



ASP.NET and ViewState Security

September 9th 2010

cyrill.brunschwiler@csnc.ch

thomas.roethlisberger@csnc.ch

Compass Security AG
Glärnischstrasse 7
Postfach 1628
CH-8640 Rapperswil

Tel +41 55-214 41 60
Fax +41 55-214 41 61
team@csnc.ch
www.csnc.ch

Multiplatform View State Tampering

From: Trustwave Advisories

Sent: Tuesday, February 9th 2010 23:41

...SpiderLabs has documented view state tampering vulnerabilities ... View states are used by some web application frameworks to store the state of HTML GUI controls. View states are typically stored in hidden client-side input fields, although server-side storage is widely supported.

Credit: David Byrne of Trustwave's SpiderLabs

Raise awareness of the ViewState flaw

Understand limitations of the ASP.NET framework

Know how to avoid the ViewState flaw

Adopt issue to other technologies

ViewState Intro

ViewState Flaw Demo

Remediation

- ✦ Input Validation
- ✦ Output Encoding
- ✦ Avoid ViewState Tampering

Conclusion

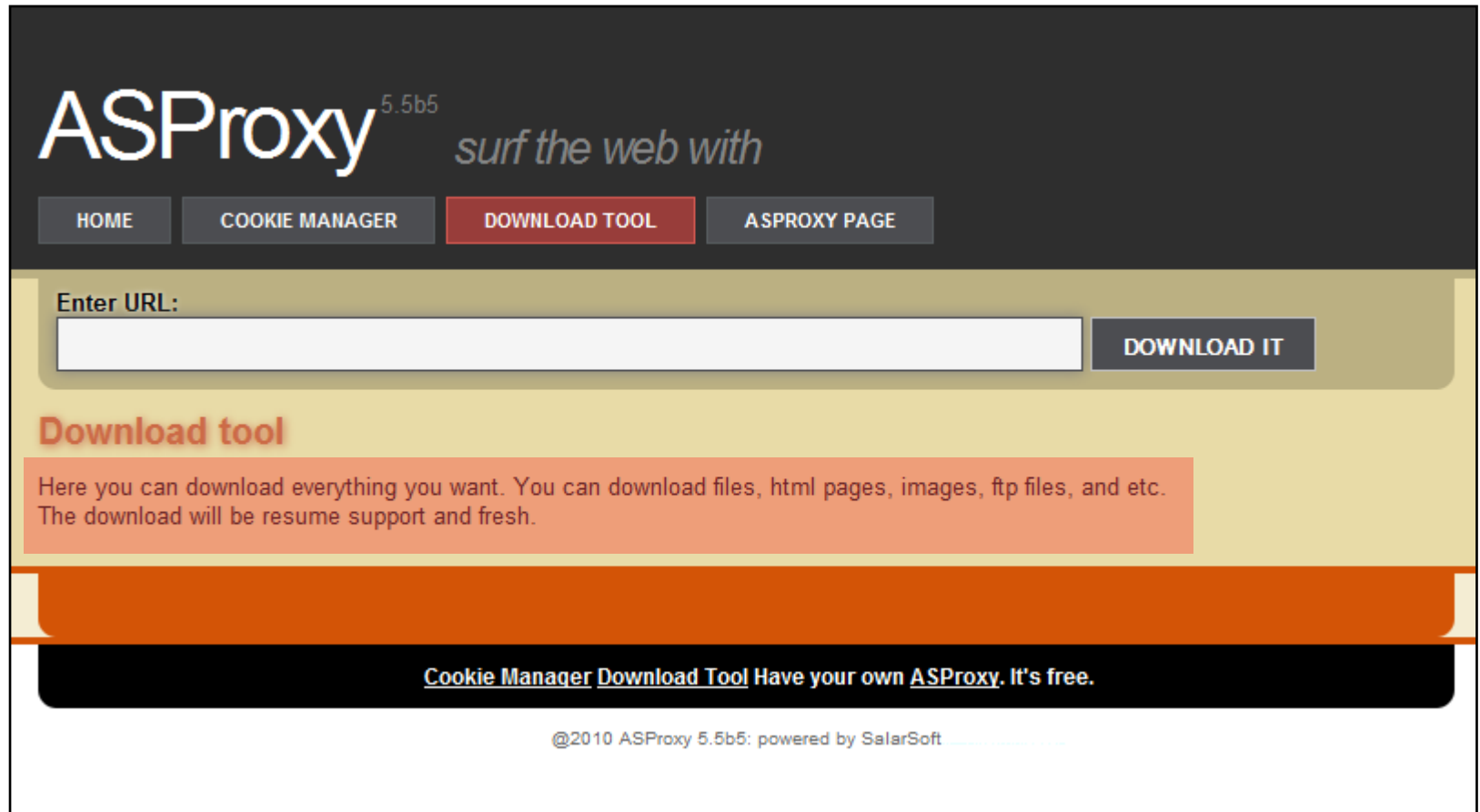


ViewState Intro

Compass Security AG
Glärnischstrasse 7
Postfach 1628
CH-8640 Rapperswil

Tel +41 55-214 41 60
Fax +41 55-214 41 61
team@csnc.ch
www.csnc.ch

Sample Application

A screenshot of the ASProxy 5.5b5 web application interface. The page has a dark header with the "ASProxy 5.5b5" logo and the tagline "surf the web with". Below the header is a navigation menu with buttons for "HOME", "COOKIE MANAGER", "DOWNLOAD TOOL" (highlighted in red), and "ASPROXY PAGE". The main content area features a "Enter URL:" label, a text input field, and a "DOWNLOAD IT" button. Below this is a section titled "Download tool" with a description: "Here you can download everything you want. You can download files, html pages, images, ftp files, and etc. The download will be resume support and fresh." At the bottom, there is a black banner with white text: "Cookie Manager Download Tool Have your own ASProxy. It's free." and a footer: "@2010 ASProxy 5.5b5: powered by SalarSoft".

Sample Code Snippet in C#

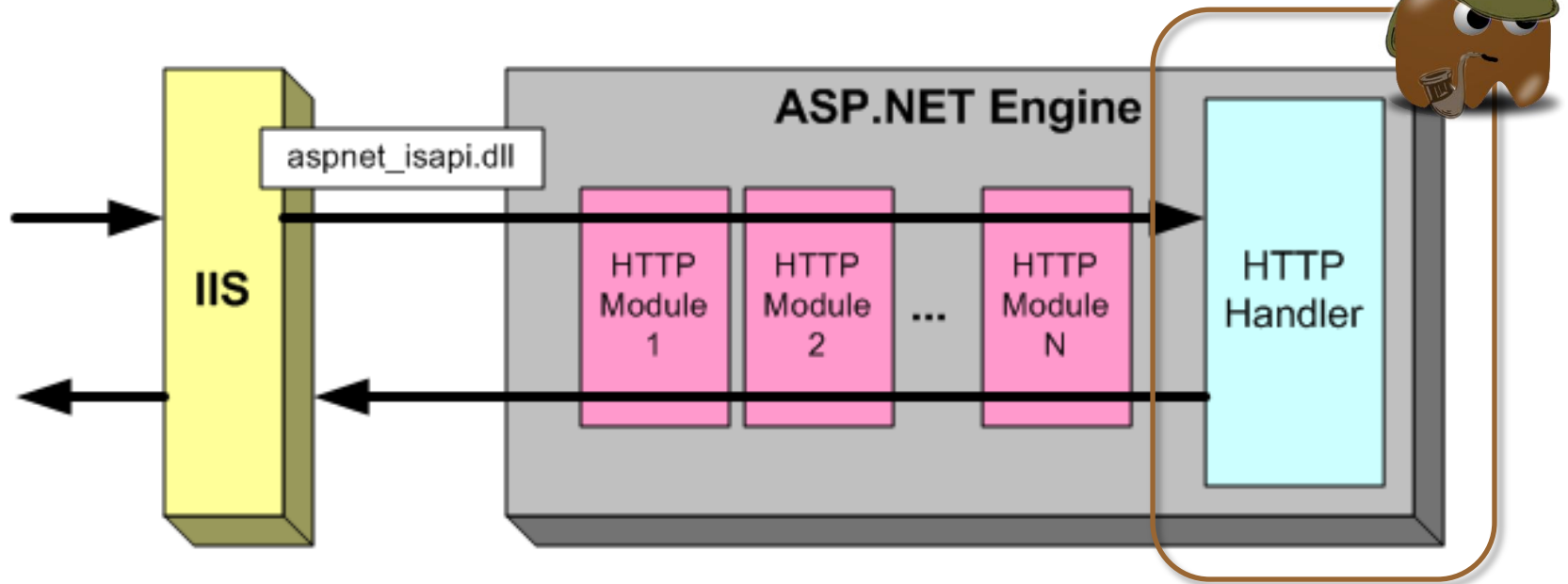
```
<script runat="server">
    protected void Page_Load(object sender, Event...
        if (!IsPostBack) {
            myLabel.Text = "Here you can download..."
        }
    }
</script>
```

```
<asp:Content runat="server" ContentPlaceHolderID...
    <asp:Label ID="myLabel" runat="server">
    </asp:Label>
```

Sample HTML Snippet

```
<form name="aspnetForm" method="post" id="asp...  
  <input type="hidden" name="__VIEWSTATE" id="__V...  
    value="/wEP0aWpA45OkQLP9+4sT2...YW1lcw=" />  
  ...  
    Download tool</span></h1>  
</div>  
  ...  
<div class="entry">  
  <span id="ctl100_plhContent_myLabel">  
    Here you can download everything you wan...  
  </span>
```

ViewState Flow

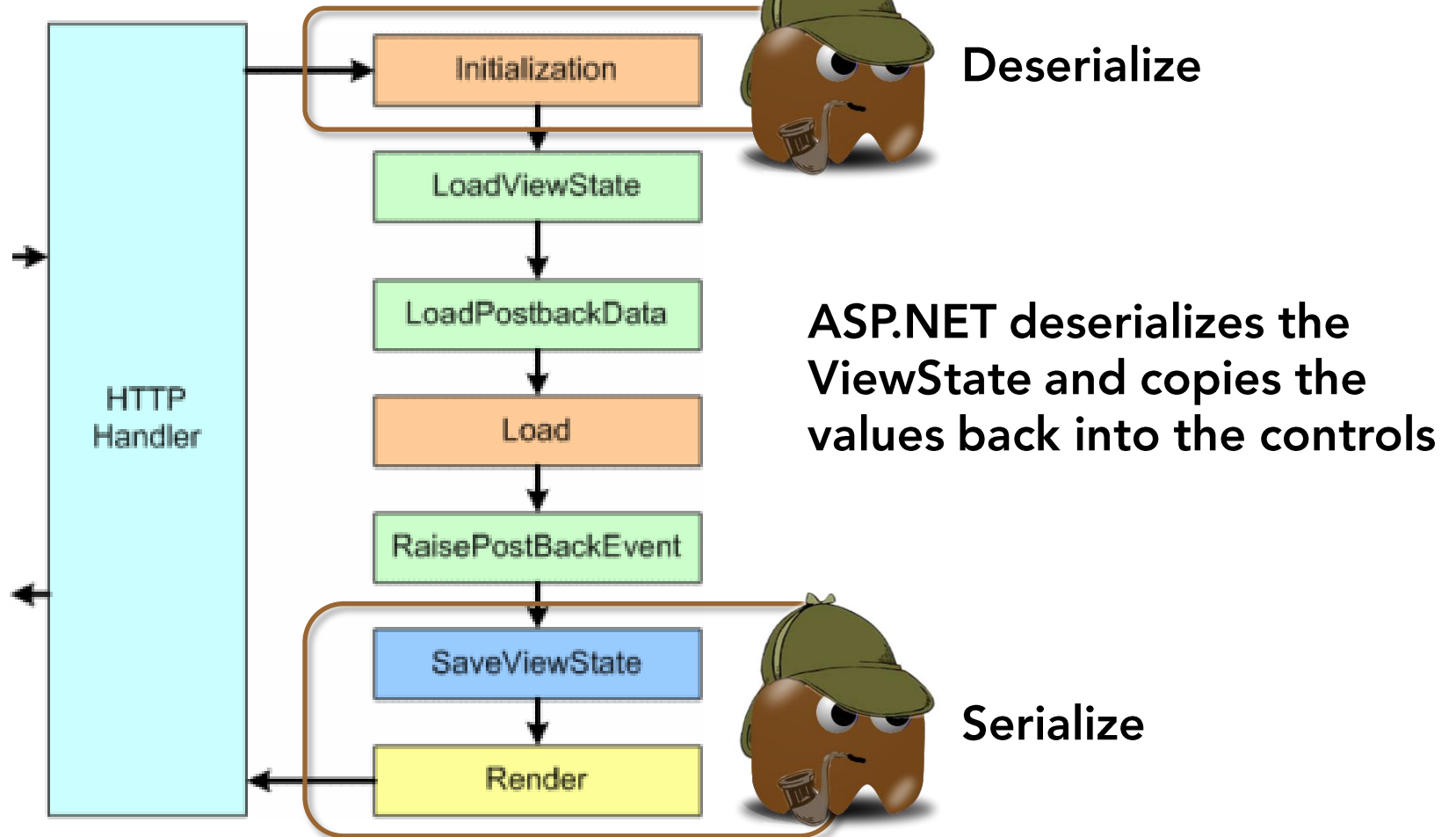


...

```
<input type="hidden" name="__VIEWSTATE" id="__V...  
value="/wEP0aWpA45OkQLP9+4sT2...YW11cw=" />
```

The field `__VIEWSTATE` contains value of myLabel in encoded form

ViewState Handling



ViewState Facts

- ✦ Passive controls (eg. Labels) are not rendered as HTML input fields
- ✦ Passive controls need their value to be posted back to the server
- ✦ Disabling the ViewState will complicate the way to allow a user to work with multiple windows
- ✦ Disabling the ViewState will destroy the advantages of the framework
- ✦ Disabling the ViewState will result in overhead during development

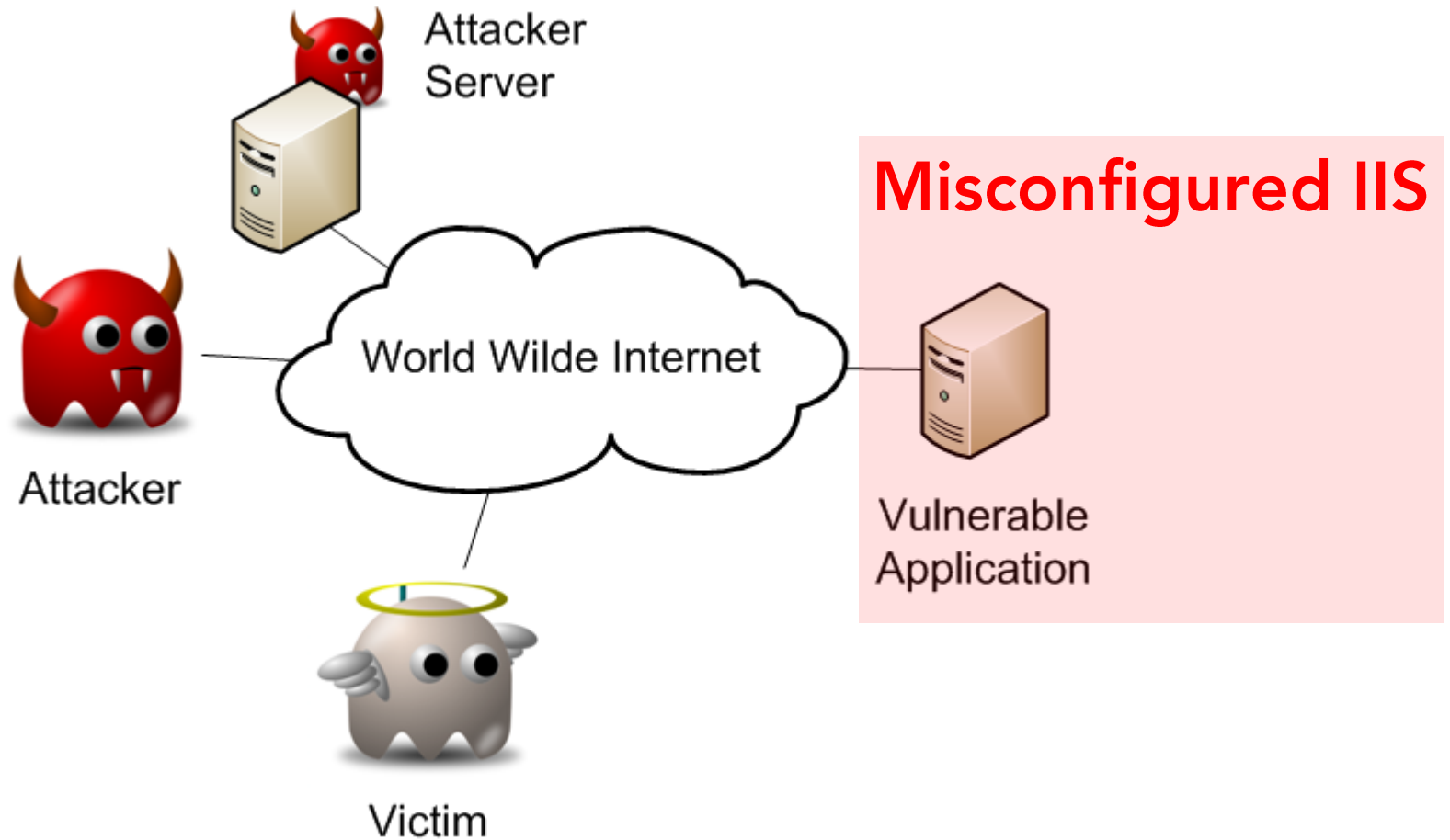
A vertical decorative strip on the left side of the slide features a close-up image of a computer keyboard with a yellow padlock resting on one of the keys.

ViewState Flaw Demo

Compass Security AG
Glärnischstrasse 7
Postfach 1628
CH-8640 Rapperswil

Tel +41 55-214 41 60
Fax +41 55-214 41 61
team@csnc.ch
www.csnc.ch

Involved Parties



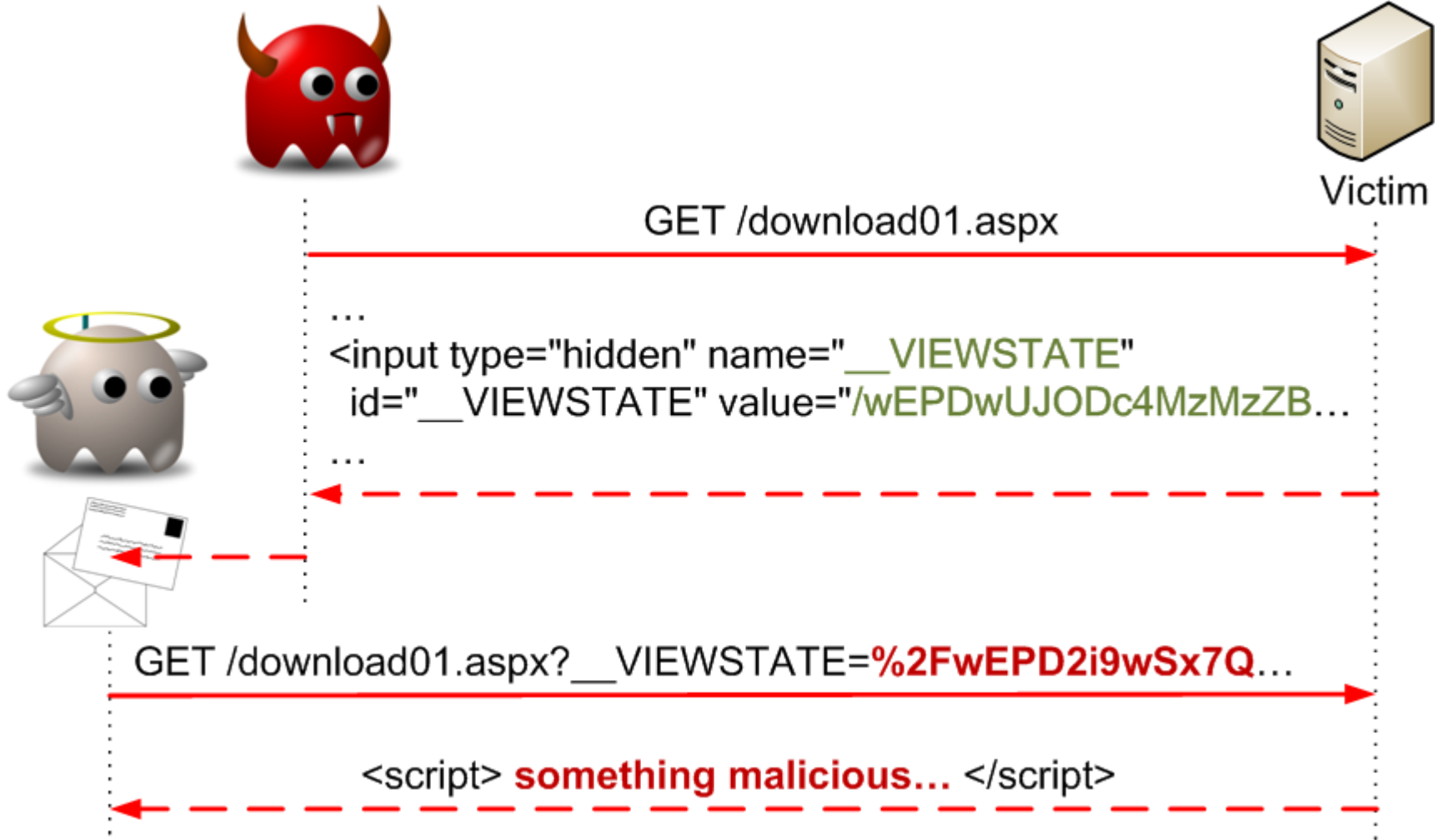
A vertical decorative strip on the left side of the slide features a close-up photograph of a computer keyboard with a yellow padlock resting on one of the keys. The background of the slide is white with a horizontal dotted line near the top.

ViewState Flaw Demo

Compass Security AG
Glärnischstrasse 7
Postfach 1628
CH-8640 Rapperswil

Tel +41 55-214 41 60
Fax +41 55-214 41 61
team@csnc.ch
www.csnc.ch

ViewState Flaw - Sequence



An attacker can inject malicious HTML or JavaScript into the ViewState

JavaScript can be injected by tampering the value of the hidden field `__VIEWSTATE`. The server copies the tampered values back into the server controls. As ASP.NET renders the resulting HTML response the malicious code is included in the page.

Identify a property of a control inside of the ViewState which makes sense to modify (Several properties are vulnerable to such an attack: E.g. `Text` of a label, `InnerHTML` of the main form, etc.)

ViewState Flaw - Text Injection



```
<System.Collections.ArrayList>
  <System.Int32>1</System.Int32>
  <System.Web.UI.Pair>
    <System.Web.UI.Pair>
      <System.Collections.ArrayList>
        <System.Web.UI.IndexedString>Text</System.Web.UI.IndexedString>
        <System.String xml:space="preserve">Hello &lt;script&gt;alert('xss')&lt;/script&gt;X
      </System.Collections.ArrayList>
      <Null>True</Null>
    </System.Web.UI.Pair>
    <Null>True</Null>
  </System.Web.UI.Pair>
  <System.Int32>3</System.Int32>
  <System.Web.UI.Pair>
    <System.Web.UI.Pair>
      <System.Collections.ArrayList>
        <System.Web.UI.IndexedString>Text</System.Web.UI.IndexedString>
      </System.Collections.ArrayList>
    </System.Web.UI.Pair>
  </System.Collections.ArrayList>
```

Reserialized and Base64 encoded

```
/wEPDwUKMTI2NTY4ODI3MQ9kFgICAw9kFgQCAQ8PFgIeBFRIeHQFJkhIbGxvIDxzY3JpcH0
```

ViewState Flaw - InnerHTML Injection



```
<System.Collections.ArrayList>
```

```
<System.Int32>3</System.Int32>
```

```
<System.Web.UI.Pair>
```

```
<System.Collections.ArrayList>
```

```
<System.Web.UI.IndexedString>innerHTML</System.Web.UI.IndexedString>
```

```
<System.String xml:space="preserve">Hello &lt;script>alert('xss')&lt;/script></System.String>
```

```
</System.Collections.ArrayList>
```

```
<Null>True</Null>
```

```
</System.Web.UI.Pair>
```

```
</System.Collections.ArrayList>
```

```
</System.Web.UI.Pair>
```

```
</System.Web.UI.Pair>
```

```
<Null>True</Null>
```



**Reserialized and
Base64 encoded**

```
/wEPDwUKMTI2NTY4ODI3MQ9kFgICAw9kFgQCAQ8PFgleBFRIeHQFJkhlbGxvIDxzY3JpcH0
```

How to tend the patient?



Compass Security AG
Glärnischstrasse 7
Postfach 1628
CH-8640 Rapperswil

Tel +41 55-214 41 60
Fax +41 55-214 41 61
team@csnc.ch
www.csnc.ch

The left side of the slide features a vertical image strip. It shows a close-up of a computer keyboard with a yellow padlock resting on one of the keys. The background is a light blue color with a subtle pattern of keyboard keys.

Input Validation

Compass Security AG
Glärnischstrasse 7
Postfach 1628
CH-8640 Rapperswil

Tel +41 55-214 41 60
Fax +41 55-214 41 61
team@csnc.ch
www.csnc.ch

ASP.NET Request Validation

- ✦ Validation against query-string and form variables as well as cookie values
- ✦ ASP.NET raises an error if a request contains HTML-encoded elements or certain HTML characters (e.g. `<script>`)
- ✦ Does not validate contents of the ViewState

Search:



Server Error in '/XSSViewState' Application.

A potentially dangerous Request.QueryString value was detected from the client (search="test'><script>alert(1)</sc...").

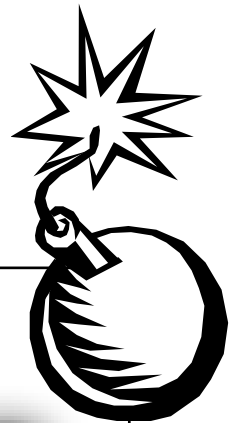
Description: Request Validation has detected a potentially dangerous client input value, and processing of the request

ASP.NET Request Validation Dangers

- ✦ Feature often disabled by developers as quick fix for this "strange errors".
- ✦ It doesn't filter all dangerous characters (E.g. ', " or &)

Search:

Search:



Exploitable Code

```
<form id="Form1" method="GET" runAt="server,...  
  <label for="inpSearch">Search: </label>  
  <input value='<%=Request.QueryString["search"]%>'  
    type='text' id='search' name='search'>  
  <input type="submit" />  
</form>
```

Applications that embed user input in JavaScript are vulnerable as well



Looks like Request Validation is
not the solution...



Compass Security AG
Glärnischstrasse 7
Postfach 1628
CH-8640 Rapperswil

Tel +41 55-214 41 60
Fax +41 55-214 41 61
team@csnc.ch
www.csnc.ch

A vertical decorative strip on the left side of the slide features a close-up photograph of a computer keyboard with a yellow padlock resting on one of the keys.


Output Encoding

Compass Security AG
Glärnischstrasse 7
Postfach 1628
CH-8640 Rapperswil

Tel +41 55-214 41 60
Fax +41 55-214 41 61
team@csnc.ch
www.csnc.ch

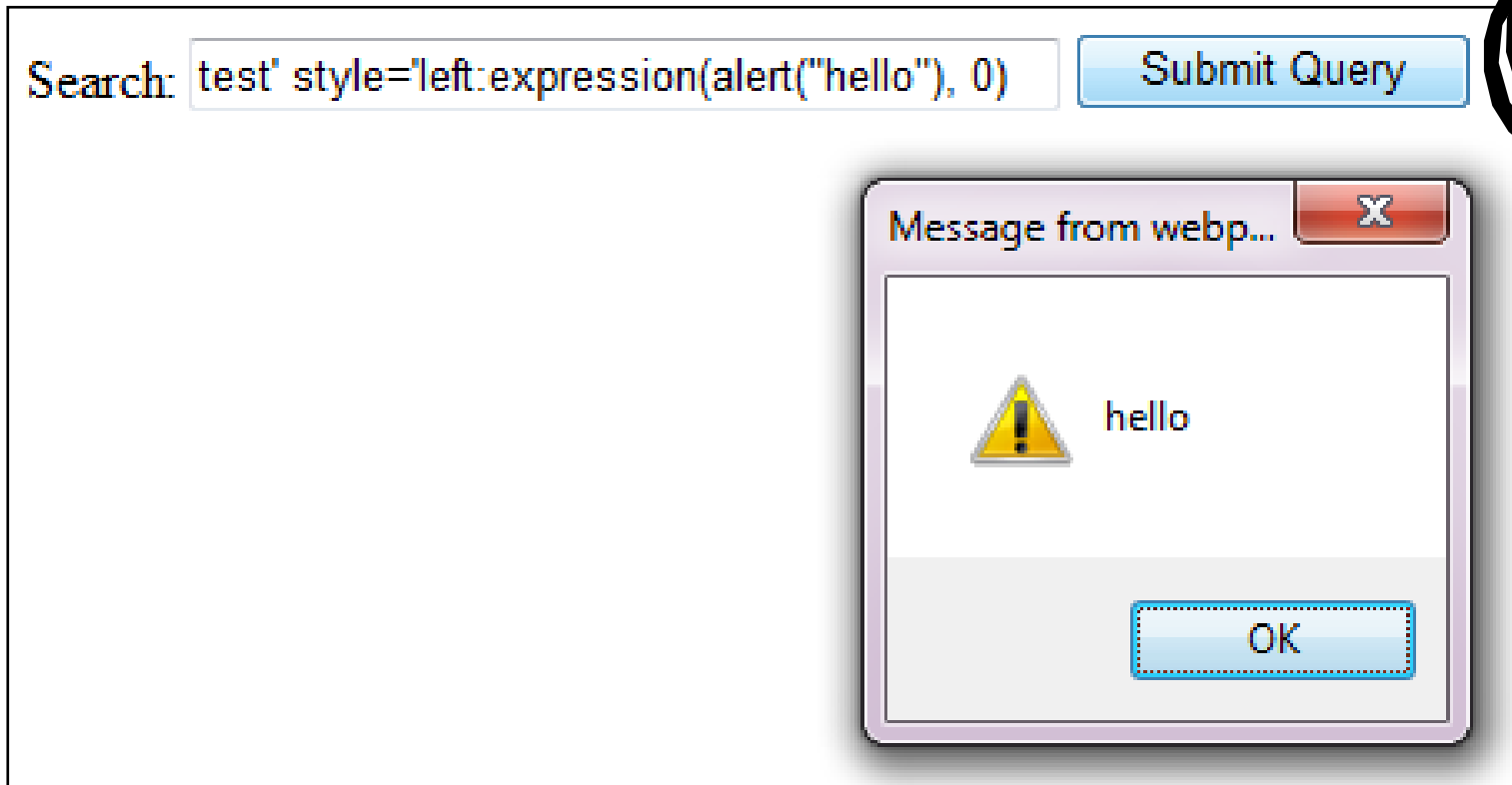
ASP.NET HTML Encoding

- ✦ Some of the controls encode their content automatically:
 - ✦ Button.Text
 - ✦ TextBox.Text
 - ✦ ...

 - ✦ But the rest has to be encoded manually:
 - ✦ Label.Text
 - ✦ Literal.Text
 - ✦ ...
- A black and white icon of a bomb with a lit fuse and a starburst effect, symbolizing a security warning or a critical issue.
- ✦ HtmlEncode converts critical characters using HTML entities
`Server.HtmlEncode("") => `

HTML Encoding Dangers

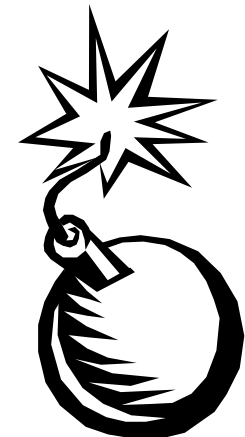
- ◆ Developers often forget to use this method.
- ◆ HtmlEncode doesn't encode all dangerous characters (E.g. ', ")

A screenshot of a web search interface. The search input field contains the text "test' style='left:expression(alert(\"hello\"), 0)". To the right of the input field is a blue button labeled "Submit Query". Below the search field, a small dialog box titled "Message from webp..." is displayed. The dialog box has a yellow warning triangle icon and the text "hello". At the bottom of the dialog box is a blue button labeled "OK".

Exploitable Code

```
<form id="Form1" method="GET" runAt="server...  
  <label for="inpSearch">Search: </label>  
  <input value='<%=Server.HtmlEncode (  
    Request.QueryString["search"]) %>'  
    type='text' id='search' name='search'>  
  <input type="submit" />  
</form>
```

Especially text embedded in JavaScript or JSON fragments will be prone to Cross-site Scripting



Okay, how do we really fix it?



Compass Security AG
Glärnischstrasse 7
Postfach 1628
CH-8640 Rapperswil

Tel +41 55-214 41 60
Fax +41 55-214 41 61
team@csnc.ch
www.csnc.ch

Avoid ViewState Tampering

Compass Security AG
Glärnischstrasse 7
Postfach 1628
CH-8640 Rapperswil

Tel +41 55-214 41 60
Fax +41 55-214 41 61
team@csnc.ch
www.csnc.ch

Approaches

- ✦ Do not store the ViewState within the HTML page => Implement Handler
- ✦ Protect the ViewState contents => Configuration

Avoid ViewState in HTML

- ✦ ViewState can be stored on the server
- ✦ Override the framework methods
 - ✦ `SavePageStateToPersistenceMedium()`
 - ✦ `LoadPageStateFromPersistenceMedium()`
- ✦ Approach leads to challenges
 - ✦ Users may work with multiple windows => multiple ViewStates
 - ✦ ViewState must be unique for each user and for each page
 - ✦ ViewState has to be cleaned up when no longer needed
 - ✦ Ensure users cannot access each others ViewStates

ViewState Protection

- ✦ **MAC integrity check**
The hash of the ViewState data is signed with a key and stored along with the data in the ViewState field. The MAC key might be configured to be recreated on startup, which will cause issues in balanced environments. Moreover, applications could be isolated.
- ✦ **Event Validation**
For each control on the page a unique number is generated (XOR of the hashes of all valid values and the Uniqueid of the control). All these numbers are stored in another hidden field on the page called `__EVENTVALIDATION`.
- ✦ **ViewStateUserKey**
If provided a user session specific id is used as a salt in the MAC. This avoids the ViewState being used by a different user.
- ✦ **Encryption**
The data of the ViewState is encrypted.
- ✦ Lockdown the ViewState configuration within the machine.config.



Conclusion

Compass Security AG
Glärnischstrasse 7
Postfach 1628
CH-8640 Rapperswil

Tel +41 55-214 41 60
Fax +41 55-214 41 61
team@csnc.ch
www.csnc.ch

Input Validation

- ✦ Activate the RequestValidation but not solely trust this input filter.
- ✦ Use a well tested filtering framework such as the OWASP Enterprise Security API (ESAPI) or the Microsoft AntiXSS library.

Output Encoding

- ✦ Consequently encoding the content of every control before it gets rendered into the page.
- ✦ Use a well tested encoding framework such as the OWASP Enterprise Security API (ESAPI) or the Microsoft AntiXSS library

Avoid ViewState Tampering

- ✦ Enable MAC integrity check to prevent tampering
- ✦ Enable EventValidation to restrict values for each specific control
- ✦ Enable ViewStateUserKey to bind ViewStates to the user session
- ✦ Encrypt ViewState to avoid disclosure and caching of confidential information

Assure

- ✦ Developers need to understand the security implications and features
- ✦ Enforce input filtering and output encoding
- ✦ Make use of approved filter and encoding frameworks
- ✦ Enroll a secure development life cycle (use CAT.NET to analyze assemblies)

Mind

- ✦ Server and client logic may be exploited due to malicious data formats
- ✦ Entry server, WAF protection may be bypassed using custom data formats
- ✦ Other frameworks (Java Server Faces) know the concept of a ViewState as well
 - ✦ Apache MyFaces
 - ✦ SUN Project Mojarra

Implement Best Practices

- ✦ Guard Against Malicious User Input
 - ✦ ViewState Security
 - ✦ Input Filter
 - ✦ Output Encoding

- ✦ Run Applications with Least Privileges and Know Your Users
 - ✦ Authentication
 - ✦ Authorization
 - ✦ ACLs

- ✦ Keep Sensitive Information Safely
 - ✦ SSL protect traffic
 - ✦ Safeguard configuration
 - ✦ Keep sensitive information assets at the server side
 - ✦ Strong encryption (System.Security.Cryptography)

Implement Best Practices

- ✦ Use Cookies Securely
 - ✦ No sensitive information
 - ✦ Secure cookies settings

- ✦ Access Databases Securely
 - ✦ Inherent security
 - ✦ Parameterized queries
 - ✦ Protected configuration

- ✦ Create Safe Error Messages
 - ✦ Custom error pages
 - ✦ No detailed error messages



References

Compass Security AG
Glärnischstrasse 7
Postfach 1628
CH-8640 Rapperswil

Tel +41 55-214 41 60
Fax +41 55-214 41 61
team@csnc.ch
www.csnc.ch

ViewState Documentation and Analysis Tools

- ✦ ViewState
<http://msdn.microsoft.com/en-us/library/ms972976.aspx>
- ✦ Trustwave's SpiderLabs Security Advisory TWSL2010-001
<https://www.trustwave.com/spiderlabs/advisories/TWSL2010-001.txt>
- ✦ ViewStateViewer
<http://labs.neohapsis.com/2009/08/03/viewstateviewer-a-gui-tool-for-deserializingreserializing-viewstate/>
- ✦ Fiddler 2 & ViewState Interceptor Plugin
<http://www.fiddler2.com/>
<http://www.binaryfortress.com/aspnet-viewstate-helper/>

Security Guides

- ✦ Microsoft Web Application Best Practices
<http://msdn.microsoft.com/en-us/library/zdh19h94.aspx>
<http://msdn.microsoft.com/en-us/library/330a99hc.aspx>
- ✦ OWASP Development Guide
http://www.owasp.org/index.php/Category:OWASP_Guide_Project

Security Tools and Frameworks

- ✦ OWASP Enterprise Security API (ESAPI) for .NET
http://www.owasp.org/index.php/Category:OWASP_Enterprise_Security_API
- ✦ AntiXSS API from Microsoft
<http://www.microsoft.com/downloads/details.aspx?familyid=051EE83C-5CCF-48ED-8463-02F56A6BFC09>
- ✦ CAT.NET (Assembly Analyzer)
<http://www.microsoft.com/downloads/details.aspx?FamilyID=0178e2ef-9da8-445e-9348-c93f24cc9f9d>

.NET Hints

- ✦ List of .NET Controls and Encoding
<http://blogs.msdn.com/b/sfaust/archive/2008/09/02/which-asp-net-controls-automatically-encodes.aspx>
- ✦ Page State Persister Example
<http://msdn.microsoft.com/en-us/library/aa479403.aspx>
- ✦ Securing the ViewState
[http://msdn.microsoft.com/en-us/library/ms178199\(VS.85\).aspx](http://msdn.microsoft.com/en-us/library/ms178199(VS.85).aspx)
- ✦ Configuration Locking
<http://learn.iis.net/page.aspx/145/how-to-use-locking-in-iis-70-configuration/>

Sample CAT.NET output

Result #9			
Summary			
Problem	A cross-site redirection vulnerability was found through a user controlled variable that enters the application at adminlogin.aspx variable stack0 which eventually leads to a cross-site redirection issue at adminlogin.aspx.cs:60.		
Resolution	Do not allow off-site redirections to absolute URLs that can be specified by the user.		
Entry Variable	stack0		
Confidence	High		
Source Context	Line	Input Variable	Statement
adminlogin.aspx.cs	57		string returnUrl = Request.QueryString["ReturnUrl"];
adminlogin.aspx.cs	57	Return from HttpRequest.get_QueryString	string returnUrl = Request.QueryString["ReturnUrl"];
adminlogin.aspx.cs	60	returnUrl	Response.Redirect(returnUrl, true);
Cross-Site Scripting (ACESEC05)			
35 results			
Result #10			
Summary			
Problem	A cross-site scripting vulnerability was found through a user controlled variable that enters the application at noscript.aspx:23 variable stack1 which eventually leads to a cross-site scripting issue at noscript.aspx:160.		
Resolution	Use the Anti-XSS library to properly encode the data before rendering it		
Entry Variable	stack1		
Confidence	Low (External call to non-system method, cannot verify data taints result.)		
Source Context	Line	Input Variable	Statement
noscript.aspx	232		engine.RequestInfo.RequestUrl = txtUrl.Text;
noscript.aspx	232	Return from TextBox.get_Text	engine.RequestInfo.RequestUrl = txtUrl.Text;

