公衆無線 LAN を用いた HTTPS 通信に対する MITM 攻撃の新手法の開発とその実装

神戸大学工学部電気電子工学科 木村圭一朗

2022 年 3 月 29 日

概要

ここに概要を記述する。この文章は LaTeX の機能と表示結果を表示する説明書です。概要は 1 コラム、本文は 2 コラムで書かれています。図の挿入については 2 コラムに収まるように表示サイズを調整していますが、表については 2 コラムのレイアウトに納まらないので table 環境使用時に table*とすることで 1 コラムで表示するようにしています。

目次

1 背景

2 目的

偽の SSID を用いた公衆無線 LAN からクライアントと正規サーバとの通信に割り込み，デバイスに警告を出さずに通信の傍受・改竄を行う事が主な目的である．

3 前提知識

3.1 無線 LAN 関連について

Captive Portal

無線及び有線ネットワークに接続したユーザーが，ネットワークへのアクセスを許可される前に表示される Web ページを指す．
ユーザーは，アクセス許可に必要な情報の入力或いは利用規約への同意を行い，その認証を行ってもらう事でインターネットへのアクセスが可能になる．

4 前提条件と仮説

4.1 前提条件

今回，MITM 攻撃を仕掛けるにあたり以下の前提条件を設ける．

被害者は

- 偽 SSID を有する無線 LAN アクセスポイント（以下，AP）を使用する．
- https 通信で任意のサイトを閲覧する．
- 通信先を確認しない．

4.2 仮説

上記の前提条件をもとに，

5 検証内容

5.1 検証対象

MITM 攻撃が可能か否かの検証にあたり，昨今急激な普及が見られるインターネット通販サイトを対象とした．その中でも国内 EC モールの売上ランキング上位 2 つを占める「楽天」と「Amazon」を対象とした．

5.2 検証フロー

具体的な検証フローは次の通り.

1. CaptivePortal を検知させて，予め攻撃者が用意しておいた Captive Portal サイトに誘導させる.
2. Captive Portal サイトで表示された偽の検索エンジンから，被害者が検索したワードを攻撃者サーバで取得する.
3. 攻撃者は取得したワードをバックエンドで検索し，その検索結果を取得.
4. 取得した HTML ファイル内にあるハイパーリンクを書き換え，そのコピーを被害者に提示する.
5. 以後，クリックした URL を攻撃者サーバで取得する.
6. 攻撃者は取得した URL を利用して正規サーバにアクセスする.
7. 正規サーバから返却された HTML ファイル内のハイパーリンクを書き換えたものを被害者に提示する.

5.3 実装

攻撃者サーバの実装は Golang で実装した．以下に該当の GitHub リポジトリとデモサイトのリンクを載せている.

5.3.1 3 について

ここでは, 最も利用されている検索エンジン Google を用いた．Google での検索結果を取得する為には, 検索クエリを作成する必要がある．基本的に任意のクエリ (query) に対して https://google.com/search?q=query という URL が Google の検索 URL となっているが，クエリに空文字列が存在する場合は，その空文字列を + に置換する必要があることに留意しなければならない．例えば,「神戸大学　工学部」と検索する場合には, その URL は https://google.com/search?q=神戸大学 + 工学部となる.

5.3.2 4,5 について

取得した HTML ファイル内にあるハイパーリンクがそのままであれば，攻撃者サーバではなく正規サーバとの通信に切り替わってしまう． 従って，既存の URL を攻撃者サーバへ通信するように書き換え，且つ既存の URL を正確に抽出する必要がある． これを実現する為には，既存のハイパーリンクをルールに則って書き換える必要がある． 具体的に，https://example.com という URL に対して処理を行うことを考える． サーバには予め，URL を受け取る為のエンドポイントを設置する．今回の場合は「/templates」というエンドポイントに対して,「url」というパラメータを受け取るものとする． このエンドポイントに対して適切に URL を飛ばすために，サーバ側で予め https://mitm.es3.com/templates?url=https://example.com のように書き換える． その結果，被害者に提示した HTML ファイル内にあるハイパーリンクをクリックすると攻撃者サーバに飛び，且つサーバ側で遷移しようとしたページのリンクを取得できる.

5.3.3 6,7 について

フローの 4 と 5 で得た正規 URL を用いて，バックエンドで正規サーバとの通信及び HTML ファイルの取得を行う． 通常の通信であれば HTML ファイルの取得のみでよいが，個人アカウントへのログインを行う際は，その ID とパスワードを取得し且つ得られた情報と実際に登録されている情報との整合性の確認を行わなければならない． 登録情報の整合性に関しては，取得した個人情報を攻撃者が手動で入力・確認を行わず，バックエンドでブラウザのインスタンスを生成して入力・確認を行う． この処理に関して，Chromedp という Golang で実装されているパッケージを用いた． 具体的な実装については GitHub レポジトリを参照して頂きたい．ここでは，処理の流れを以下に簡単に示す．

1. Chrome インスタンスを生成する．
2. 指定 URL を叩き，JS Path を指定して該当部分に取得した ID やパスワードを入力する．
3. 個人情報の入力が完了すれば，同じくログインボタンの JS Path を指定してログイン処理を行う．
4. 正規サーバに情報を整合させ，返却された HTML ファイル内のハイパーリンクを，4 と 5 と同じ要領で書き換えて被害者に返却する．

JS Path の指定に関しては，予め正規サイトのログイン画面を見てから確認・指定をする必要がある．

6 検証結果

検証結果は以下のようになった．

6.1 楽天の場合

6.2 Amazon の場合

7 考察

8 参考文献・サイト