Shortcuts for Understanding Malicious Scripts

Evan H Dygert
Principal Consultant

Dygert Consulting, Inc.
evand@dygertconsulting.com

SANS

from the most trusted name in information security

About Me

- SANS Instructor
- Blue Cross Blue Shield Association Senior Security Engineer
- Software Developer, Expert Witness
- B.S. in Computer Science, M.B.A
- SANS CyberGuardian Red and Blue teams
- Certifications: GSE, CISSP, CCE, GCFA, GPEN, GREM, etc.

Goals

- Understand the actions taken by malicious scripts.
- Don’t spend a lot of time doing it.

Overview

- We will focus on JavaScript.
- Concepts and techniques apply generally.
- Obfuscation techniques and how to deobfuscate.
- Will not cover extracting scripts from documents.
- Will not cover reverse engineering.

Obfuscation Techniques

- Minification
- Visual Noise
- Function name/keyword substitution
- Obscure language features (e.g. JS tuples)
- Encoding/Encryption
- Multiple levels of obfuscation
- JavaScript obfuscation web sites

Minification

- Remove whitespace from script.
- Rename variables and functions with smaller names.
- JSCompress.com
- To reformat use beautification tool like js-beautify or website.

Visual Noise

- Increase difficulty of reading code without changing its functionality.
  - Spurious comments
  - Dead code
  - Long names
  - String splitting
  - Character substitution (e.g. replace)

Removing Visual Noise

- How to deobfuscate
  - Manually remove noise.
  - Write a script.
  - Extract meaningful code.

Character Encoding

- Encodings
  - Hex (just hex characters)
  - Backslash Hex (\x<n>)
  - Ampersand Hex (&H<n>)
  - Backslash Unicode (\u<n>)
  - Percent Unicode (%u<n>)
  - Octal (\<n>)

Deobfuscating Character Encoding

- Normalize encoded chars to readable characters.
- Didier Stevens tools (base64dump.py, numbers-to-string.py, etc.)
- Custom script

Other Encoding/Encryption

- Base64
- Encryption (e.g. AES)
- Custom
- How to deobfuscate:
  - Let the script do it for you.
  - Custom script

Deobfuscation Principles

- Make the script do the work.
- Don’t sweat the details.
- Beautify the script.
- Look for anything recognizable.
- Peel back the layers.

Tools

- Didier Stevens Suite
  - oledump.py: Analyse MS Office files.
  - pdfid.py/pdf-parser.py: Analyze PDF files.
  - base64dump.py: Extract base64 and hex encoded strings.
  - js-file/js-ascii (modified SpiderMonkey): Run JavaScript outside browser.

More Tools

- My custom scripts
  - urldecode.py
  - strip_xml
  - combine_strings
  - decode_chars.py: Decode mixed encodings.
- js-beautify
- Linux tools (grep, sed, awk, cut, etc.)

Sample 1: Embedded DDE

- Commands:
  - 1. oledump.py 1.docx (finds nothing)
  - 2. unzip 1.docx -d sample1
  - 3. cd sample1/word
  - 4. less document.xml
  - 5. strip_xml document.xml
- Lesson: Strip out the noise.

Sample 2: Web Page JavaScript

- Commands:
  - 1. wget http://cfarm.com.tw/wp-setting.js (no longer there)
  - 2. js-beautify wp-setting.js
  - 3. js-file wp-setting.js
  - 4. less write.log
  - 5. search for <script> and <form> tags

Sample 2: Web Page JavaScript Lessons

- Lessons:
  - Download payloads ASAP (SAFELY!)
  - Do recon on file (js-beautify).
  - Use js-file to see what happens

Sample 3: Raw JavaScript

- Commands:
  - 1. less 20170504...
  - 2. decode_chars.py 20170504...
  - 3. Step 2 plus | urldecode.py
  - 4. Step 3 plus | js-beautify -
  - 5. Step 4 plus | combine_strings

Sample 3: Raw JavaScript Lessons

- Lessons
  - Decode encoded characters.
  - Chain commands to accomplish goal.
  - Learn Linux text commands.
  - Write custom scripts.
  - Learn obfuscation techniques.

Sample 4: Web Page JavaScript 2

- Commands:
  - 1. less UpdateForm.html
  - 2. extractscripts.py UpdateForm.html
  - 3. vi script.1. (from step 2)
  - 4. Edit bottom 4 lines
    - a. Delete document.createElement
    - b. Replace next line with print(genr51ac)
    - c. Delete last two lines
  - 5. js-file script.1.

Sample 4: Web Page JavaScript 2 (cont.)

- Commands:
  - 6. wget <url from step 5> (no longer there)
  - 7. js-beautify 151f... > 1.js
  - 8. vi 1.js
  - 9. Understand logic
  - 10. Delete last line (window.onload)
  - 11. Delete function PP1s7UduU() line
  - 12. Delete last bracket

Sample 4: Web Page JavaScript 2 (cont.)

- Commands:
  - 13. js-file 1.js
  - 14. less write.log (from step 13)
  - 15. Analyze form submission logic

Sample 4: Web Page JavaScript 2 Lessons

- Lessons
  - Extract scripts from HTML.
  - Look for x.src = and print expression on right hand side.
  - Delete unnecessary lines.
  - Find logic that prevents script from running outside of browser and edit so it will.
  - Search for <script> and <form> tags.
  - Search for submit.

Sample 5 Raw JavaScript 2

- Commands:
  - 1. cp 2872.js 1.js
  - 2. js-beautify 1.js > 2.js (need to delete 1st line)
  - 3. js-file 2.js
  - 4. gedit 2.js and fix error (delete if statement and closing brace)
  - 5. js-file 2.js
  - 6. cat eval.001.log
  - 7. cat eval.001.log | urldecode.py
  - 8. Step 7 plus | js-beautify

Sample 5 Raw JavaScript 2 Lessons

- Lessons
  - Find errors that prevent script from running outside of browser and edit so it will.
  - Resolve errors (not always easy).

Sample 6 Raw JavaScript 3

- Commands:
  - 1. js-file 1.js
  - 2. less eval.001.log
  - 3. js-beautify eval.001.log | less
  - 4. Note that Base64 data is present.
  - 5. base64dump.py eval.001.log
  - 6. base64dump.py -s 8 -d eval.001.log
  - 7. Examine result (PowerShell)

Sample 6 Raw JavaScript 3 Lessons

- Lessons
  - Errors can sometimes be ignored.
  - Always look for base64 strings and extract them.

Sample 7 Raw JavaScript 4

- Commands:
  - 1. less DOC1...
  - 2. decode_chars.py DOC1... | less
  - 3. cp DOC1... 1.js
  - 4. vi 1.js
  - 5. delete Line 304 (ActiveXObject)
  - 6. Replace _0x57064b['\x52\x75\x6e'] with print and remove last parameter to function call.
  - 7. js-file 1.js

Sample 7 Raw JavaScript 4 Lessons

- Lessons
  - Decode characters for better visibility.
  - Look for key locations to use print such as ActiveXObject Run, WScript.Run, IEX/Invoke-Expression, etc.).

Sample 8 Raw JavaScript 5

- Commands:
  - 1. less 20170110...
  - 2. js-file 20170110...
  - 3. vi 20170110...
  - 4. Find WScript on line 6
  - 5. Find edeb
  - 6. Find cqorobcit
  - 7. Find tdurot ("run")
  - 8. Copy line 338 to clipboard

Sample 8 Raw JavaScript 5 (cont.)

- Commands:
  - 9. Create new 1.txt file with copied line.
  - 10. cat 1.txt | sed -r 's/;/;\n/g' | tr –d "^" > 2.txt
  - 11. subit.py 2.txt | combine_strings
- Lessons
  - Focus on what matters (i.e. WScript).
  - Ignore garbage code.
  - Extract and work with meaningful code.
  - Use Linux tools to clean up results for clarity.
  - Follow the clues.

Conclusion

- You can efficiently analyze malicious scripts.
- Don’t worry if you don’t understand all the details.
- Focus on the key functionality.
- Go have fun!

- Email for slides and files: evand@dygertconsulting.com

References I

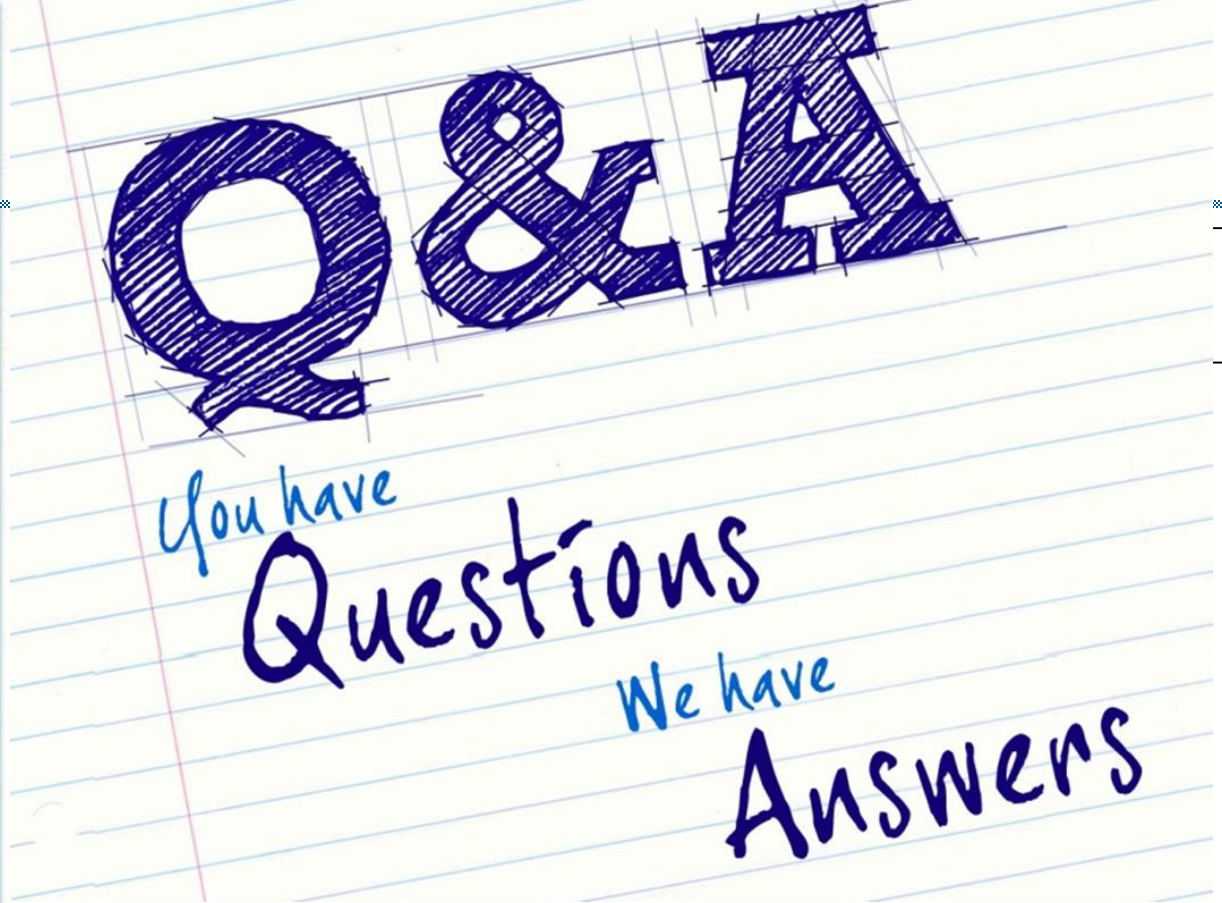
- CuteSoft Components, (2018). Javascript obfuscator. https://javascriptobfuscator.com/Javascript-Obfuscator.aspx
- JSCompress.com. (2018). Jscompress - the javascript compression tool. http://jscompress.com/
- Lielmanis, E. (2018). Online javascript beautifier. http://jsbeautifier.org/
- SophosLabs. (2018). Malware with your mocha? https://www.sophos.com/en-us/medialibrary/PDFs/technical%20papers/malware_with_your_mocha.pdf?la=en.pdf?dl=true
- Stevens, D. (2018a). Didier Stevens spidermonkey. https://blog.didierstevens.com/programs/spidermonkey/
- Stevens, D. (2018b). Didier Stevens suite. https://blog.didierstevens.com/didier-stevens-suite/
- Tools, D. (2018). Javascript obfuscator. https://www.danstools.com/javascript-obfuscate/index.php

References II

- Various. (2018a). Javascript malware collection. https://github.com/HynekPetrak/javascript-malware-collection
- Various. (2018b). Js malicious dataset. https://github.com/geeksonsecurity/js-malicious-dataset

Slides

- **Come learn more about malware at SANS San Francisco Fall 2018: Nov. 26-Dec. 1 where I will be teaching FOR610: Reverse Engineering Malware.**
- **Slides: evand@dygertconsulting.com.**



from the most trusted name in information security

Q & A

- Please use GoToWebinar's Questions tool to submit questions to our panel.

- Send to "Organizers" and tell us if it's for a specific speaker.