情報ネットワーク応用 専門科目演習 OS コマンドインジェクション

細川 夏風

2025 年 12 月 27 日

1 目的

近年，計算機の発展は目覚ましい．それに起因してオペレーティング・システム (以下:OS) 等の低レイヤのサービスがより拡充された．それがいくつかの問題を引き起こしている．その中で最も大きな問題がセキュリティに関する問題である．OS を構成しているコードの中に難解なアルゴリズムが散見される．これにより，問題の重要度に関係なく少なからずセキュリティホールが発生してしまう．中でも最も易く，危険度が高いのが “OS コマンドインジェクション” である．それに対する，理解を深め，それから防御するということを理解するため攻撃演習を行う．

2 内容

2.1 構成

今回の環境は “CYNEX” から提供されたものであり，大まかに以下のように作られている．

- 受講者用サーバ: windows(おそらく 10)
- 受講者用コンテンツサーバ: Linux(Ubuntu，ver 不明)
- サーバ構築のために Docker を用いて，解析環境を立ち上げる

2.2 環境

今回の Docker のコンテナ内の環境を以下に示す (コンテナ名等は念の為，ここに記さない)．

- app.py によって Web ページから情報を取得し，それに対応した値を返す
- ローカル DNS サーバ (ローカル IP であるがここには記さない)

2.3 手順

1). 受講者サーバへ Apache Guacamole から接続
2). 受講者用サーバから受講者用コンテンツサーバに ssh からログインする
3). スーパーユーザになる
4). 作業ディレクトリから Docker を起動し，コンテナ内に入る
5). コンテナ内に存在している app.py のソースコードと/var/log/nginx/に存在している access.log のファイルから解析，ログの確認を行う
6). ブラウザから URL を入力し，サーバにアクセスしてから攻撃する
7). 攻撃はサイトのテキストボックスに「; cat /etc/passw」と入力する
8). app.py を書き換えて，API サーバを再起動後，結果を確認する

3 結果

以上の手順を行うと，テキストボックスの下にあるボックスに「dig; cat /etc/passwd」の結果が出力される (結果はここに記さない)．上記ファイルには以下の情報が格納されており，以下のような規則で記されている．

- 第 1 フィールド: ユーザ名
- 第 2 フィールド: パスワード
- 第 3 フィールド: ユーザ ID
- 第 4 フィールド: グループ ID
- 第 5 フィールド: コメント欄
- 第 6 フィールド: ホームディレクトリ
- 第 7 フィールド: ログインシェル

これの防御策として以下のようなものがある

- 今回は app.py に Shell を利用してコマンドを実行するようになっていたが，その部分を Shell を用いずとも実行できるように書き換える
- エスケープ処理を行う

Python の Web アプリ制作によく用いられるあるモジュールでは OS コマンドを実行する際，Shell を経由せずに Python が直接そのプログラムを実行するというモードが搭載されている．これにより，Shell が可能なことでも Python には不可能なことが存在する．今回の「cat /etc/passed」が不可能なのも一つの例だ．

次にエスケープ処理であるが，これは Python であれば，単純に記述可能である．Python 内に存在する in というコマンドを用いれば，in の前述の文字列が後述の文字列の中に含まれているかの

結果を boolean で返す．これに or，and 等を用いれば更に条件を絞ることも可能である．

```
    if ";" in cmd or "'" in cmd or "\" in cmd:
      何かしらの処理
```

図 1　エスケープ処理のプログラムの例

4　考察

今回，OS コマンドインジェクションの攻撃演習を行った．以前，Web アプリケーションを作成した際にもこれと SQL インジェクションに留意して開発を行った．このようなインジェクション攻撃は直接的なアルゴリズムを用いて開発を行うと，必ず攻撃の隙を生んでしまう．この攻撃は歴史が長く，仮に攻撃が成功した時の被害がとても大きい攻撃手段であるために多くのプログラミング言語やサービスで処置がされている．様々な対策が取られているために，この攻撃を許すようなサービスを実装してしまった場合，そのサービスの信頼が地に落ちることになってしまう．よってこのような攻撃には常に気を張って対処すべきであり，開発者であれば知らないでは済まされないようなセキュリティ的事案なのである．そして，私が次に作るサービスは前回のサービスより低レイヤのサービスなどを多く用いる．自戒をもって，本稿の結びとする．