

SECCON の過去問から学ぶ

Server Side Request Forgery(SSRF)

～IPアドレスの様々な表現方法～

SSRFとは

外部に公開していないサーバに対して、攻撃を仕掛ける手法の一つ。

CTFでもいくつか主題されている。

基本的にはWebサーバの機能の脆弱性を利用して、内部サーバへアクセスする問題となる。

SSRF過去被害事例(参考)

2019年に米金融大手企業が不正アクセスにより1億人を超える個人情報流出した事件がある。

分かりやすく解説しているサイトがあるので、詳しくはそちらを参照。

[SSRF攻撃によるCapital Oneの個人情報流出についてまとめた](#)
(piyolog)

SECCON に登場した2例

今回は、SECCON 2019 と SECCON Beginners 2021 に登場した2つの問題を紹介。

SECCON 2018

[Web][GhostKingdom](#)

SECCON Beginners 2021

[Web][check url](#)

problem 1

GhostKingdom (SECCON 2018)

GhostKingdom (SECCON 2018)



Login (from internet)

User:

Password:

Login

[Create new user](#)

GhostKingdom (SECCON 2018)

概要

1. ユーザを作成してログインすると3つのメニューがある
 - Message to admin(メッセージを入力 or preview機能がある)
 - Take a screenshot(指定したURLにアクセスしScreenshotを取得する)
 - Upload image * Only for users logged in from the local network
2. 3つめのメニューはローカルネットワークからしかアクセスできない

GhostKingdom (SECCON 2018)

概要

- Message to admin
→メッセージを入力 or previewする機能
- Take a screenshot
→指定したURLにアクセスしScreenshotを取得する機能
- Upload image * Only for users logged in from the local network
→画像ファイルをアップロードする機能

攻略手法(ざっくり)

Upload image にたどり着くため、「Message to admin」と「Take a screenshot」に存在するそれぞれの脆弱性を利用する。

1. 「Take a screenshot」の機能を利用して**サーバ自身からアクセスさせる**
2. CSS Injectionを使い、セッションIDを取得して管理者でアクセス
3. GhostScriptの脆弱性を利用するPoCを使って画像ファイルを作成し、Uploda imageにアップしフラグを取得する

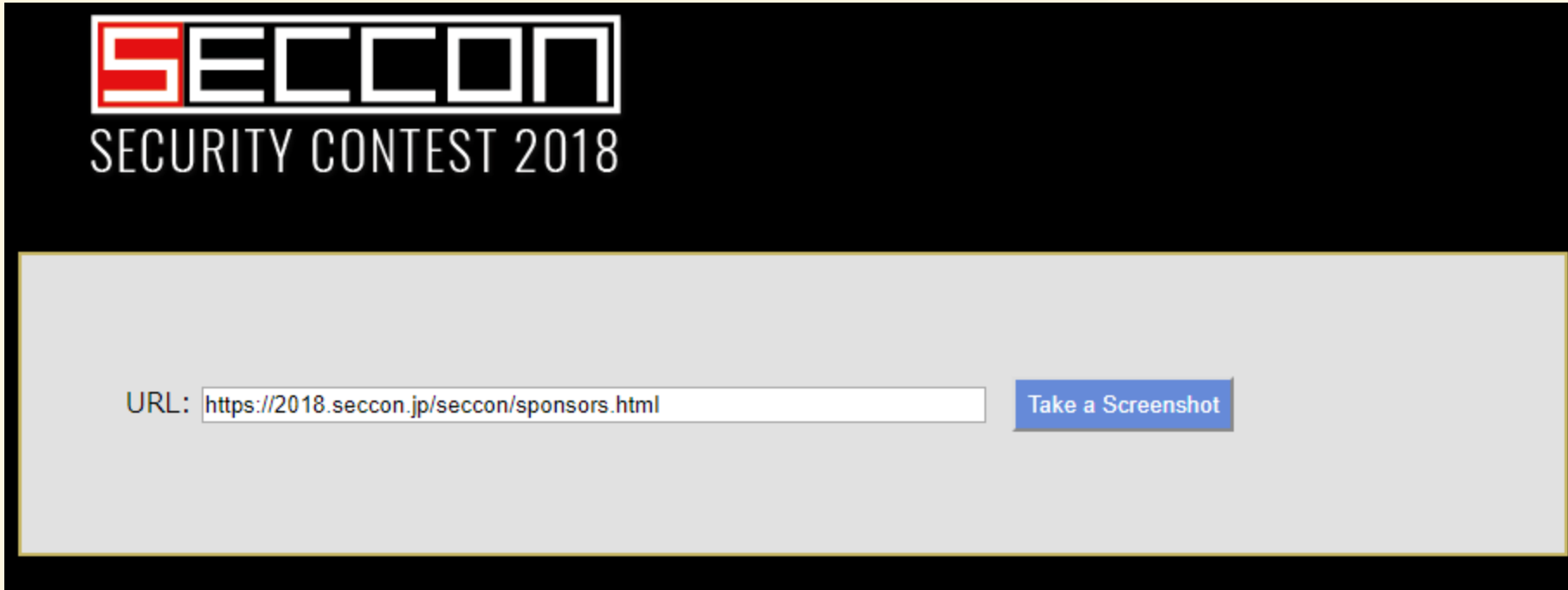
今回の注目ポイント

“ 1. 「Take a screenshot」の機能を利用して**サーバ自身からアクセスさせる** ”

これは具体的にどういうことをやっているのか？

Take a screenshotの機能おさらい

- 指定したURLのスクリーンショットを取り、表示する機能
- URLを指定する際の画面は以下



The image shows a web interface for SECCON SECURITY CONTEST 2018. At the top, the logo "SECCON" is displayed in a stylized font with a red square on the left, followed by "SECURITY CONTEST 2018" in white text on a black background. Below this, there is a light gray rectangular area containing a form. The form has a label "URL:" followed by a text input field containing the URL "https://2018.secon.jp/secon/sponsors.html". To the right of the input field is a blue button with the text "Take a Screenshot".

今回の注目ポイント

- 上記で指定するURLに**サーバ自身のアドレスを指定**した際、機能を動作させている権限(管理者であることが分かっている)で動作する
- しかし、単純に「localhost」や「127.0.0.1」はブロックされている

今回の注目ポイント

ブロックされている際に表示されるメッセージは以下の通り。

“ <http://localhost/> を入力すると、You can not use URLs that contain the following keywords: 127, ::1, localの表示。 ”

出典：[SECCON 2018 Quals - GhostKingdom\(こんとろーるしーこんとろーるぶい\)](#).

問われていること

Q. 「localhost」や「127.0.0.1」意外でループバックアドレスを指定する方法、知ってますか？

解き方

「127」, 「::1」, 「local」が入らない方法で、サーバ自身(ループバックアドレス)を指定するよう表記を変換する。

実は、127.0.0.1を10進数値に変換してアドレス指定してもループバックアドレスと判断される仕様がある。今回、これを利用する。

127.0.0.1 -> 10進変換 -> 2130706433

<http://localhost/> -> <http://2130706433/>

解き方の詳細について

正確には細かい指定が必要だったり、CSS Injection や GhostScript で頑張る必要がありますが、今回のテーマから外れてしまうので割愛。

GhostKingdom (SECCON 2018)の詳しい解き方について知りたい方は、以下に解説しているサイトがあるのでそちらをご覧ください。

- [SECCON 2018 Quals - GhostKingdom\(こんとろーるしーこんとろーるぶい\)](#)
- [json 作問者の解説](#)

problem 2

check_url (SECCON Beginners 2021)

check_url (SECCON Beginners 2021)

c(heck)_url

c(heck)_url

You can run `curl` like this!

Here, take this
URL: <https://www.example.com>

Example Domain

check_url (SECCON Beginners 2021)

概要

- 「like this」 をクリックすると、URLのパラメータにFQDNが指定される
- 直後、パラメータに指定されたURLのページが画面に表示される
- 動作しているソースコードのサンプルが配布されている
- URLを表示する機能には curl コマンドが使用されている
- ループバックアドレスを指定することでフラグを出力することが分かっている

check_url (SECCON Beginners 2021)

概要

```
if ($_SERVER["REMOTE_ADDR"] === "127.0.0.1"){  
    echo "Hi, Admin or SSSSRFer<br>";  
    echo "*****FLAG*****";  
}
```

check_url (SECCON Beginners 2021)

概要

- 127.0.0.1を指定してもサニタイズされ失敗する。
- 他にもサニタイズするルールが色々指定されていることを以下のソースより確認できる。

```
if ($url !== "https://www.example.com"){  
    $url = preg_replace("/[^\a-zA-Z0-9\:\/\/:]+\u", "🐼", $url); //Super sanitizing  
}  
if(stripos($url,"localhost") !== false || stripos($url,"apache") !== false){  
    die("do not hack me!");  
}
```

問われていること

Q. 「localhost」や「127.0.0.1」意外でループバックアドレスを指定する方法、「どれだけ」知ってますか？

解き方

こちら先ほど紹介した問題と同様、ループバックアドレスの形を変化させることで回避することが可能。

ただし、この問題では先ほど使った10進数表現では上手くアクセスできない（サーバの設定でアクセスできないようにしていたため）

そのため、10進数以外で指定する必要がある。

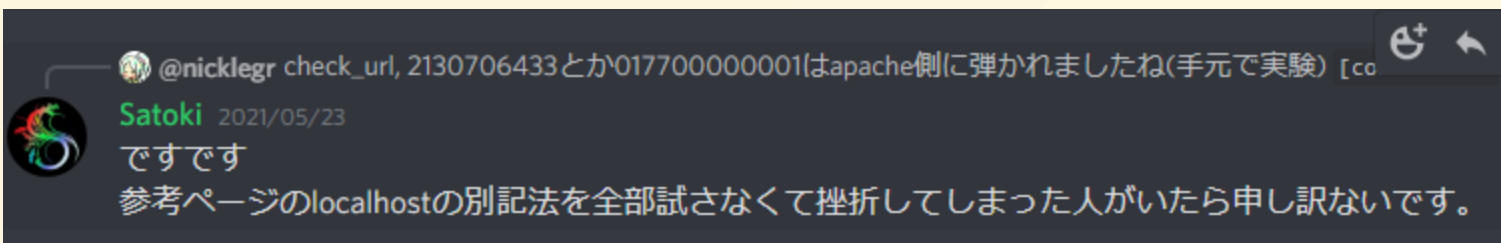
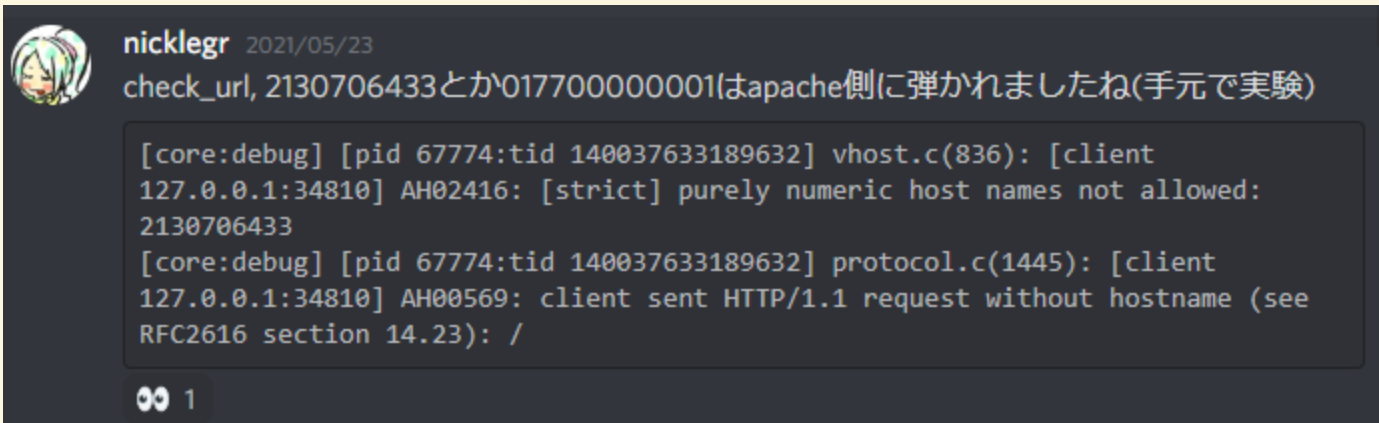
解き方

実は、10進数以外にもIPアドレスを表現することは可能。

- 16進数 → 0x7F000001
- 8進数 → 017700000001

この2つのいずれかを指定してあげることによってフラグを取得できる...と思ったら、16進数表現以外はダメだったみたいです。

解き方



何故その指定方法でうまくいくのか？

RFC3986にそのことについて触れている箇所がある。

“ IPv4address のための URI 構文は一般的なドット付き数字形式の IPv4 アドレスリテラルのみを認めているが、URI を処理する多くの実装は**文字列リテラルを実際の IP アドレスへと変換するために、gethostbyname() や inet_aton() のような、プラットフォーム依存のシステムルーチンを利用**する。残念ながら、そのようなシステムルーチンは、しばしば Section 3.2.2 にて記述されるものよりはるかに大きな集合の形式を認め、そして処理を行う。 ”

何故その指定方法でうまくいくのか？

“ 例えば、多くのアプリケーションは**三つの番号のドット付き形式**を認め、この場合、最後の部分が **16 ビットの量**として解釈され、**ネットワークアドレスの最も右 2 バイト**に置かれる(例えば、クラス B ネットワーク)。

同様に、**二つの番号のドット付きの形式**は、最後の部分が **24 ビットの量**として解釈され、**ネットワークアドレスの最も右 3 バイト**に置かれ(クラス A)、**(ドットなしの) 単一の数字**は **32 ビットの量**として解釈され、**直接ネットワークアドレスに格納される事**を意味する。

”

何故その指定方法でうまくいくのか？

更に混乱させる事に、いくつかの実装では、C 言語にて規定されるように、**ドット付けされた部分が 10 進、8 進、16 進として解釈される事も認める** (すなわち、先行する 0x や 0X は 16 進を意味する; 先行する 0 は 8 進を意味する; さもなくば、その数は 10 進であると解釈される)。

[出典：RFC3986 7.4 珍しい IP アドレス形式](#)

まとめ

SSRFについて過去に出題されたケースについて紹介。

GhostKingdom は残念ながら問題ファイルが見つからないが、`check_url` は作問者のgithubに問題に使われたファイルが残っている。

環境を構築する必要があるが、構築さえできれば今からも問題にチャレンジすることが可能。

また、SSRFには他にも様々なテクニックが存在する。
他のCTFでもいくつか出しているケースがあるので興味があれば調べてみることをおすすめする。