Author: G.Moresi	Date: 12.7.2023

Reverse Proxy Mitigation

Introduction

Phishing attacks frequently focus on senior management, account managers, human resources, and finance personnel—precisely the individuals with privileged access to extremely sensitive information. When such attacks are successful, malicious actors gain permanent entry to emails, files, contacts, notes, Microsoft Teams chats, and other valuable data. In certain instances, they may even redirect users to a phishing site subsequent to obtaining their consent to utilize the application.

During this kind of attacks (called AiTM), a phished user interacts with an impersonated site created by the attacker**. This allows the attacker to intercept credentials and session cookies and bypass multifactor authentication (MFA), which can then be used to initiate other attacks.**

Attack in the Middle (AiTM)

Attack in the Middle (AiTM) is a sophisticated cyber-attack where an attacker intercepts and alters communications between two parties without their knowledge. It typically occurs when a user's device or network connection is compromised, allowing the attacker to eavesdrop, manipulate, or inject malicious content into the communication flow. AiTM attacks employ various techniques to intercept and manipulate communications. Some common methods include:

Multi-Factor Authentication (MFA) is a vital security measure used to protect sensitive information and systems by adding an extra layer of verification. However, cybercriminals are continuously evolving their techniques to bypass MFA. In this chapter, we will explore the concept of MFA bypass using a reverse proxy, a method employed by attackers to circumvent MFA protections.

Understanding MFA

MFA is crucial in mitigating the risk of unauthorized access to sensitive information. By requiring additional factors beyond a simple password, MFA adds an extra layer of protection against stolen credentials or brute-force attacks.

Reverse Proxy and MFA Bypass

A reverse proxy is a server or service that sits between client devices and backend servers, acting as an intermediary for requests. It receives client requests and forwards them to the appropriate backend server, allowing for load balancing, caching, or security purposes.

Attackers can exploit the functionality of a reverse proxy to bypass MFA. Here's how it works:

Step 1: Initial Authentication.

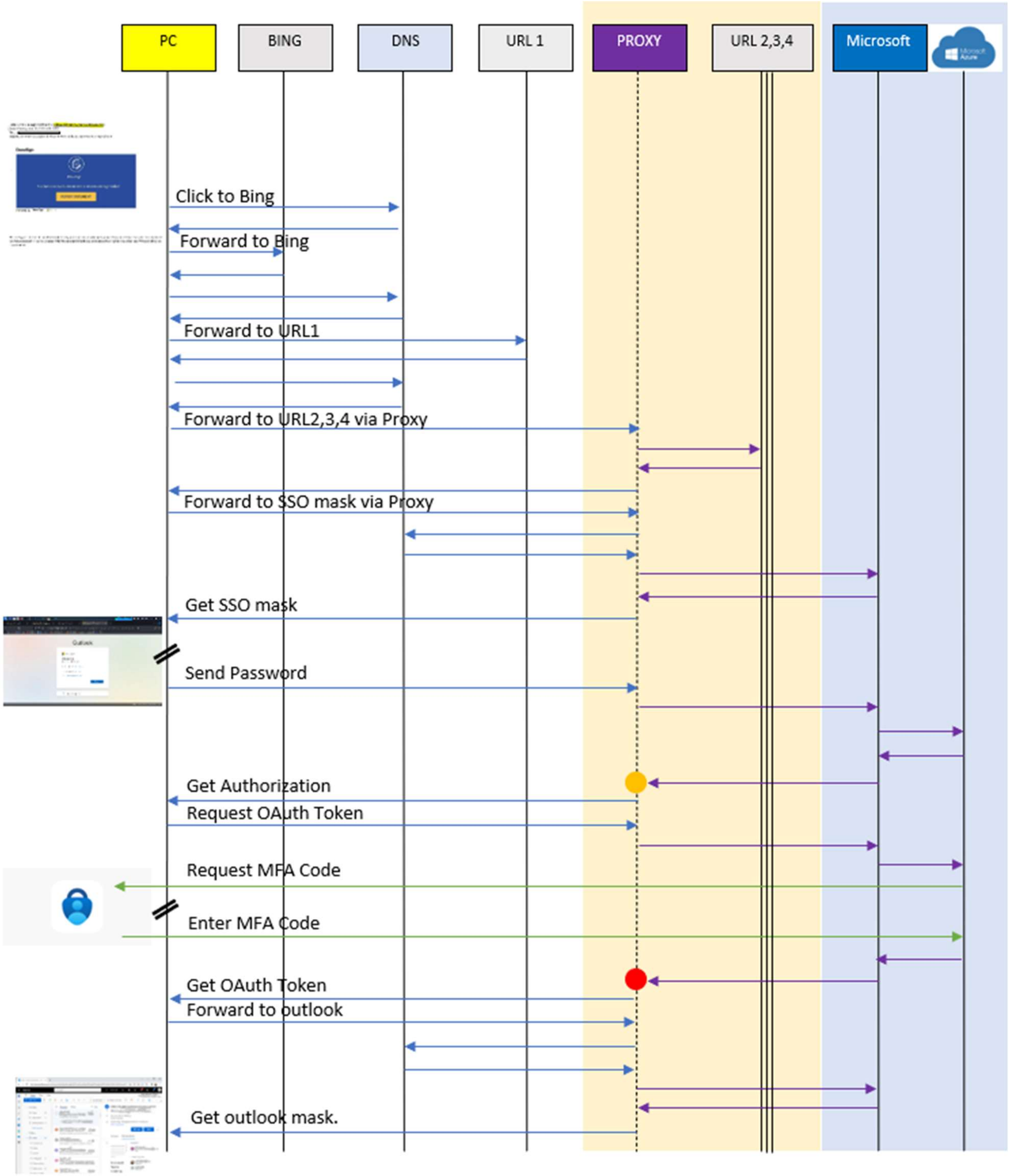
Step 2: Intercepting the Request

Step 3: Establishing the Bypass

Step 4: Forwarding the Request

Step 5: Gaining Unauthorized Access

MFA bypass - Simplified TCP-UDP/DNS Protocol Flow – reverse Engineering (probable TCP flow)



Reverse Engineering crafted by G.Moresi

Open Source Reverse Proxy based on NGINX

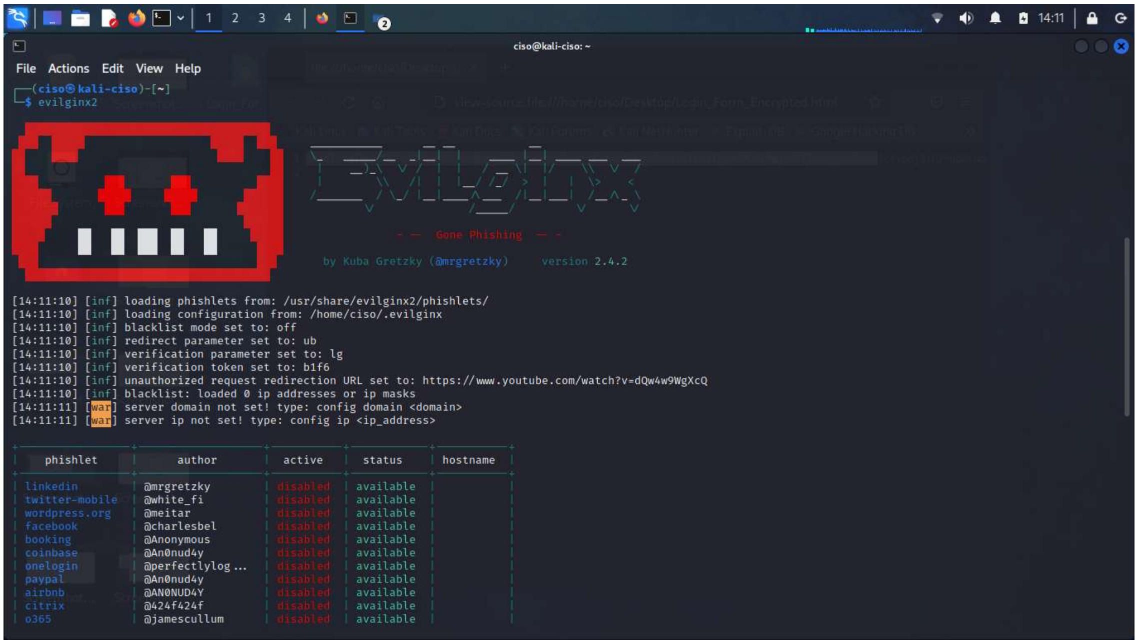
NGINX is a popular open-source web server and reverse proxy server that is widely used to handle high-traffic websites and applications. NGINX can also act as a reverse proxy, which means it can sit between client devices and backend servers, forwarding requests and responses between them. Here's an overview of NGINX's reverse proxy functionality:

Reverse Proxy Basics:

A reverse proxy is a server that receives client requests and forwards them to backend servers. It acts as an intermediary, providing various benefits such as load balancing, caching, SSL termination, and security.

NGINX as a Reverse Proxy:

NGINX is known for its efficient handling of concurrent connections and high scalability, making it an excellent choice for reverse proxying. By configuring NGINX as a reverse proxy, you can offload certain tasks from the backend servers and optimize the overall performance and security of your application.



```
┌──(ciso㉿kali-ciso)-[~]
└─$ evilginx2

                    - --  Gone Phishing  -- -

      by Kuba Gretzky (@mrgretzky)     version 2.4.2

[14:11:10] [inf] loading phishlets from: /usr/share/evilginx2/phishlets/
[14:11:10] [inf] loading configuration from: /home/ciso/.evilginx
[14:11:10] [inf] blacklist mode set to: off
[14:11:10] [inf] redirect parameter set to: ub
[14:11:10] [inf] verification parameter set to: lg
[14:11:10] [inf] verification token set to: b1f6
[14:11:10] [inf] unauthorized request redirection URL set to: https://www.youtube.com/watch?v=dQw4w9WgXcQ
[14:11:10] [inf] blacklist: loaded 0 ip addresses or ip masks
[14:11:11] [war] server domain not set! type: config domain <domain>
[14:11:11] [war] server ip not set! type: config ip <ip_address>

+----------------+-------------------+----------+-----------+----------+
|   phishlet     |      author       |  active  |  status   | hostname |
+----------------+-------------------+----------+-----------+----------+
| linkedin       | @mrgretzky        | disabled | available |          |
| twitter-mobile | @white_fi         | disabled | available |          |
| wordpress.org  | @meitar           | disabled | available |          |
| facebook       | @charlesbel       | disabled | available |          |
| booking        | @Anonymous        | disabled | available |          |
| coinbase       | @An0nud4y         | disabled | available |          |
| onelogin       | @perfectlylog ... | disabled | available |          |
| paypal         | @An0nud4y         | disabled | available |          |
| airbnb         | @AN0NUD4Y         | disabled | available |          |
| citrix         | @424f424f         | disabled | available |          |
| o365           | @jamescullum      | disabled | available |          |
```

Evilginx reverse proxy / https://github.com/kgretzky/evilginx2

NGINX can act as a caching reverse proxy, storing static or dynamic content in memory and serving it directly to clients, without hitting the backend servers. Caching reduces the load on backend servers and improves response times, particularly for content that doesn't change frequently.

In summary, NGINX's reverse proxy functionality allows it to sit between clients and backend servers, efficiently forwarding requests, balancing the load, caching content, terminating SSL, and providing security features. NGINX's flexibility, performance, and extensive feature set make it a popular choice for managing high-traffic websites and applications.

Detection of a reverse proxy

Detecting a reverse proxy or transparent proxy can indeed be challenging, but it is not necessarily impossible. While proxies are designed to forward network traffic on behalf of clients, making the original source of the request less apparent, there are several techniques and indicators that can potentially help in identifying their presence. Here are some considerations:

- **Header analysis**: Proxies often add or modify HTTP headers. By inspecting the headers of incoming requests, you may find indications of proxy involvement. For example, headers like "X-Forwarded-For," "Via," or "Proxy-Connection" may contain information about the proxy server.
- **IP address analysis**: Proxies typically replace the client's IP address with their own when forwarding requests. By examining the source IP address in the network packets, you can sometimes identify the presence of a proxy. However, this may be more challenging if the proxy is operating at the transport layer (e.g., using NAT techniques).
- **Time delays**: Proxies can introduce additional latency due to the extra processing involved in forwarding requests. By comparing the response times of requests, you may notice delays that could suggest the presence of a proxy.
- **SSL/TLS certificates**: Proxies that terminate SSL/TLS connections and establish new ones may use their own certificates. By analyzing the certificate chain during an SSL/TLS handshake, you might detect the presence of a proxy server.
- **Behavior analysis**: Proxies may exhibit certain behavioral patterns that can be used to identify them. For example, they may modify the User-Agent header or exhibit consistent connection behavior across different requests.
- **Network monitoring**: Deep packet inspection (DPI) or network monitoring tools can help analyze network traffic and identify patterns associated with proxy usage. These tools can detect unusual packet structures, traffic patterns, or anomalies that may suggest the presence of a proxy.

Overall, detecting a reverse proxy or transparent proxy requires a combination of technical analysis, network monitoring, and understanding of proxy behavior. It often requires a comprehensive approach rather than relying on a single technique.

Here is a link to an informative whitepaper that discusses standard techniques: http://www.icir.org/christian/publications/2016-acsac-revprobe.pdf

Real examples

Based on my recent analysis of real examples, it has become evident that detecting proxies using the previously mentioned parameters is exceedingly challenging. The instances I examined over the past few weeks have shown that time delays are minimal. This is primarily due to reverse proxies promptly caching phishing websites upon the initial request, resulting in exceptionally quick responses. Additionally, the headers remain unaltered and unchanged in these cases.

Approach via Traffic Analysis

The phishing website offers an excellent user experience, making it extremely challenging, if not impossible, for users to determine whether they are connected to the genuine site, often resembling the Microsoft Login Form. However, by conducting traffic analysis and capturing TCP/UDP packets, it becomes possible to ascertain if a visitor has been redirected multiple times before reaching the reverse proxy.

The new solution can be implemented as app, plugin or agent in a browser and quickly redirect the user to a warning page indicating that he/she is connected through a transparent proxy.

The purpose of this whitepaper is not to provide an exhaustive and intricate solution to this challenging task. However, it aims to assist in the development of new strategies for user protection. As I have emphasized repeatedly, even in various conventions, the most secure approach would be to adopt the concept of Zero Internet. This involves maintaining a whitelist of URLs and IPs that corporate users are permitted to connect to. Depending on the risks and desired level of exposure, this list can be expanded or adjusted accordingly.

Example: Green List for Microsoft Login Form

The process of merging all correct DNS requests and selecting only the top domains yields the following green list outcome:

```
.office365.com
.office.com
.msn.com
.live.com
.bing.com
.microsoft.com
.microsoftonline.com
.skype.com
.msftauth.net
.msauth.net
.azureedge.net
```

As evident from the three presented examples, correlating DNS queries with the green list provides a distinct indication of whether the user has undergone multiple redirections. A notable difference of at least +33% (1/3) serves as a strong indicator for detecting proxy-based redirections. Conversely, the analysis based on the time taken to send TCP packets is currently an outdated method of analysis.

Algorithm for detecting a reverse Proxy

The new approach focuses on detecting the number of DNS translations and evaluating the reputation of individual IPs. Additionally, it involves correlating the DNS translation with the "normal" path used when users directly access the Microsoft login.

It can be resumed as follows:

- The TCPDUMP application is launched listening port 53 or 853 (secure DNS)
- The user visits a URL.
- DNS queries are captured and recorded.
- The DNS queries are correlated with the green list.
- If the correlation is less than 100%, the user is redirected to a secure site.
- If the HTML code is encrypted, the user is redirected to a secure site.
- Otherwise, the website is displayed.

Performing a correlation between the green list with the DNS requests collected by visiting a URL gives a difference of at least +33%.