Declaration

Hereby I declare that this paper is my original authorial work, which I have worked out on my own. All sources, references, and literature used or excerpted during elaboration of this work are properly cited and listed in complete reference to the due source.

Bc. Andrej Staruch

Advisor: RNDr. Marek Kumpošt, Ph.D.

Abstract

The goal of this master's thesis is to design and implement a system, which will compute potential risk for a given URL. The computation of potential risk is based on extendable series of individual test suites, and the result of weighted tests is a number called 'phishing score'. Based on this number, the application can automatically allow or block the given communication. Optionally, the end user could be warned and decide if he wants to proceed to this website.

Keywords

phishing detection, Phishtank, security, cpp, javascript, URL tests, Trusted Network Solutions

Contents

1 Introduction **1**

2 Phishing **3**
2.1 *Attack strategies* 3

3 Detection methods **5**
3.1 *Phishing detection methods* 5
3.1.1 List based approach 5
3.1.2 URL analysis 5
3.1.3 Analysis of a content 5
3.1.4 Visual analysis 5
3.1.5 Source-code analysis 5
3.1.6 Behavioral analysis 5

4 Implementation **7**
4.1 *System design* 7
4.2 *Modules for detection* 7
4.2.1 Google safe browsing 7
4.2.2 Phishtank 7
4.2.3 URL anomalies detection 8
4.2.4 Behaviorial 8

5 Testing **9**
5.1 *Deployment* 9
5.2 *Integration into proxy* 9
5.3 *Testing on real traffic* 9
5.4 *Results* 9

6 Summary **11**

Bibliography **13**

A Documentation **15**
A.1 *Phishtank record* 15
A.2 *Phishurl fetcher* 15

1 Introduction

In today's world, the threat of a cyber attack can't be ignored. There are a plethora of companies that try to protect their customers from potential damage, such as getting infected by a virus, getting ransomware or malware or protect their business intelligence.

This thesis is concerned with another part of the cyber crime called phishing. Phishing is a social engineering attack to obtain sensitive information such as user names, passwords, credit card details, bank account credentials for malicious reasons /* cite here */. Because of a large attack vector, there isn't a reliable way or a tool to prevent such an attack consistently in every sector.

This thesis will review several ways to detect a phishing attack, implement some of them and test them in the production environment with an association with a Trusted Network Solutions company. The final software could be easily used with another proxy or tool.

First of all, we'll look around current market for existing solutions for phishing protection and compare them. After that, we will look for academic papers with phishing oriented thematic and review different approaches and choose suitable ones. Following these steps, we will design and implement phishing detection system. After that, we will test it on real customer traffic and conclude to result whether security of users has increased or not.

Designing an anti-phishing system has similar character as designing an anti-virus system - we are trying to protect user before malicious attempts of an attacker but as we can see, users are still infected when they are using anti-virus, so designing correct and powerful anti-phishing system is an uneasy task. We will focus on extendability for future editions.

2 Phishing

There isn't a precise scientific definition of what phishing is. Various literature is taking a different approach to explain this term:

> "Phishing is a trap where any targeted individual, is communicated by someone impersonating as a legitimate and a reputed organization to entice the individual into providing sensitive information such as banking information, credit card details, and passwords." [1]
>
> "PHISHING is a social engineering attack that aims at exploiting the weakness found in system processes as caused by system users." [2]
>
> "The technique used to perform on-line robbery/stealing of person credentials is called phishing in cyber international." [3]

In this thesis, we will define phishing as a fraudulent attempt combining social engineering skills with a technical trickery to obtain sensitive data from other parties.

2.1 Attack strategies

Phishing attacks are usually divided into several categories:

1. Deceptive phishing - this is the most common attack where an attacker is trying to steal money from the victim, usually done by sending a fake email from a bank with a fake URL link, where account details are exposed to the attacker
2. Spear phishing - this attack involves more social engineering skills and is targeted on single units instead of a wide group like in deceptive phishing
3. Whaling - similar to spear phishing, but attackers take a considerable time to prepare the attack and usually targets executive officers because they have more privileges and knowledge than a common employer

3 Detection methods

3.1 Phishing detection methods

3.1.1 List based approach

3.1.2 URL analysis

3.1.3 Analysis of a content

3.1.4 Visual analysis

3.1.5 Source-code analysis

3.1.6 Behavioral analysis

4 Implementation

4.1 System design

4.2 Modules for detection

4.2.1 Google safe browsing

4.2.2 Phishtank

Phishtank [4] is an online collaborative tool to track and share phishing data. Anyone can submit fraudulent URLs, users then verify if a given website is a phishing site or not. Phishtank provides a public API where a client can perform a lookup for one URL through the POST request [1], or he can request a current snapshot of the database [2].

Our system will do lots of lookups so we will use the latter option and build our database on top of these data. The snapshot has following structure:

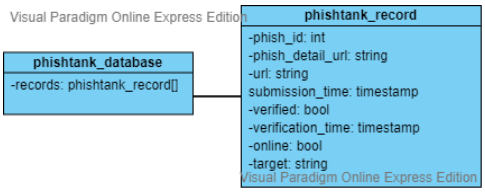

Figure 4.1: Phishtank scheme

Columns are documented in the appendix A.1.

Phishtank only provides the current state of its database, but we are also interested in the information of how long was website accessible. We will build our database alongside with this information in the following matter:

1. Request a current state from Phishtank service
2. If in our database are records with column `online=true` while they are not present in requested records, set them to `online=false` and create a new column `end_time` with a current timestamp.

1. https://www.phishtank.com/api_info.php
2. https://www.phishtank.com/developer_info.php

3. Insert all records, which `phish_id` isn't present in our database, into the database

We had implemented this into JavaScript command-line utility, which will be periodically executed to update our database table `Phishtank`. Program with instructions is attached in the sources (A.2).

4.2.3 URL anomalies detection

4.2.4 Behaviorial

5 Testing

5.1 Deployment

5.2 Integration into proxy

5.3 Testing on real traffic

5.4 Results

6 Summary

Bibliography

1. ARAVINDHAN, R.; SHANMUGALAKSHMI, R.; RAMYA, K.; C., Selvan. Certain investigation on web application security: Phishing detection and phishing target discovery. In: *2016 3rd International Conference on Advanced Computing and Communication Systems (ICACCS)*. 2016, vol. 01, pp. 1–10. Available from DOI: 10.1109/ICACCS.2016.7586405.
2. KHONJI, M.; IRAQI, Y.; JONES, A. Phishing Detection: A Literature Survey. *IEEE Communications Surveys Tutorials*. 2013, vol. 15, no. 4, pp. 2091–2121. ISSN 1553-877X. Available from DOI: 10.1109/SURV.2013.032213.00009.
3. CHURI, T.; SAWARDEKAR, P.; PARDESHI, A.; VARTAK, P. A secured methodology for anti-phishing. In: *2017 International Conference on Innovations in Information, Embedded and Communication Systems (ICIIECS)*. 2017, pp. 1–4. Available from DOI: 10.1109/ICIIECS.2017.8276081.
4. *PhishTank* [online] [visited on 2019-05-18]. Available from: https://www.phishtank.com/.

A Documentation

A.1 Phishtank record

phish_id	The ID number by which Phishtank refers to a phish submission. All data in PhishTank is tied to this ID. This will always be a positive integer.
phish_detail_url	PhishTank detail url for the phish, where you can view data about the phish, including a screenshot and the community votes.
url	The phish URL. This is always a string, and in the XML feeds may be a CDATA block.
submission_time	The date and time at which this phish was reported to Phishtank. This is an ISO 8601 formatted date.
verified	Whether or not this phish has been verified by our community. In these data files, this will always be the string 'yes' since we only supply verified phishes in these files.
verification_time	The date and time at which the phish was verified as valid by our community. This is an ISO 8601 formatted date.
online	Whether or not the phish is online and operational. In these data files, this will always be the string 'yes' since we only supply online phishes in these files.
target	The name of the company or brand the phish is impersonating, if it's known.

A.2 Phishurl fetcher

```
phishurl-fetcher
├── README.md
├── docker-compose.yaml
├── ormconfig.json
└── package-lock.json
```

```
├── package.json
├── src
│   ├── config
│   │   └── index.js
│   ├── database
│   │   ├── entity
│   │   │   ├── LastUpdated.js
│   │   │   └── Phishtank.js
│   │   ├── index.js
│   │   ├── migration .5 1555874663933-CreatePhishtank.js
│   │   │   └── 1555897004589-CreateLastUpdated.js
│   │   └── model
│   │       ├── LastUpdated.js
│   │       └── Phishtank.js
├── index.js
├── operations
│   ├── last-updated.js
│   └── phishtank.js
└── utils
    ├── logger.js
    └── utils.js
```