Webサイト解析訓練

－クローキング技術を利用したwebサイトの解析－

(座学)

訓練概要

- 目的
  - 攻撃対象を限定する攻撃の仕組みを理解し、マルウェア解析を行う足掛りとする
- 種別
  - マルウェア解析
- 前提知識
  - JavaScriptやhtmlなどのwebプログラミングの基礎知識・JavaScriptの難読化解除知識・ネットワークツールの知識
- 習得技術
  - 攻撃対象を限定する攻撃の仕組みの理解、解析手法の習得

目次

1. Webサイトを利用した攻撃
2. クローキングを利用した改ざんされたWebサイトの解析

1. WEBサイトを利用した攻撃

Webサイトを利用した攻撃

攻撃の目的

マルウェアを感染させる
情報窃取

攻撃の対象

不特定多数
特定のユーザ・組織

組み合わせ

攻撃の目的  攻撃の対象

参考:最近の傾向

- 正規Webサイトが、ドライブ・バイ・ダウンロード攻撃を行う悪性Webサイトの踏台となる事例が多発
  - 正規Webサイト本体を改ざん
  - 外部Webサイトから読み込んでいるコンテンツ（広告コンテンツなど）を改ざん
- 攻撃対象を限定した攻撃
  - 水飲み場攻撃

水飲み場攻撃 概要

攻撃の目的

マルウェアを感染させる

攻撃の対象

特定のユーザ・組織

背景

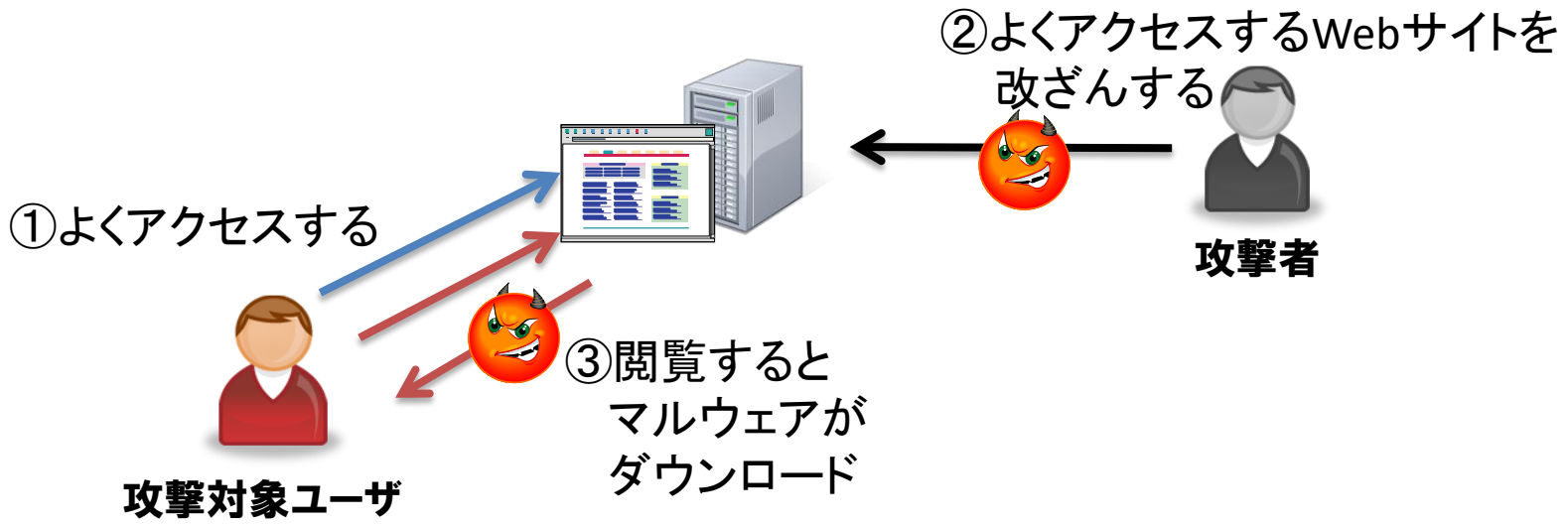
2012年RSAセキュリティにより発表

ライオンが水飲み場のそばで獲物を待ち伏せ

英名：watering hole attack

2013年頃から日本でも確認

水飲み場攻撃概念図



①攻撃対象ユーザ(以下ユーザ)には、
よくアクセスするWebサイトがある

②攻撃者が、ユーザがよくアクセスするWebサイトを
推測または観測により特定し、改ざんする

③ユーザが、改ざんされたWebサイトにアクセスすると、
攻撃が行われ、マルウェアが、ダウンロード・実行される

水飲み場攻撃の特色

基本的な手段

攻撃対象がよく利用するサイトを改ざん
ドライブ・バイ・ダウンロード攻撃

＋

新たな手段

クローキング(特定ユーザの判定)

↓

特色

アンチウイルスベンダなどに発見されにくい

クローキング(cloaking)

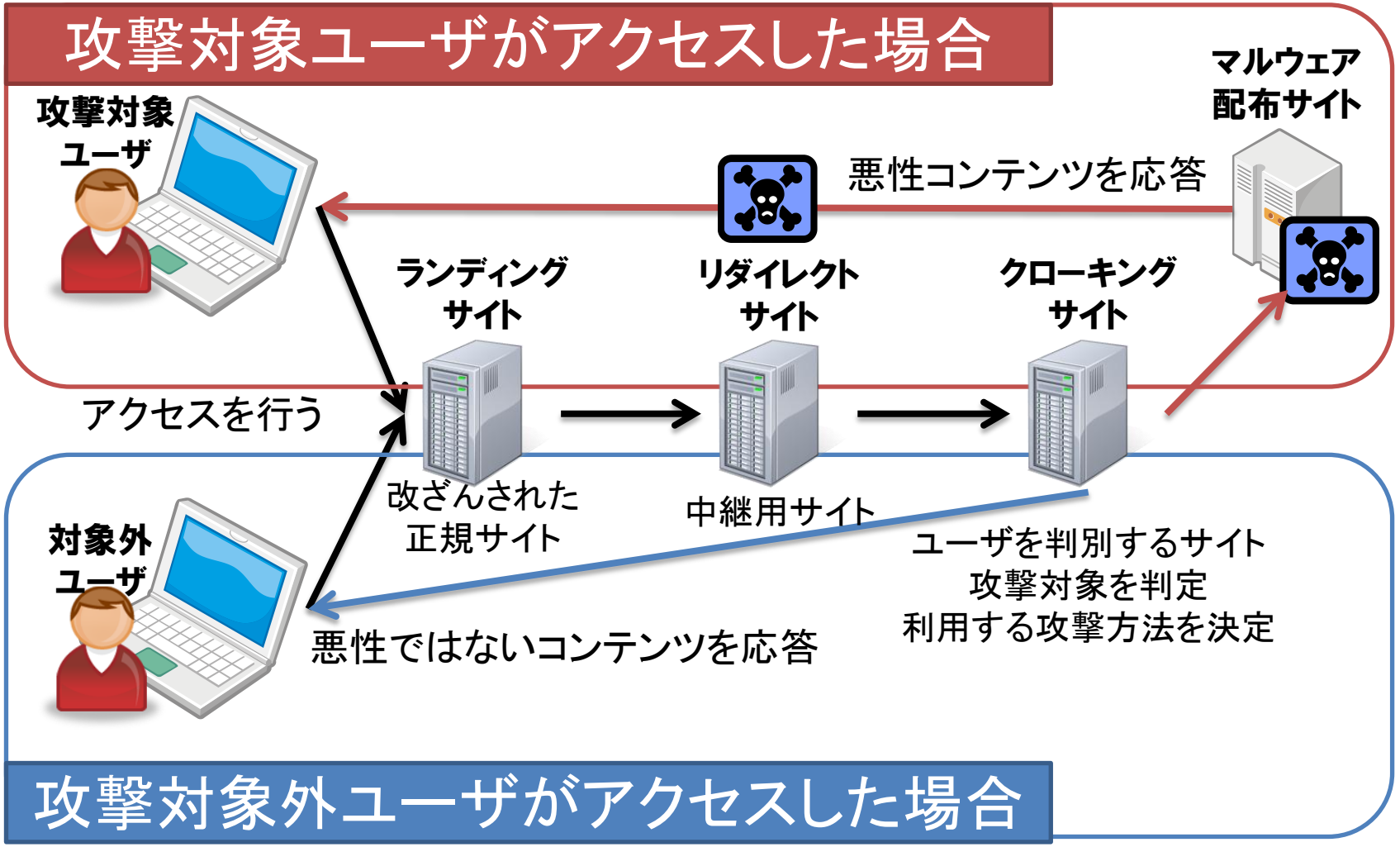
概要

Webサイトにアクセスしてきた
クライアントを判別し
クライアントによって異なる応答を行う技術

Webサイトの改ざんに利用

解析者やクローラーからの
アクセスを判定し、攻撃コードの検知回避

攻撃対象を限定した攻撃の機能的概念図



クローキングを利用した サイト改ざんの構成サイト

ランディングサイト

正規サイトを改ざんし攻撃の入口サイトとして
他のサイトへリダイレクトを実行

リダイレクトサイト

踏み台としてリダイレクトを多段で実行

クローキングサイト

クライアントの環境を識別
攻撃を行うか、どの脆弱性を利用するか判定

マルウェア配布サイト

実際にマルウェアを配布し攻撃を実行

クローキングの実装

サーバサイド

PHP、ASPなど

取得する情報
IPアドレス・ドメイン名・アクセス回数や間隔・
HTTPヘッダ（リファラ，ユーザエージェント等）

クライアントサイド

JavaScript、VBScript、Java Appletなど

取得する情報
OS・ユーザエージェント・ブラウザ・
プラグイン・リファラ・Cookie

2. クローキングを利用した改ざんされたWEBサイトの解析

改ざんされたWebサイトの特徴

外見

改ざんされる前のWebサイトと
表示上の大きな違いはない

サイズ0pxのiframeの挿入

悪性サイトや、自ドメインと異なるサイトへの
リダイレクトなどを実行

難読化されたJavaScriptの挿入

1行の文字列の長さが数百文字以上ある行、
関数、8/10/16進数、Base64、%#!:[,
などが多用

クローキングの解析の流れ

様々な条件でアクセスしてくる対象が
攻撃対象の条件に一致しているかを判定

解析に利用する端末の条件を攻撃条件に
一致させないと攻撃コードを入手できない

攻撃条件の解析

クローキングの解析時の注意(1/2)

サーバサイドクローキング

クローキングの実行結果しか
わからないので詳細な解析は難しい

クライアントサイドクローキング

クローキングのソースコードが
クライアントに読み込まれるので解析可能

クローキングの解析時の注意(2/2)

IPアドレスクローキング

既に攻撃を行ったIPアドレスから、
2回目以降アクセスしても
攻撃サイトに遷移しない場合もある

プロキシ、ローミングサービス、
モバイル回線、別契約回線などを利用

解析手順

1. 解析対象ページを取得
2. 解析対象ページより実行コードを特定
3. 難読化されている実行コードを解読
4. リダイレクト先を特定
5. リダイレクト先より解析対象ページを取得・解析
6. 攻撃対象判定部分を特定
7. クローキングによる判定の回避
8. 最終的なアクセス先を特定

解析対象ページを取得

取得方法

Webサイトにアクセスしhtmlソースを取得

Webサイトにアクセスしパケットをキャプチャ
キャプチャファイルからデータを取得

解析対象ページより実行コードを特定

サイズ0pxのiframe

<iframe src="hxxp://google-analyz .cn/ count.php?o=1" **width=0 height=0** style="hidden" frameborder=0 marginheight=0 marginwidth=0 scrolling=no></iframe>

難読化されたJavaScript

正規スクリプトの軽量化の可能性

本来のページにないコード

コンテンツを別途保存している場合は比較可能

難読化されている実行コードを解読

難読化されたJavaScriptの特徴

1行の文字列の長さが数百文字以上ある行、
関数名(function, eval, unescape,charCodeAt,
fromcharCode, replace, split など)、
8/10/16進数、Base64、%#!:[, などが多用

正規のスクリプトの可能性

難読化は高速化としても利用される
実際に解除してみないとわからない

難読化解除の詳細は別訓練
(「JavaScript難読化解除訓練」)で実施

リダイレクト先の特定

調査方法

難読化解除したコードを調査
通信をキャプチャして通信先を調査

解析対象ページの取得・解析を繰り返す
　多段リダイレクト

攻撃対象判定部分を特定

クライアントの情報を取得

ユーザエージェント(navigator.userAgent)

OSの種類・バージョン
ブラウザの種類・バージョン

プラグイン(navigator.plugins)

Flashのバージョン
PluginDetect.getVersion("Flash")
IEではActiveXObject("ShockwaveFlash.ShockwaveFlash")
Adobe Readerのバージョン
PluginDetect.getVersion("AdobeReader")
Javaのバージョン
PluginDetect.getVersion(“Java”, jarfile, verifyTagsArray)

HTTPリファラ

実際の例

IE8ユーザだけをターゲットにするJavaScript の例

```
if(navigator.userAgent.toLowerCase().indexOf("msie 8")!=-1)
{
        exp();   SetInterval("word_()",4000);
} else {
        ok();   SetInterval("word_()",4000);
}
```

その他のバージョン情報を取得する例

```
ActiveXObject("ShockwaveFlash.ShockwaveFlash.7");
navigator.plugins["Shockwave Flash"].description;
navigator.userAgent;
PluginDetect.getVersion("Java", "include/getJavaInfo.jar");
PluginDetect.getVersion("AdobeReader");
```

クローキングによる判定の回避

回避方法

判定の条件部分をアクセスする端末で偽装

プロキシを仲介し情報を改変してアクセス

クライアントの情報を偽装　　など

実行コードを改変し判定コード全体を回避

最終的なアクセス先を特定
実際の攻撃コードを取得