MWS Cup 2014

事前課題1 「Drive-by Download 攻撃解析」 回答例

1. 出題の意図

MWS [1] で提供している研究用データセット [2] に収録されている D3M (Drive-by Download Data by Marionette) [3, 4] は、高対話型のクライアント型ハニーポット Marionette [5] でドライブバイダウンロード攻撃を検知した際の悪性通信や、その際に感染したマルウェアおよびマルウェアが行う通信が収録されたデータセットである。本課題は二問で構成されており、攻撃の動向把握のため、Drive-by Download 攻撃における解析妨害、検知回避技術の一つであるクローキングについて出題した。一問目は、クローキングに関する公開情報を参考に調べた上で、攻撃者の視点に立ってクローキングを行う意味について考察する問題となっている。二問目は、一問目の内容を踏まえ、反対に対策者の視点に立って、クローキングを回避しつつ悪性 Web サイトを検知する方法を考案する問題となっており、いずれも実用的な知識の獲得を目的としている。

提供している D3M の分析や本課題を通じて、Drive-by Download 攻撃を複雑化・巧妙化させる一要因であるクローキングの存在や仕組みを理解し、今後の対策技術研究の検討や攻撃対策能力の向上が促進されることを期待する。

2. 出題内容

Drive-by Download 攻撃を行う悪性 Web サイトについて以下の設問に答えよ。

設問1:

Web サイトにアクセスしてきたクライアント(ユーザ)の情報に基づき、攻撃者が設定した条件に一致した特定クライアントには悪性コンテンツを応答し、それ以外のクライアントには悪性ではないコンテンツを応答する、クローキングという手法がある。

Drive-by Download 攻撃を仕掛ける悪性 Web サイトの中には、クローキングによる解析妨害や検知回避をする Web サイトが存在する。このような Web サイトは、クローキングによってどのような情報を、どのように取得し、解析妨害や検知回避に利用しているか、攻撃者におけるメリット(例:発見のされにくさ)やデメリット(例:攻撃失敗の可能性)という観点から考え、解答用紙 1 ページ以内で述べよ。なお、文章だけでなく適宜図を用いて説明してもよい。

設問2:

正規 Web サイトが、Drive-by Download 攻撃を行う悪性 Web サイトの踏台となる事例が多発している。例えば、正規 Web サイトが改ざんされたり、外部 Web サイトから読み込んでいるコンテンツ(例えば、広告コンテンツ)が改ざんされたりすること等が挙げられる。このような改ざんによる Drive-by Download 攻撃に、設問 1 のようなクローキングが使用されると、正規 Web サイトが改ざんされたことを発見するまでに時間を要する場合がある。そこで自身の管理下にある Web サイトが Drive-by Download 攻撃に加担していないか検知する方法を考え、解答用紙 1 ページ以内で述べよ。なお、文章だけでなく適宜図を用いて説明してもよい。

3. 回答例

MWS Cup 2014 参加チームから提出された事前課題レポートより、一部優秀であった解答内容を抜粋・要約する。【】内はチーム名を示す。

3．1　設問1

【人海戦術チーム】

クローキングを行う方法として、(1)サーバサイドで行う方法と(2)クライアントサイドで行う方法の 2 つが考えられる。

サーバサイド	PHP や ASP などのサーバ側で実行されるプログラムを用いて，下記の情報を取得する ・IP アドレス、ドメイン名 ・HTTP ヘッダー(リファラー，ユーザエージェント等) ・アクセス回数や間隔
クライアントサイド	JavaScript や VBScript，Java Applet などのクライアント側で実行されるプログラムを用いて，下記の情報を取得する ・OS，ユーザエージェント ・ブラウザ，プラグイン，リファラー　} などの種類やバージョン ・Cookie

サーバサイド及びクライアントサイドで取得できる情報に関してメリット・デメリットとしては，以下が考えられる.

サーバサイド	メリット	・解析者やクローラーからのアクセスなどを判定して、無害なページを表示することで解析妨害や検知回避を行える ・検索エンジンなどのクローラーに対して正常なページを応答することで検索結果に表示させることが出来る。これにより、多くのユーザを攻撃サイトに誘導することができる。 ・特定の組織やユーザに絞った攻撃が可能になる。また、攻撃対象が絞られることで、発覚を遅らせることが出来る
	デメリット	・コストが高くなる(条件の精査、テスト、メンテナンスなど) ・条件により攻撃対象が絞り込まれるため、攻撃の機会が減る可能性がある ・偽装可能な情報も存在するため、攻撃回避もしくは動的解析される可能性がある **・サーバサイドではアクセスしてきたクライアントに関する得られる情報が少ない(高田追記)**
クライアントサイド	メリット	・ユーザの端末に存在する脆弱性に合致する攻撃コードを配信することで攻撃成功率を上げることができる ・該当する脆弱性が無い場合は、正常なページに遷移させることで、攻撃失敗を回避及び解析回避することができる
	デメリット	・攻撃対象の環境(Cookie や JavaScript の有効判定等)によっては、攻撃が成立しない可能性がある。 ・条件により攻撃対象が絞り込まれるため、攻撃の機会が減る可能性がある ・クローキングを実装したスクリプトがユーザ側に見えてしまうため、クローキングの内容を容易に解析されてしまう

下記に D3M2014 に含まれているクローキングの例を示す。

- JavaScript で IE のバージョン情報を取得して、IE7 であれば攻撃を行う例

```
if(navigator.userAgent.toLowerCase().indexOf("msie 7")==-1)
{
        MD2C();  SetInterval("word_()",4000);  // 攻撃に遷移
} else  {
        ok();  SetInterval("word_()",4000);    // 攻撃に遷移しない
}
```

- 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");
```

【フレックス・スヴェンソン】

＜メリット＞

(1) 悪性サイトであると検知されにくい

クローキングを使用した悪性サイトは、検知システムやクローラに対しては、無害なコンテンツを提供するため、検知されにくい。また、検知されにくいことにより、検体を入手しにくいため、解析が難しい。クローキングに利用される情報は、Web サイトへのリクエストパケットや JavaScript の navigator オブジェクトを参照することで取得できる。取得した情報をもとに、以下の手法により、クローキングを行う [6]。

(A) ブラウザフィンガープリンティング

OS、Web ブラウザ、プラグインの種類に応じて応答の可否を決定する。このとき、標的とするアプリケーションが搭載されたクライアントを選択的に攻撃する。攻撃者がアプリケーションの脆弱性を利用した不正なコードを埋め込んでいた場合、アプリケーションのバージョンが異なるとコードが正常に実行されず、予期せぬ不具合が発生する可能性がある。クライアントがこの不具合を検知することで、悪性サイトを検知できる可能性がある。しかし、クローキングにより、対象を限定して不正なコード配信することで、不正なコードの不具合が原因で検知される可能性が低くなる。また、D3M 2014 の pcap ファイルから難読化された JavaScript を抽出し、Caffeine Monkey[7]を使用して難読化を解除した。また、難読化を解除した JavaScript の一部を以下に示す。

```
if(navigator.userAgent.toLowerCase().indexOf("msie 7")==-1)
```

上記の処理は、アクセスしたユーザの User-Agent が IE7 か否かで処理を変化させている。この処理は、ブラウザフィンガープリンティングの 1 種であると考えられる。なお、サーバ側の設定ファイルやモジュールによるクローキングは、D3M 2014 から検知できない。

(B) リファラ検査

攻撃者は、改ざんした Web サイトの URL がリファラに設定されているか確かめることでアクセスの可否を決定する。このとき、攻撃者が意図するリファラが付与されている場合のみ応答する。検査目的で悪性 Web サイトにアクセスする際、攻撃者の意図する経路で悪性 Web サイトにアクセスする必要があるため、検査が困難だと考えられる。

(C) クライアント IP 検査

クライアントの IP アドレスに基づいて、特定 IP アドレスやネットワークアドレス単位もしくは GeoIP のカントリーコード単位でフィルタリングを行う。多くのセキュリティベンダや研究機関の IP アドレスは攻撃者によってブラックリスト化されているといわれており、上記のフィルタリングに用いられる。また、同一、または隣接した IP アドレスで検査し続けると、攻撃者側の IP ブラックリストに掲載される場合がある。

(2) 攻撃の成功率が上がる

クローキングにより、攻撃が成功する可能性が高い標的に対してのみ攻撃を試みるため、攻撃の成功率が上がる。

＜デメリット＞

(1) 攻撃の機会が減少する可能性がある

クローキングが発覚すると、検索エンジンからペナルティを受け、Web サイトが検索にかかりにくくなることにより、サイト訪問者が減少し、攻撃の機会も減少する。

(2) 攻撃の幅が狭まる

クローキングを用いた場合、特定の条件を満たす対象にのみ攻撃するため、不特定多数への攻撃には不向きである。

【n00b】

The true motivation for using cloaking is to avoid detection from analyst which makes your exploit site's lifespan longer and increases the possibility of infecting users. Hiding your exploit also slows the discovery of zero-day attacks and keeps researchers from learning new exploit hosting methods. Since exploits generally impact only a targeted vulnerable component of a system it is advantageous for attackers to only attack systems that have that vulnerable component installed and avoid unnecessary interaction with systems that don't have the vulnerability.

The information gathering method called fingerprint, a process in which the browser and plugin versions of the host machine are acquired, is commonly used in conjunction with cloaking because it can find evidence of emulation, a technique used by analyst. Also, it is normal for malsites to check for the use of virtual machines, debuggers, and static/dynamic analysis programs as they are commonly used by researchers. Crawlers used by analyst to find malware will often attempt shortcuts in order to load pages faster, yet cloaking sites can detect these shortcut attempts. [8] gives an example of code that will only serve malware if all the images on the page are loaded. Crawlers often skip this step to save time and can be easily detected. If signs of researchers or malware analyst are found the host will likely be redirected to a decoy, benign webpage while users exhibiting normal behavior with the necessary versions of browser or plugins will be redirected to the malware distribution site.

Client-side cloaking techniques like those mentioned above can be accomplished in different ways but are commonly done by a combination of fingerprinting and JavaScript code that searches for signs of crawlers or emulated browsers. PluginDetect [9] is a JavaScript library commonly used by exploit kits in order to determine what versions of plugins the browser is using. PluginDetect's harvested information can be used to determine when cloaking should be used. Server-side cloaking, a technique which looks at the IP address of clients, can also be used to avoid detection because it can block IP's associated with analyst.

Some disadvantages of cloaking include limiting your attack area and increasing the difficulty of coding required to create and inject the exploit. Users can download specific programs or make fake registry entries in order to fool malware into thinking they are a researcher or analyst, subsequently tricking a cloaking site into not serving malware.

【GOTO Lobe with m1z0r3】

クローキングに関しては、サーバ側およびクライアント側の 2 つのクローキングに分かれている。

・サーバ側のクローキング

IP アドレスや HTTP ヘッダといったクライアント側の情報を PHP 等のサーバサイドプログラムにより取得している。そして、取得した情報に基づいてクライアントがセキュリティベンダーやセキュリティに関する研究所である場合、悪性ではないコンテンツを応答し、そうでない場合、悪性コンテンツを応答することで解析妨害、検知回避を実現している。

また、D3M2014 のデータセットに含まれる URL にアクセスし、Windows では悪性のあるコードを含むサイトが表示されるが Ubuntu では何も表示されないというように、クライアントで使用されている OS に応じて処理の変更を行っている事例も確認した。

・クライアント側のクローキング
クライアント側のクローキングについては、OS 情報、ブラウザのバージョン、言語、プラグインの種類やバージョン、ActiveX コントロールの有無などの情報を、JavaScript コード(navigator.userAgent, navigator.plugins, navigator.browserlanguage など)から取得している。これによって、try/catch,if/else を用いることで、シェルコードの実行に条件を付ける。

この手法を行う上での攻撃者にとってのメリットやデメリットについて考えると、メリットとしては、特定のターゲットを狙うことができる。また、良性コンテンツを応答することによってセキュリティベンダーやセキュリティに関する研究所は悪性コンテンツを早期かつ簡単に解析することができない。さらに、様々なユーザ情報を用いることができるため、コード作成には柔軟性・多様性がある。デメリットとしては、セキュリティベンダーやセキュリティに関する研究所がクローキングを行う Web サイトにアクセスする際、ブラックリストされていない IP アドレスもしくは HTTP ヘッダを利用した場合、攻撃に該当したクライアントだと判断し，悪性コンテンツを送ってしまう恐れがある。

【urandom】
以下にクローキングの手法として考えられるものを挙げる。

・IP アドレスを取得する
クライアントの IP アドレスを取得して、サーバーサイドのアプリケーションで 一部の IP アドレスにのみ悪意あるコンテンツを返答するように制限したり、 特定の IP アドレス（例えばセキュリティ会社などが利用するレンジ）には、 悪意あるコンテンツを表示しないといったクローキングをすることで、 解析者に悪意あるコンテンツを見せないようにすることができる。 これにより、Web サイトが悪意あるコンテンツを配信している事が発覚しにくくなる。
デメリットとしては、セキュリティ会社などが利用するレンジをブラックリストで弾くという方法は、まず制限対象のレンジを特定する必要がある点が挙げられる。 また、Tor や VPN Gate といったソフトウェア等でアクセス元 IP アドレスを偽装された場合には効果が無い。

・JavaScript によって DOM の変化を調べる
解析者は、脆弱な特定の Web ブラウザのように振る舞うエミュレーター（クローラー）を用いて、 悪意あるコンテンツの発見を試みる。そこで、悪意あるコンテンツを配信する Web サイトは、 結果がブラウザに依存するような DOM 操作を行う JavaScript 等をまず配信してから JavaScript で DOM を調べて、意図通りに変更されていれば エミュレーターではないと判断して悪意あるコンテンツを配信し、そうでなければ攻撃を停止するといった振る舞いが出来る。 これにより、エミュレーターによる検知を回避でき、解析者に発見されにくくなると考えられる。

・CSS が正しく DOM に反映されているか調べる
エミュレーターの一部は実際に Web ページを描写しない場合があり、その場合には CSS をロードしてもそれが正しく DOM に反映されない可能性がある。 しかし、一般的なブラウザの場合は、CSS をロードしたならばその結果が DOM に反映される。 悪意ある Web サイトは、まず CSS を読み込み、その後に DOM に CSS が反映されているか検査するような JavaScript を実行して、 もし DOM に CSS が反映されていなければ処理を停止するといった振る舞いが出来る。

デメリットとしては、この方法はエミュレーターに対して一定の効果があるが、一方で攻撃対象である脆弱なブラウザとそうでないブラウザを識別することはできない点が挙げられる。

・Web ブラウザのパーサの違いを利用する

Web ブラウザによる JavaScript パーサ等の違いを使って、実行したい環境ではパーズできるが、実行したくない環境ではパーズできないような JavaScript 等を使う事で、攻撃対象を一部の環境に絞る事が出来る。

デメリットとして、実行できなかった（パースに失敗した）場合、その痕跡がエラーとしてデバッグコンソールから容易に特定できる上、パーザの問題を使う事から少なくともその部分は難読化などができない。その為、解析者は、あるブラウザでのみパーズに失敗するようなパターンを集めておいて、そのパターンを調べることで悪意のある Web サイトを検知できる。

・JavaScript から脆弱ではないブラウザが持つ関数を実行する

console.timeStamp()といった、特定のブラウザには存在するが、脆弱のブラウザには存在しない関数を呼んで try-catch することで、脆弱なブラウザでだけ悪意のあるコンテンツを呼び出し、他のブラウザでは 悪意のない Web サイトとして振る舞うことができる。多くの場合、脆弱なブラウザはバージョンが低いなど 古い ブラウザであるので、

```
try {
        console.timeStamp();
} catch (e) {
        // malicious code
}
```

というような方法になる。すると、脆弱ではないブラウザからは正しくconsole.timeStamp()といった関数が実行され、悪意あるコードは実行されない。また、ブラウザが持つ関数を利用したブラウザの特定は、広告会社などが普通に行っているので、ブラウザを特定するような JavaScript があるということだけで、悪意あるコンテンツと判断することは難しい。

３．２　設問2

【人海戦術チーム】

Web サイトが Drive-by Download 攻撃に加担していないか検知する方法は、大きく分けて (1)サーバ内部で検知する方法、(2)外部からのアクセスで検知する方法、(3)セキュリティ装置で検知する方法の 3 つが考えられる。

(1) Web サーバ内部での検知する方法

Web サーバ内のコンテンツや設定ファイル、ログを、下記の条件で検査して総合的に判断する。

① コンテンツに対して改ざん検知
- ハッシュ値による検知
- iFrame が追加されていないかを検査
- 存在しないコンテンツが追加されていないかを検査

② コンテンツに対して以下の項目をヒューリスティックに検査
- 改ざんで利用される既知の特徴的な文字列パターンやコメント
- 関数名(function, eval, unescape, charCodeAt, fromcharCode, replace, split など)、8/10/16 進数、Base64、%#!:[, などの有無や出現回数
- 1 行の文字列の長さが数百以上ある行の有無

③ コンテンツに対してシグネチャを利用して検査
- アンチウイルスソフトで検査

④ 設定・ログファイルに対して検査
- .htaccess などの設定ファイルに変更がないか検査
- コンテンツ更新用の FTP アカウントが不正利用の検査
- ログに想定外のサーバからのリファラーが大量にないかを検査

(2) 外部からのアクセスで検知する方法

外部から Web サーバのコンテンツを閲覧した際のクライアントの挙動から検知する。

① クローキングの判定に利用される情報を偽装できるクローラーを作成し、ブラウザやユーザエージェント、プラグイン名などを変更・組み合わせたパターンを大量に作成してクロールを行い、他のサイトにリダイレクトするなどの不審挙動がないかを判定して検出する(クライアントハニーポットのようなもので、項目をクロールする)

② 既に攻撃を行った IP アドレスについては、2 回目以降アクセスしても攻撃サイトに遷移しないようにクローキングされている場合、プロキシや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する

(3) セキュリティ装置(ソフトウェアも含む)で通信内容を監視して検知する方法

Web サーバとクライアントの間(Web サーバ内も含む)に設置した IDS や WAF などのセキュリティ装置で、下記のようなシグネチャを用いて検出する。

- 既知の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
- (1)の②の項目など

【フレックス・スヴェンソン】

以下に、クローキングを検知する方法と、それぞれの検知方法が対象とするクローキング手法を示す。

(1) 定期的に Web サイトのバックアップとの差分をとる

定期的に Web サイトのバックアップとの差分を取ることにより、攻撃者から不正な改変を受けていないか検査する。また、Tripwire[10]や inotifywait といったファイルの改変ツールを利用することで、Web サイトの改ざんを検知できる。このとき、以下の 3 点に注目して検査する。

(A) Web サイトに不正なコードが挿入されていないかを検査する。

本検査により、クローキングによって動作が動的に変化する Web サイトであっても、直接不正なコードを埋め込む攻撃であれば検知できる。

(B) 不正な設定ファイルが存在しないか検査する

攻撃者は、攻撃対象 Web サーバに.htaccess などの設定ファイルを不正に配置することで、Web サイトにアクセスしたクライアントを不正な Web ページにリダイレクトさせることができる。また、.htaccess などの設定ファイルは、アクセスしたユーザの情報に基づいて動作を変更できるため、クローキングに利用される場合がある。これを検知するために、Web サーバに不正な設定ファイルがないか、また改ざんされていないかを検査する。設定ファイルを利用した攻撃は、Web ページに直接コードを埋め込まないため、(A)の手法では検知できないが、本検査により検知できる。

(C) 不正なモジュールが組み込まれていないか検査する。

Darkleech[11]と呼ばれる Apache のモジュールを利用したクローキング手法が存在する。Darkleech は、アクセス元のクライアントの情報を参照し、設定した条件を満たす場合のみ、Web サイトに不正なコードを挿入し、配信するモジュールである。このような不正なモジュールを利用した攻撃を検知するために、 Web サーバに不正なモジュールが組み込まれていないか検査する。

(2) ユーザ情報を変更して Web サイトにアクセスし、表示内容を比較する

攻撃にクローキングが利用されると、ユーザの情報に応じて配信されるコンテンツが異なる。このため、異なるユーザでアクセスした際に配信されるコンテンツが異なっていると、Web サイトが攻撃に利用されている可能性が高いと判断できる。よって、IP アドレスや User-Agent などのユーザの情報を様々に変えて自身の Web サイトにアクセスし、配信されるコンテンツが異なっているか検査することで、Drive-by Download 攻撃に加担していないかを検知する手法が考えられる。

(3) 定期的に参照先 URL のパターンを比較する

Web サイトに配置しているリンクやリンクを含むコンテンツ(広告など)を参照し、参照先 URL を定期的に比較する。例えば、広告を掲載している場合、その広告が参照するはずの URL とは別の URL が参照された場合、広告が改ざんされている可能性がある。

ブログサービスのように、設定ファイル等の操作に制限がある場合、上記の(1)-(A)、(2)、および(3)を実施する。

【GOTO Lobe with m1z0r3】

Webサイトにおいては、Webサーバ管理者にて管理する静的なコンテンツと外部Webサイトから読み込んでいる動的なコンテンツの2つの種類が存在する。各種類のコンテンツに対して下記のように対策を講じると考えている。

(1)[正規 Web サイト(ホームページ)が改ざんされた場合の対策]

静的なコンテンツの検知手法として、サイトの HTML コンテンツのハッシュを記録しておく。その後、10 分ごとに同様なコンテンツのハッシュ値を改めて取得し、保存しておいたハッシュ値と比較する。二つのハッシュ値が一致しない場合、Web サイトが改ざんされたと判定する。また、Web サーバ管理者の視点から Web サイト攻撃の検出ツール iLogScannerV3.0[12]を用いて Web サーバログの解析を行うことにより、Web サイト攻撃の早期発見に役に立つと考えている。しかし、この手法は、管理者の管理下にある部分に対してのみ有効である。サイト内の広告部分等、管理者の管理できない部分は以下に示す手法を用いる。

(2)[外部 Web サイトから読み込んでいるコンテンツが改ざんされた場合の対策]

ハニーポット（サンドボックス環境)を利用し、外部 Web サイトから読み込んでいるコンテンツを検知する。シェルコードを含むなど、怪しいファイルをダウンロードしたら感染しているとみなす。しかし、この手法では多種多様な脆弱性の組み合わせを網羅することはできない。より高精度に検知するために、サンドボックスが生成した PCAP データから HTTP ステータスコード（例えば、301、302）によってリダイレクトに関する HTML と JavaScript コンテンツを抽出し、リダイレクトチェーンを構築する。リダイレクトチェーンに属するコンテンツに不審な長文列(文字数が 128 以上且つ空白の数が文字列の 5%より少ない)の存在や iframe タグの hidden 属性等の特徴が存在した場合、外部 Web サイトが改ざんされた可能性が高いと判定し、そのコンテンツに関する情報をサイト管理者に送付する。

このとき、ハニーポット（サンドボックス環境）は、OS やブラウザのバージョンが異なる環境を用意し管理対象のサイトにアクセスさせる。

【n00b】

One way to determine if your website is hosting a malware injected ad or another form of injection is to use a honeyclient to test your website. Honeyclients can detect JavaScript injected into a webpage which is the common attack vector of malicious ad injection. However, it should be recognized that webmasters cannot control all the possible ads given to their page by third party ad vendors due to ad syndication. In this situation it is important to check the credentials of the ad network used on your site or you may be forced to police your site endlessly as new ads will always come and go from the ad network.

If ad syndication is used one method of detection is proposed in [13]. The method of detection is done by analyzing URL features along redirection chains of ads. The features obtained, number of ad networks an ad appears on, etc., are not conclusive unless they are grouped by

content and neighboring nodes and then analyzed concurrently. This method must be done by crawling your website and using various personalities, browser and plugin settings, as cloaking may be deployed along with selective redirection based on client system. Though, this method can be used to detect malicious ads it is more beneficial to use a highly trusted ad publisher who doesn't use ad syndication. Other vectors of inject can be monitored by checking for malicious iframes or other suspicious redirects that appear on your site.

【TDU ISL with 親方 リターンズ】

現時点で自身の管理下にある Web サイトが改ざんされているかを確認する方法として、IPA では以下の3つの手法を挙げている。

(1) サーバ上の HTML ソースと、手元に有るオリジナルの HTML ソースを比較する
(2) サーバ上の HTML ソースをセキュリティソフトでスキャンする
(3) ftp アクセスログを確認する

(1)の方法では、サーバ上の HTML ソースが頻繁に更新される場合、オリジナルの HTML ソースがその都度更新されない可能性が考えられる。そのため厳密には、サーバ上の HTML ソースのスクリプトの中で、不正なリダイレクトやクローキングの命令が記述されていないかを確認する必要があると考えた。また、(3)の方法は、加えて管理者アカウントのアクセスログと書き込みログの比較することで、より検知精度を高めることができると考えた。以上のことを踏まえると、以下の3点の手法が有効であると思われる。

(1) サーバ上の HTML ソース中で、スクリプト等に不正なリダイレクトやクローキングの命令が記述されていないかを確認する
(2) サーバ上の HTML ソースをセキュリティソフトでスキャンする
(3) ftp アカウント及び管理者アカウントのログを確認する

これらを即日もしくは定期的に確認すれば、Web サイトの改ざんを早期に発見する可能性が高くなると言える。しかし、問題として即時性が無い事と誤検知が少なからず発生する恐れがある。そこで、Web サイトの改ざんをリアルタイムに発見する方法として、以下の 2 つのツールを利用した方法が有効であると考えた。

① Tripwire
② inotifywait

これらのツールは、指定したファイル群を監視しファイルイベントを検出、そのイベントをメールで通知の機能があるので、HTML ファイルや Web サーバの設定ファイルを監視することで、リアルタイムで改ざんを検知することができる(オープンソースの Tripwire はバッチ処理による監視なので、完全なリアルタイム検知とは言えないが、IPA の対策よりは早急な発見が期待できる)。

ただし、Web サイトを頻繁に更新する企業では、イベント数が多くなるため誤検知が多発する恐れがある。そこで、そのような企業では更新を行う時間を定め、その時間内のログの収集をしないもしくは収集してもメールでの通知は行わないといった方法で誤検知や可用性の低下を防ぐことが有効であると考えた。

4. 総評

本課題は、攻撃者および対策者両方の観点からクローキングの仕組みや構造を理解した上で、効果的かつ実用的な対策技術を考案する課題であった。

設問1は、多くのチームが図表を用いて体系的にクローキングについてまとめていた。一方で、主にクローラーに対するクローキングである Black Hat SEO について記述しているチームがいくつか見られた。攻撃者の立場で考えると、攻撃者はセキュリティに従事する解析者、技術者、研究者等による攻撃の検知、対策を防ぐため、クローキングのような解析妨害・検知回避技術を使用すると考えられる。すなわち、クローキングの対象は、クローラーだけではないと言える。クローキングは、いかにセキュリティに従事する者を検知し、回避しつつ、攻撃者にとって利益となる攻撃対象（主に一般ユーザ）に対してのみ攻撃できるかがポイントとなる。その点を踏まえた回答として、従来の主に仮想環境を使用するような攻撃検知技術と一般ユーザが使用する実環境との間に生じる実装の差異を利用したクローキング手法について記述している回答が見られた。そのほか、実際に D3M2014 を解析し、クライアントサイドでクローキングを行うコードを抽出し、具体的なコードを基に仕組みや構造を記述しているチームもあった。

設問2は、多くのチームがシグネチャやヒューリスティックを用いた定期的なコンテンツの検査、ハニークライアントによる Web サイトアクセス等、クローキングの特徴を考慮した上で、効率的に改ざんを検知する方法を回答していた。また、単純な Web コンテンツの改ざんだけでなく、Web サーバを構成するモジュールや設定ファイルの改ざんも含めた回答も見られた。さらに、検知方法に加え、検知機構のシステム化や既存ツールを用いた解析の自動化等といった、コストパフォーマンスや実現性を意識した回答も多く見られたため、回答の中には実運用に耐えられる手法も存在する可能性がある。

各チームにおける本課題への取り組みが、新しい研究や技術として深化し、MWS 等での発表、延いてはマルウェア対策研究の発展に繋がることを期待したい。

参考文献

[1] マルウェア対策研究人材育成ワークショップ（MWS: anti-Malware engineering WorkShop）, http://www.iwsec.org/mws/2014/

[2] 秋山, 他, “マルウェア対策のための研究用データセット ～MWS Datasets 2014～”, 情報処理学会研究報告コンピュータセキュリティ（CSEC）Vol. 2014-CSEC-66, No. 19, pp. 1 -- 7, 2014.

[3] 高田, 秋山, “MWS2014 意見交換会 D3M (Drive-by Download Data by Marionette) 2014”, http://www.iwsec.org/mws/2014/files/D3M_2014.pdf

[4] 秋山, “MWS2013 意見交換会 D3M (Drive-by Download Data by Marionette) 2013”, http://www.iwsec.org/mws/2013/files/D3M_Dataset_2013.pdf

[5] M. Akiyama, et al., “Design and Implementation of High Interaction Client Honeypot for Drive-by-download Attacks,” IEICE Transactions on Communication, Vol.E93-B, No.05, pp.1131-1139, May. 2010.

[6] 秋山, 他, “改ざん Web サイトのリダイレクトに基づく悪性 Web サイトの生存期間測定”, 信学技報, Vol. 113, No. 502, ICSS2013-71, pp.53--58（2014）.

[7] “Caffeine Monkey”, http://www.secureworks.com/cyber-threat-intelligence/tools/caffeinemonkey/

[8] C. Kolbitsch, et al, “Rozzle: De-cloaking internet malware.” In Proc. of IEEE Symposium on Security and Privacy, 2012.

[9] E. Gerds, “Browser plugin detection with PluginDetect,” http://www.pinlady.net/PluginDetect/

[10] tripwire, http://www.tripwire.org/

[11] FireEye: Darkleech Says Hello, http://www.fireeye.com/blog/technical/cyber-exploits/2013/09/darkleech-says-hello.html

[12] “IPA ウェブサイト攻撃の検出ツール iLogScanner V3.0”, https://www.ipa.go.jp/security/vuln/iLogScanner/

[13] Z. Li, et al. “Knowing your enemy understanding and detecting malicious web advertising,” In Proc. of ACM Conference on Computer and Communications Security, 2012.