What is this talk about?
Agenda

What is SharePoint?
- Demo SharePoint

SharePoint Web Security
- Demo XSS and CSRF

Hardening Considerations
- Demo Hidden List and REST API

Conclusion

Quiz and Q&A
The Voting Device
The Voting Device

It enables you to participate on votings

The device has no batteries, so it works autarkic

You power it by shaking it until green light flashes
The Voting

Let’s give it a try...

Question

Answer A  A  Answer B

Answer C  C  Answer D
What is SharePoint?
What is SharePoint?

Browser-based collaboration and content management platform

The latest release of the product is SharePoint 2013
Terminology

Web Application
Site Collection
Site
Subsite
List
Custom List
Document Library
Web Parts
Apps
Web Application, Site Collection, Site

Web application

Top-level site

Subsite

Subsite

Subsite

Site collection
Web Application, Site Collection, Site

Web application

http://sps13.csnclab.ch/sites/teams/bigbang/Lists/Favourite%20Physicists/AllItems.aspx

SharePoint

BROWSE WEB APPLICATIONS

http://sps13.csnclab.ch/sites/teams/bigbang/Lists/Favourite%20Physicists/AllItems.aspx

Site Collection overview

Big Bang Theory

Team Sites Big Bang Theory IT Crowd

Home Tasks
Contacts Watch Star
Documents 2 upcoming
Pictures
Document Libraries

Document Libraries contain files:

- Team Sites
- Big Bang Theory
- IT Crowd
- Subsites
  - Big Bang Theory
  - IT Crowd
- Site Contents
- Edit Links

[Image of SharePoint interface showing documents and file management]
Lists and Custom Lists contain structured information:

- **Favourite Physicists** list in SharePoint
  - Name: Carl Sagan, Year: 11/9/1934, Still alive: No
  - Name: Richard Feynman, Year: 5/11/1918, Still alive: No
  - Name: Wolfgang Pauli, Year: 4/25/1900, Still alive: No
  - Name: Stephen Hawking, Year: 1/8/1942, Still alive: Yes
  - Name: Brian Cox, Year: 3/3/1968, Still alive: Yes
Web Part Pages

A web site in SharePoint is built with Web Part Pages

Web Part Pages host Web Parts and App Parts on a given layout
Web Parts

Web Parts render data from Doc. Libraries, Lists and other sources
Apps

Apps do the same as Web Parts, but are much more limited.

Apps do not allow any code behind to access SharePoint server-side objects.

Apps can create Lists and Doc Libraries, but they are stored in a dedicated Site Collection on a different domain.

Apps are isolated on client side in iFrames (The Same Origin Policy prevents access to the data of your web site).
Apps do the same as Web Parts, but are much more limited.

Apps do not allow any code behind to access SharePoint server-side objects.

Apps can create Lists and Doc Libraries, but they are stored in a dedicated Site Collection on a different subdomain.

Apps are isolated on client side in iFrames (The Same Origin Policy prevents access to the data of your web site).

==> To make it easier for end-users, Microsoft now calls everything an app, but technically this is not true.

==> Built-in “apps” like Tasks, Contacts or Calendar are not isolated in a dedicated site collection.
## Site Templates

<table>
<thead>
<tr>
<th>Type</th>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Collaboration</td>
<td>Team Site</td>
<td>A place to work together with a group of people.</td>
</tr>
<tr>
<td></td>
<td>Project Site</td>
<td>A site for managing and collaborating on a project.</td>
</tr>
<tr>
<td>Publishing</td>
<td>Publishing Portal</td>
<td>A starter hierarchy for an Internet-facing site or a large intranet portal.</td>
</tr>
<tr>
<td></td>
<td>Enterprise Wiki</td>
<td>A site for publishing knowledge that you capture and want to share.</td>
</tr>
</tbody>
</table>
Demo: Team Sites

IT Crowd

Project Summary

Big Bang Theory

Tasks

Dinner with Penny due in 13 days

Calendar

© Compass Security AG

Search this site

Project Summary

Big Bang Theory

Tasks

Dinner with Penny due in 13 days

Calendar

© Compass Security AG
Internet-facing Web Sites are published for Anonymous Users:

The main characters in *Lost* have killed many other people over the course of the show. Many kills were cases of self defense and the defense of others either individually or in combat. Many were murder. Of those some are *premeditated*, some are *spur of the moment*; some are *manslaughter*. At least one, that of US Marshal Edward Mars, was *euthanasia*.

*From Lostpedia Theory Policy*: “A theory is an attempt to explain a certain mystery using logic backed up with logically consistent observations and facts. Without supporting evidence, statements are merely speculation. Speculation is similar to theories except there are no facts or logic to back the theory up.” - This is a
Web Site (Publishing Portal)

Editors use an internal URL to author the content:
Flight 815 was a scheduled flight from Sydney, Australia, to Los Angeles, California, United States, on a Boeing 777. On September 22, 2004, the airliner, carrying 324 passengers, deviated from its original course and disappeared over the Pacific Ocean. This is the central moment in the series *Lost* and the personal chronological beginning of the main characters' exploits on the Island.

Flight 815's breakup and crash was due to Desmond Hume failing to enter a code into the Swan station computer, causing a large burst of electromagnetic energy. The true cause of the plane's off-course deviation and arrival to the Island's airspace was Jacob, the supernatural entity who protected the Island. Jacob brought the plane because many of the flight's passengers were candidates to replace him as protector.

Two months after the crash, wreckage was found in the Sunda Trench in the Indian Ocean near B passengers were presumed dead. In reality, however, the discovered wreckage was staged by C.

The real plane had suffered a mid-air break-up and crashed on an uncharted Island, with more than 2 crew members surviving the crash itself. Later, six of those survivors made it off the Island known as the Oceanic Six.

By late 2007, the Island's treacherous conditions and the violent battles between passengers, the inhabitants, and other factions killed nearly every passenger on the plane. As of Lost's finale episode, Austen, Hugo "Hurley" Reyes, James "Sawyer" Ford, Walt Lloyd, Rose Nadler, Bernard Nadler, Chr. Emma, and Zach were still alive. Cindy Chandler was the only crew member of the flight to survive. Some of the survivors were able to escape a failing Flight 815, which they needed to return to the...
SharePoint Web Security

Cross-Site Scripting (XSS)
Cross-Site Scripting

With Cross-Site Scripting vulnerabilities, attackers are able to execute JavaScript code in the users' context
Cross-Site Scripting – Threats

Stealing user sessions (cookie value) to gain access to the application.

Usually not relevant on SharePoint Web Applications (Windows Integrated Authentication or HttpOnly Flag set).

However, cookies of other applications on the same domain can be stolen.
Cross-Site Scripting – Threats

Sending requests on behalf of the currently logged in user.

On SharePoint 2013 the JavaScript Client-Side Object Model (CSOM) allows comfortable access to all the data stored in SharePoint.
Cross-Site Scripting – Navigation

Default SharePoint Features: **Navigation Links** (2010 only)
Cross-Site Scripting – Web Parts

Default SharePoint Features: Web Parts
(e.g. Script / Content Editor Web Parts)

Cross-Site Scripting Test Page

This is a sample script about how to insert JavaScript code in SharePoint 2013 Web Part Pages

Content Editor

test
Cross-Site Scripting – ASP.NET Pages

Default SharePoint Features: ASP.NET / Web Part Pages
(Edit or upload .aspx page)
Cross-Site Scripting – Web Parts

Pre-Conditions I:

- Users with sufficient permissions can include JavaScript by Design
- “Add & Customize Pages” Permission is required

- Not included in “Contribute” permission level and below (Since SP 2010)
- Included in “Designer” and “Full Control” permission levels

<table>
<thead>
<tr>
<th>Home</th>
<th>Type</th>
<th>Permission Levels</th>
</tr>
</thead>
<tbody>
<tr>
<td>Documents</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Subsites</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Big Bang Theory</td>
<td></td>
<td></td>
</tr>
<tr>
<td>IT Crowd</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

- Add and Customize Pages - Add, change, or delete HTML pages or Web Part Pages, and edit the Web site using a Microsoft SharePoint Foundation-compatible editor.
Cross-Site Scripting – Web Parts

Pre-Conditions II:

- Only applied consequently if Web Part Security Setting is not changed:

  - And Web Parts are declared correctly in the web.config

```xml
<SafeControl ... TypeName="*" Safe="True" SafeAgainstScript="False" ... />
<SafeControl ... TypeName="ListViewWebPart" Safe="True" SafeAgainstScript="True" ... />
<SafeControl ... TypeName="XsltListViewWebPart" Safe="True" SafeAgainstScript="True" ... />
<SafeControl ... TypeName="ImageWebPart" Safe="True" SafeAgainstScript="True" ... />
<SafeControl ... TypeName="PageViewerWebPart" Safe="True" SafeAgainstScript="True" ... />
<SafeControl ... TypeName="PictureLibrarySlideshowWebPart" Safe="True" SafeAgainstScript="True" ... />
...```

© Compass Security AG  www.csnc.ch
## Cross-Site Scripting – Web Parts

<table>
<thead>
<tr>
<th>Permission Level</th>
<th>SharePoint Version</th>
<th>Web Part Security Setting</th>
<th>SafeAgainstScript</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Full Control (Owner)</strong></td>
<td>2007 and older</td>
<td>N/A</td>
<td>N/A</td>
</tr>
<tr>
<td></td>
<td>2010 and newer</td>
<td>Allow Contributors</td>
<td>True</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Prevent Contributors</td>
<td>True</td>
</tr>
<tr>
<td><strong>Contribute (Member)</strong></td>
<td>2007 and older</td>
<td>N/A</td>
<td>N/A</td>
</tr>
<tr>
<td></td>
<td>2010 and newer</td>
<td>Allow Contributors</td>
<td>True</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Prevent Contributors</td>
<td>True</td>
</tr>
</tbody>
</table>

**Note:** The table shows the recommended security settings for different permission levels and SharePoint versions.
Cross-Site Scripting – Web Parts

Trying to inject JavaScript code as Contributor without permissions
Cross-Site Scripting – Web Sensitive Files

Web Sensitive Files can be used to embed malicious content like JavaScript

Therefore only users with “Add & Customize Pages” permissions are allowed to add and modify Web Sensitive Files

By default, Web Sensitive Files are:

- ascx
- asmx
- Aspx (ASP.NET pages)
- Jar (Java)
- master
- Swf (Flash)
- xap
- xsf
- xsn
Cross-Site Scripting – Web Sensitive Files

The Web Sensitive File Types are not complete. Other dangerous files like `.html` can be uploaded without the “Add & Customize Pages” permission.

However, SharePoint Setting “Browser File Handling” prevents the browser from automatically render malicious files by default:

- **Strict**: Content-Disposition Header is set on every file type to force the file to be downloaded instead of rendering it in the browser.

- **Permissive**: Some file types, like `*.html` are rendered in the browser. If these files contain javascript it gets executed in the context of SharePoint.
Cross-Site Scripting – Web Sensitive Files

Trying to upload ASP.NET page with JavaScript code as Contributor without permissions
Cross-Site Scripting – Web Sensitive Files

Trying to download malicious HTML page uploaded by a Contributor
Cross-Site Scripting – Mitigation

**Minimal trustful Owners:**
- Only provide Owner permission level to trustful users
- Do not allow everyone to own a Site
- Disable Self-Site Creation or at least limit it to a dedicated domain

**Careful Configuration:**
- Do not allow untrusted users to “Add & Customize Pages” *(Custom Permission Levels)*
- Do not allow Contributors to use Web Parts not declared as “SafeAgainstScripts” *(Web Part Security Settings)*
- Do not change the “SafeAgainstScripts”-declaration of preinstalled Web Parts *(Web Config Settings)*
- Do not misconfigure the list of Web Sensitive Files
- Do not misconfigure the Browser File Handling
Cross-Site Scripting – Mitigation

3rd Party Code:
- Check correct declaration of 3rd party Web Parts which by design allow to embed JavaScript ("SafeAgainstScript=false")
- Test 3rd party Web Parts declared as "SafeAgainstScript", if they do not accidently allow embedding of JavaScript (XSS Vulnerability)
SharePoint Web Security

Cross-Site Request Forgery (CSRF)
Cross-Site Request Forgery

An attacker must not be able to predict a valid request, which could be used to trick a victim to execute a given action in the already authenticated context. E.g.:

- Submitting a form to add a new user
- Submitting a form to delete a document
- ...

Common best practice is to include an unpredictable element within every request which triggers immediate actions.

This so called “Anti-XSRF” token should be included in every important html form.
Cross-Site Request Forgery

An attacker must not be able to predict a valid request, which could be used to trick a victim to execute a given action in the already authenticated context. E.g.:

- Submitting a form to add a new user
- Submitting a form to delete a document
- ...

Common best practice is to include an unpredictable element within every request which triggers immediate actions. This so-called “Anti-XSRF” token should be included in every important HTML form.
Cross-Site Request Forgery

Web Page Security Validation = Anti-XSRF Token

This feature is enabled by default on every ASP.NET page based on a SharePoint Master Page

3rd party solutions, which do not inherit from the SharePoint Master Page, must include the “FormDigest” control and check it by calling the “ValidateFormDigest” method before they execute an action
DEMO: Cross-Site Request Forgery

Google

⚠️ There was a problem.
Some of your changes may have been saved. The security validation for this page is invalid. Click Back in your Web browser, refresh the page, and try your operation again.
SharePoint Web Security

SQL Injection
The default pages and web parts of SharePoint are most likely not vulnerable to SQL Injection attacks

However, 3rd Party Solutions need to be checked carefully

- Which data sources do these pages / web parts use?
- Do they access the database?
- Do they use Stored Procedures / Prepared Statements?
- ...
Hardening Considerations

Lockdown Anonymous Access
Lockdown Feature (DE: Sperrmodus)

Lockdown mode is a Site Collection Feature that you can use to secure published sites:

Limited-access user permission lockdown mode

When this feature is enabled, permissions for users in the "limited access" permissions level (such as Anonymous Users) are reduced, preventing access to Application Pages.

By enabling lockdown mode on a site, you can remove unnecessary permissions for anonymous users.
Lockdown Feature (DE: Sperrmodus)

When lockdown mode is turned on, fine-grain permissions for the limited access permission level are reduced:

<table>
<thead>
<tr>
<th>Permission</th>
<th>Limited access — default</th>
<th>Limited access — lockdown mode</th>
</tr>
</thead>
<tbody>
<tr>
<td>List permissions: View Application Pages</td>
<td>Yes</td>
<td>No</td>
</tr>
<tr>
<td>Site permissions: Browse User Information</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>Site permissions: Use Remote Interfaces</td>
<td>Yes</td>
<td>No</td>
</tr>
<tr>
<td>Site permissions: Use Client Integration Features</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>Site permissions: Open</td>
<td>Yes</td>
<td>Yes</td>
</tr>
</tbody>
</table>

- Removes the permission to access application pages like `_layouts/viewlsts.aspx`, which can be used to shows all available lists in a site
- Removes the permission to use remote interfaces like SOAP and REST API
Lockdown Feature (DE: Sperrmodus)

View lists as authenticated user (e.g. Editor):
Lockdown Feature (DE: Sperrmodus)

View lists as anonymous user (not authorized):

```plaintext
401 UNAUTHORIZED
```

==> Access still possible with Remote Interfaces if Web Application is poorly configured!
Hardening Considerations

Remote Interfaces
Remote Interfaces – SOAP

Traditional SharePoint SOAP Web Services:

- `<site>/_vti_bin/lists.asmx`
- `<site>/_vti_bin/sites.asmx`
- ...

![Image of SharePoint SOAP Web Services](image_url)
Remote Interfaces – CSOM

Client-Side Object Model CSOM:

- Client Applications can use the web service `<site>/_vti_bin/client.svc` to access the SharePoint Object Model
- They provide better performance because they batch requests and perform all operations asynchronously
- The semantics are more familiar and easier to be used for SharePoint developers
Remote Interfaces – CSOM + REST

The CSOM can be used with...

- JavaScript or .NET APIs
- REST Endpoints on URL: `<site>/_api` (new in SP 2013)
Demo: Remote Interfaces

Passenger Information is pulled from a list called “Passenger List”
Demo: Remote Interfaces

Trying to access Passenger List as Editor (sensitive information)
Demo: Remote Interfaces

Trying to access Passenger List as Anonymous User

![Browser window displaying a 401 Unauthorized error](image-url)

- Flight 815
- `oceanic.csnclab.ch/Lists/F`
- `oceanic.csnclab.ch/Lists/Passenger%20List/AllItems.aspx`
- 401 UNAUTHORIZED
Demo: Remote Interfaces

Access Passenger List with CSOM REST API:

- http://oceanic.csnclab.ch/_api/lists
- http://oceanic.csnclab.ch/_api/lists/?$select=Title
- http://oceanic.csnclab.ch/_api/lists/getbytitle('Passenger%20List')
- http://oceanic.csnclab.ch/_api/lists/getbytitle('Passenger%20List')/Items?$select=Title
- http://oceanic.csnclab.ch/_api/lists/getbytitle('Passenger%20List')/Items?$select=Title,Credit_card_No

Raw in browser (e.g. chrome):
Demo: Remote Interfaces

REST Chrome Apps like Dev HTTP Client, Postman – REST Client:
Cause: Misconfiguration

The Lockdown Feature:
- prevents access to /_vti_bin and /_layouts folders
- removes the Remote Interface Permissions for anonymous users

But there is an **important flag on two separate locations**, which overrules the Remote Interface Permission if set wrongly.
**Cause: Misconfiguration**

## #1: Authentication Provider (Web Application Settings)

<table>
<thead>
<tr>
<th>Central Administration</th>
<th>Name</th>
<th>URL</th>
<th>Port</th>
</tr>
</thead>
<tbody>
<tr>
<td>Application Management</td>
<td>SharePoint - 80</td>
<td><a href="http://sps13.csnlab.ch/">http://sps13.csnlab.ch/</a></td>
<td>80</td>
</tr>
<tr>
<td>System Settings</td>
<td>SharePoint Central Administration v4</td>
<td><a href="http://sps13.csnlab.ch:4443/">http://sps13.csnlab.ch:4443/</a></td>
<td>4443</td>
</tr>
</tbody>
</table>

## #2: Anonymous Access Configuration (Site Permissions)

Some content on this site has different permissions from what you see here. [Show these items.](#)
Mitigation I – Remote Interface Perm.

Client Object Model Permission Requirement
You can require that the user must have the Use Remote Interfaces permission in order to use the Client Object Model to access the server. The Client Object Model is used by some parts of the UI. Enabling this prevents users from performing some tasks using the UI if they do not have the Use Remote Interfaces permission.

```xml
<?xml version="1.0" encoding="utf-8"?>
    <code>2147024891</code>, System UnauthorizedAccessException(m.code)
    <message xml:lang="en-US">Access denied. You do not have permission to perform this action or access this resource.</message>
</error>
```
Mitigation II – List Permissions

If you need CSOM for some client side features like search or other highly dynamic Ajax features:

==> Explicitly deny access for Anonymous Users on “hidden” lists
Hardening Considerations

Server-Side Controls, Sandboxing and further hardening recommendations
Server-Side Controls

Make sure that only controls from trustful sources are in the list of safe ASP.NET controls in the web.config:

- Dangerous controls which should not be usable within .aspx pages are marked as `Safe="False"` by default. (E.g. controls allowing the execution of server-side code or similar)
- Controls / Web Parts marked as `Safe="True"` can be used in .aspx pages (whitelisting).

```xml
<SafeControl ... TypeName="*" Safe="True" SafeAgainstScript="False" ... />
<SafeControl ... TypeName="ListViewWebPart" Safe="True" SafeAgainstScript="True" ... />
<SafeControl ... TypeName="XsltListViewWebPart" Safe="True" SafeAgainstScript="True" ... />
<SafeControl ... TypeName="ImageWebPart" Safe="True" SafeAgainstScript="True" ... />
<SafeControl ... TypeName="SqlDataSource" Safe="False" SafeAgainstScript="False" />
<SafeControl ... TypeName="Xml" Safe="False" SafeAgainstScript="False" />
<SafeControl ... TypeName="DataViewWebPart" Safe="False" SafeAgainstScript="False" />
```

==> The list of default controls should only be extended with a very good reason!
.NET Sandbox

Make sure that only trustful code (Web Parts / ASP.NET Controls) deployed with a Full Trust Level

Global Assembly Cache:
- Runs with a full trust level
- Can be used by every SharePoint Web Application
- Assemblies are signed and need to be installed by an Administrator

Bin directory of Web Application:
- Runs with the trust level `WSS_Minimal` (<= SP 2010)
- Only available to that single Web Application
- Can be deployed by everyone with write access to that directory
- Developers need to explicitly specify which privileges their code need (CAS polices)
.NET Sandbox

With SharePoint 2013, Microsoft changed the default trust level of code in the Bin directory to **full trust**!

```xml
</sitemap>
<trust level="Full" originUrl="" legacyCasModel="true" />
<webParts>

==> Only allow 3rd party code which is reviewed and trusted to run with a full trust level

==> Try to change trust level back to **WSS_Minimal**

==> Use Sandboxed Solutions or Apps if possible
Sandboxed Solutions / Apps

Sandboxed Solutions:
- Run isolated in the User Code Service in a dedicated process
- Running under a very strict CAS policy only allowing a minimum of calls

Apps:
- Are hosted in an isolated SharePoint site, farm or either in the cloud
- Are not allowed to use server-side code
- Need to use the client object model to connect back to the SharePoint farm

Best Practices:
- Only deploy trusted code (Farm Solutions) to the GAC or the Bin directory
- 3rd party developers should use Sandboxed Solutions or Apps
- Sandbox solutions are deprecated but are still working in SharePoint 2013
- Use SharePoint Apps for 3rd party code in SharePoint 2013
Hardening Recommendations

Is SharePoint Designer disabled?

Are unnecessary or suspicious features disabled?

Is Self-Service Site Creation disabled?

Check the list of blocked file types?

Check the Web Part gallery, which really need to be used?

Is the Developer Dashboard disabled?

Is the Audit Log configured properly?

Is HTML Field Security configured properly (SP 2013)?

Who are your Site Collection Administrators?
Hardening Recommendations

Do we use a proper architecture for our deployment?

Split back-to-back

About this diagram:
- Application servers are hosted inside the perimeter network. This option is illustrated by blue servers inside the dashed line.
- Application servers can optionally be deployed inside the corporate network, with the database servers. This option is illustrated by the gray servers inside the dashed line.
- To optimize search performance and crawling, place the application servers inside the corporate network with the database servers. You can also add the Web server role to the index server inside the corporate network and configure this Web server for dedicated use by the index server for content crawling.
Conclusion
Conclusion

SharePoint 2010+ by default secure

-> Do not misconfigure!
Conclusion

Do not trust 3rd party code (e.g. Web Parts)

- Wrong declarations in web.config?
- XSS, CSRF and SQL Injection vulnerabilities?
Quiz and Q&A
How much do you still know?
References

- MS Technet: SharePoint 2013
  http://technet.microsoft.com/de-de/library/cc303422.aspx

- Allow or prevent Contributors ability to edit scriptable Web Parts

- Are you Safe Against Script?
  http://www.bluedoglimited.com/SharePointThoughts/Lists/Posts/Post.aspx?ID=303

- Web Sensitive Files and SharePoint 2010
  http://support.microsoft.com/kb/2483447
  https://www.nothingbutsharepoint.com/sites/itpro/Pages/Web-Sensitive-Files-and-SharePoint-2010.aspx

- SharePoint Security Best Practices: Cross-Site Request Forgery

- Security Validation and Making Posts to Update Data
References

- Plan security for an external anonymous access environment

- How to lock down external anonymous access SharePoint sites

- Deciding Which SharePoint 2010 API to Use

- Programming using the SharePoint 2013 REST service

- Understanding and Programming with SharePoint Web Services
  http://de.slideshare.net/newsteplearning/understanding-and-programming-with-sharepoint-web-services

- Basic operations with the SharePoint .NET client object model
References

- Choose the right API set in SharePoint 2013

- Safe Controls list
  http://technet.microsoft.com/en-us/library/cc261736#BKMK_SafeControls

- Security for SharePoint Solutions

- Configure and deploy Web Parts

- Microsoft Windows SharePoint Services and Code Access Security

- What to do? Farm solution vs Sandbox vs App
References

- Using the Developer Dashboard

- SharePoint Hacking Diggity Project
  http://www.stachliu.com/resources/tools/sharepoint-hacking-diggity-project/

- Research for SharePoint (MOSS)
  https://www.owasp.org/index.php/Research_for_SharePoint_%28MOSS%29

- ...