Exploiting and Defending: Common Web Application Vulnerabilities
Introduction: Steve Kosten

Principal Security Consultant
SANS Instructor
Denver OWASP Chapter Lead

Certifications
CISSP, GWAPT, GSSP-Java, CISM

Contact Info
Steve.kosten@cypressdefense.com
@skosten
Introduction: Aaron Cure

Principal Security Consultant
SANS Instructor & Contributing Author

Certifications
CISSP, GSSP.NET, GWAPT, GMOB, GPEN

Contact Info
aaron.cure@cyressdefense.com
@curea
Disclaimer

Using real attack tools

Illegal to attack targets without written contractual consent

Obey all state and federal laws

Cypress Data Defense assumes no liability
Introduction
A6: Sensitive Data Exposure
A5: Security Misconfiguration
A1: Injection
A3: Cross-Site Scripting (XSS)
A8: Cross-Site Request Forgery (CSRF)
Secure Software Development LifeCycle (SSDLC)
Software Development LifeCycle (SDLC)

• Software Development Life Cycle
• Process for planning, creating, testing, and deploying an information system
What is a Secure SDLC?

Security considered at each phase

Initial and ongoing Security Training

Overall security is the priority

Testing and evaluation of security throughout
Secure Software Development LifeCycle (SSDLC)

- SECURITY TRAINING
  - Core Security Training
  - Specialized Training
  - Ongoing Training

- REQUIREMENTS
  - User Stories
    - Security Stories
    - Abuse Stories
  - Risk Analysis

- PLANNING & DESIGN
  - Attack Surface
  - Threat Modeling

- DEVELOPMENT
  - Peer Review
  - Static Analysis

- VERIFICATION & TESTING
  - Penetration Testing
  - Attack Surface Review

- RELEASE
  - Continuous Monitoring
  - Continuous Feedback

©2017 – Cypress Data Defense, LLC
:: Details

Your notice of insecure password and/or log-in automatically appearing on the log-in for my website, Oil and Gas International is not wanted and was put there without our permission. Please remove it immediately. We have our own security system and it has never been breached in more than 15 years. Your notice is causing concern by our subscribers and is detrimental to our business.
:: Details

Your notice of insecure password and/or log-in automatically appearing on the log-in for my website, Oil and Gas International is not wanted and was put there without our permission. Please remove it immediately. We have our own security system and it has never been breached in more than 15 years. Your notice is causing concern by our subscribers and is detrimental to our business.
Oh, THAT notice...
It Just Gets Worse...

Individual Subscriber Log on
Email
Password
Log on

If you're not a subscriber, click here for more information.

Corporate Subscriber Log on
Company
ID
Email
Password
Log on

Forgot your password?
Click here to receive it via email.
Sensitive Data Exposure occurs when an application does not adequately protect sensitive information. The data can vary and anything from passwords, session tokens, credit card data to private health data and more can be exposed.
A6: Mitigation

HTTPS (TLS Cert)
HTTP Security Headers
HSTS (HTTP Strict Transport Security)
Server Error in '/' Application.

Input string was not in a correct format.

Description: An unhandled exception occurred during the execution of the current web request. Please review the stack trace for more information about the error and where it originated in the code.

Exception Details: System.FormatException: Input string was not in a correct format.

Source Error:

Line 49:
{
    OGILogin objLogin = new OGILogin(System.Configuration.ConfigurationSettings.AppSettings["cnString"]);
    bool binLogin = objLogin.UserLogin( txtEmailCorporateLogin.Text, txtPasswordCorporateLogin.Text, int32.Parse(txtCompanyName.Text));
    if (binLogin == true)
    {
    
Source File: g:\inetpub\OilandGasInternational\html\login.aspx.cs  Line: 51

Stack Trace:

[FormatException: Input string was not in a correct format.]
   System.Number.StringToNumber(String str, NumberStyles options, NumberBuffer& number, NumberFormatInfo info, Boolean parseDecimal) +10169507
   System.Number.ParseInt32(String s, NumberStyles style, NumberFormatInfo info) +207
   g\inetpub\OilandGasInternational\login2.btncorporatelogin_OnClick(Object sender, EventArgs e) in g:\inetpub\OilandGasInternational\html\login.aspx.cs:51
   System.Web.UI.WebControls.Button.OnClick(EventArgs e) +113
   System.Web.UI.Page.ProcessRequestMain(Boolean includeStagesBeforeAsyncPoint, Boolean includeStagesAfterAsyncPoint) +2981

Version Information: Microsoft .NET Framework Version:2.0.50727.5485; ASP.NET Version:2.0.50727.5485

©2017 – Cypress Data Defense, LLC
Good security requires having a secure configuration defined and deployed for the application, frameworks, application server, web server, database server, and platform. Secure settings should be defined, implemented, and maintained, as defaults are often insecure. Additionally, software should be kept up to date.
A5: Mitigation

Custom Error Handler

Single Error Message/Page

No Error Information – Including Return Code

Internal Error Logging
What Threw the Stack Trace?

Server Error in '/' Application.

Input string was not in a correct format.

Description: An unhandled exception occurred during the execution of the current web request. Please review the stack trace for more information about the error and where it originated in the code.

Exception Details: System.FormatException: Input string was not in a correct format.

Source Error:

Line 48:
Line 50:
{  
    object objLogin = new OGIILogin(System.Configuration.ConfigurationSettings.AppSettings["cnString"]);  
    bool bIsLogin = objLogin.UserLogin( txtEmailCorporateLogOn.Text, txtPasswordCorporateLogOn.Text, Int32.Parse(txtCompName.Text));  
    if( bIsLogin == true )  
{

Source File: g:\inetpub\OilandGasInternational\html\login.aspx.cs  Line 51

Stack Trace:
[FormatException: Input string was not in a correct format.]
System.Number.StringToNumber(String str, NumberStyles options, NumberBuffer& number, NumberFormatInfo info, Boolean parseDecimal) +10169507
System.Number.ParseInt32(String s, NumberStyles style, NumberFormatInfo info) +207
0x7ffdbd2f3ea0 +10 OilandGasInternational.aulogin.aulogin_onClick(Object sender, EventArgs e) in g:\inetpub\OilandGasInternational\html\login.aspx.cs:51
System.Web.UI.Page.ProcessRequestMain(Boolean includeStagesBeforeAsyncPoint, Boolean includeStagesAfterAsyncPoint) +2981

A1: Injection

Text-based attacks that exploit the syntax of the targeted interpreter.

Almost any source of data can be an injection vector, including internal sources.

Injection flaws occur when an application sends untrusted data to an interpreter.
A1: SQL Injection
In The News (Target)

110 million customer records
Email, Mailing addresses, other Personally Identifiable Information (PII)
In The News (Living Social)

50 million customer records
Email, DOB, Password Hashes, Challenge Questions & Answers
Command Injection

```java
Runtime.getRuntime().exec(String.format("myTestProcess.exe %s", request.getParameter("employeeId")))
```

Inline SQL

```java
rs = statement.executeQuery(
"Select EmployeeId, LastName, FirstName, PhoneNumber " +
"From Employees " +
"Where EmployeeId = " + request.getParameter("employeeId")
)
sqlmap DEMO

http://sqlmap.org/

Written in Python
A1: Mitigation

Parameterized Queries
Object Relation Mappers (ORM)
Remember Me?

Server Error in '/' Application.

Input string was not in a correct format.

Description: An unhandled exception occurred during the execution of the current web request. Please review the stack trace for more information about the error and where it originated in the code.

Exception Details: System.FormatException: Input string was not in a correct format.

Source File: g:\inetpub\OilandGasInternational\html\login.aspx.cs   Line 51  

Stack Trace:

[FormatException: Input string was not in a correct format.]

System.String.ToString(String str, NumberStyles options, NumberFormatInfo info, Boolean parseDecimal) +10169507
System.Number.ParseInt32(String s, NumberStyles style, NumberFormatInfo info) +207
OilandGasInternational.login2.0nCorporatelogOn.0nclick(Object sender, EventArgs e) in g:\inetpub\OilandGasInternational\html\login.aspx.cs:51
System.Web.UI.WebControls.Button.0nclick(EventArgs e) +115
System.Web.UI.Page.RaisePostBackEvent(EventArgs e, String eventArgument) +29
System.Web.UI.Page.ProcessRequestMain(Boolean includeStagesBeforeAsyncPoint, Boolean includeStagesAfterAsyncPoint) +2981

Version Information: Microsoft .NET Framework Version:2.0.50727.5485; ASP.NET Version:2.0.50727.5483
A3: Cross-Site Scripting (XSS)

XSS flaws occur whenever an application takes untrusted data and sends it to a web browser without proper encoding.

- Execute scripts in the victim’s browser
- Hijack user sessions
- Deface web sites
- Redirect the user to malicious sites.
In The News (Sears)
Site defaced to contain flashing images designed to cause seizures
Some victims required hospital care
Reflected Example

**HTML Context**

```
<td><%= request.getParameter("Name") %></td>
```

**URL Context**

```
<a href='<%= String.format("details.aspx?id=%s", request.getParameter("Name")) %>'></a>
```

**JavaScript Context**

```
<a href='<%= String.format("javascript:redirect('{%s}')", request.getParameter("Name")) %>'>View</a>
```
Exploitation DEMO

Browser Exploitation Framework (BeEF)

http://beefproject.com/

Written in Ruby
Mitigations

Encoding, encoding, encoding

**Validation is not the solution**

Contexts to consider

Html, Url, JavaScript

HtmlAttribute, Css, Xml, XmlAttribute
Mitigations (2)

Language Specific Encoding Libraries

HTTP Security Headers

  X-XSS-Protection

  Content-Security-Policy (CSP)
CSRF

Cross Site Request Forgery
Admin console vulnerable to CSRF allowing attackers to perform the following:

- Modify automatic renewals
- Edit zone files
- Name server management
Multiple manufacturers
4.5 Million Routers Compromised
in Brazil
A CSRF attack forces a logged-on victim’s browser to send a forged HTTP request, including the victim’s session cookie and any other automatically included authentication information.

Audit logs will show the user made the transaction
User has no knowledge of the transaction
Cross-Site Request Forgery (CSRF) Example

Multiple Tabs

Authenticated Session
Cross-Site Request Forgery (CSRF) Example (2)

Payload on attack page

```html
<form id="csrfForm"
    action="http://localhost:8080/csrf/content/vulnerable/changepassword" method="POST">
  <input type="hidden" name="newPassword"
         value="StorageRoomB" />
  <input type="hidden" name="confirmPassword"
         value="StorageRoomB" />
</form>
```
Cross-Site Request Forgery (CSRF) Example (3)

Request triggered from authenticated session

```
POST /csrf/content/vulnerable/changepassword HTTP/1.1
Host: localhost:8080
Cookie: JSESSIONID=2E7F523BE6E086F5EEB593B2B69842D2
Content-Type: application/x-www-form-urlencoded
Content-Length: 53

newPassword=StorageRoomB&confirmPassword=StorageRoomB
```
200 Response from web site

HTTP/1.1 200 OK

<div class="alert alert-dismissable alert-success">
  <span>Your password was successfully changed.</span>
</div>
Exploitation DEMO

Simple Javascript Post
Mitigations

CSRF Mitigations

- Random nonce for each request
- Anti-Forgery Tokens
- CSRF Guard (OWASP Project)
- Browsers looking at headers (e.g., Origin)
Cross-Site Request Forgery (CSRF) Mitigation (1)

Payload with incorrect csrf token

```html
<form id="csrfForm"
  action="http://localhost:8080/csrf/content/vulnerable/changepassword"
  method="POST">

  <input type="hidden" name="newPassword"
        value="StorageRoomB" />

  <input type="hidden" name="confirmPassword"
        value="StorageRoomB" />

  <input type="hidden" name="&\#95;csrf"
        value="103ae2a3&\#45;d4d6&\#45;46e9&\#45;8ba6&\#45;92188ff998c2" />

</form>
```
Cross-Site Request Forgery (CSRF) Mitigation (2)

Request with invalid token submitted

```plaintext
POST /csrf/content/vulnerable/changepassword HTTP/1.1
Host: localhost:8080
Cookie: JSESSIONID=2E7F523BE6E086F5EEB593B2B69842D2
Content-Type: application/x-www-form-urlencoded
Content-Length: 53

newPassword=StorageRoomB&confirmPassword=StorageRoomB&_csrf=103ae2a3-d4d6-46e9-8ba6-92188ff998c2
```
Cross-Site Request Forgery (CSRF) Example (3)

403 response from web site

HTTP/1.1 403 Forbidden

<div class="alert alert-dismissable alert-danger">
    <span>java.lang.NullPointerException</span>
</div>
<table>
<thead>
<tr>
<th>Category</th>
<th>Security Training</th>
<th>Requirements</th>
<th>Planning &amp; Design</th>
<th>Development</th>
<th>Verification &amp; Testing</th>
<th>Release</th>
</tr>
</thead>
<tbody>
<tr>
<td>Core Security Training</td>
<td>Specialized Training</td>
<td>Ongoing Training</td>
<td>User Stories</td>
<td>Risk Analysis</td>
<td>Peer Review</td>
<td>Penetration Testing</td>
</tr>
<tr>
<td></td>
<td>Security Stories</td>
<td>Abuse Stories</td>
<td>Attack Surface</td>
<td>Static Analysis</td>
<td>Attack Surface Review</td>
<td>Testing</td>
</tr>
<tr>
<td></td>
<td>Risk Analysis</td>
<td></td>
<td>Threat Modeling</td>
<td>Peer Review</td>
<td>Continuous Feedback</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Secure Software Development LifeCycle (SSDLC)**

©2017 – Cypress Data Defense, LLC
Secure Lifecycle

Involve security through lifecycle

Security Training
Requirements
Design
Automated testing during implementation
Manual testing of critical security components during implementation
Secure Code Review and Penetration Testing
What Can I Do TODAY?

- Security Headers
- Parameterized Queries/ORM
- Treat Untrusted Data Appropriately
Questions?

Aaron
Twitter: @curea
Email: aaron.cure@cypressdefense.com

Steve
Twitter: @skosten
Email: steve.kosten@cypressdefense.com

Thanks for attending!
Thanks for attending!

Questions?

Aaron

Twitter: @curea
Email: aaron.cure@cypressdefense.com

Steve

Twitter: @skosten
Email: steve.kosten@cypressdefense.com