情報ネットワーク応用 演習 B

学籍番号 1280391
細川 夏風

2026 年 2 月 8 日

1 背景

このレポートは先日の演習 A レポートの続きであるため，背景，内容ともに必要である部分のみを記述する．

2 内容

2.1 使用ツール

curl

```
Usage: curl [options...] <url>
```

curl とは、URL 構文を使用してデータを転送するためのコマンドラインツールである。HTTP, HTTPS, FTP, SCP など多数のプロトコルをサポートしている [1]。

- -d, --data <data>
  - HTTP POST data
    - HTTP サーバーへ POST リクエストでデータを送信する
    - コンテンツタイプはデフォルトで application/x-www-form-urlencoded となる
  - <data>
    - 送信するデータを文字列で指定する
    - @filename のように @ を先頭につけることで、ファイルの内容を読み込んで送信できる
- -f, --fail
  - Fail fast with no output on HTTP errors
    - サーバーエラー時 (400 番台や 500 番台のレスポンス) に、エラーページの内容を出力せず、即座に失敗として終了する

    - スクリプト等で終了コード (exit code 22) によるエラー判定を行いたい場合に用いる
- -h, --help <subject>
  - Get help for commands
    - コマンドの使い方やオプションの一覧を表示する
  - <subject>
    - all を指定すると全てのオプションを表示し、カテゴリ名 (例: http) を指定するとそのカテゴリのオプションのみを表示する
- -o, --output <file>
  - Write to file instead of stdout
    - 取得したデータを標準出力 (画面) ではなくファイルに書き出す
  - <file>
    - 保存先のファイル名を指定する
- -O, --remote-name
  - Write output to file named as remote file
    - リモートの URL に含まれるファイル名部分をそのままローカルの保存ファイル名として使用する
    - 例: curl -O http://example.com/foo.html を実行すると foo.html というファイル名で保存される
- -i, --show-headers
  - Show response headers in output
    - レスポンスボディ (データ本体) だけでなく、HTTP レスポンスヘッダーも含めて出力する
- -s, --silent
  - Silent mode
    - 進捗状況 (プログレスメーター) やエラーメッセージを表示しないようにする
- -T, --upload-file <file>
  - Transfer local FILE to destination
    - ローカルのファイルを指定した URL へアップロードする (通常は PUT メソッドが使用される)
  - <file>
    - アップロードするローカルファイルのパスを指定する
- -u, --user <user:password>
  - Server user and password
    - Basic 認証や Digest 認証などで使用するユーザー名とパスワードを指定する
  - <user:password>
    - user:password の形式で指定する。パスワードを省略した場合はプロンプトで入力を求められる

- -A, --user-agent <name>
  - Send User-Agent <name> to server
    - サーバーに送信する User-Agent ヘッダーの値を偽装または指定する
  - <name>
    - ブラウザ名やボット名などの文字列を指定する
- -v, --verbose
  - Make the operation more talkative
    - 詳細表示モード。リクエストとレスポンスのヘッダー情報、TLS ハンドシェイクの様子などを表示し、デバッグに用いる
    - > は送信データ、< は受信データを表す
- -V, --version
  - Show version number and quit
    - インストールされている curl のバージョンと、サポートしているプロトコルやライブラリの一覧を表示して終了する

2.2 方法

今実習のこの IoT デバイスには深刻な脆弱性が存在する．このように，深刻な脆弱性には CVE という記号とともに番号がつけられ管理されている．その CVE と用いられた IoT デバイスを照らし合わせたとき，CVE-2019-12289 という記号であることがわかる．このデバイスはとある sh ファイルによって起動ごとに IoT デバイスを起動していることがわかる．この sh ファイルの内容に telnet を起動するようなコマンドを追記することによって外部から非常にこの IoT デバイスに接続可能になる．

```
1  echo -e "<CMD>" >> /system/init/ipcam.sh
```

このようなプログラムであれば bash コマンドの任意のプログラムが実現可能である．

この機能を利用した Python を前に作成し，それを利用し攻撃用ファームウェアである fw.bin というファイルを作成する．

```
1  # curl -X POST -F “file1=@fw.bin;filename=fw.bin” “http://< Iotデバイス
       のIP >:45678/upgrade\_firmware.cgi?loginuse&loginpas”
```

というコマンドによって，特定のファイルを指定した Web サイトに送信する．今回送信した Web サイトはその IoT デバイス専用のアップデートファイルの置き場である．

しかし，これによって管理画面の認証を突破できる訳では無い．そのようなとき，検索エンジンでこの IoT デバイスについて調べるとなんと情報が掲載されているケースがある．それを使用すれば簡単にこのデバイスを乗っ取ることができる．

3 考察

今回の攻撃のようにまだセキュリティパッチが当たっていないデバイスが狙われるケースは非常に多い．理由は単純である．攻撃のマニュアルは前の攻撃者たちがいくらでも発掘しているからである．そのため，攻撃者の苦労はほとんど存在しない．このような攻撃の対策は非常に単純であり，セキュリティパッチを当てることである．これがなされていないケースは非常に多い．また，セキュリティパッチが当てられないのであれば使用しない．これを徹底するしか無いのだ．今回は IoT デバイスであったが，それによって他の多くのデバイスが犠牲になる可能性を秘めていることは今実習で理解できた．また，そこからそれに接続されているルータ等のネットワークデバイスに接続を許すのだ．これによって発生する問題は非常に単純であり．情報の多くを奪取される可能性が非常に高いことである．しかし，ここに更に大きな問題が隠れており，それがさらなる上位ネットワークの存在である．本社サーバ等が存在した場合それがそのネットワークへの侵入が可能になる可能性がある．そうなってしまえばそのサービス自体が停止しかねない．また，その会社の業務自体が機能停止に追い込まれる可能性がある．このようにたったひとつのデバイスから全体の機能停止の可能性までデバイスの脆弱性がもたらす被害は計り知れない．アップデートの重要性の理解が更に深まることを理想に本稿の終文とする．

参考文献

[1] https://curl.se/docs/manpage.html