

A RELIABLE & ROBUST DDoS ATTACK DETECTION VEHICULAR NETWORK USING NEURAL NETWORKS

Submitted in partial fulfillment of the requirements for
the award of
Bachelor of Engineering Degree in Computer Science and Engineering

by

HRITHIK HARSHA.G (37110276)
G. RISHIDHAR REDDY (37110227)



**DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING
SCHOOL OF COMPUTING**

SATHYABAMA

INSTITUTE OF SCIENCE AND TECHNOLOGY

(DEEMED TO BE UNIVERSITY)

Accredited with Grade "A" by NAAC

**JEPPIAAR NAGAR, RAJIV GANDHI SALAI,
CHENNAI – 600119.**

APRIL 2021



SATHYABAMA

**INSTITUTE OF SCIENCE AND TECHNOLOGY
(DEEMED TO BE UNIVERSITY)**

Accredited with Grade "A" by NAAC

JEPPIAAR NAGAR, RAJIV GANDHI SALAI, CHENNAI 600 119

www.sathyabama.ac.in



SCHOOL OF COMPUTING

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

BONAFIDE CERTIFICATE

This is to certify that this Professional Training Report is the Bonafide work of **HRITHIK HARSHA.G (Reg.No.37110276)** and **G. RISHIDHAR REDDY (Reg.No.37110227)** who underwent the professional training in "**A Reliable & Robust DDoS Attack Detection in Vehicular networks using Neural Network**" under my supervision from April 2020 to June 2020.

Internal Guide

Mr. S. Dhamodaran, M.Tech., (Ph.D.),

Head of the Department

Dr. S. VIGNESHWARI M.E., Ph.D.,

Submitted for Viva voce Examination held on _____

Internal Examiner

External Examiner

DECLARATION

I, G. RISHIDHAR REDDY (Reg.No.37110227) and HRITHIK HARSHA.G (Reg. No 37110276) hereby declare that the Professional Training Report on **“A Reliable & Robust DDoS Attack Detection in Vehicular networks using Neural Network”** done by me under the guidance **Mr. S. Dhamodaran, M.Tech., (Ph.D.)**, at Sathyabhama Institute of Science and Technology is submitted in partial fulfillment of the requirements for the award of Bachelor of Engineering degree in Computer Science and Engineering.

DATE:

PLACE:

SIGNATURE OF THE CANDIDATE

ACKNOWLEDGEMENT

I am pleased to acknowledge my sincere thanks to **Board of Management of SATHYABAMA** for their kind encouragement in doing this project and for completing it successfully. I am grateful to them.

I convey my thanks to **Dr. T. SASIKALA., M.E., Ph.D., Dean, School of Computing** **Dr. S. VIGNESHWARI, M.E., Ph.D., and Dr.L. Lakshmanan M.E., Ph.D. Head of the Department, Dept. of Computer Science and Engineering** for providing me necessary support and details at the right time during the progressive reviews.

I would like to express my sincere and deep sense of gratitude to my Project Guide **Mr. S. Dhamodaran, M.Tech., (Ph.D.)**, for his valuable guidance, suggestions and constant encouragement paved way for the successful completion of my project work.

I wish to express my thanks to all Teaching and Non-teaching staff members of the **Department of Computer Science and Engineering** who were helpful in many ways for the completion of the project.

ABSTRACT

Application Layer Distributed Denial of Service (DDoS) attacks are very challenging to detect and mitigate. The various possible application layer attacks are HTTP flooding, XML attack, DNS attacks, etc. The most common and renowned application layer attack is HTTP flooding. The HTTP flooding detection and mitigation is an interesting research topic in computer networks. There are various research solutions proposed by validating against HTTP flooding; using tools such as Golden Eye, LOIC, proprietary tools, etc. HTTP flooding attacks generated using any existing tools may not exhibit similar characteristics of the real time HTTP flooding attack. Various methods were used to defend these attacks based on distributed schemes with certain difficulties to count the packets or duplicates sent by a node.

This is due to lack of communication infrastructure. Two limits are used to mitigate packet flood and replica flood attacks, respectively. Violation of both the limits can be easily noticed by claim-carry-and check. The inconsistency check against full claims is trivial. This is designed to work in a distributed system. Moreover, it allows tolerating a little number of attackers for collision.

LIST OF FIGURES

FIGURE No.	FIGURE NAME	PAGE No.
1.1	MACHINE LEARNING	4
1.2	SUPERVISED LEARNING	5
3.1	WATERFALL MODEL	9
3.2	RAD DIAGRAM	11
4.1	PYTHON RELATIONS	17
4.2	PYTHON LINE STRUCTURE HIERARCHY	37
5.1	ARCHITECTURE DIAGRAM	32

TABLE OF CONTENTS

CHAPTER NO	TITLE	PAGE NUMBER
	ABSTRACT	v
	LIST OF FIGURES	vi
1	INTRODUCTION	1
	1.1 OVERVIEW	1
	1.2 SCOPE OF THE PROJECT	1
	1.3 DOMAIN OVERVIEW	1
	1.3.1 MACHINE LEARNING	2
2	LITERATURE SURVEY	8
3	AIM AND SCOPE	15
	2.1 EXISTING SYSTEM	20
	2.2 DRAWBACKS OF EXISTING SYSTEM	20
	2.3 PROPOSED SYSTEM	21
	2.4 ADVANTAGES OF PROPOSED SYSTEM	22
4	MATERIAL AND METHODS	23
	4.1 INTRODUCTION	23
	4.2 HARDWARE & SOFTWARE SPECIFICATION	23
	4.2.1 HARDWARE SPECIFICATION	23
	4.2.2 SOFTWARE SPECIFICATION	23
	4.3 SOFTWARE DESCRIPTION	24
	4.3.1 ABOUT PYTHON	24
	4.3.2 ABOUT IDE	35
5	SYSTEM DESIGN	39
	5.1 INTRODUCTION	39
	5.2 ARCHITECTURE DIAGRAM	39
6	SYSTEM IMPLEMENTATION	41
	6.1 LIST OF MODULES	41
	6.1.1 EXPLORATORY OF DATA EVALUATION	41
	6.1.2 FEATURE EXTRACTION	42
	6.1.3 FEATURE ENGINEERING	42

	5.1.4 PREDICTION	43
7	CONCLUSION & FUTURE ENHACEMENT	45
	7.1 CONCLUSION	45
	7.2 FUTURE ENHANCEMENT	45
	REFERENCES	46

CHAPTER 1

INTRODUCTION

A new attack, the Ad Hoc Flooding Attack, results in denial of service when used against all on demand ad hoc networks routing protocols. In this attack, the attacker either broadcasts a lot of Route Request packets for node ID who is not in networks, or sends a lot of DATA packets to consume the bandwidth so as to congest in links.

Position falsification attack: An attacker may modify the position information in broadcast beacons using a technique such as GPS spoofing.

1.1 Scope of the Project

The main contributions of this project therefore are:

- Data Analysis
- Dataset Preprocessing
- Training the Model
- Testing of Dataset

1.2 Domain Overview

Machine Learning is the most popular technique of predicting the future or classifying information to help people in making necessary decisions. Machine Learning algorithms are trained over instances or examples through which they learn from past experiences and also analyze the historical data. Therefore, as it trains over the examples, again and again, it is able to identify patterns in order to make predictions about the future.

Data is the core backbone of machine learning algorithms. With the help of the historical data, we are able to create more data by training these machine learning algorithms. For example, Generative Adversarial Networks are an advanced concept of Machine Learning that learns from the historical images through which they are capable of generating more images. This is also applied towards speech and text synthesis. Therefore, Machine Learning has opened up a vast potential for data science applications.

1.3.1 Machine Learning

Machine Learning combines computer science, mathematics, and statistics. Statistics is essential for drawing inferences from the data. Mathematics is useful for developing machine learning models and finally, computer science is used for implementing algorithms.

However, simply building models is not enough. You must also optimize and tune the model appropriately so that it provides you with accurate results. Optimization techniques involve tuning the hyperparameters to reach an optimum result.

The world today is evolving and so are the needs and requirements of people. Furthermore, we are witnessing a fourth industrial revolution of data. In order to derive meaningful insights from this data and learn from the way in which people and the system interface with the data, we need computational algorithms that can churn the data and provide us with results that would benefit us in various ways.

Machine Learning has revolutionized industries like medicine, healthcare, manufacturing, banking, and several other industries. Therefore, Machine Learning has become an essential part of modern industry. Data is expanding exponentially and in order to harness the power of this data, added by the massive increase in

computation power, Machine Learning has added another dimension to the way we perceive information. Machine Learning is being utilized everywhere. The electronic devices you use, the applications that are part of your everyday life are powered by powerful machine learning algorithms.

With an exponential increase in data, there is a need for having a system that can handle this massive load of data. Machine Learning models like Deep Learning allow the vast majority of data to be handled with an accurate generation of predictions. Machine Learning has revolutionized the way we perceive information and the various insights we can gain out of it. These machine learning algorithms use the patterns contained in the training data to perform classification and future predictions. Whenever any new input is introduced to the ML model, it applies its learned patterns over the new data to make future predictions. Based on the final accuracy, one can optimize their models using various standardized approaches. In this way, Machine Learning model learns to adapt to new examples and produce better results.

Types of Machine Learning

Machine Learning Algorithms can be classified into 3 types as follows –

- Supervised Learning
- Unsupervised Learning
- Reinforcement Learning

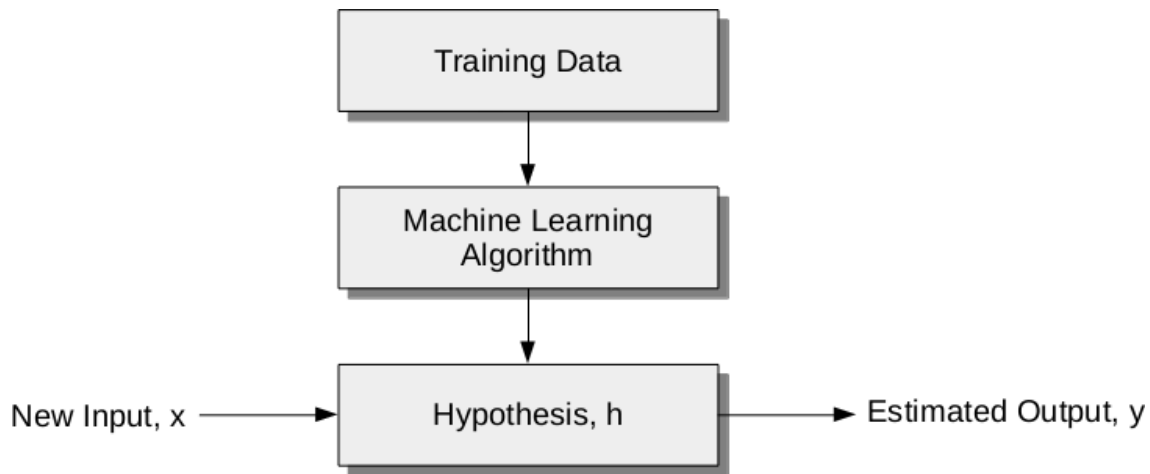


Fig 1.1 Machine Learning

Supervised Learning

In the majority of supervised learning applications, the ultimate goal is to develop a finely tuned predictor function $h(x)$ (sometimes called the “hypothesis”). “Learning” consists of using sophisticated mathematical algorithms to optimize this function so that, given input data x about a certain domain (say, square footage of a house), it will accurately predict some interesting value $h(x)$ (say, market price for said house).

$$h(x_1, x_2, x_3, x_4) = \theta_0 + \theta_1 x_1 + \theta_2 x_3^2 + \theta_3 x_3 x_4 + \theta_4 x_1^3 x_2^2 + \theta_5 x_2 x_3^4 x_4^2$$

This function takes input in four dimensions and has a variety of polynomial terms. Deriving a normal equation for this function is a significant challenge. Many modern machine learning problems take thousands or even millions of dimensions of data to build predictions using hundreds of coefficients. Predicting how an organism’s genome will be expressed, or what the climate will be like in fifty years, are examples of such complex problems.

Under supervised ML, two major subcategories are:

- Regression machine learning systems: Systems where the value being predicted falls somewhere on a continuous spectrum. These systems help us with questions of “How much?” or “How many?”.
- Classification machine learning systems: Systems where we seek a yes-or-no prediction, such as “Is this tumor cancerous?”, “Does this cookie meet our quality standards?”, and so on.

In practice, x almost always represents multiple data points. So, for example, a housing price predictor might take not only square-footage (x_1) but also number of bedrooms (x_2),

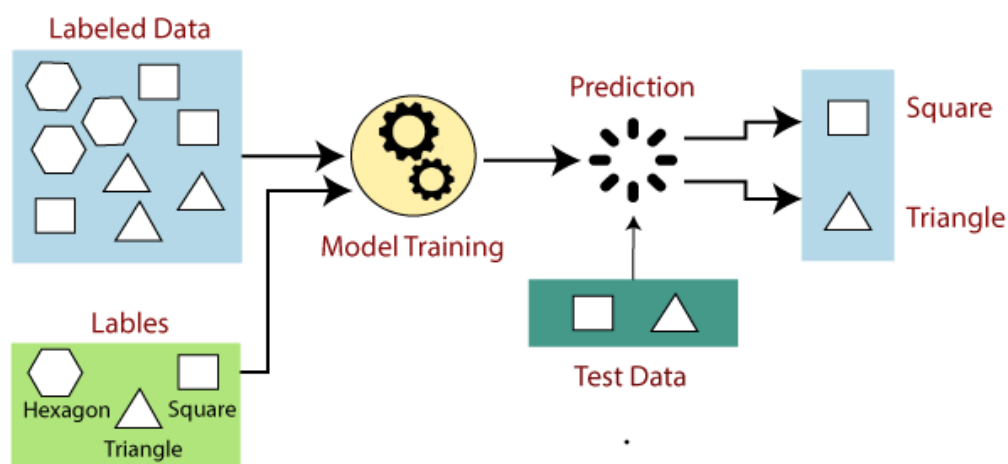


Fig 1.2 Supervised Learning

number of bathrooms (x_3), number of floors (x_4), year built (x_5), zip code (x_6), and so forth. Determining which inputs to use is an important part of ML design. However, for the sake of explanation, it is easiest to assume a single input value is used.

Steps Involved in Supervised Learning:

- First Determine the type of training dataset
- Collect/Gather the labelled training data.
- Split the training dataset into training dataset, test dataset, and validation dataset.
- Determine the input features of the training dataset, which should have enough knowledge so that the model can accurately predict the output.
- Determine the suitable algorithm for the model, such as support vector machine, decision tree, etc.
- Execute the algorithm on the training dataset. Sometimes we need validation sets as the control parameters, which are the subset of training datasets.
- Evaluate the accuracy of the model by providing the test set. If the model predicts the correct output, which means our model is accurate.

Regression

Regression algorithms are used if there is a relationship between the input variable and the output variable. It is used for the prediction of continuous variables, such as Weather forecasting, Market Trends, etc. Below are some popular Regression algorithms which come under supervised learning:

- Linear Regression
- Regression Trees
- Non-Linear Regression
- Bayesian Linear Regression
- Polynomial Regression

Classification

Classification algorithms are used when the output variable is categorical, which means there are two classes such as Yes-No, Male-Female, True-false, etc.

- Spam Filtering,
- Random Forest
- Decision Trees
- Logistic Regression
- Support vector Machines

CHAPTER 2

LITERATURE SURVEY

BRIOT: BEHAVIOR RULE SPECIFICATION-BASED MISBEHAVIOR DETECTION FOR IOT-EMBEDDED CYBER-PHYSICAL SYSTEMS

The identification of vulnerabilities in a mission-critical system is one of the challenges faced by a Cyber-Physical System (CPS). The incorporation of embedded Internet of Things (IoT) devices makes it tedious to identify vulnerability and difficult to control the service- interruptions and manage the operations losses. Rule-based mechanisms have been considered as a solution in the past. However, rule-based solutions operate on the goodwill of the generated rules and perform assumption- based detection. Such a solution often is far from the actual realization of IoT runtime performance and can be fooled by zero-day attacks. Thus, this paper takes this issue as a motivation and proposes better lightweight behavior rule specification-based misbehavior detection for IoT-embedded cyber-physical systems (BRIoT). The key concept of our approach is to model a system with which misbehavior of an IoT device manifested as a result of attacks exploiting the vulnerability exposed may be detected through automatic model checking and formal verification, regardless of whether the attack is known or unknown. Automatic model checking and formal verification are achieved through a 2-layer Fuzzy-based Hierarchical Context-Aware Aspect-Oriented Petri Net (HCAPN) model, while effective misbehavior detection to avoid false alarms is achieved through a Barycentric-coordinated based center of mass calculation method. The proposed approach is verified by an unmanned aerial vehicle (UAV) embedded in a UAV system. The feasibility of the proposed model is demonstrated with high reliability, low operational cost, low false-positives, low false- negatives, and high true positives in comparison with existing rule- based solutions.

Our behavior-rule based IDS approach relies on the use of monitoring nodes. We assume that a monitoring node performs misbehavior detection on a target node. One possible design is to have a sensor (actuator) monitor another sensor (actuator respectively) within the same CPS. This may require each sensor (actuator) to have multiple sensing functionalities. Note that a malicious embedded IoT device cannot

bypass misbehavior detection because our approach is based on a device being monitored by a peer device (or more than one peer IoT devices to increase the detection strength). If a peer monitoring IoT device is itself malicious and performs attacks, its misbehavior would be detected by another peer IoT device. Further, whenever an IoT device is identified as malicious, its monitoring duty would be reassigned to another IoT device. Therefore, no malicious IoT device can bypass detection in our approach. Another possibility is that each IoT device is built on top of secure computational space.

Once the behavior rules are formally verified and proven correct, we transform the corresponding ABIs into a C-language state machine for misbehavior detection of the specified target IoT device. This part is also performed at static time. Then we preload the state machine into the memory of a monitoring node and assign the monitoring node the duty of monitoring and detecting misbehavior of the target IoT device. This misbehavior detection part is performed at runtime. During runtime, misbehavior data detected if any are collected by the monitoring node via an on-board lightweight data collector. Subsequently the data collected are fed into a lightweight statistical analyzer (also on-board as it is lightweight) to judge if the target IoT device is malicious.

We propose to use “operational profile”, which essentially is a mission assignment statement generated according to the probabilities with which events will happen to an embedded IoT device during its mission execution as input to BRIoT. A mission assignment statement explicitly defines a set of security requirements for the mission to be successful, from which a set of threats as well as a set of behavior rules to cope with the threats may be automatically derived. If the “operational profile” of an embedded IoT device (by software engineers who developed the IoT device) which defines the security requirements is available, then it can be modeled to automatically identify the complete set of threats and consequently derive the complete set of behavior rules. Otherwise, the system designer would be guided to define the “anticipated operational profile” as input.

HYBRID AND MULTIFACETED CONTEXT-AWARE MISBEHAVIOR DETECTION MODEL FOR VEHICULAR AD HOC NETWORK

Vehicular Ad Hoc Networks (VANETs) have emerged mainly to improve road safety and traffic efficiency and provide user comfort. The performance of such networks' applications relies on the availability of accurate and recent mobility-information shared among vehicles. This means that misbehaving vehicles that share false mobility information can lead to catastrophic losses of life and property. However, the current solutions proposed to detect misbehaving vehicles are not able to cope with the dynamic vehicular context and the diverse cyber-threats, leading to a decrease in detection accuracy and an increase in false alarms. This paper addresses these issues by proposing a Hybrid and Multifaceted Context-aware Misbehavior Detection model (HCA-MDS), which consists of four phases: data-collection, context-representation, context-reference construction, and misbehavior detection. Data-centric and behavioral-detection-based features are derived to represent the vehicular context. An online and timely updated context-reference model is built using unsupervised nonparametric statistical methods, namely Kalman and Hampel filters, through analyzing the temporal and spatial correlation of the consistency between mobility information to adapt to the highly dynamic vehicular context. Vehicles' behaviors are evaluated locally and autonomously according to the consistency, plausibility, and reliability of their mobility information. The results from extensive simulations show that HCA-MDS outperforms existing solutions in increasing the detection rate by 38% and decreasing the false positive rate by 7%. These results demonstrate the effectiveness and robustness of the proposed HCA-MDS model to strengthen the security of VANET applications and protocols.

The proposed misbehavior detection model is host-based; i.e., it is deployed for each vehicle to detect local misbehavior early in its initial stages before it develops into a sophisticated attack. Due to the absence of labeled attack data, the proposed solution uses an unsupervised statistical method to detect unknown attacks. This method is based on the mobility information, which is the main building block of many critical VANET applications. In addition, such mobility information is the target for many types of difficult-to-detect attacks that can easily disrupt VANET applications and services. The proposed model is composed of a set of multifaceted base classifiers to consolidate the diversity of the ensemble, which

in turn, improves the capability of the model to detect many types of misbehaviors that can tamper with VANET data. Furthermore, the model is context-aware, in which the context-based features are used to train the base classifiers. Therefore, the model is able to adapt to any changes in the context.

The neighbor predictor algorithm of the DSA-ABR is responsible for collecting the mobility messages broadcast by neighboring vehicles, as well as predicting the lost and/or the omitted mobility messages. It uses a modified version of the Kalman Filter, in which the parameters of the self-predictor algorithm of the neighboring vehicles are shared along with the mobility information to improve the prediction accuracy [21]. The output of this phase is a dataset for each neighboring vehicle. Each dataset contains a history of vehicles' mobility information at each 100ms. In this phase, the features that represent the context are derived. Three context factors were considered for the construction of the context references: the driver's behavior, environmental noises, and communication status. These three factors have a direct impact on the data-consistency, plausibility, and vehicles' behavior, raising the need for context-aware systems. Due to drivers' maneuvering behavior, vehicles may drive in and out of communication ranges of each other, causing intermittent communication and loss of context information. Similarly, due to high vehicle mobility and variable traffic density, communication channels become congested, causing intermittent communication and loss of context information and thus an incomplete context.

SURVEY ON MISBEHAVIOR DETECTION IN COOPERATIVE INTELLIGENT TRANSPORTATION SYSTEMS

Cooperative Intelligent Transportation Systems (cITS) are a promising technology to enhance driving safety and efficiency. Vehicles communicate wirelessly with other vehicles and infrastructure, thereby creating a highly dynamic and heterogeneously managed ad-hoc network. It is these network properties that make it a challenging task to protect integrity of the data and guarantee its correctness. A major component is the problem that traditional security mechanisms like PKI based asymmetric cryptography only exclude outsider attackers that do not possess key material. However, because attackers can be insiders within the network (i.e., possess valid key material), this approach cannot detect all possible attacks. In this survey, we present misbehavior detection

mechanisms that can detect such insider attacks based on attacker behavior and information analysis. In contrast to well-known intrusion detection for classical IT systems, these misbehavior detection mechanisms analyze information semantics to detect attacks, which aligns better with highly application-tailored communication protocols foreseen for cITS. In our survey, we provide an extensive introduction to the cITS ecosystem and discuss shortcomings of PKI-based security. We derive and discuss a classification for misbehavior detection mechanisms, provide an in-depth overview of seminal papers on the topic, and highlight open issues and possible future research trends.

In contrast to routing in classical IT systems such as the Internet, multicast and broadcast are the predominant networking patterns in cITS. These patterns are much more suitable for the envisioned applications, which vary from lane change warnings to cooperative adaptive cruise control (CACC) and city-scale traffic flow optimization. CACC is an application that has gained significant attention in recent years, and is essentially an extension of adaptive cruise control (ACC), where the safety distance between vehicles is reduced. In CACC, vehicles periodically exchange position information to form a very tight formation that would normally be susceptible to collisions. This is a form of partially autonomous driving that goes beyond the potential benefits from sensor-equipped vehicles. The data exchanged between the network participants, vehicles and infrastructure alike, share the property that they are relevant to all receivers. Therefore, addressing packets to specific vehicles does not make sense; instead, addressing refers to the local neighborhood (1-hop broadcast), specific regions (geocast) and infrastructure. This style of addressing allows the network to exploit the simple fact that wireless networks are a broadcast medium by nature.

Beyond the communication model under standardization, some authors propose that clustering might be a viable approach to organize the communication. Current standards do not foresee this approach because it has significant disadvantages in terms of communication overhead to create and maintain clusters. These clusters are typically created and maintained in an ad-hoc fashion, to facilitate deployment, which requires vehicles to agree on one or more cluster heads that manage a cluster. The disadvantage of this approach is that it requires a fundamental inequality between different vehicles to take advantage of clustering, i.e., the cluster heads must have an authoritative status. Because this network model security challenges that are widely different from the standard cITS communication models, we do not address them in detail in this survey.

If vehicles attach their long-term certificates to all outgoing messages, they can be tracked using a trace of received messages with attached location information. For this reason, long-term certificates can be replaced by short-term identifiers, referred to as pseudonyms. These pseudonyms make it harder to collect location traces of specific vehicles while still providing the same authenticity guarantees. While pseudonyms enhance privacy, their parallel use enables potential attacks, often referred to as Sybil attacks [46]. This is one of the classes of attacks that we will discuss extensively in this survey (in section IV-A2), as it is one of the key challenges that misbehavior detection helps address. However, in this section we first present a discussion of some proposals for certificate management.

RELIABLE MULTISERVICE DELIVERY IN FOG-ENABLED VANETS: INTEGRATED MISBEHAVIOR DETECTION AND TOLERANCE

Vehicular ad hoc networks (VANETs) in which vehicles act as the mobile nodes provide a wide variety of services, such as audio and video surveillance. However, such networks suffer from an important problem of service delivery reliability as the network performance degrades significantly in the presence of misbehaving vehicles. Most of the existing works either assume that vehicles' misbehaviors are constant or ignore heterogeneous traffic for multiservice (a mix of video, voice and data traffic). In this paper, we investigate the problem of trust-based multiservice delivery via integrated misbehavior detection and tolerance for fault-aware VANETs. To model the effects of time-varying misbehaviors, modern fog computing could help analyzing and storing related data in VANETs while evaluating the dynamic trust weights based on attribute parameters of each vehicle. Then we incorporate these weights into our service delivery framework that integrates trustworthy vehicle selection for misbehavior detection and uses differential resource allocation to achieve misbehavior tolerance. We present a multi-path selection criterion based on the trust evaluation and design a trust-aware heterogeneous traffic allocation algorithm over multiple routing paths. Finally, our scheme is evaluated using extensive simulations where we show that

(i) approximately 12% 40% higher successful service delivery can be achieved by our routing algorithm at the expense of acceptable delay loss compared to other three routing protocols;

(ii) our trust aware traffic allocation algorithm can gain 10% 20% higher effective network utility and better fairness than that of the baseline solution;

(iii) the integrated scheme significantly improves both effective utility ratio and path usage proportion by comparing it with only routing scheme and only traffic allocation scheme.

The proposed routing and traffic allocation algorithms utilize the fog server to evaluate the trust weights of every vehicle. The trust weights are calculated in terms of faulty behavior factors upload beach vehicle and its neighbors. We assume that the vehicles information can be acquired, transmitted, stored and analyzed through Internet to Vehicles. In the route discovery and maintenance, the source finds or reroutes multiple paths to the destination node according to the trust records of intermediate nodes of sensors which generate massive data referred to as Big Data from sensing the status of roads and vehicles [10]. The huge amount and various types of data generated by vehicles can be divided into two groups: data from intrinsic aspect and data from extrinsic aspect. Data from intrinsic aspect that directly collects the vehicle information such as numbers of violation, engine parameters, numbers of accident, mileage, velocity, etc. Data from extrinsic aspect refers to the information about the relationship with other vehicles or the environmental conditions, e.g., distance between vehicles and neighboring ones, warning notice, blind spots, trajectory, road map, etc. The principal component analysis (PCA) approach is a common data analysis method that decrease dimensions of multi-variable space. In the big data era of vehicles, we can use PCA approach to perform the trust value in terms of vehicle “big data” which is a multi-variable data table.

In the precious section, we introduce the trust-aware multipath routing protocol for the VANET. To make effective utilizing of multiple routing paths, each source node must be able to allocate traffic intelligently across the routing paths while considering the potential impact of misbehaviors on service delivery. In this section, we present a trust-aware traffic allocation algorithm to generate the optimal effective rate control among multiple services.

CHAPTER 3

SYSTEM ANALYSIS

One of the primary choices we face for our project implementations is “Which development methodology should we use?” There are various software development methodologies available in the ICT industry. They are

- Agile Software Development
- Crystal Methods
- Dynamic Systems Development Model (DSDM)
- Extreme Programming (XP)
- Feature Driven Development (FDD)
- Joint Application Development (JAD)
- Lean Development (LD)
- Rapid Application Development (RAD)
- Rational Unified Process (RUP)
- Scrum
- Spiral
- Systems Development Life Cycle (SDLC)
- Waterfall (a.k.a. Traditional)

Among all these methodologies, we have decided to compare the two most popular

and widely used methodologies:

- Waterfall Model
- RAD Model

Waterfall Model

It is called as traditional approach. Waterfall is a linear method (sequential model) for any software application development. Here application development is segregated into a sequence of pre-defined phases.

Waterfall Model

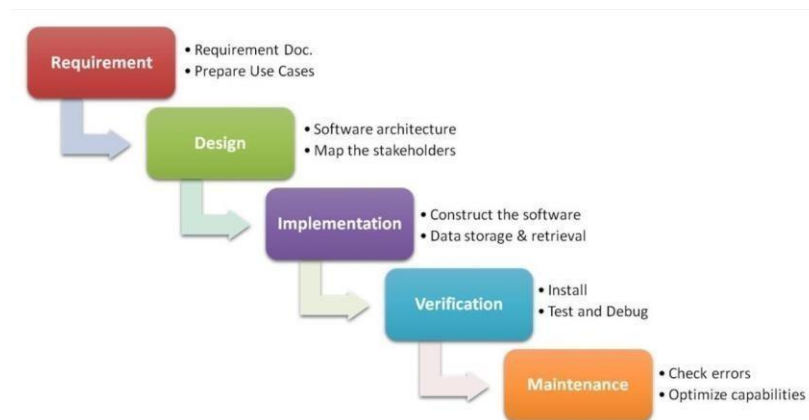


Fig 3.1 Waterfall Model

1. Requirement gathering and documentation

In this stage, you should gather comprehensive data approximately what this challenge requires. You may gather this data in a diffusion of ways, from interviews to questionnaires to interactive brainstorming. By means of the stop of this section, the task requirements have to be clean, and also you have to have a necessities file that has been allotted on your team.

2. System design

Using the well-known requirements, your team designs the solutions. During this phase, no development will be happening. But the project team starts specification such as programming language or hardware requirements.

3. Implementation

During this phase software development coding will be happening. Webapplication programmers take data from the previous stage and create a functional product. Web application programmers write source code in small pieces, which are integrated at the end of this phase or the beginning of the next.

4. Testing

Once all coding is done, testing of the product can begin. Testers methodically find and report any problems. If serious issues arise, your project may need to return to phase one for reevaluation.

5. Delivery/deployment

In this phase, the solution is complete, and your project team submits the deliverables to be deployed or released.

6. Maintenance

The final solution has been implemented to the client and is being used. As troubles arises, the project team might also want to create patches and updates might also to deal with them. Again,huge troubles may also necessitate a return to segment one.

RAD Model

Agile is an iterative, team-based approach to development. This method highlights the speedy delivery of a software application in comprehensive functionalcomponents

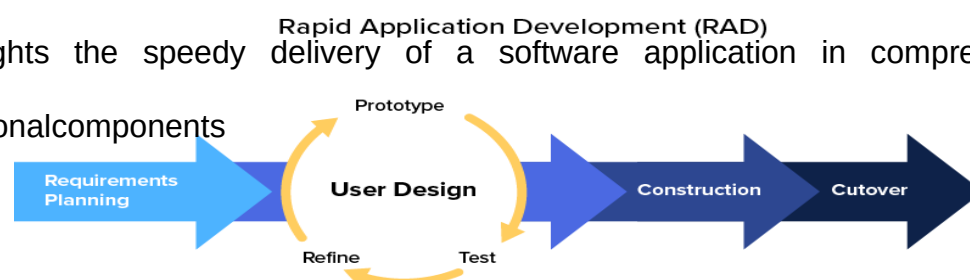


Fig3.2RAD Diagram

- ***Phase 1: Requirements planning***

This phase is equivalent to a project scoping meeting. Although the planning phase is condensed compared to other project management methodologies, this is a critical step for the ultimate success of the project. During this stage, web application programmers, clients (software users), and team members communicate to determine the goals and expectations for the project as well as current and potential issues that would need to be addressed during the build.

A basic breakdown of this stage involves:

- Finalizing the requirements with each stakeholder's approval
- Researching the current project

It is important that everyone has the opportunity to evaluate the goals and expectations for the project and weigh in. By getting approval from each key stakeholder and web application programmer, teams can avoid miscommunications and costly change orders down the road.

Phase 2: User design

Once the project is scoped out, it's time to jump right into development, building out the user design through various prototype iterations. This is the meat and potatoes of the RAD methodology—and what sets it apart from other project management strategies. During this phase, clients work hand in hand with web application programmers to ensure their needs are being met at every step in the design process. It's almost like customizable software development where the users can test each prototype of the product, at each stage, to ensure it meets their expectations.

All the bugs and kinks are worked out in an iterative process. The web application programmer designs a prototype, the client (user) tests it, and then they come together to communicate on what worked and what didn't.

This method gives web application programmers the opportunity to tweak the model as they go until they reach a satisfactory design. Both the software web application programmers and the clients learn from the experience to make sure there is no potential for something to slip through the cracks.

Phase 3: Rapid construction

Phase 3 takes the prototypes and beta systems from the design phase and converts them into the working model. Because the majority of the problems and changes were addressed during the thorough iterative design phase, web application programmers can construct the final working model more quickly than they could by following a traditional project management approach.

The phase breaks down into several smaller steps:

- Preparation for rapid construction
- Programming application
- Development, Coding
- Unit, integration, and system testing

The software development team of programmers, coders, testers, and web application programmers work together during this stage to make sure everything is working smoothly and that the end result satisfies the client's expectations and objectives.

This third phase is important because the client still gets to give input throughout the process. They can suggest alterations, changes, or even new ideas that can solve problems as they arise.

Phase 4: Cutover

This is the implementation phase where the finished product goes to launch. It includes data conversion, testing, and changeover to the new system, as well as user training.

All final changes are made while the coders and clients continue to look for bugs in the system. Benefits of RAD methodology RAD is one of the most successful software development programs available today, with numerous benefits for both software development teams as well as their clients.

Here are just a few advantages:

- ▶ RAD lets you break the project down into smaller, more manageable tasks.
- ▶ The task-oriented structure allows project managers to optimize their team's efficiency by assigning tasks according to members' specialties and experience.
- ▶ Clients get a working product delivered in a shorter time frame.
- ▶ Regular communication and constant feedback between team members and stakeholders increases the efficiency of the design and build process.

3.1 Existing System

The existing system misbehavior detection system (MDS) can be deployed to detect and prevent internal attacks. In this paper, we propose a machine learning and reputation based MDS to enhance the detection accuracy as well as to ensure the reliability of both vehicles and messages. Existing system MDS is trained using datasets generated through extensive simulation based on the realistic vehicular network environment. To improve the accuracy of the detection, we have employed the Dempster Shafer (DS) theory-based collaborative misbehavior detection system. In the existing system scheme, the reputation score of each vehicle is used as a belief value for Dempster-Shafer based feedback combination. In addition, existing system a beta distribution-based reputation update and revocation scheme.

An attacker can control some vehicles or groups of vehicles to broadcast false information. An attacker can broadcast false alerts such as emergency vehicle approaching or icy road conditions to disrupt the normal operation of the traffic. These kinds of false alerts can cause huge accidents; thus, it is very crucial to detect them.

3.2 Drawbacks of Existing System

- It achieves low-latency in delivery with less energy consumption.
- The drawback is not improved detection performance
- Neural network must be trained in advance

- Parties need to be involved
- Cannot overcome inter-packet dependency

3.3 Proposed System

The proposed system presents a period-based defense mechanism (PDM scheme is based on the periods and uses a blacklist to efficiently prevent the data flooding attack, by checking the data packet floods at the end of each period in order to enhance the throughput of burst traffic. Therefore, it can guarantee the Quality of Service (QoS) of burst traffic. As a result of which many data packets are forwarded at a high rate for the whole duration. Flood attacks are launched by malicious or selfish nodes. Malicious nodes, which can be the nodes purposely setup by the opponent or subverted by the opponent via mobile phone worms begin attacks to congest the network and misuse the resources of other nodes.

Selfish nodes may also develop flood attacks to increase their communication throughput. In DTNs, a single packet usually can only be carried to the destination with chances smaller than 1 due to the opportunistic connectivity. If a selfish node floods many replicas of its own packet, it can increase the probability of its packet being delivered, since the delivery of any replica means successful delivery of the packet. With packet flood attacks, selfish nodes can also boost their throughput, although in a subtler manner.

In the proposed Single-copy routing after sending a packet out, a node deletes its own copy of the packet. Thus, each packet only has one copy in the network. In the proposed Multicopy routing to the source node of a packet sprays a certain number of replicas of the packet to other nodes and each copy is separately routed using the single-copy strategy.

The maximum number of copies that each packet can have is set. In the proposed, Propagation routing (when a node locates it appropriate (according to the routing algorithm) to send a packet to another encountered node, it replicates that packet to the encountered node and keeps its own fake. There is no preset limit over the number of copies a packet can have. In Propagation, a node duplicates a packet to another encountered node if the latter has more regular

contacts with the destination of the packet.

3.4 Advantages of Proposed System

- Reduce false alarm probability.
- Cost reduction for its users.
- Higher accuracy.
- Opaqueness since routers need not be involved.
- Successfully minimize the amount of traffic.

CHAPTER 4

MATERIALS AND METHODS

4.1 Introduction

Clearly defined requirements are essential signs on the road that leads to a successful project. They establish a formal agreement between a client and a provider that they are both working to reach the same goal. High-quality, detailed requirements also help mitigate financial risks and keep the project on a schedule. According to the Business Analysis Body of Knowledge definition, requirements are a usable representation of a need.

Creating requirements is a complex task as it includes a set of processes such as elicitation, analysis, specification, validation, and management. In this article, we'll discuss the main types of requirements for software products and provide a number of recommendations for their use.

4.2 Hardware & Software Specification

4.2.1 Hardware Specification

Hardware Requirement:

- Hardware Minimum Requirement
- Disk Space 32 GB or more
- Processor 1.4 GHz 64 bit
- Memory 512 MB
- Display (800 × 600) Capable video adapter and monitor

4.2.2 Software Specification

Backend Technologies

- Python
- Numpy
- Sci-learn
- Eclipse IDE

Frontend Technologies

- Web Technologies
- Bootstrap

4.3 Software Description

4.3.1 About Python

Python is a free, open-source programming language. Therefore, all you have to do is install Python once, and you can start working with it. Not to mention that you can contribute your own code to the community. Python is also a cross-platform compatible language. So, what does this mean? Well, you can install and run Python on several operating systems. Whether you have a Windows, Mac or Linux, you can rest assure that Python will work on all these operating systems.

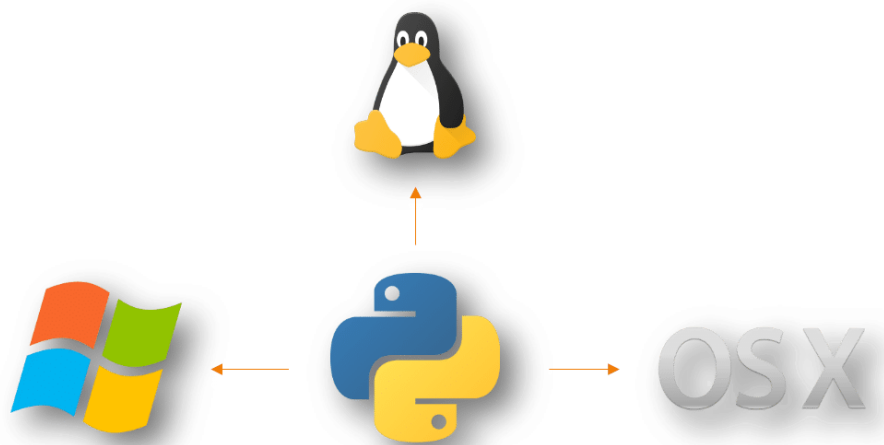


Fig 3.1 Python Relations

Python is also a great visualization tool. It provides libraries such as Matplotlib, seaborn and bokeh to create stunning visualizations.

In addition, Python is the most popular language for machine learning and deep learning. As a matter of fact, today, all top organizations are investing in Python to implement machine learning in the back-end.

Python Line Structure

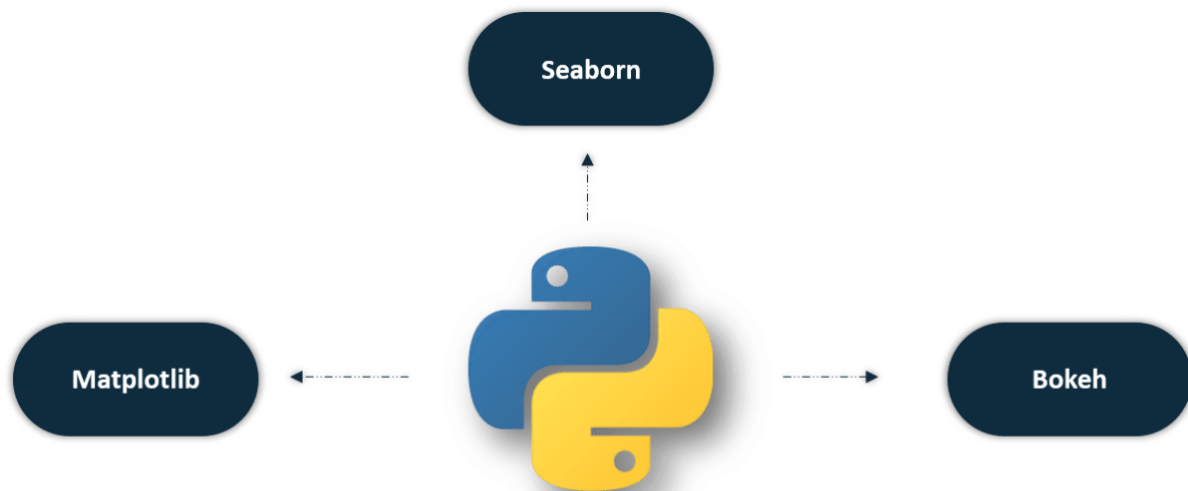


Fig 4.2 Python Line Structure Hierarchy

Python coding style comprises physical lines as well as logical lines or statements. A physical line in a Python program is a sequence of characters, and the end of the line terminates the line sequence as opposed to some other languages, such as C and C++ where a semi-colon is used to mark the end of the statement. A logical line, on the other hand, is composed of one or more physical lines. The use of a semi-colon is not prohibited in Python, although it's not mandatory. The NEWLINE token denotes the end of the logical line. A logical line that only contains spaces, comments, or tabs are called blank lines and they are ignored by the interpreter.

As we saw that in Python, a new line simply means that a new statement has started. Although, Python does provide a way to split a statement into a multiline statement or to join multiple statements into one logical line. This can be helpful to increase the readability of the statement. Following are the two ways to split a line

into two or more lines:

Explicit Line Joining

In explicit line joining, we use a backward slash to split a statement into a multiline statement. Implicit Line Joining Statements that reside inside [], {}, or () parentheses can be broken down into two or more physical lines without using a back slash. Multiple Statements on a Single Line in Python, it is possible to club multiple statements in the same line using a semi-colon; however, most programmers do not consider this to be a good practice as it reduces the readability of the code.

Whitespaces and Indentation

Unlike most of the programming languages, Python uses indentation to mark a block of code. According to Python coding style guideline or PEP8, we should keep an indent size of four. Most of the programming languages provide indentation for better code formatting and don't enforce to have it. But in Python it is mandatory. This is why indentation is so crucial in Python. Comments in any programming language are used to increase the readability of the code. Similarly, in Python, when the program starts getting complicated, one of the best ways to maintain the readability of the code is to use Python comments.

It is considered a good practice to include documentations and notes in the python syntax since it makes the code way more readable and understandable to other programmers as well, which comes in handy when multiple programmers are simultaneously working on the same project.

The code can only explain how it does something and not why it does that, but Python comments can do that. With Python comments, we can make documentations for various explanations in our code itself. Comments are nothing but tagged lines of codes which increase the readability of a code and make it self-explanatory. There are different ways of creating comments depending on the type of comment we want to include in our code. Following are different kinds of comments that can be included in our Python program:

1. Single Line Comments
2. Docstring Comments

3. Multiline Comments

Single line Python comments are marked with # character. These comments end at the end of the physical line, which means that all characters starting after the # character (and lasts till the end of the line) are part of the comment.

Python has the documentation strings (or docstrings) feature which is usually the first statement included in functions and modules. Rather than being ignored by the Python Interpreter like regular comments, docstrings can actually be accessed at the run time using the dot operator.

It gives programmers an easy way of adding quick notes with every Python module, function, class, and method. To use this feature, we use triple quotes in the beginning of the documentation string or comment and the closing triple quotes at the end of the documentation comment. Docstrings can be one-liners as well as multi-liners. Unlike some programming languages that support multiline comments, such as C, Java, and more, there is no specific feature for multiline comments in Python. But that does not mean that it is totally impossible to make multiline comments in Python. There are two ways we can include comments that can span across multiple lines in our Python code.

Python Block Comments: We can use several single line comments for a whole block. This type of comment is usually created to explain the block of code that follows the Block comment. Python Block comment is the only way of writing a real comment that

can span across multiple lines. It is supported and preferred by Python's PEP8 style guide since Block comments are ignored by Python interpreter or parser. However, nothing is stopping programmers from using the second 'non-real' way of writing multiline comments in Python which is explained below:

Using Docstrings

Docstrings are largely used as multiline comments in Python by many programmers since it is the closest thing to having a multiline comment feature in Python. While it is not wrong to use docstrings when we need to make multiline comments, it is important to keep in mind that there is a significant difference between docstrings and comments. Comments in Python are totally ignored by the Python Interpreter, while docstrings, when used inside the Python function, can be

accessed at the run time.

Python Data Types

One of the most crucial part of learning any programming language is to understand how data is stored and manipulated in that language. Users are often inclined toward Python because of its ease of use and the number of versatile features it provides. One of those features is dynamic typing. In Python, unlike statically typed languages like C or Java, there is no need to specifically declare the data type of the variable. In dynamically typed languages such as Python, the interpreter itself predicts the data type of the Python Variable based on the type of value assigned to that variable

Advantages of Python

- Universal Language Construct
- Support both High Level and Low-Level Programming
- Language Interoperability
- Fastest Development life cycle therefore more productive coding environment
- Less memory used because a single container holds multiple data types and each type doesn't require its own function
- Learning Ease and open-source development
- Speed and user-friendly data structure
- Extensive and extensible libraries.
- Simple & support IoT

Based on the 3 languages that are described above, we decided to use Python as programming language for developing this e-Voting web-based application. The

important motives are

- Easy to learn, even non experienced programmers can use it. Ex: spacing and tabbing instead of extra syntax
- Interactive mode
- Large and comprehensive standard libraries
- Python programs resemble to that of pseudo-code. This makes it a basis and a musthave for beginner programmers due to its extreme ease and difficulty when compared to C++ Java, Perl, and so forth

About Pandas

Pandas is a popular Python package for data science, and with good reason: it offers powerful, expressive and flexible data structures that make data manipulation and analysis easy, among many other things. The Data Frame is one of these structures.

Pandas is a high-level data manipulation tool developed by Wes McKinney. It is built on the NumPy package and its key data structure is called the Data Frame. Data Frames allow you to store and manipulate tabular data in rows of observations

and columns of variables.

Pandas is built on top of the NumPy package, meaning a lot of the structure of NumPy is used or replicated in Pandas. Data in pandas is often used to feed statistical analysis in SciPy, plotting functions from Matplotlib, and machine learning algorithms in Scikit-learn. Jupyter Notebooks offer a good environment for using pandas to do data exploration and modeling, but pandas can also be used in text editors just as easily.

Jupyter Notebooks give us the ability to execute code in a particular cell as opposed to running the entire file. This saves a lot of time when working with large datasets and complex transformations. Notebooks also provide an easy way to visualize pandas' Data Frames and plots. As a matter of fact, this article was created entirely in a Jupyter Notebook.

There are two types of data structures in pandas: Series and Data Frames.

1. Series: a pandas Series is a one-dimensional data structure (“a one-dimensional array”) that can store values — and for every value it holds a unique index, too.
2. Data Frame: a pandas Data Frame is a two (or more) dimensional data structure – basically a table with rows and columns. The columns have names and the rows have indexes.

Those who are familiar with R know the data frame as a way to store data in rectangular grids that can easily be overviewed. Each row of these grids corresponds to measurements or values of an instance, while each column is a vector containing data for a specific variable. This means that a data frame’s rows do not need to contain, but can contain, the same type of values: they can be numeric, character, logical, etc. Now, Data Frames in Python are very similar: they come with the Pandas library, and they are defined as two-dimensional labeled data structures with columns of potentially different types. In general, you could say that the Pandas Data Frame consists of three main components: the data, the index, and the columns.

Firstly, the Data Frame can contain data that is:

- a Pandas Data Frame
- a Pandas Series: a one-dimensional labeled array capable of holding any data type with axis labels or index. An example of a Series object is one column from a data Frame.
- a NumPy Nd array, which can be a record or structured
- a two-dimensional ndarray
- dictionaries of one-dimensional ndarray’s, lists, dictionaries or Series.

Some of the key features of Python Pandas are as follows:

It provides data Frame objects with default and customized indexing which is very fast and efficient. There are tools available for loading data of different file formats into in-memory data objects.

- It is easy to perform data alignment and integrated handling of missing data in Python Pandas.
- It is very simple to perform pivoting and reshaping of data sets in Pandas.
- It also provides indexing, label-based slicing, and sub-setting of large data sets.
- We can easily insert and delete columns from a data structure.
- Data aggregation and transformations can be done using group by.
- High-performance merging and joining of data can be done using Pandas.
- It also provides time series functionality.
- Inserting and deleting columns in data structures.
- Merging and joining data sets.
- Reshaping and pivoting data sets.
- Aligning data and dealing with missing data.
- Manipulating data using integrated indexing for data Frame objects.
- Performing split-apply-combine on data sets using the group by engine.
- Manipulating high-dimensional data in a data structure with a lower dimension using hierarchical axis indexing.
- sub setting, fancy indexing, and label-based slicing data sets that are large in size.
- Generating data range, converting frequency, date shifting, lagging, and other time-series functionality.
- Reading from files with CSV, XLSX, TXT, among other formats.
- Arranging data in an order ascending or descending.
- Filtering data around a condition.
- Analyzing time series.
- Iterating over a data set.

NumPy

NumPy is the core library for scientific computing in Python. It provides a high-performance multidimensional array object, and tools for working with these arrays. If you are already familiar with MATLAB, you might find this tutorial useful to get started with NumPy. A NumPy array is a grid of values, all of the same type, and is indexed by a tuple of nonnegative integers. The number of dimensions is the rank

of the array; the shape of an array is a tuple of integers giving the size of the array along each dimension.

NumPy is, just like SciPy, Scikit-Learn, Pandas, etc. one of the packages that you just can't miss when you're learning data science, mainly because this library provides you with an array data structure that holds some benefits over Python lists, such as: being more compact, faster access in reading and writing items, being more convenient and more efficient.

NumPy is a Python library that is the core library for scientific computing in Python. It contains a collection of tools and techniques that can be used to solve on a computer mathematical models of problems in Science and Engineering. One of these tools is a high-performance multidimensional array object that is a powerful data structure for efficient computation of arrays and matrices. To work with these arrays, there's a vast amount of high-level mathematical functions operate on these matrices and arrays. An array is basically nothing but pointers. It's a combination of a memory address, a data type, a shape, and strides: The data pointer indicates the memory address of the first byte in the array The data type or d-type pointer describes the kind of elements that are contained within the array, The shape indicates the shape of the array, and the strides are the number of bytes that should be skipped in memory to go to the next element. If your strides are (10,1), you need to proceed one byte to get to the next column and 10 bytes to locate the next row. Or, in other words, an array contains information about the raw data, how to locate an element and how to interpret an element. With NumPy, we work with multidimensional arrays.

We'll dive into all of the possible types of multidimensional arrays later on, but for now, we'll focus on 2-dimensional arrays. A 2- dimensional array is also known as a matrix, and is something you should be familiar with. In fact, it's just a different way of thinking about a list of lists. A matrix has rows and columns. By specifying a row number and a column number, we're able to extract an element from a matrix.

We can create a NumPy array using the NumPy. Array function. If we pass in a list of lists, it will automatically create a NumPy array with the same number of rows and columns. Because we want all of the elements in the array to be float elements for easy computation, we'll leave off the header row, which contains strings. One of the limitations of NumPy is that all the elements in an array have to be of the same type, so if we include the header row, all the elements in the array will be read in as strings. Because we want to be able to do computations like find the average quality of the wines, we need the elements to all be floats.

NumPy has several advantages over using core Python mathematical functions, a few of which are outlined here:

1. NumPy is extremely fast when compared to core Python thanks to its heavy use of C extensions.
2. Many advanced Python libraries, such as Scikit-Learn, SciPy, and Kera, make extensive use of the NumPy library. Therefore, if you plan to pursue a career in data science or machine learning
3. NumPy is a very good tool to master. NumPy comes with a variety of built-in functionalities, which in core Python would take a fair bit of custom code.

Matplotlib

Plotting of data can be extensively made possible in an interactive way by Matplotlib, which is a plotting library that can be demonstrated in Python scripts. Plotting of graphs is a part of data visualization, and this property can be achieved by making use of Matplotlib. Matplotlib makes use of many general-purpose GUI toolkits, such as wxPython, Tkinter, QT, etc., in order to provide object-oriented APIs for embedding plots into applications. John D. Hunter was the person who originally wrote Matplotlib, and its lead developer was Michael Droettboom. One of the free and open-source Python libraries which is basically used for technical and scientific computing is Python SciPy. Matplotlib is widely used in SciPy as most scientific calculations require plotting of graphs and diagrams. Matplotlib is a plotting library like GNU plot. The main advantage towards GNU plot is the fact

that Matplotlib is a Python module. Due to the growing interest in python the popularity of matplotlib is continually rising as well.

Another reason for the attractiveness of Matplotlib lies in the fact that it is widely considered to be a perfect alternative to MATLAB, if it is used in combination with NumPy and SciPy. Whereas MATLAB is expensive and closed source, Matplotlib is free and open-source code. It is also object-oriented and can be used in an object- oriented way. Furthermore, it can be used with general-purpose GUI toolkits like wxPython, Qt, and GTK+.

There is also a procedural "pylab", which designed to closely resemble that of MATLAB. This can make it extremely easy for MATLAB users to migrate to matplotlib. Matplotlib can be used to create publication quality figures in a variety of hardcopy formats and interactive environments across platforms.

Another characteristic of matplotlib is its steep learning curve, which means that users usually make rapid progress after having started. The official website has to say the following about this: "matplotlib tries to make easy things easy and hard things possible. You can generate plots, histograms, power spectra, bar charts, error charts, scatterplots, etc., with just a few lines of code."

4.3.2 About IDE

IDE – Integrated Development Environment. It is software suite. It contains different instruments for web application software engineers to compose and test applications.

Programming Web application software engineers utilize wide scope of devices habitually all through application source code creation, building and application testing. Programming Development apparatuses frequently incorporate word processors for source code creation, code libraries for reuse, compilers and test stages.

An Integrated Development Environment (IDE) is a product application that gives a programming domain to streamline creating and troubleshooting

programming. Instead of playing out every one of the means required to make an executable program as disconnected individual assignments, it brings every one of the devices required into one application and workspace. Every one of the devices has a familiarity with the earth, and they cooperate to display a consistent improvement set for the designer. An IDE typically has an editor and a compiler window. There, we can write our code, run it, and compile it. Hence, it becomes much easier to develop applications using an IDE. The major reasons for using an IDE are given below:

- We can easily write our code in its text editor window.
- To store the resource files of a project, we have a project editor window.
- It makes debugging of the code easier.
- An IDE's autosuggest feature helps us in writing efficient code.
- We can import other projects for integrating with the existing one.

Without an IDE, a product web application software engineer must choose, send, incorporate and deal with these instruments independently. An IDE brings huge numbers of those improvement related instruments all together system, application or administration. The incorporated toolset is intended to streamline programming advancement and can distinguish and limit coding missteps and grammatical errors.

- Komodo
- NetBeans
- PyCharm
- Python for Visual Studio.NET
- PyDev
- PyStudio
- LiClipse (Eclipse)

LiClipse is at the core for all Python based application development activities. It's been available for many years. LiClipse is a set of plugins to enhance Eclipse. It improves the overall Eclipse experience. It is best IDE tool to develop Python Applications.

Advantages of LiClipse

- Perfect Debugger
- Error checking features
- An enhanced ability to analyze the code for quality and security concerns
- Auto complete feature

Based on the various IDE that are described above, we decided to use LiClipse as IDE (Integrated Development Environment) for developing this e-Voting web-based application Important Features That Make LiClipse our IDE of choice:

- Rich Reusable Libraries are available
- Model Driven Development
- Enterprise Python Tooling
- Powerful Debugging options

IDE is a program that is used to write applications (other programs) in a variety of programming languages. It allows you to complete many distinct program development steps in one place (program window).

A program which allows you to do this is known as an integrated development environment, or IDE. The program is free and easy to import onto your home computer. LiClipse is used for writing programs in Python. There are several steps that must be completed to set up LiClipse and further steps to start up a new Python project (set of Python components which work together to make up an application). PyCharm is an Integrated Development Environment (IDE) that provides tools for writing efficient code. It supports various frameworks such as Django, Flask, and Pyramid. PyCharm is majorly used by developers for application development with Python language.

Let's suppose, we are making a project using Python programming language. We know that Python is extensively popular among the developer's community, as well as in industries, due to its following features:

- Simple to code: Python is a very simple language compared to other programming languages. Also, it is a developer-friendly language.
- Object-oriented language: Python is an object-oriented language that uses concepts of classes and objects.
- GUI programming support: We can create graphical user interfaces using Python Tkinter library. Besides, we can use modules such as PyQt5,

PyQt4, and wxPython to create a GUI application.

- High-level language: Python is a high-level programming language. We do not need to take care of managing memory or the system architecture while writing Python codes.
- Portability: Python is a portable programming language, i.e., we can run Python codes on any platform. For example, we can run a code written for Windows in Linux, UNIX, and Mac, without changing it.
- Integrated language: Python is an integrated programming language. We can easily integrate C, C++, etc. with Python.
- Interpreted language: Python is an interpreted programming language as well. Python's source code is changed to bytecode. Due to this, it is directly executed without compiling.

Large set of standard libraries: Python consists of a rich set of modules, packages, and libraries. It helps in various stages of application development. "notebook" or "notebook documents" denote documents that contain both code and rich text elements, such as figures, links, equations, ... Because of the mix of code and text elements, these documents are the ideal place to bring together an analysis description, and its results, as well as, they can be executed perform the data analysis in real time. The Jupyter Notebook App produces these documents. As a server-client application, the Jupyter Notebook App allows you to edit and run your notebooks via a web browser. The application can be executed on a PC without Internet access, or it can be installed on a remote server, where you can access it through the Internet. Its two main components are the kernels and a dashboard. A kernel is a program that runs and introspects the user's code. The Jupyter Notebook App has a kernel for Python code, but there are also kernels available for other programming languages. The dashboard of the application not only shows you the notebook documents that you have made and can reopen but can also be used to manage the kernels: you can which ones are running and shut them down if necessary.

CHAPTER 5

SYSTEM DESIGN

5.1 Introduction

A UML diagram is a diagram based on the UML (Unified Modeling Language) with the purpose of visually representing a system along with its main actors, roles, actions, artifacts or classes, in order to better understand, alter, maintain, or document information about the system. Mainly, UML has been used as a general-purpose modeling language in the field of software engineering. However, it has now found its way into the documentation of several business processes or workflows. For example, activity diagrams, a type of UML diagram, can be used as a replacement for flowcharts. They provide both a more standardized way of modeling workflows as well as a wider range of features to improve readability and efficiency.

5.2 Architecture Diagram

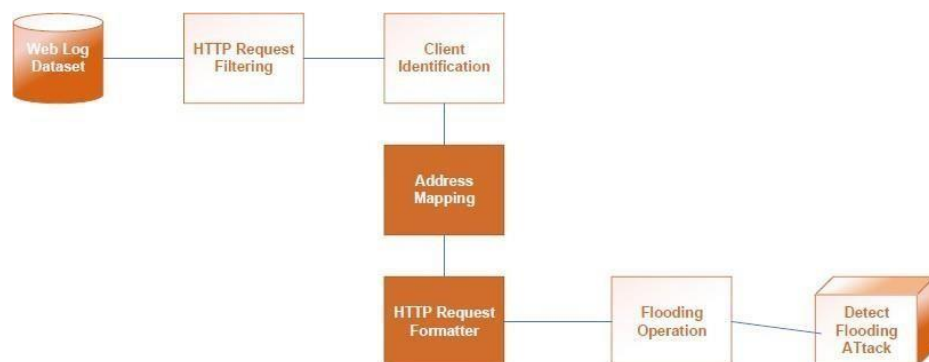


Fig 5.1 Architecture Diagram

An architecture diagram is a graphical representation of a set of concepts, that are part of an architecture, including their principles, elements and components. What is a diagram? What are the types of diagrams for architecture? The Dragon1 open EA Method makes it very clear: if a diagram does not show a concept, principle or part of a principle, it is NOT an architecture diagram, because it does not show

(a part of) the architecture. There are many kinds of architecture diagrams, like a software architecture diagram, system architecture diagram, application architecture diagram, security architecture diagram, etc. For system developers, they need system architecture diagrams to understand, clarify, and communicate ideas about the system structure and the user requirements that the system must support. It's a basic framework can be used at the system planning phase helping partners understand the architecture, discuss changes, and communicate intentions clearly.

CHAPTER 6

SYSTEM IMPLEMENTATION

6.1 List of Modules

Module 1: Data Evaluation

Module 2: Feature Extraction

Module 3: Feature Engineering

Module 4: Prediction.

6.1.1. Module 1: Exploratory Data Evaluation

Data Evaluation process seeks the unexpected in a data-driven manner, it is crucial that these tools are seamlessly integrated so analysts can flexibly select and compose tools to use at each stage of analysis. Few systems have integrated all these capabilities either architecturally or at the user interface level. Visage's information-centric approach allows coordination among multiple application user interfaces. It uses an architecture that keeps track of the mapping of visual objects to information in shared databases.

It will typically generate dozens, if not hundreds, of exploratory graphs in the course of analyzing a dataset. Of these graphs, you may end up publishing one or two in a final format. One purpose of Data Evaluation is to develop a personal understanding of the data, so all your code and graphs should be geared towards that purpose. Important details that you might add if you were to publish a graph² are not necessary in an exploratory graph.

Visual data analysis methods in use in a wide range can be traced back too many centuries ago, it is because that human eyes and brains possess strong structural ability to detect that occupy such important position in data exploring. And visual analysis is to play a variety of human models in the processing capacity of the special way to display data.

6.1.2. Module 2: Feature Extraction

Feature extraction and selection methods are used isolated or in combination with the aim to improve performance such as estimated accuracy, visualization and comprehensibility of learned knowledge. Generally, features can be categorized as: relevant, irrelevant, or redundant. In feature selection process a subset from available features data is selected for the process of learning algorithm. The best subset is the one with least number of dimensions that most contribute to learning accuracy. Information gain of a feature is calculation the difference of entropy whether it appears in the text. The larger information gain, the greater contribution the characteristics to the text. Characteristics with high information gain will be selected as feature.

Dimensionality reduction is wide spread preprocessing in high dimensional data analysis, visualization and modeling. One of the simplest ways to reduce dimensionality is by Feature Selection; one selects only those input dimensions that contain the relevant information for solving the particular problem. Filter methods have low computational cost and faster but with inefficient reliability in classification as compared to wrapper methods and better suitable for high dimensional data sets. Hybrid/embedded methods are recently developed which utilize advantages of both filters and wrappers approaches.

A hybrid approach uses both an independent test and performance evaluation function of the feature subset. Filter's methods can be further categorized into two groups, namely feature weighting algorithms and subset search algorithms. Feature weighting algorithms assign weights to features individually and rank them based on their relevance to the target concept.

6.1.3 Module 3: Feature Engineering

Filter methods are generally used as a preprocessing step. The selection of features is independent of any machine learning algorithms. Instead, features are selected on the basis of their scores in various statistical tests for their correlation with the outcome variable. The correlation is a subjective term here. For basic guidance, you can refer to the following table for defining correlation co-efficient.

- **Pearson's Correlation:** It is used as a measure for quantifying linear dependence between two continuous variables X and Y. Its value varies from -1 to +1.
- **LDA:** Linear discriminant analysis is used to find a linear combination of features that characterizes or separates two or more classes (or levels) of a categorical variable.
- **ANOVA:** ANOVA stands for Analysis of variance. It is similar to LDA except for the fact that it is operated using one or more categorical independent features and one continuous dependent feature. It provides a statistical test of whether the means of several groups are equal or not.
- **Chi-Square:** It is a statistical test applied to the groups of categorical features to evaluate the likelihood of correlation or association between them using their frequency distribution

6.1.4 Module 4: Prediction

Neural networks are multi-layer networks of neurons (the blue and magenta nodes in the chart below) that we use to classify things, make predictions, etc. More complex neural networks are just models with more hidden layers and that means more neurons and more connections between neurons. And this more complex web of connections (and weights and biases) is what allows the neural network to “learn” the complicated relationships hidden in our data.

In a neural network, changing the weight of any one connection (or the bias of a neuron) has a reverberating effect across all the other neurons and their activations in the subsequent layers. Once training is complete, it's time to see if the model is any good, using Evaluation. This is where that dataset that we set aside earlier comes into play. Evaluation allows us to test our model against data that has never been used for training. This metric allows us to see how the model might perform

against data that it has not yet seen. This is meant to be representative of how the model might perform in the real world.

A good rule of thumb I use for a training-evaluation split somewhere on the order of 80/20 or 70/30. Much of this depends on the size of the original source dataset. If you have a lot of data, perhaps you don't need as big of a fraction for the evaluation dataset.

Once you've done evaluation, it's possible that you want to see if you can further improve your training in any way. We can do this by tuning our parameters. There were a few parameters we implicitly assumed when we did our training, and now is a good time to go back and test those assumptions and try other values.

A tree has many analogies in real life, and turns out that it has influenced a wide area of machine learning, covering both classification and regression. In decision analysis, a decision tree can be used to visually and explicitly represent decisions and decision making. As the name goes, it uses a tree-like model of decisions. A decision tree is drawn upside down with its root at the top. In the image on the left, the bold text in black represents a condition/internal node, based on which the tree splits into branches/ edges. The end of the branch that doesn't split anymore is the decision/leaf, in this case, whether the passenger died or survived, represented as red and green text respectively.

Although, a real dataset will have a lot more features and this will just be a branch in a much bigger tree, but you can't ignore the simplicity of this algorithm. The feature importance is clear and relations can be viewed easily. This methodology is more commonly known as learning decision tree from data and above tree is called Classification tree as the target is to classify passenger as survived or died. Regression trees are represented in the same manner, just they predict continuous values like price of a house. In general, Decision Tree algorithms are referred to as CART or Classification and Regression Trees.

CHAPTER 7

CONCLUSION AND SUMMARY

7.1 Conclusion

Reduce the flood attacks by employing rate limitation and probabilistically detect the number of attackers by developing the claim-carry-check. The learning automata algorithm is used to detect the approximate counting number of packets which are violating the rate limits. This works is implemented in a distributed manner. Also, it can tolerate a little number of attackers to collude. They easily reduce the throughput of burst traffic by comparing with the simple threshold. Hence, the main aim to enhance the throughput of burst traffic under the data flooding attack. This is achieved by using proposed scheme and it also gives guarantee the QOS compared to old scheme.

REFERENCES

- Amandeep Verma Manpreet Singh Gujral “A Comprehensive Appraisal of Ad hoc Networks” International Journal of Computer Applications (0975–8887) Volume 49–No.22, July 2012.
- Afanasyev, & al., “Interest flooding attack and countermeasures in named data networking,” in IFIP Networking Conference. IEEE, 2013, pp.1–9.
- Compagno, & al., “Poseidon: Mitigating interest flooding ddos attacks in named datanetworking,” in Local Computer Networks (LCN), Intl’ Conf. on. IEEE, 2013, pp. 630–638.
- Ghali, & al., “Closing the floodgate with stateless content-centric networking,” in 2017 26th International Conference on Computer Communication and Networks (ICCCN), July 2017, pp.1–10.
- K. Wang, & al., “On the urgency of implementing Interest NACK into CCN: from the perspective of countering advanced interest flooding attacks” in IET Networks, vol. 7, no. 3, pp. 136–140, 2018.
- S. DiBenedetto and C. Papadopoulos, “Mitigating poisoned content with forwarding strategy,” in 2016 IEEE Conference on Computer Communications Workshops (INFOCOM WKSHPS), April 2016, pp. 164–166.
- S. Umarani, D. Sharmila, “Predicting Application Layer DDoS Attacks Using Machine Learning Algorithms”, International Journal of Computer, Electrical, Automation, Control and Information Engineering Vol.8, No.10, 2014, pp. 1912-1917.
- Shahanaz Begum I, Geetharamani G, “DDoS Attack detection and Prevention in Private Cloud Environment “, International Journal of Innovations in Engineering and Technology (IJJET), Vol.7 Issue.3, Oct 2016, pp. 527- 531.
- T. Zhi, H. Luo, and Y. Liu, “A gini impurity-based interest flooding attack defence mechanism in ndn,” IEEE Communications Letters, vol. 22, no. 3, pp. 538–541, March 2018.

- Y. Xin, & al., “Detection of collusive interest flooding attacks in named data networking using wavelet analysis,” in IEEE Military Communications Conference (MILCOM), Oct 2017, pp. 557–562.
- Yi, & al., “A case for stateful forwarding plane,” Computer Communications, vol. 36, no. 7, pp.779–791, 2013.