Department of Computer Science

UNIVERSITY of York

Submitted in part fulfilment for the degree of MSc

Topics in Privacy and Security Open Individual Assessment

Exam number: Y3864454

12 March 2019

Supervisor: Dr Radu Calinescu

TABLE OF CONTENTS

1 Online Reputation Systems

The popularity of online social reputation systems have significantly increased during the last years. This is due to the fact that they provide users with great opportunity to have deeper knowledge about a particular service, which is based on experience of other people. Such systems accelerate the decision-making process, enhance the purchasing power and contribute to overall trustworthiness of online retailers. As Q. Feng emphasizes "social rating systems not only increase influence on consumers' learning and decision-making behavior, but also offer incentives for good behaviour and positive impact on information quality" [1]. However, with the increase of social reputation systems more cyber attacks have arisen which could significantly influence online booking, buying tickets processes and lower consumers' confidence.

In this section we will firstly analyse the types of cyberattacks that may lead to fake hotel, restaurant and museum reviews ending up on a travel website. Secondly, we will look at two approaches which could significantly limit the fraction of fake reviews due to the cyberattacks (for registered and unregistered users). Lastly, we will compare the identified approaches, underlying their advantages, disadvantages and propose the most suitable approach for the company GENuine REviews (GENRE).

1.1 Cyberattacks analysis

Before defying the classes of attacks, we will determine the main characteristicks of the attacker which could damage holidaymakers ratings and reputation. First, attackers can be insider, i.e. they acts according to the system' regulations – authenticated users, and outsiders who are unauthorized users. Second, attackers can act alone or form coalitions of people whose main objective is to intendedly destroy the system. According to K. Hoffman, "coordinated attacks are more difficult to detect and defend against because attackers can exhibit multi-faceted behavior that allows them to partially hide within their malicious coalition" [2]. Thirdly, every attacker has their primary purpose which can be:

1. Selfishly promote a particular service (in the context of GENRE it could be hotel, museum, restaurant website and other types of travel-related venues) due to their own benefit, i.e. the increase of reputation, the elevated rating etc. Having higher reputation, sellers would have more users and, thus, financial gain.
2. Maliciously demote a particular service due to degrade the reputation of an item or "impact the availability of the reputation system itself" [2]. Similarly, eliminating potential "victims" would result in higher reputation and increased incoming.

From identified above features, characteristics, motives and goals of attackers, we can identify the types of cyber attacks in the context of GENRE:

1. **Direct attack** is a simplest way to reduce the rating of a particular service. This naïve attack is based on the following principle: dishonest users register in the system and each of them rate a particular item in the target set unfairly, i.e. providing false rating. The most well-known incident of this type of attack is "the bad-mouthing attack" [3] where the attackers conspire to provide negative feedback on the victim which would consequently lead to lowered or damaged reputation. Overall, this type of attack is easy to eliminate because "the number of malicious users is the same as the number of malicious ratings in the target set" [1]
2. **Self-promoting attack** which represents more complex type of attack (in comparison with a direct attack) when "attackers manipulate their own

reputation by falsely increasing it" [2]. This type of attack is related to the class of *disguise* cyber attacks where malicious users register with the system and behave as honest ones, providing fair feedback and marking items honestly. However, once they gain a necessary high trust scores from other participants, they begin to act dishonestly, providing deteriorating marks to targeted items. One of the most known way to undertake a self-promoting attack is through a Sybil attack [4] which requires dishonest users to participate in events which allow them to improve their reputation faster than honest participants and, thus, promote each other (colluding users) and gain higher reputation. In summary, this kind of attack happens in the "systems lacking mechanisms to provide data authentication and integrity… as they are not able to discern between fabricated and legitimate feedbacks" [2].

3. **Whitewashing attack** is the kind of a cyber attack when malicious users reset their bad reputation in order to rejoin the system with a new account and continue to damage a targeted item reputation. As Friedman and Resnick underline, "the attack is facilitated by the availability of cheap pseudonyms and the fact that reciprocity is much harder to maintain with easily changed identifiers" [5].
4. **Slandering attack** is very similar to self-promoting attack with the only difference in the feedback provided: in the case of a slandering attack malicious users give only ratings of 1 in order to decrease the reputation of the victim items. The system is vulnerable against slandering attacks when it has the lack of authentication. Like self-promoting attack, slandering attack is usually organized in a coalition of people whose goal is to provide wrong feedback on a particular item [6].
5. **Denial of Service.** This kind of attack destroy a particular service "by overloading its network or computational resources)" [2]. For example, a user wants to know more about a particular hotel/museum/restaurant. He/she goes to an item's website but as it was intendedly corrupted by hackers, the user cannot access the website and, thus, essential information. This could potentially lead the user left this website and search for another. Thus, hackers could significantly decrease the number of users who will benefit from a hacked website and, consequently, result in low reputation of the website because of the its unavailability.

All identified above threats could destroy tourist-venues websites, spoilt reputation, decrease rating of a particular organization which could result in loss of customers, decline in income and other significant consequences for GENRE.

1.2 Approach for addressing fake reviews (anonymous users)

Now when we have already defined different types of cyber attacks which could be used by malicious users in order to gain their own benefit from travel-venues online reputation system, we are able to propose an approach which combines several security controls to decrease the number of fake reviews from anonymous users. To defend items from attackers, GENRE should increase the cost of every attack, which consists of two parameters: number of dishonest users and the number of malicious ratings. According to Feng, "by requiring attackers to gain the trust of different individual users, it greatly increases the cost of attacks" [1]. All security controls described below are focused on the increase of the cost of every attack.

In order to limit the number of fake reviews posted by anonymous users, the system should use a different security control methods. One of them is to allow users rate a particular item only after buying a ticket to a museum, restaurant, trip or after booking a place in a hotel. This way of defense would require attackers to pay money in order to spoilt or improve the reputation of a particular service which is not suitable for them. This technique of security control is used by a popular e-Commerce Internet service "Amazon" [1]. However, the main disadvantage of this method is that sometimes users want to rate a particular hotel/service/museum after visiting it. This could lead to the GENRE website unpopularity due to the lack of ability to rate an item without buying a ticket or booking a place. To enhance this security control, GENRE can allow users to leave review on a particular item only once per registered cookie. This could significantly contribute to overall system's resistance against malicious attacks. At the same time, an advanced user can simply delete this cookie or use another computer to rate the same item again.

Another way to eliminate fake reviews is to introduce time windows which provides "a social recommendation score of an item based only on the ratings valid in the given time window" [1]. This technique is based on the principle of dividing the time into intervals, e.g. one week. For example, as it shown in Figure 1, the recommendations on a particular item are divided into three periods of time. First, GENRE should calculate the reputation of the service/item in each of the three time intervals. As negative rates outweigh positive ones, the total social recommendation is negative (six positive reviews versus seven negative). Second, GENRE should compute the results from different windows. As we can see from the Figure 1, first and second window are positive while third is negative (two positive versus one negative). Thus, final social recommendation of a particular item is positive.

According to Q. Feng, "incorporating time dimension not only helps us detect dishonest ratings more easily and consistently but also strengthens the attack resilience of the social rating systems" [1]. It means that the dishonest users have to take into consideration not only which services/items to attack, but also when and which time windows the attack should be implemented. However, if the quality of a particular service/item changes frequently which affects its reputation, this way of defense should be combined with other techniques "to balance present and historical time windows" [1].

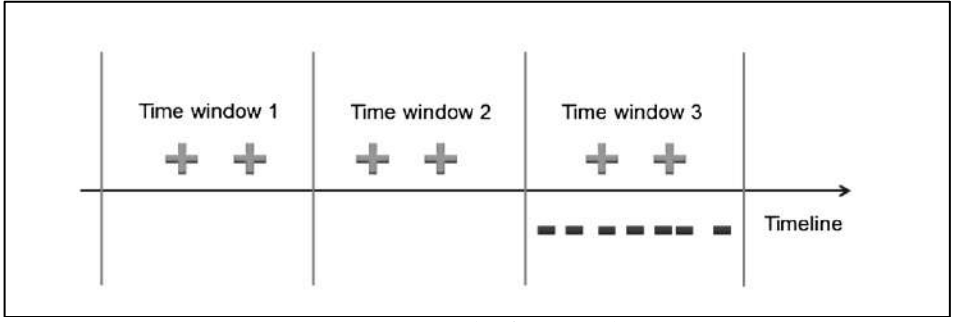


Figure 1 Example of time-window

To summarize, we proposed an approach that combines different countermeasures against anonymous fake reviews such as:

- rating a particular item only after buying a ticket;
- one vote per registered cookie for any given review;
- introduction of time windows.

All these security control techniques would significantly contribute to online reputation systems resistance against identified above cyber attacks.

1.3 Approach for addressing fake reviews (registered users)

In order to enhance the resistance of the system against fake reviews posted by registered users, GENRE should introduce the trust users system which provides all users with their own status (e.g., a scale of 1 to 10). When a user register with the system, he/she receives 5 points to his/her rating. To obtain a higher status, a particular user should be honest, i.e. "the trust of a user is evaluated based on whether her rating matches the overall social recommendation of the item, computed based on the majority of the users and their ratings" [1]. This way of security control would lead to malicious people would require more time and efforts to provide a dishonest feedback on a particular item which would consequently increase the cost of cyber attacks. What is more, other people would be able to distinguish honest users from malicious ones as they would have low ratings. Additionally, GENRE could automatically block users whose ratings lower than 2 which would subsequently result in the decrease of the dishonest users number. The main disadvantage of this security control method is that malicious users could artificially obtain a high rater status by incorporating ballot stuffing. As a consequence, they could significantly affect a particular item/service reinforcing unfair rating.

Described above approach of introducing trust status for every user can be strengthen by using a system of *Favorite People* [7]. In other words, all users can have a list of favorite reviewers "and the number of other members who has a specific reviewer listed as favourite person also influences that reviewer's rank" [7]. This approach would make the process of obtaining a high reputation more complicated for dishonest persons. This way of security control is also used by Amazon.

In order to prevent whitewashing attack where users can "clean" their ratings simply by creating a new account, GENRE could make the registration process a little bit sophisticated: as well as nicknames users have to provide their proofs of identity such as passport, driving license, birth certificate. Such approach is used by Airbnb online reputation system – community of hosts and travelers. This method also could prevent multiple identities – Sybil attack [2]. Additionally, using reCAPTCHA before registration could also be used against multiple identities [8]. However, this could result in the decrease of the number of registered users as not all people are ready to provide unknow website with their passport credentials and other important private information.

Another way to defend the system against fake-reviews is to introduce experts and professional reviewers. This approach was discussed by X. Amatrian [9] who showed that the use of external experts opinions could significantly decrease the number of unfair reviews. Q. Feng summarized his findings and the basic idea: "the systems can choose to use the ratings of experts as the authorized recommendation complimented by the majority based social recommendation scores". Thus, a dishonest review will not affect the overall rating of the item. The problem arises when experts or professional reviewers were bribed by people who want to gain a high reputation. As a result, unfair experts could significantly influence the rating of a particular item or service.

What is more, by allowing users to rate the review as being *helpful* or *not helpful* could "contribute to determining how prominently the review will be placed, as well as to giving the reviewer a higher status" [7]. Thus, each review would have its own "*weight*" (the number of "helpful" and the number of total votes) which would enable people to distinguish fake reviews from honest ones. The significant drawback of this technique is that malicious users could artificially increase the helpfulness of dishonest review and, thus, affect the reputation of honest raters who marked this review as helpful.

The main vulnerability of online reputation systems, which dishonest users exploit while cyber attacks described above, is the access to the knowledge about actual state of the systems. That is to say, knowing information about users, their trust ratings, reviews, number of honest and dishonest raters of a particular item, an attacker can find the weakest points in the system and "launch a sequence of detrimental attacks to items or users selectively at affordable cost" [1]. As a result, it could lead to the drop of the ratings of honest persons/items and increase of ratings of dishonest ones. To increase a cost of a cyber attack and to make it more difficult to create a structure of a powerful and influential attack, GENRE should introduce a defense by hiding information which is important for attackers but less valuable to honest users. In other words, GENRE should hide "the user-item graph" [1] which represents the relationships between users and items, i.e. which user rates a particular item. This strategy was proposed by Q. Feng, according to whom: "This information-hiding defense will, in turn, increase both the number of malicious users needed and the number of malicious ratings needed for a successful attack" [1]. What it means that the probability of attacks success will be reduced and the cost of attacks increased. However, this way of defense could affect the utility of the GENRE reputation website because people tend to trust more to the rating of people who are more like them, i.e. the same nationality, the same religion, the same country etc. This information is very important to know in the context of travel-venues online reputation system.

In conclusion, all identified before security controls would strengthen the online reputation system against identified above cyber attacks. Below you can find the list of the all proposed security controls:

- introduction of the user's rating system;
- introduction of the system of *Favorite People;*
- a complicated process of registration requiring proofs of identity;
- using reCAPTCHA before registration;
- introduction of experts and professional reviewers;
- introduction of review' statuses (*helpful* or *not helpful*);
- introduction of information-hiding defense.

1.4 Approach to adopt for GENRE

Now we can compare two described before approaches emphasizing their advantages and disadvantages. The system in which all users are anonymous has a number of advantages. Firstly, it allows users to rate a particular item or someone's review avoiding a registration process. As a result, it contributes to a positive user's impression because they do not have to spend too much time on filling the forms, generating a password, providing proofs of identity, confirming registration by email etc. Additionally, some users do not want to reveal their identities for personal security reasons or because they want to avoid biases based on nationality or race. Thus, anonymous reviews would allow them to rate a particular item without revealing their real names, i.e. anonymously.

On the other hand, a study conducted by C. Forman showed that "online community members rate reviews containing identity-descriptive information more positively" [10] which means that users trust "real" people more that to anonyms. It is very important in the context of travel-venues reputation system because such characteristics as nationality, country of origin, race are often considered by people while making decision about what places to visit. What is more, the probability to implement successful cyber attack in anonymous environment and to avoid all countermeasures is more likely due to the lack of security controls mechanisms in comparison with the registered users system.

In contrary, the system where users have to create an account in order to rate a particular item is more resistance against different types of cyber attacks. The main advantage of such systems is that all users are verified through providing the proofs of their identities which increases the cost of every attack. People can access the information about raters, their reviews, their trust ratings and, hence, the process of identifying fake reviews as well as their authors becomes easier. Additionally, in such systems moderators can track behavior of the members. A. Josang gives an example: “an entity that has behaved well for a long time but suddenly goes downhill can be quickly recognized as untrustworthy” [7]. That is to say, every dishonest person while loosing their reputation contributes to their quicker identification. Moreover, the combination of multiple security mechanisms identified in part 1.3 require dishonest persons make more efforts to implement cyber attacks.

To conclude, GENRE should implement the online reputation system which requires users to be registered because:

- People need to know the authors of reviews, their nationality, country etc. Information provided by people which are similar to them helps to choose a destination easier and quicker;
- Rating of every user will increase the trustworthiness of every review provided by a the person. It will assist people to distinguish fakes reviews from honest ones. Rating system is impossible to implement in anonymous environment.
- Users with high rating could provide reviews to emulate – experts opinions. It is also impossible to achieve in anonymous websites.
- Whitewashing attack would become difficult to implement because users have to provide their real names, proofs of identities which makes the process of the creation of multiple accounts almost impossible.
- Direct, self-promoting and slandering attacks would be easy to eliminate as every dishonest review decreases the rating of a user.

All countermeasures identified before would help GENRE to defend their online reputation system against fake reviews and make it more cyber-attacks resistant.

2 Anomaly Detection

2.1 Comparison of rule-based and anomaly detections

There are two different techniques for identifying cyber attacks in web servers log files: rule-based and anomaly detection. Rule-based method implies that static rules or patterns are defined before the analysis phase. These rules are usually focused on detection certain characters in a particular request or seek for specific words. Static rules remain the same during the detection phase.

This approach allows to identify different types of cyber attacks. One of them is *Cross Site Scripting* [11] cyber attack where attackers embed script tags in requests and, thus, force user to execute malicious javascript on their machines. This allows attackers have access to private user's information such as card details, passwords and other important data. This cyber attack is in third place in the ranking of key risks for Web applications according to OWASP [12]. Another cyber attack which can be identified through rule-based detection is *SQL injections* [13]. Here attackers inject SQL scripts in requests and, thus, modifies an origin SQL request which allows them to execute malicious SQL scripts and extract secure information from databases. Moreover, rule-based detection could prevent web-servers from *insecure object references* [14] attacks where attackers substitute origin objects' references by unauthorized objects. The effective way to identify beforementioned types of cyber attack is to use regular expressions which should be defined in advance, carefully validate users' inputs and use strings with fixed lengths.

The main advantage of the rule-based detection is that they could prevent numerous predetermined, specified and well-known cyber attacks which have been identified before. However, validation files could not contain rules for all types of cyber attacks or developers could simply not take into account a particular cyber attack and miss the rule against it. In this scenario, cyber attacks would pass unnoticed which could lead to hazardous consequences. What is more, every year a new cyber attack emerges which might not be defined in rules. This was emphasized by P. Garcı´a-Teodoro, "they are not capable of detecting new, unfamiliar intrusions, even if they are built as minimum variants of already known attacks" [15].

In contrast, anomaly rules are not static, they are dynamic and usually defined in a learning phase. In our case, a learning phase is an analysis of a "training_log" file containing "good" traffic, i.e. clean and free of attacks entries. Once analysis of a good traffic is performed, particular behavior is being identified which can be consequently used in detection malicious cyber attacks in other web services' log files. Anomaly-based detection assists in identifying previously unseen intrusion events. In other words, anomaly based protection is not focused on particular cyber attacks, they just "generate an anomaly alarm whenever the deviation between a given observation at an instant and the normal behaviour exceeds a predefined threshold" [15]. This is one of the significant advantages of anomaly-based detection: independently of the type of cyber attack it would be always identified and appropriate countermeasures would be undertaken. However, on the other hand, this approach cannot point out which type of cyber attack has been undertaken in comparison with rule-based detection. What is more, attackers could learn how to generate the malicious network traffic which would be considered as "normal". Additionally, if the data is not consistent and there are a great number of outlier values, the results of anomaly detection might be inaccurate. For this reason, outlier coefficients should be chosen carefully and precisely.

2.2 Description of the tool's principal of work

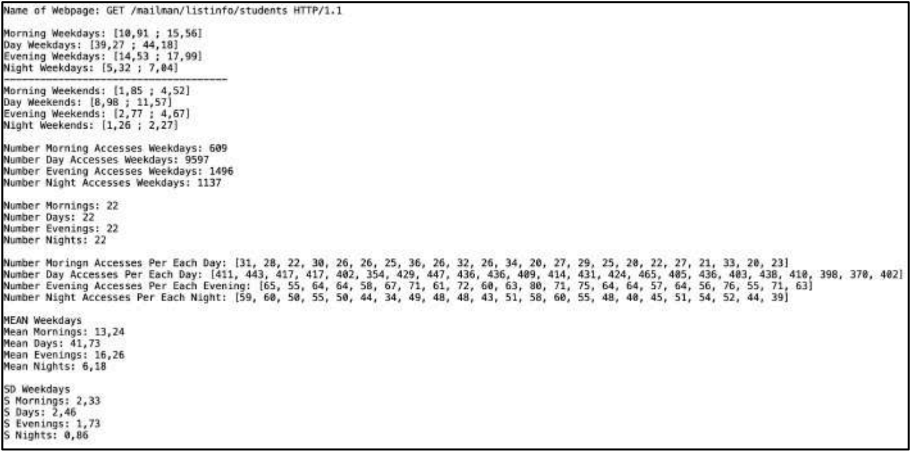
The software consists of the three Java classes:

1. **Interval.java**. This class stores information about mean (M), standard deviation (S), minimum and maximum number of requests for a particular webpage per each day (weekdays – morning, day, evening, night and weekends - morning, day, evening, night). It also perform calculations of beforementioned parameters.
2. **WebPage.java.** This class represent a particular webpage, stores the name of the webpage, the information about user's accesses distributed by time of day/type of day combinations and the results of statistical analyses, i.e. allowed intervals (Interval.java) - minimum and maximum number of requests.
3. **AnomalyDetector.java.** This class is the entry point to the application which processes the arguments and performs analysis of the files. In the constructor, it firstly validates the correctness of the arguments, then it extracts names of all webpages from training_log.txt (extractWebpages method) and collects statistical data (collectStatistics() method): the number of user's accesses for each webpage per each day (weekdays – morning, day, evening, night and weekends - morning, day, evening, night). After webpages were stored in class attributes (webpages), the tool extracts names of potentially attacked webpages from text_log.txt and stores them in attackedWebpages class's attribute. Then, the tool identifies "good traffic" based on the entries and data extracted from training_log.txt and prints the results of the statistical analysis: e.g., mean, standard deviation, number of accesses for each webpage etc. The last step is to calculate the number of accesses for attacked webpages per each hour and find out whether this number is in between the allowed interval (analyse() method). AnomalyDetector class has a number of helper methods which are used while performing statistical calculation and anomaly detection analysis. The most important are:
   - *isTimeInBetween*(String startPeriod, String endPeriod, String timeOfAccess) returns true if the time of webpage's access was between startPeriod and endPeriod;
   - *containsName*(final List<WebPage> list, final String name) returns true if the List<WebPage> contains a WebPage with the name name;
   - *isWeekend*(String date) returns true if the date is weekend;
   - *getTime*(String line) returns time in format hh:mm:ss;
   - *getDate*(String line) returns date in format DD/Mmm/YYYY.

The execution time of the program is 3000 ms which is fine in this case: the tool performs analysis of two files, which sizes are 5 634 690 bytes (training_log.txt) and 1 161 457 bytes (test_log.txt) respectively, and performs essential statistical calculations. Overall, the program shows good performance. The time complexity of this tool is quadric - O(n^2). This is because we some methods which have loop in loop: the running time of the two loops is proportional to the square of input parameters - N. When N doubles, the running time increases by N * N. The space complexity of the tool is also quadric - O(n^2).

2.3 Results of an experiment

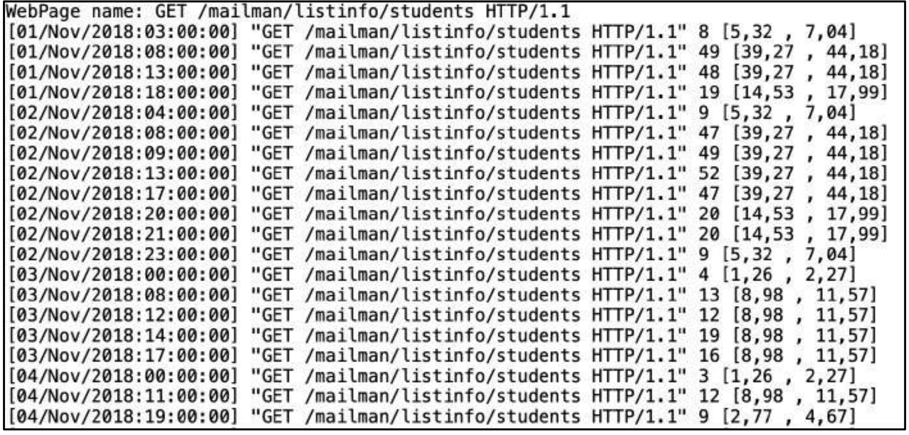
Figure 2 shows the collected statistics from training_log.txt file for webpage "`GET /mailman/listinfo/students HTTP/1.1`". In the very top we can observe allowed intervals (number of minimum and maximum requests) for eight time of day/type of day combinations. Then, overall number of accesses per each time of day is calculated (morning, day, evening and night). Next, it is shown the overall number of days when requests were made followed by number of morning/day/evening/night requests per each day. Lastly, the statistical data is displayed for each of these eight time of day/type of day combinations: mean and standard deviations. Figure 2 shows statistical information only for weekdays while the tool displays statistical information for both weekdays and weekends for each page.



```
Name of Webpage: GET /mailman/listinfo/students HTTP/1.1

Morning Weekdays: [10,91 ; 15,56]
Day Weekdays: [39,27 ; 44,18]
Evening Weekdays: [14,53 ; 17,99]
Night Weekdays: [5,32 ; 7,04]
----------------------------------------
Morning Weekends: [1,85 ; 4,52]
Day Weekends: [8,98 ; 11,57]
Evening Weekends: [2,77 ; 4,67]
Night Weekends: [1,26 ; 2,27]

Number Morning Accesses Weekdays: 609
Number Day Accesses Weekdays: 9597
Number Evening Accesses Weekdays: 1496
Number Night Accesses Weekdays: 1137

Number Mornings: 22
Number Days: 22
Number Evenings: 22
Number Nights: 22

Number Moringn Accesses Per Each Day: [31, 28, 22, 30, 26, 26, 25, 36, 26, 32, 26, 34, 20, 27, 29, 25, 20, 22, 27, 21, 33, 20, 23]
Number Day Accesses Per Each Day: [411, 443, 417, 417, 402, 354, 429, 447, 436, 436, 409, 414, 431, 424, 465, 405, 436, 403, 438, 410, 398, 370, 402]
Number Evening Accesses Per Each Evening: [65, 55, 64, 64, 58, 67, 71, 61, 72, 60, 63, 80, 71, 75, 64, 64, 57, 64, 56, 76, 55, 71, 63]
Number Night Accesses Per Each Night: [59, 60, 50, 55, 50, 44, 34, 49, 48, 48, 43, 51, 58, 60, 55, 48, 40, 45, 51, 54, 52, 44, 39]

MEAN Weekdays
Mean Mornings: 13,24
Mean Days: 41,73
Mean Evenings: 16,26
Mean Nights: 6,18

SD Weekdays
S Mornings: 2,33
S Days: 2,46
S Evenings: 1,73
S Nights: 0,86
```

Figure 2 Collected statistics from training_log.txt

Having this statistical data, the tool learns normal pattern of access for all webpages of the webserver and calculates allowed number of requests. After "good" traffic was identified, the tool analyses data in test_log.txt file and prints the webpages which have been attacked. Part of the results is shown in Figure 3.

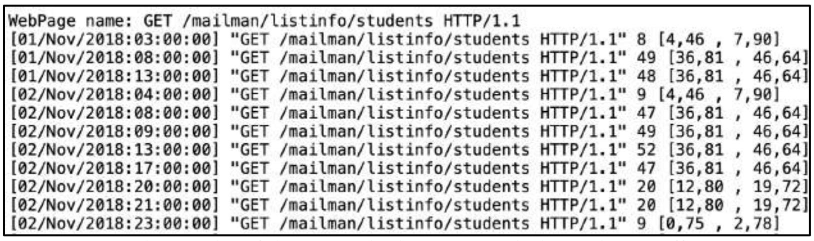


```
WebPage name: GET /mailman/listinfo/students HTTP/1.1
[01/Nov/2018:03:00:00] "GET /mailman/listinfo/students HTTP/1.1" 8 [5,32 , 7,04]
[01/Nov/2018:08:00:00] "GET /mailman/listinfo/students HTTP/1.1" 49 [39,27 , 44,18]
[01/Nov/2018:13:00:00] "GET /mailman/listinfo/students HTTP/1.1" 48 [39,27 , 44,18]
[01/Nov/2018:18:00:00] "GET /mailman/listinfo/students HTTP/1.1" 19 [14,53 , 17,99]
[02/Nov/2018:04:00:00] "GET /mailman/listinfo/students HTTP/1.1" 9 [5,32 , 7,04]
[02/Nov/2018:08:00:00] "GET /mailman/listinfo/students HTTP/1.1" 47 [39,27 , 44,18]
[02/Nov/2018:09:00:00] "GET /mailman/listinfo/students HTTP/1.1" 49 [39,27 , 44,18]
[02/Nov/2018:13:00:00] "GET /mailman/listinfo/students HTTP/1.1" 52 [39,27 , 44,18]
[02/Nov/2018:17:00:00] "GET /mailman/listinfo/students HTTP/1.1" 47 [39,27 , 44,18]
[02/Nov/2018:20:00:00] "GET /mailman/listinfo/students HTTP/1.1" 20 [14,53 , 17,99]
[02/Nov/2018:21:00:00] "GET /mailman/listinfo/students HTTP/1.1" 20 [14,53 , 17,99]
[02/Nov/2018:23:00:00] "GET /mailman/listinfo/students HTTP/1.1" 9 [5,32 , 7,04]
[03/Nov/2018:00:00:00] "GET /mailman/listinfo/students HTTP/1.1" 4 [1,26 , 2,27]
[03/Nov/2018:08:00:00] "GET /mailman/listinfo/students HTTP/1.1" 13 [8,98 , 11,57]
[03/Nov/2018:12:00:00] "GET /mailman/listinfo/students HTTP/1.1" 12 [8,98 , 11,57]
[03/Nov/2018:14:00:00] "GET /mailman/listinfo/students HTTP/1.1" 19 [8,98 , 11,57]
[03/Nov/2018:17:00:00] "GET /mailman/listinfo/students HTTP/1.1" 16 [8,98 , 11,57]
[04/Nov/2018:00:00:00] "GET /mailman/listinfo/students HTTP/1.1" 3 [1,26 , 2,27]
[04/Nov/2018:11:00:00] "GET /mailman/listinfo/students HTTP/1.1" 12 [8,98 , 11,57]
[04/Nov/2018:19:00:00] "GET /mailman/listinfo/students HTTP/1.1" 9 [2,77 , 4,67]
```

Figure 3 The results of the anomaly detection with the outlier coefficient = 1

For example, Figure 3 reports that the first entry - webpage "`GET /mailman/listinfo/students HTTP/1.1`" was accessed `8` times between 03:00:00 am and 03:59:59 am on the 1nd of November 2018, instead of a number of accesses in the expected [$M - \alpha s$, $M + \alpha s$] interval, which is `[5.32,7.04]` in this case.

However, if we increase the outlier coefficient by 1 (the outlier coefficient = 2), the interval expands, i.e. more requests become allowed in a particular period of time, i.e. the interval is less strict. It is demonstrated for the same webpage for the same period of time (first line) in Figure 4. As M. Rahman underlines, "an outlier coefficient is an observation in a dataset that is unusually large or small compared to the rest of the data" [16]. It means that an outlier coefficient helps us to achieve more accurate results in a particular case. In other words, we should choose outlier coefficient depending on the number of unusually large or small values which could significantly influence the results of the analysis or even spoilt them. For example, if the difference between unusually large/small values is too big compared to the rest of the data, we should choose a higher outlier coefficient in order to obtain more precise results.



```
WebPage name: GET /mailman/listinfo/students HTTP/1.1
[01/Nov/2018:03:00:00] "GET /mailman/listinfo/students HTTP/1.1" 8 [4,46 , 7,90]
[01/Nov/2018:08:00:00] "GET /mailman/listinfo/students HTTP/1.1" 49 [36,81 , 46,64]
[01/Nov/2018:13:00:00] "GET /mailman/listinfo/students HTTP/1.1" 48 [36,81 , 46,64]
[02/Nov/2018:04:00:00] "GET /mailman/listinfo/students HTTP/1.1" 9 [4,46 , 7,90]
[02/Nov/2018:08:00:00] "GET /mailman/listinfo/students HTTP/1.1" 47 [36,81 , 46,64]
[02/Nov/2018:09:00:00] "GET /mailman/listinfo/students HTTP/1.1" 49 [36,81 , 46,64]
[02/Nov/2018:13:00:00] "GET /mailman/listinfo/students HTTP/1.1" 52 [36,81 , 46,64]
[02/Nov/2018:17:00:00] "GET /mailman/listinfo/students HTTP/1.1" 47 [36,81 , 46,64]
[02/Nov/2018:20:00:00] "GET /mailman/listinfo/students HTTP/1.1" 20 [12,80 , 19,72]
[02/Nov/2018:21:00:00] "GET /mailman/listinfo/students HTTP/1.1" 20 [12,80 , 19,72]
[02/Nov/2018:23:00:00] "GET /mailman/listinfo/students HTTP/1.1" 9 [0,75 , 2,78]
```

Figure 4 The results of the anomaly detection with the outlier coefficient = 2

Hence, if the outlier coefficient equals 5, the number of attacked entries in our case significantly reduce because the interval (the minimum and the maximum number of requests) expands. With outlier coefficient equals 3, 4, 5 for particular entries we have a negative minimum number of requests in the interval. It means that the alpha number is too big for a particular case. In this case we just would change the minimum of allowed requests to zero.

Overall, as the tool has demonstrated, all webpages were attacked with slight differences in a number of attacks. However, as the training_log.txt file doesn't have statistical information for the webpage "`GET /mailman/listinfo/dean HTTP/1.1`" , we cannot make a conclusion whether this website was attacked or not. In addition, with the increase of the outlier coefficient, the interval becomes less strict which results in more requests become allowed.

2.4 When the tool should be used

This tool might be suitable for websites with high users' activity/density (attendance is high and consistent from day to day, i.e. the number of requests from day to day is approximately the same) and where it is important to know when the number of accesses for a particular webpages has significantly exceed the regular number of daily accesses. As an example of such webpages, online reputation systems (e.g, online shops, travel-venue websites and others) may be cited. This tool would help to elicit suspicious activity of a particular item in the case of slandering, self-promoting, direct attacks and many others. These attacks implies that a malicious user accesses a particular webpage a lot of times in order to increase/decrease its reputation which result in an enormously increased number of requests in a given period of time. Every single access of a particular webpage is recorded in a web-server log file which our tool uses to detect anomaly. By comparing "normal" activity with suspicious activity on a particular webpage, this tool would assist online reputations system developers to identify which web pages have been attacked and which items could potentially have fake ratings.

However, this tool could show bad results and be not very useful in such web sites where number of requests vary significantly from day to day. In order to obtain good results with this tool, the number of requests should be consistent. For example, news and informational portal websites will not be suitable for the tool because the number of users depend on the type of news: if the news is a sensation, then the number of requests will be high and visa versa. The issue is that we cannot predict which news will be tomorrow and, thus, can't predict the number of users who would attend this news website. The number of requests on such informational portals significantly varies from day to day and there is not any attendance pattern which could be defined as "good traffic".

In conclusion, the tool should be used in such websites where a number of accesses is roughly equal from day to day and should not be used where the number of requests is unpredictable and could significantly different every day.

Bibliography

[1] Q. Feng, L. Liu, and Y. Dai, “Vulnerabilities and countermeasures in context-aware social rating services,” *ACM Trans. Internet Technol.*, vol. 11, no. 3, pp. 1–27, Jan. 2012.

[2] K. Hoffman, D. Zage, and C. Nita-Rotaru, “A survey of attack and defense techniques for reputation systems,” *ACM Comput. Surv.*, vol. 42, no. 1, pp. 1–31, 2009.

[3] J. Moskaliuk, J. Kimmerle, and U. Cress, “Handbook of Research on Web 2.0, 3.0 and X.0: Technologies, Business, and Social Applications.,” 2009, pp. 573–595.

[4] R. J. Douceur, “The Sybil attack. In Proceedings of the 1st International Workshop on Peer-to-Peer Systems (IPTPS),” *Springer*, 2002.

[5] E. Friedman and P. Resnick, “The Social Cost of Cheap Pseudonyms,” *J. Econ. Manag. Strateg.*, vol. 10(1), pp. 173–199, 2001.

[6] J. Feng, Y. Zhang, S. Chen, and A. Fu, *RepHi: A Novel Attack against P2P Reputation Systems*. .

[7] A. Jøsang, R. Ismail, and C. Boyd, “COVER SHEET A Survey of Trust and Reputation Systems for Online Service Provision,” 2007.

[8] L. Von Ahn, B. Maurer, C. Mcmillen, D. Abraham, and M. Blum, “reCAPTCHA: Human-Based Character Recognition via Web Security Measures.”

[9] X. Amatriain, N. Lathia, J. M. Pujol, H. Kwak, and N. Oliver, “The Wisdom of the Few A Collaborative Filtering Approach Based on Expert Opinions from the Web.”

[10] C. Forman, A. Ghose, and B. Wiesenfeld, “Examining the Relationship Between Reviews and Sales: The Role of Reviewer Identity Disclosure in Electronic Markets,” *Inf. Syst. Res.*, vol. 19, no. 3, pp. 291–313, 2008.

[11] M. Ter Louw and V. N. Venkatakrishnan, “Scripting attacks for existing browsers,” *Proc. - IEEE Symp. Secur. Priv.*, pp. 331–346, 2009.

[12] “OWASP.” [Online]. Available: https://www.owasp.org/index.php/Main_Page. [Accessed: 27-Feb-2019].

[13] S. W. Boyd and A. D. Keromytis, “SQLrand: Preventing SQL Injection Attacks,” pp. 292–302, 2010.

[14] R. Meyer, “SANS Institute Information Security Reading Room Detecting Attacks on Web Applications from Log Files,” 2019.

[15] P. García-Teodoro, J. Díaz-Verdejo, G. Maciá-Fernández, and E. Vázquez, “Anomaly-based network intrusion detection: Techniques, systems and challenges,” *Comput. Secur.*, vol. 28, no. 1–2, pp. 18–28, 2009.

[16] M. S. Rahman and K. Al-Amri, “Effect of Outlier on Coefficient of Determination Age distribution model for PNG population View project.”