November 2011

OWASP

The Open Web Application Security Project

Table of Contents

Notes from the Editor

Deepak Subramanian

OWASP Newsletter is changing. The changes in OWASP Newsletter are aimed at making the newsletter more mature and more responsible. These changes are currently being carried out by Tom Halleway, Kate Hartmann and me. We request the OWASP Project Leaders to kindly help us with these changes by keeping us updated on your various projects. We would also like to take this opportunity to welcome all security enthusiasts to contribute articles and interesting pieces of information.

The OWASP Newsletter aims to have many research publications and we welcome research articles with a great deal of enthusiasm.

Any suggestions and changes to making this NewsLetter better are appreciated.

Email: deepak.subramanian@owasp.org

Notes on Internal Projects

OWASP Projects Committee

The OWASP Projects are an integral part of the OWASP Foundation. Several tools and projects get updated regularly with frequent releases and assessments. The information provided below is updated till end of July by the OWASP Projects Committee. OWASP Newsletter will bring you the information about these projects in more detail in every issue to come.

NEW PROJECTS

OWASP Web Application Security Accessibility Project, Petr Závodský

https://www.owasp.org/index.php/OWASP_Web_Application_Security_Accessibility_Project

OWASP Cloud - 10 Project, Vinay Bansal, Shankar Babu Chebrolu, Pankaj Telang, Ken Huang, Ove Hansen

https://www.owasp.org/index.php/Category:OWASP_Cloud_-_10_Project

OWASP Web Testing Environment Project, Matt Tesauro

https://www.owasp.org/index.php/Projects/OWASP_Web_Testing_Environment_Project

OWASP iGoat Project, Kenneth R. van Wyk

https://www.owasp.org/index.php/OWASP_iGoat_Project

Opa Project, Henri Binsztok, Adam Koprowski

https://www.owasp.org/index.php/Opa

OWASP Mobile Security Project - Mobile Threat Model, Project leader not yet defined

https://www.owasp.org/index.php/Projects/OWASP_Mobile_Security_Project_-_Mobile_Threat_Model

OWASP Codes of Conduct, Colin Watson

https://www.owasp.org/index.php/OWASP_Codes_of_Conduct

OWASP GoatDroid Project, Jack Mannino

https://www.owasp.org/index.php/Projects/OWASP_GoatDroid_Project

OWASP WhatTheFuzz Project, Joe Basirico

https://www.owasp.org/index.php/OWASP_WhatTheFuzz_Project

OWASP ESAPI C++ Project, Project leader not yet defined

https://www.owasp.org/index.php/Projects/OWASP_ESAPI_C%2B%2B_Project

OWASP ESAPI C Project, David Anderson

https://www.owasp.org/index.php/Projects/OWASP_ESAPI_C_Project

OWASP Security Tools for Developers Project, Mark Curphey

https://www.owasp.org/index.php/OWASP_Security_Tools_for_Developers_Project

OWASP Data Exchange Format Project, Psiinon, Dinis Cruz

https://www.owasp.org/index.php/OWASP_Data_Exchange_Format_Project

OWASP Cheat Sheets Project, Sherif Koussa

https://www.owasp.org/index.php/Cheat_Sheets

ASSESSED RELEASES

OWASP Zed Attack Proxy Project – Release ZAP 1.3.0, Psiinon

https://www.owasp.org/index.php/Projects/OWASP_Zed_Attack_Proxy_Project/Releases/ZAP_1.3.0

NEW RELEASES

OWASP Zed Attack Proxy Project – Release ZAP 1.3.1, Psiinon

https://www.owasp.org/index.php/Projects/OWASP_Zed_Attack_Proxy_Project/Releases/ZAP_1.3.1

OWASP Hatkit Datafiddler Project – Release Hatkit Fiddler v 0.5.0, Martin Holst Swende

https://www.owasp.org/index.php/Projects/OWASP_Hatkit_Datafiddler_Project/Releases/Hatkit_Fiddler_v_0.5.0

OWASP Hatkit Proxy Project - Release Hatkit Proxy 0.5.1, Martin Holst Swende

https://www.owasp.org/index.php/Projects/OWASP_Hatkit_Proxy_Project/Releases/Hatkit_Proxy_0.5.1

OWASP Mantra - Security Framework – Release Mantra Security Toolkit – 0.61, Abhi M BalaKrishnan

https://www.owasp.org/index.php/Projects/OWASP_Mantra_-_Security_Framework/Releases/Mantra_Security_Toolkit_-_0.61

OWASP ESAPI Objective - C Project – Release Alpha, Deepak Subramanian

http://code.google.com/p/owasp-esapi-objective-c/downloads/detail?name=ESAPI_ObjC_Framework_v0.0.1_Alpha.tar.gz

OWASP X5s Project – Release - x5s v1.0.1, ChrisWeber

https://www.owasp.org/index.php/Projects/OWASP_X5s_Project/Releases/x5s_v1.0.1

OWASP ModSecurity Core Rule Set Project - Release - ModSecurity 2.2.0, Ryan Barnett

https://www.owasp.org/index.php/Projects/OWASP_ModSecurity_Core_Rule_Set_Project/Releases/Current

OWASP Esapi- Ruby, Release 0.30.0, Paolo Perego

https://rubygems.org/gems/owasp-esapi-ruby/versions/0.30.0

OWASP Communities | Builders, Breakers, Defenders

Michael Coates michael.coates@owasp.org

https://www.owasp.org/index.php/User:MichaelCoates

OWASP provides a wealth of application security materials. From the popular OWASP Top 10 document, to the learning application WebGoat, to the ESAPI web application security control library, OWASP projects and guides span many focus areas and phases of the secure software development lifecycle. Despite the quantity and quality of OWASP materials, one formidable challenge for individuals is navigating the sea of material to find projects, resources and experts that are relevant to a specific individual's objectives and responsibilities within an organization.

OWASP Communities was created to address this problem and has three primary objectives:

1. Provide a logical grouping of materials to enable individuals to easily find OWASP materials relevant to their interests and responsibilities within an organization
2. Link together people that have similar security objectives within their organization to better facilitate knowledge sharing throughout OWASP
3. Establish a focus area and target audience for each community so created projects can be customized for the intended audience

A review of the OWASP materials showed three logical groupings for the communities; builders, breakers and defenders.

The builders community consists of individuals that are involved in the creation of applications, for example, developers and application architects. OWASP projects that are geared toward secure development would fall into the builders category.

The breakers community includes individuals that have the objective of identifying flaws and vulnerabilities in applications. This includes penetration testers and quality assurance teams that focuses on security. OWASP projects that assist in locating security vulnerabilities in applications and code would fall into the breakers category.

The defenders community is the group of people that are tasked with protecting the deployed application from attacks and investigating potential intrusions. Members of the defenders community would include security monitoring teams, incident response and even system operations individuals that focus on security. OWASP projects that target security monitoring for applications, secure deployment and incident response for applications would fall into the defenders category.

Finally, it is important to realize that the goal of OWASP Communities is to help bring talented experts together on projects that directly relate to their specific line of work within the broader application security field. Although OWASP Communities divide projects into these logical groups, the goal is not to create silos of information. It is recommended for projects to select a primary OWASP community to best reach out to their target audience, but many projects will provide benefits across multiple OWASP Communities and the project should therefore provide information on how other audiences can leverage the particular OWASP project in their respective phase of the security lifecycle.

OWASP Communities are a new idea with hopes of bringing together talented individuals for the overall success of OWASP projects. By providing a scalable mechanism to group together projects with a similar target audience, increasing security communication among experts in their respective phases of application security, and providing a target audience for OWASP materials, the OWASP Communities concept can greatly enhance the quality and usability of OWASP projects, tools and guides.

For all Enquiries on this project please contact:

Michael Coates michael.coates@owasp.org

If there is any error you would like to see rectified and/or republish the contents of this article, kindly contact:

Deepak Subramanian deepak.subramanian@owasp.org

Kate Hartmann kate.hartmann@owasp.org

Protecting against XSS

Gareth Heyes, gazheyes@gmail.com

The problem as I see it

Where to start? Let me start by telling you that most of the books you read are wrong. The code samples you copy of the internet to do a specific task are wrong (the wrong way to handle a GET request), the function you copied from that work colleague who in turn copied from a forum is wrong (the wrong way to handle redirects). Start to question everything. Maybe this blog post is wrong this is the kind of mindset you require in order to protect your sites from XSS. You as a developer need to start thinking more about your code. If a article you are reading contains stuff like echo $_GET or Response.Write without filtering then it's time to close that article.

Are frameworks the answer? I think in my honest opinion no. Yes a framework might prevent XSS in the short term but in the long term the framework code will be proven to contain mistakes as it evolves and thus when it is exploited it will be more severe than if you wrote the code yourself. Why more severe? A framework hole can be easily automated since many sites share the same codebase, if you wrote your own filtering code than an attacker would be able to exploit the individual site but find it hard to automate a range of sites using different filtering methods. This is one of the main reasons the internet works today, not because everything is secure just because everything is different.

One of the arguments I hear is that a developer can't be trusted to create a perfect filtering system for a site and using a framework ensures the developer follows best guidelines. I disagree, developers are intelligent they write code and understand code, if you can build a system you can protect it because you're in the best position to.

How to handle input

When you handle user input just think to yourself “a number is a vector”, imagine a site that renders a image server side and allows you to choose the width and height of the graphic, if you don't think a number is a vector then you might not put any restrictions on the width and height of the generated graphic but what happens when an attacker requests a 100000×100000 graphic? If you're code doesn't handle the maximum and minimum inputs then an attacker can DOS your server with multiple requests. The lesson is not to be lazy about each input you handle, you need to make sure each value is validated correctly.

The process should be as follows.

1. Validate type – Ensure the value your are getting is what you were expecting.
2. Whitelist – Remove any characters that should not be in the value by providing the only characters that should.
3. Validate Length – Always validate the length of the input even when the value isn't being placed in the database. The less that an attacker has to work with the better.
4. Restrict – Refine what's allowed within the range of characters you allow. For example is the minimum value 5? 5. Escape – Depending on context (where your variable is on the page) escape correctly.

You can make things easier for yourself by placing these methods into a function or a class but don't overcomplicate keep each method as simple as possible and be very careful and descriptive with your function names to avoid confusion.

HTML context

Lets look at an example of the method above with a code sample in PHP.

```
<?php
$x = (string) $_GET[‘x’]; //ensure we get a string not array
$x = preg_replace(“/[^\w]/”,””, $x); //remove any characters that are not a-z, A-Z, 0-9 or _
$x = substr($x, 0, 10);//restrict to a maximum of 10 characters
if(!preg_match(“/^a/i”, $x)) {//this value must only begin with a or A
       $x = ‘’;
}
echo ‘<b>’ . htmlentities($x, ENT_QUOTES) . ‘</b>’; //escape everything according to context of $x
?>
```

You might be wondering why I used (string) in the code above. Lets try it without it.

Using the following: test.php?x[]=123

Results in: “Warning: substr() expects parameter 1 to be string, array given”

Because of the PHP feature which allows you to pass arrays over a GET request you can create a warning in PHP over unexpected type when trying to whitelist the value. Using type hinting ensures you get the expected type.

Great so we now understand how to restrict and escape a value. Lets look at another context.

Script context

When not in XHTML/XML mode a script tag does not decode HTML entities. If you have a value within a variable inside a script tag, question is what do you escape?

example:

```
<script>x=’value here’;</script>
```

Inside a JavaScript variable like this you have to watch out for the following ‘ and </script> using these vectors it's possible to XSS the value. The two examples are listed below.

Vector 1: `‘,alert(1),//`

Vector 2: `</script><img src=1 onerror=alert(1)>`

The second example requires no quotes and a lot of developers assume it won't be executed because it's still inside a JavaScript variable, this is clearly wrong as it executes because the browser doesn't know where the script begins and ends correctly.

To escape a value inside a script context you should JavaScript escape the value. The best way of doing this is using unicode escapes, a unicode escape in JavaScript looks like the following:

```
 <script> alert('\u0061');//"a" in a unicode escape </script>
```

You can experiment with unicode escapes using my Hackvertor tool. Please understand how they work as they will be very important to you when understanding how to protect many contexts.

```
<?php
function jsEscape($input) {
        if(strlen($input) == 0) {
                return '';
        }
        $output = '';
        $input = preg_replace("/[^\\x01-\\x7F]/", "", $input);
        $chars = str_split($input);
        for($i=0;$i<count($chars);$i++) {
                $char = $chars[$i];
                if(preg_match("/^\t$/", $char)) {
                        $output .= '\\t';//don't unicode escape but using a shorter \t instead.
Double escape remember!
                        continue;//skip a line and move on the the next char
                }
                $output .= sprintf("\\u%04x", ord($char));
         }
         return $output;
}
?>
```

It's very important you follow the same procedure as before (Validate type, Whitelist, Validate Length, Restrict, Escape) for the specific variable you're working on but this time we will convert our value into unicode escapes. A simple function to do that is as follows:

I've purposely designed this function with a few little optimisations missing, for example instead of using unicode you could use hex escapes since we restrict the range of allowed characters, alphanumeric characters are even converted when they could be replaced by their literal characters and new lines/tabs are encoded too when you could use the shorter equivalent. Lets add a line to use a literal tab character instead of \u0009. Why would you want to do this? To reduce the characters sent down the wire.

```
<?php
if(preg_match("/^\t$/", $char)) {
   $output .= '\\t';
   continue;
}
?>
```

Code to handle tab

This converts a tab specifically to “\t”, notice how we separate input and output and by using continue we can skip the input character and override it with something more specific. The full code is now below for clarity.

```
<?php
function jsEscape($input) {
       if(strlen($input) == 0) {
              return ‘’;
       }
       $output = ‘’;
       $input = preg_replace(“/[^\\x01-\\x7F]/”, “”, $input);//remove any characters outside 
the range 0x01-0x7f
       $chars = str_split($input);
       for($i=0;$i<count($chars);$i++) {
              $char = $chars[$i];
              $output .= sprintf(“\\u%04x”, ord($char));//get the character code and convert to 
hex and prefix with \u00
       }
       return $output;
}
?>
```

Exercises for this code:

1. Can you handle characters outside the ascii range?
2. Convert any non dangerous character to their escaped or literal representation.

Script context in XHTML

In the previous section you might have wondered about XHTML when I stated, “When not in XHTML/XML mode a script tag does not decode HTML entities”. In XHTML entities can be decoded even inside script blocks! Fortunately the code I provided for that section will handle that since unicode escapes are used. If you followed the exercises in that section did you make the “&” safe? That is something to think about when you are working on XHTML page. In order for XHTML to be used in the browser you have to serve the pages with the correct XHTML header. I recommend you don’t use the XHTML header.

Even though the previous examples still protect you against attack, I will show you a couple of vectors for XHTML sites.

```
<script>x=’&#39;,alert(/This works in XHTML/)//’;</script>
<script>x=’&apos;,alert(/This also works in XHTML/)//’;</script>
```

This would work in any XML based format, entities can be used to break out of strings and just a simple &lt/ will also do the trick. Don’t use XHTML or if you do unicode escape and don’t allow literal “&”.

JavaScript events

Now you know what happens in XHTML, you might be interested to know it also happens in HTML attributes. Any HTML attribute including events such as onclick will automatically decode entities and use them as if they were literal characters. Best demonstrated with a code example.

```
<div title=”&gt;” id=”x”>test</div> <script> alert(document.getElementById(‘x’).title); </
script>
```

As you can see instead of the value of the title attribute of the div element returning “&gt;” it returned “>” because it was automatically decoded. This whole process is one of the root causes of XSS, the developer didn’t understand that. Lets look at what happens with a onclick event and a variable of “x”.

Clicking on the link fired the alert because like XHTML the entities are decoded, when you are in the attribute context you need to do exactly the same as if you were in the XHTML context. Reusing your jsecape function will fully protect you from XSS in attributes and variables like this.

```
<a href=”#” onclick=”x=’&#39;,alert(1),&#39;’;”>test</a>
```

innerHTML context

I hope you’ve grasped the previous concepts because now it’s going to get slightly confusing. If you’re in the script context and you are assigning a value which writes to the dom in some way then the previous rules of escaping break down. Because although you are escaping the value correctly for the context, the context shifts once it’s applied to innerHTML. As always here is an example:

```
 <div id=”x”></div> <script> //this is bad don’t do this with innerHTML document.
getElementById(‘x’).innerHTML=’<?php echo jsEscape($_GET[‘x’]);?>’;</script>
```

Even though the string is “\u003c\u0069\u006d\u0067\u0020\u0073\u0072…” and so on it will still cause XSS because the innerHTML write will actually see the decoded characters from the JavaScript string. You need to escape for the HTML context as well as the script context, if you add XHTML to that too then it gets really really complicated. My advice is not to allow HTML when using the innerHTML context, whitelist and restrict your values and use innerText or textContent instead. If you really need HTML inside innerHTML follow the tutorial at the end on how to write a basic HTML filter for innerHTML.

CSS context

The same rules I’ve stated previously apply to CSS, a style block will not decode entities except when in XHTML/XML mode and style attributes will decode HTML entities automatically. This makes protecting against injections in the CSS context hard if you don’t know what you’re doing. In addition to the regular entities, CSS also supports it’s own format of hex escapes. The format is a backslash followed by a hex number of the required character padded optionally with zeros from 2-6 in length (vendors also supported a large amount of zero padding over the 6 length restriction). To see how it looks let use Hackvertor again to build our string.

As you can see there are quite a few combinations you can use, there are more. The CSS specification states that comments can be used and consist of C style /* */ and any hex escape can include a space after the escape to avoid the next character continuing the hex escape. E.g. to CSS \61 \62 \63 is still “abc” regardless of the spaces. Hopefully you’ve read my blog for a while and know about using entities as well as hex escapes or maybe you’ve just realised? Well yeah it’s correct you can use hex escapes, comments and html entities to construct a valid execute css value.

This leaves you with a nightmare scenario with regard to protecting css property values, IE7 and IE7 compat (on newer builds of IE) supports expressions in CSS. Which basically allows you to execute JavaScript code inside CSS values. A simplistic example here:

```
 <div style=”xss:expression(open(alert(1)))”></div>
```

I use the open() function call to avoid the annoying client side DOS of continual alert popups. Anything inside “(” and “)” of the expression is a one line JavaScript call. In the example I use a invalid property called “xss” but it’s more likely to be “color” or “font-family”. Lets take it up a notch and start to encode the CSS value and see what executes. I’ll just encode the “e” of expression to make it easier to follow.

```
Hex escape: <div style=”xss:\65xpression(open(alert(1)))”></div> Hex escape with trailing
space: <div style=”xss:\65 xpression(open(alert(1)))”></div> Hex escape with trailing
space and zero padded: <div style=”xss:\000065 xpression(open(alert(1)))”></div> Hex escape
with trailing space and zero padded and comment: <div style=”xss:\000065 /*comment*/
xpression(open(alert(1)))”></div> Hex escape with trailing space and zero padded and HTML encoded
comment: <div style=”xss:\000065 &#x2f;&#x2a;comment*/xpression(open(alert(1)))”></div> and
finally hex escape with encoded backslash with trailing space and zero padded and HTML encoded
comment: <div style=”xss:&#x5c;000065 &#x2f;&#x2a;comment*/xpression(open(alert(1)))”></div>
```

I’m sure you’ll agree that’s hard to follow and there are literally millions of combinations. Unfortunately you can’t simply hex escape the value and expect it to be safe from injection, since even encoded CSS escapes as you’ve seen can be used as vectors. The option you’re left from a defensive point of view is to whitelist every CSS property value, luckily I’ve already done that with CSS Reg and Norman Hippert kindly converted it to PHP.

Serving your pages

Every single page that’s available on the web for your site should include a doc type and a UTF-8 charset in a meta tag, now we have a shortened HTML5 header we can use the following:

```
 <!doctype html> <html> <head> <meta http-equiv=”Content-Type” content=”text/html;
charset=UTF-8” /> ... your content ....
```

This is to prevent charset attacks and E4X vectors and force your document into standards mode on IE which is also important. I also recommend you enforce standards mode by following this blog post from Dave Ross.

The last section of this long blog post will be how to write you're own filter. I don't think I'm the world's greatest programmer but I think I've worked out a cool technique to filtering content using little code and by only matching the content you want you won't get anything bad. I hope you take the basis of this code and improve it and learn from it. This code is intentionally incomplete. I wrote a more complete HTML filter called HTMLReg which you can examine if you want to improve this basic filter. But I recommend you try and improve the filter yourself and learn to break it too.

```
<script>
function yourFilter(input) {
        var output = ‘’ , pos = 0;
        input = input + ‘’; //ensure we have a string
        function isNewline(chr) {
                return /^[\f\n\r\u000b\u2028\u2029]$/.test(chr);
        }
        function outputSpace(chr) {
                if(!/^\s$/.test(output.slice(-1))  &&  !isNewline(chr))  {  //skip  new  lines  and
multiple spaces
                        output += chr;
                }
        }
        function outputChars(chrs) {
                output += chrs;
        }
        function error(m) {
                throw {
                     description: m
                   };
        }
        function parseHTML() {
                var allowedTags = /^<\/?(?:b|i|strong|s)>/,
                        match;
                        if(allowedTags.test(input.substr(pos))) {
                                match = allowedTags.exec(input.substr(pos));
                                if(match === null) {
                                        error(“Invalid tag”);
                                } else {
                                        pos += match[0].length;
                                        outputChars(match[0]);
                                }

                        } else {
                                outputChars(‘&lt;’);
                                pos++;
                        }
        }
        function parseEntities() {
                var allowedEntities = /^&(?:amp|gt|lt);/,
                        match;
                        if(allowedEntities.test(input.substr(pos))) {
```

```
                match = allowedEntities.exec(input.substr(pos));
                if(match === null) {
                    error(“Invalid entity”);
                } else {
                    pos += match[0].length;
                    outputChars(match[0]);
                }

            } else {
                outputChars(‘&amp;’);
                pos++;
            }
    }

    while(pos < input.length) {
        chr = input.charAt(pos);
        if(chr === ‘<’) {
            parseHTML();
        } else if(chr === ‘&’) {
            parseEntities();
        } else if(/^\s$/.test(chr)) {
            outputSpace(chr);
            pos++;
        } else if(chr === ‘>’) {
            outputChars(‘&gt;’);
            pos++;
        } else if(chr === ‘”’) {
            outputChars(‘&quot;’);
            pos++;
        } else if(chr === “’”) {
            outputChars(‘&#39;’);
            pos++;
        } else if(/^[\w]$/.test(chr)) {
            outputChars(chr);
            pos++;
        } else {
            pos++;//move to the next character but don’t output it
        }
    }
    return output;
}
</script>
```

The code above separates input and output and shows how to move along the input and produce a different output without losing track of the position. New lines are dropped from the HTML and more than one space this is to demonstrate how to use the output to prevent repeated characters you can and should change the behavior to suit your needs. The code is written in JavaScript but can be easily customized into your language.

Exercises

1. Can you handle attributes safely?
2. Can you convert new lines into <br> where appropriate.

For all Enquiries on this article please contact:

Gareth Heyes, gazheyes@gmail.com

If there were any error you would like to see rectified and/or republish the contents of this article, kindly contact:

Deepak Subramanian deepak.subramanian@owasp.org

Kate Hartmann kate.hartmann@owasp.org

OWASP Podcast

hosted by Jim Manico

OWASP Podcast series is hosted by Mr. Jim Manico and features a wide variety of security experts. This week we feature Mr. Jason Li from the OWASP Global Projects Committee to get a better understanding of the projects committee's working and the projects that drive OWASP.

Podcast Link: https://www.owasp.org/download/jmanico/owasp_podcast_89.mp3

OWASP Zed Attach Proxy (ZAP)

Simon Bennetts, project leader

In June the OWASP Zed Attack Proxy (ZAP) version 1.3.0 was released. This was a landmark release as it was formally assessed and granted 'Stable Release' status. It also contained significant enhancements, including:

- Fuzzing, using JBroFuzz code and including the ability to automatically regenerate Anti CSRF tokens
- Dynamic SSL certificates, so that you can generate a CA that you can trust and then intercept SSL connections transparently to the browser
- An API and 'headless' or daemon mode, which allows developers to add ZAP into their Continuous Integration environment
- BeanShell integration, so that ZAP can be dynamically extended with scripts
- Full internationalization
- Out of the box support for 10 languages (in addition to English)

Subsequent bug fix releases have demonstrated the team's commitment to supporting their users and resolving significant problems as soon as they are reported.

The project lead, Simon Bennetts (aka Psiinon), also gave talks on ZAP at both Appsec EU and AppSec USA.

With 5 developers now working on ZAP and a real community focus, ZAP continues to go from strength to strength and has provisionally be designated as one of the new 'flagship' projects.

Global Board of Directors Announced

The OWASP Foundation Board of Directors consists of six elected volunteers. These unpaid volunteers dedicate themselves to the organizational mission, supporting the OWASP Global Committees and playing a pivotal role in the software security community. OWASP conducts democratic elections of its Board Members and Committee Chairs to enable bottom-up advancement of its mission.

Each year, half of the Board seats, and all of the Committee Chair seats, are up for election by the then-current registered members. Complete information on the Global Board of Directors can be found in our current by-laws: https://www.owasp.org/images/d/d6/2011-06-OWASP-BYLAWS.pdf

The Current Board of Directors and their roles

Michael Coats
Chairman of the Board

Eoin Keary
Vice Chair

Matt Tesauro
Treasurer

Tom Brennan
Secretary

Dave Wichers
Board Member at Large

Sebastian Deleersnyder
Board Member at Large

OWASP Foundation Staff

Kate Hartmann
Global Operations
Kate.Hartmann@owasp.org

Alison Shrader
Accounting
Alison.Shrader@owasp.org

Sarah Baso
Global Conference and Chapter Committee
Sarah.Baso@owasp.org

Kelly Santalucia
Global Membership Committee
Kelly.Santalucia@owasp.org

Global Committees

Global Chapter Committee

Mission

To provide the support required by the local chapters to thrive and contribute to the overall mission and goals of OWASP.

Committee Chair: Tin Zaw

Global Conference Committee

Mission

The OWASP Global Conferences Committee (GCC) exists to coordinate and facilitate OWASP conferences and events worldwide.

Committee Chair: Mark Bristow

Global Connections Committee

Mission

To help the OWASP foundation communicate to the outside world in a unified and coherent way. We also assist with internal communication between different OWASP projects and committees.

Committee Chair: Jim Manico

Global Education Committee

Mission

Provide awareness, training and educational services to corporate, government and educational institutions on application security.

Committee Chair: Martin Knobloch

Global Industry Committee

Mission

The OWASP Global Industry Committee (GIC) shall expand awareness of and promote the inclusion of software security best practices in Industry, Government, Academia and regulatory agencies and be a voice for industry. This will be accomplished through outreach; including presentations, development of position papers and collaborative efforts with other entities.

Committee Chair: Rex Booth

Global Membership Committee

Mission

The Membership Committee recommends policies, procedures, and strategies for enhancing the membership in OWASP both numerically and qualitatively. The committee provides a written plan and recommends policies, procedures, and initiatives to assure a growing and vital membership organization.

Committee Chair: Dan Cornell

Global Projects Committee

Mission

To foster an active OWASP developer community, facilitate contributions from OWASP community members, provide support and direction for new projects, and encourage adoption of OWASP Projects by the global community at large.

Committee Chair: Jason Li

Upcoming Events

(Links to all events can be found here: https://www.owasp.org/index.php/Category:OWASP_AppSec_Conference)

- OWASP BeNeLux 2011: Nov. 30, 2011 - Dec. 1, 2011; Luxembourg
- Global AppSec AsiaPac 2012: March 12, 2012 - March 16, 2012 Sydney, Australia
- AppSec DC 2012: April 2, 2012 - April 5, 2012; Washington, DC
- Global AppSec Research 2012 (Wiki) July 10, 2012 - July 13, 2012 Athens, Greece
- Global AppSec North America 2012 Oct. 22, 2012 - Oct. 26, 2012 Austin, TX TBD
- Global AppSec Latin America 2012 Nov. 14, 2012 - Nov. 16, 2012

Organization and Barter In Trade Supporters

acunetix

ACCUVANT LABS

Adobe

ADP

aeCERT

Akamai

amazon

AppliedTrust

ascure

ASPECT SECURITY Application Security Experts

ASTECH security

art defence

BEST BUY

Black Hat

Cargill

cigital

CORE SECURITY TECHNOLOGIES

DENIM GROUP

DIGITAL DEFENSE INCORPORATED

DREAMLAB TECHNOLOGIES

ENGINEERING

fishnet SECURITY

FUJITSU

gemalto security to be free

GOTHAM DIGITAL SCIENCE

HARRIS CONNECT

HIGH-TECH BRIDGE INFORMATION SECURITY SOLUTIONS

HUAWEI

IBM

(ISC)^{2}

IMPERVA

Information Builders

INTREPIDUS GROUP

IOActive COMPREHENSIVE COMPUTER SECURITY SERVICES

mozilla Firefox

mnemonic -securing your business

nixu

NetIQ

NOKIA

ORACLE

Penta SECURITY

PRAETORIAN

protiviti Independent Risk Consulting

QUALYS On Demand Security

Quotium

rackspace

Redspin

salesforce.com Success On Demand

SD ELEMENTS

Security Innovation

SECURITY PS

Seeker

Symantec

TENABLE Network Security

Trustwave

ups

WhiteHat SECURITY

Academic Supporters

ADELPHI UNIVERSITY

Anglia Ruskin University

AUM

CIT Instituid Teicneolaíochta Chorcaí
Cork Institute of Technology

DARTMOUTH
VOX CLAMANTIS IN DESERTO
1769

DAKOTA STATE
dsu
UNIVERSITY

DCU

foi
UNIVERSITY OF ZAGREB
FACULTY OF ORGANIZATION AND INFORMATICS
VARAŽDIN

KATHOLIEKE UNIVERSITEIT
LEUVEN

hochschule mannheim

heig-vd
Haute Ecole d'Ingénierie et de Gestion
du Canton de Vaud

HOCHSCHULE FURTWANGEN UNIVERSITY HFU

IDC HERZLIYA
Efi Arazi School
of Computer Science

ISCTE IUL
Instituto Universitário de Lisboa
Lisbon University Institute

TECNOLÓGICO DE MONTERREY

isep

itb
Institute of Technology
Blanchardstown
Institiúid Teicneolaíochta
Baile Bhlainséir

NYP
Nanyang
Polytechnic

ORT
UNIVERSIDAD ORT
Uruguay

RUTGERS
THE STATE UNIVERSITY
OF NEW JERSEY

TRINITY
COLLEGE
DUBLIN

SGU®

STEVENS
INSTITUTE of TECHNOLOGY
THE INNOVATION UNIVERSITY
1870

TEMASEK
POLYTECHNIC

UCIRVINE

NKU NORTHERN KENTUCKY UNIVERSITY

northumbria
UNIVERSITY

NORTH TEXAS

THE GEORGE
WASHINGTON
UNIVERSITY
WASHINGTON DC

NYU·poly
POLYTECHNIC INSTITUTE OF NYU

ΤΕΙ
ΠΑΡΙΣΑΣ

FACULTAD
DE INGENIERIA
Universidad de Buenos Aires

UCLA

UdeMM
Universidad de la
Marina Mercante
Tu futuro es nuestra meta.

UNIVERSITI KUALA LUMPUR

UT DALLAS

The OWASP Foundation

The Open Web Application Security Project (OWASP) is an international community of security professionals dedicated to enabling organizations to conceive, develop, acquire, operate, and maintain applications that can be trusted. All of the OWASP tools, documents, forums, and chapters are free and open to anyone interested in improving application security. We advocate approaching application security as a people, process, and technology problem because the most effective approaches to application security includes improvements in all of these areas. OWASP is a new kind of organization. OWASP is not affiliated with any technology company, although we support the informed use of commercial security technology. Our freedom from commercial pressures allows us to provide unbiased, practical, cost-effective information about application security. Similar to many open-source software projects, OWASP produces many types of materials in a collaborative, open way.

Core Values

- OPEN – Everything at OWASP is radically transparent from our finances to our code
- INNOVATION – OWASP encourages and supports innovation/experiments for solutions to software security challenges
- GLOBAL – Anyone around the world is encouraged to participate in the OWASP community
- INTEGRITY – OWASP is an honest and truthful, vendor neutral, global community

OWASP Membership

The professional association of OWASP Foundation is a not-for-profit 501c3 charitable organization not associated with any commercial product or service. To be successful we need your support. OWASP individuals, supporting educational and commercial organizations form an application security community that works together to create articles, methodologies, documentation, tools, and technologies

A complete list of all OWASP members can be found here: https://www.owasp.org/index.php/Membership

Individual Supporter -$50 USD/year

- Underscore your awareness of web application software security
- Receive Discounts to attend OWASP Conferences
- Expand your personal network of contacts
- Obtain an owasp.org email address
- Allocate 40% of your membership dues to directly support your local chapter
- Participate in Global Elections and vote on issues that shape the direction of the community

Corporate Supporter - $5,000 USD/year

- Tax deductible donation
- Receive Discounts at OWASP Conferences to exhibit products/services
- Opportunity to post a rotating banner ad on the owasp.org website for 30 days at no additional cost ($2,500 value)
- Be recognized as a supporter by posting your company logo on the OWASP website
- Be listed as a sponsor in the quarterly newsletter distributed to over 10,000 individuals
- Have a collective voice via the Global Industry Committee
- Participate in Global Elections and vote on issues that shape the direction of the community
- Allocate 40% of your annual donation to directly support your choice of chapter and/or projects

For More information on sponsorship opportunities, contact Kelly Santalucia at Kelly.santalucia@owasp.org

Newsletter Advertising:

- 1/4 page advertisement $2000
- 1/2 page advertisement $2500
- 1/2 page advertisement + either a 30 rotating banner on the OWASP site or 10 copies of the Top 10 Books $3000
- full page advertisement $5000
- Year subscription (1 newsletter every quarter with the 1/2 page advertisement posted) $9000.

Please contact Kelly.Santalucia@owasp.org or Kate.Hartmann@owasp.org for details.