simply based on the predefined set of algorithms or rules. In this paper we will provide the system and methods to consider the current situation context and characteristic of the different services as part of the application level load balancing to provide optimal performance and efficient dynamic load balancing.

INTRODUCTION

L7 Load Balancer

In the OSI reference model, layer 7 is called Application Layer (Http/Https). Application Layer or Layer7(L7) Load Balancer is used for routing client traffic to all the participating server based on context switching, it uses the request HTTP header, Session ID, URI, Cookies and sometime HTML form data to route the request to different servers.

Load Balancer Algorithms

Load balancer algorithms are used for deciding which request has to be routed to which server for processing based on various parameters defined in the algorithm. Some of the commonly ised algorithms are Round Robin, Weighted Round Robin, Least Connection, Weighted Least Connection, Resource based (Adaptive), Resource based (SDN Adaptive), Fixed Weighting, Weighted Response Time, Source IP Hash, URL Hash, etc. These servers which in turn host services to process those requests and respond back. Most of the real time systems involving highly intensive traffic fails to do effective load balancing due to lack of situational context and dynamic characteristic of the server, as L7 load balancers are preload with standard algorithms mentioned above without any contextual knowledge of the system.

Round Robin

Round-robin load balancing is one of the simplest and most used load balancing algorithms. Client requests are distributed to application servers in rotation. For example, if you have three application servers: the first client request to the first application server in the list, the second client request to the second application server, the third client request to the third application server, the fourth to the first application server and so on.

This load balancing algorithm does not take into consideration the characteristics of the application servers i.e. it assumes that all application servers are the same with the same availability, computing and load handling characteristics.

Weighted Round Robin

Weighted Round Robin builds on the simple Round-robin load balancing algorithm to account for differing application server characteristics. The administrator assigns a weight to each application server based on criteria of their choosing to demonstrate the application servers' traffic-handling capability. If application server #1 is twice as powerful as application server #2 (and application server #3), application server #1 is provisioned with a higher weight and application server #2 and #3 get the same weight. If there five (5) sequential client requests, the first two (2) go to application server #1, the third (3) goes to application server #2, the fourth (4) to application server #3 and the fifth (5) to application server #1

Least Connection

Least Connection load balancing is a dynamic load balancing algorithm where client requests are distributed to the application server with the least number of active connections at the time the client request is received. In cases where application servers have similar specifications, an application server may be overloaded due to longer lived connections; this algorithm takes the active connection load into consideration.

Weighted Least Connection

Weighted Least Connection builds on the Least Connection load balancing algorithm to account for differing application server characteristics. The administrator assigns a weight to each application server based on criteria of their choosing to demonstrate the application servers' traffic-handling capability. The Loadmaster is making the load balancing criteria based on active connections and application server weighting.

Resource Based (Adaptive)

Resource Based (Adaptive) is a load balancing algorithm requires an agent to be installed on the application server that reports on its current load to the load balancer. The installed agent monitors the application server's availability status and resources. The load balancer queries the output from the agent to aid in load balancing decisions.

Resource Based (SDN Adaptive)

SDN Adaptive is a load balancing algorithm that combines knowledge from Layers 2, 3, 4 and 7 and input from an SDN Controller to make more optimized traffic distribution decisions. This allows information about the status of the servers, the status of the applications running on them, the health of the network infrastructure, and the level of congestion on the network to all play a part in the load balancing decision making.

Fixed Weighting

Fixed Weighting is a load balancing algorithm where the administrator assigns a weight to each application server based on criteria of their choosing to demonstrate the application servers' traffic-handling capability. The application server with the highest weigh will receive all the traffic. If the application server with the highest weight fails, all traffic will be directed to the next highest weight application server.

Weighted Response Time

Weighted Response Time is a load balancing algorithm where the response times of the application servers determines which application server receives the next request. The application server response time to a health check is used to calculate the application server weights. The application server that is responding the fastest receives the next request.

Source IP Hash

Source IP hash load balancing algorithm that combines source and destination IP addresses of the client and server to generate a unique hash key. The key is used to allocate the client to a particular server. As the key can be regenerated if the session is broken, the client request is directed to the same server it was using previously. This is useful if it's important that a client should connect to a session that is still active after a disconnection.

URL Hash

URL Hash is a load balancing algorithm to distribute writes evenly across multiple sites and sends all reads to the site owning the object.

PROBLEM STATEMENT

Overview

While most of the load balancing algorithms focus on the decisions based on the predefined logic or the current state data of the servers, it lacks in proper decision making due to the unavailability of the situational context and importance of the difference services running in the different servers. Though weighted round robin algorithms take the weight value of the server for decision making, it lacks in situational context of the server and the current service state. Let's say a server is running few services like provisional device service, registration device service, device api service, device telemetry processing service, etc. Out of the above services, registration and provisional device service are more important at any point in time for the end-users to get their first unbox experience of their consumer IOT device. So, in addition to providing weightage to server, its also important to provide dynamic weightage for those critical services based on its current context and situational state for making load balancing decision making, in order to effectively use the system resource without being compromising the customer user experience and trust.

Research Question/Hypothesis

Researches on Adaptive Software Defined Network(SDN) based SDN architecture that decouples the data and control planes, seems to be promising, but the algorithm used are of common ones which does not considers the situational metrics of those critical services running the server.
In the Research Paper "Adaptive Consistency for Distributed SDN Controllers", Mohamed Aslan etal.[1], use of adaptive controllers into software-defined networking (SDN) and propose the use of adaptive consistency models in the context of distributed SDN controllers. The comparison of their research showed Adaptive controller are more resilent and effective in load balancing even during sudden burst in network conditions.

OBJECTIVES AND AIMS

Overall Objective

With the advancement of technology and Internet Of Things (IoT) making its place everywhere, the importance of the IoT device software providers QoS and QoE becomes very important and chanllenging to achieve. Though make larger cloud providers are bringing in effective cloud solutions through their PaaS and IaaS offerings, the need for tailored load balancing algorithm in combination of their IaaS offering is inevitable for bringing the better customer QoE.

Specific Aims

Though hardware level and network level improvements can be brought in for effective load balancing, the need for the more granular level decision making based on the situational context of the service running in different participating server is much required, and this need a shift in load balancing algorithm design from a traditional standard algorithms which just focus on distributing the load to different server either randomly or using the predefined weightage setting and static resource state rules like number of request the server is servicing, number of system cores unutilized, etc.

Through this research a more systematic and dynamic decision making algorithm is proposed considering the current situational context of the services the server is currently running.

RESEARCH DESIGN AND METHODS

Overview

This research is to propose a novel enhanced adaptive target context and situation aware SDN based load balancing algorithm, where granular level weightage is also given to the services running based on their importance in the solution and also its situational behaviour is considered for load balancer decision making of routing traffic to the server which host/runs those services.

Algorithm design and method evaluation is in progress…

REFERENCES

[1] M. Aslan and A. Matrawy, "Adaptive consistency for distributed SDN controllers," *2016 17th International Telecommunications Network Strategy and Planning Symposium (Networks)*, Montreal, QC, 2016, pp. 150-157, doi: 10.1109/NETWKS.2016.7751168.