Reference Sheet

Hack The Podsについて

Hack The Podsは、プロジェクト内部で開くオフラインCTFです。

通常のオンラインCTFとは異なり、実際のマシンに問題を配置して解いてもらうのが特徴です。

進め方

全ての問題で、

1. `htp-ctf{xxx}` のという形式のFlagを探す
2. それぞれの問題の解答Formに入力
3. Submit
4. 合っていれば得点！

のような形で進めていきます。

色々な問題からFlagを見つけ、CTFdに入力してポイントを獲得しましょう！

また、途中わからない単語や事象に遭遇したら、「xxx とは」のような感じで検索してみることをオススメします。

それでもわからなかったら、巡回している運営メンバーに聞いてくれれば助言します！

Hintについて

問題によってはHintが設定されています。

これは自分の獲得したポイントを使って見ることができます。

ご利用は計画的に！

CTFdについて

CTFのコンテストWebアプリのOSSです。

脆弱性があるようなのでパブリックな場面では使えませんが、
基本的に必要な機能は揃っているので，内部での利用には問題ないです。

ユーザ登録をして各種問題を解いていきましょう！（まずはWelcome問がおすすめです）

問題サーバへのSSH

このHack The Podsでは問題サーバに遠隔でアクセスして，解く必要のある問題がいくつかあります。

そのアクセスにはSSHを用います．下記がそのコマンドです。

また，SSHに成功すると，問題「SSH CONNECTION」のFlagを入手できます。

```
ssh [username]@[ip-addr]
```

```
ssh kit@192.168.0.1
```

例: `kit` というユーザ名で `192.168.0.1` のサーバにSSHしている

よく使うツール／コマンド一覧

CTFをやる上でよく使うツールやコマンドについて示します。

全ては示しませんが，ここを参照するだけでWelcome問は解けると思います。

もちろんここに存在しないコマンドやツールを使用する必要のある問題も存在します。

基本コマンド

- `ls` — ファイルやディレクトリを一覧するコマンド

```
ls path/to/dir
```

第一引数に対象を指定 引数なしでカレントディレクトリの中身を一覧します

```
ls -l path/to/dir
```

`-l` optionはファイルのサイズや作成日、権限などを表示します

```
ls -a path/to/dir
```

`-a` optionは隠しファイルも同時に表示できます

- `cd` — カレントディレクトリを移動するコマンド

```
cd path/to/dir
```

第一引数に宛先を指定 引数なしでホームディレクトリに移動します

- `echo` — テキストの行を表示する

```
echo "Hello World"
```

第一引数に指定したものを標準出力に表示する

- `mv` — ファイルを移動するコマンド

```
mv ./file1 dir/
```

第一引数に対象ファイル 第二引数に移動先を指定します

- `chmod` — ファイルの権限を設定する

```
chmod 644 file
```

第一引数に権限設定、第二引数に対象ファイルを指定。例は `rw-r--r--` の権限を設定している

- `cat` — ファイルの中身を表示するコマンド

```
cat file.txt
```

第一引数に対象ファイルを指定 複数のファイルを指定すると中身を連結して出力します

- `less` — ファイルの中身を表示するコマンド

```
less file.txt
```

第一引数に対象ファイルを指定 `cat` と違い移動や検索が可能

- `file` — ファイルタイプを表示するコマンド

```
file file1
```

第一引数に対象ファイルを指定 ファイルの種類などを表示します

- `zip` — ファイルやディレクトリを圧縮するコマンド

```
zip -r file.zip dir/
```

`-r` optionは再帰的にという意味で、ディレクトリを対象にするときは必要

- `unzip` — zipファイルを伸長（解凍）するコマンド

```
unzip file.zip
```

第一引数に対象を指定します。他にも様々な解凍系コマンドがあるので調べてみよう。

ネットワーク系コマンド

- `ping` — 疎通を確認するコマンド

```
ping 192.168.0.1
```

第一引数にIPアドレスやドメイン名を指定すると、そこへの疎通を確認してくれます
疎通確認には `ICMP` というプロトコルが使われています

- `curl` — URLを叩くコマンド

```
curl https://www.google.com
```

第一引数にURLを指定するとそこへアクセスし、レスポンスを表示します

```
curl -X GET https://www.google.com
```

`-X` optionでHTTPメソッドを指定可能。 `GET` / `POST` / `PUT` / `DELETE` などが存在します

- `nmap` — サービスが建っているポートをスキャンするコマンド

```
nmap 192.168.0.1
```

第一引数にIPアドレスやドメイン名を指定すると、そのサーバのポートをスキャンします

- `nc` — netcatコマンド、TCP/UDPの簡易クライアント/サーバプロセスを起動するコマンド

```
nc 192.168.0.1 10000
```

第1引数にIPアドレス、第2引数にポート番号を指定すると、
クライアントとして引数に指定したサーバーに接続を行います

バイナリ系コマンド

- `strings` — ファイルから表示可能文字を表示するコマンド

```
strings file
```

第一引数か標準入力から対象を受け取ります

- `hexdump` — ファイルの中身を16進ダンプしてくれるコマンド

```
hexdump file hexdump -C file
```

第一引数か標準入力から対象を受け取ります
`-C` optionで1Byte区切り、ASCIIを追加表示できます（デフォルトは2Byte区切り）

- `objdump` — 実行形式のファイルを逆アセンブルして表示します

```
objdump -d file objdump -d -M intel file
```

`-d` optionで機械語を含むとされているセクションを逆アセンブル
`-M` optionで逆アセンブラ出力のオプションを指定します。ここでは表示をintel記法に置き換えています（デフォルトはAT&T記法）

テキストエディタ系コマンド

- `nano` — CLIエディタ

```
nano file
```

`Ctrl-S` — 保存
`Ctrl-X` — 終了

- `vim` — CLIエディタ

```
vim file
```

`i` — insertモードへ
`Esc` or `Ctrl-[` — normalモードへ
`:w` — normalモード時、保存
`:q` — normalモード時、終了

- `emacs` — CLIエディタ

```
emacs file
```

`Ctrl-X Ctrl-S` — 保存
`Ctrl-X Ctrl-C` — 終了
`Ctrl-g` — コマンドの中断

- `gedit` — GUIエディタ

```
gedit file
```

- `hexedit` — ファイルをバイト単位で編集できるエディタ（バイナリエディタ）

```
hexedit file
```

第一引数に指定したファイルを編集します。
`F1` — ヘルプ
`F2` — 保存
`/` — 検索
`Ctrl-X` — 保存して終了
`Ctrl-C` — 保存せずに終了

ツール

- Webブラウザ — HTML/CSS/JavaScriptを解釈してWebページを表示してくれるツール
  - Chrome / Firefox / Microsoft Edge / Safari などいろいろ
  - 右クリックから「ソースを表示」などでページのHTMLを読める
    - Webページには描画されていないが、重要な情報がある場合がある
  - 右クリックから「開発者ツール」などでページのリクエスト詳細やスタイルシートが見える
    - Webページの構造やネットワーク越しにやりとりしている内容を調べたい時に有用
- WireShark — パケットキャプチャファイルを開くツール
  - https://www.wireshark.org/#download からDL可能
  - Wiresharkでググれば一番上にでてくるはず
  - `.pcap` や `.pcapng` などを開くことができる
  - 一行一行がパケット一つ一つに対応していて，やりとりしている内容をみることができる

- GDB（Gnu DeBugger） — CLI上で動作するデバッガ

```
gdb file
```

  - 実行形式のファイルを実行しながら、プログラムの動作確認やバグの発見が行えます。
  - アセンブリで表示されるため、アセンブリ言語の知識が必要です。
  - 解析の基本的な流れはこの様になってます。
    ブレークポイントを設置 → プログラム実行 → （ブレークポイントで停止）→
    → レジスタ/メモリ/スタック の確認 → 実行継続→ ...

パッケージマネージャ

- `apt` — debian系OS（ubuntuなど）でよく使われる

```
sudo apt install <package-name>
```

  - proxyについて
    大学のWi-Fi（KIT-WLAP2）につなげている場合は `/etc/apt/apt.conf.d/00-proxy` に下記を追記してください

```
Acquire::http::Proxy "http://wwwproxy.kanazawa-it.ac.jp:8080"; Acquire::https::Proxy
"http://wwwproxy.kanazawa-it.ac.jp:8080";
```

- `dnf` — CentOSなどRedHat系OSでよく使われる（ `yum` の後継）

```
sudo dnf install <package-name>
```

こちらの場合も同様にProxyがあるので，installできない場合は調べたり聞いたりしてください

プロトコル

- `HTTP` — 平文でWebコンテンツをやりとりする
- `HTTPS` — TLSで暗号化し、Webコンテンツをやりとりする
- `TCP` — コネクション型通信を実現するL4プロトコル
- `UDP` — コネクションレス型通信時に使われるL4プロトコル