Small Is Beautiful
How to improve security by maintaining less code
About Me

- Natalie Silvanovich AKA natashenka
- Project Zero member
- Previously did mobile security on Android and BlackBerry
- Defensive-turned-offensive researcher
Attack surface reduction
Bugs that (maybe) shouldn’t be

- Unused features
- Old features
- Code sharing
- Third-party code
- Excessive SKUs and branching
- Privilege reduction and sandboxing
Unused Features

- All code has risk, so adding a feature is a tradeoff
  - Make sure it’s worth it!
Array.species

“But what if I subclass an array and slice it, and I want the thing I get back to be a regular Array and not the subclass?”

class MyArray extends Array {
    static get [Symbol.species]() {
        return Array;
    }
}

- Easily implemented by inserting a call to script into *every single* Array native call
CVE-2016-7200 (Array.filter)

- Bug in Array conversion due to Array.species
class dummy{
    constructor(){ return [1, 2, 3]; }
}

class MyArray extends Array {
    static get [Symbol.species](){ return dummy; }
}

var a = new MyArray({}, [], "natalie", 7, 7, 7, 7, 7);

function test(i){ return true; }

var o = a.filter(test);
CVE-2016-7200 (Array.filter)

RecyclableObject* newObj = ArraySpeciesCreate(obj, 0, scriptContext);
...
newArr = JavascriptArray::FromVar(newObj);
...
if (!pArr->DirectGetItemAtFull(k, &element))
...
selected = CALL_ENTRYPOINT(callBackFn->GetEntryPoint(), callBackFn,
CallInfo(CallFlags_Value, 4), thisArg, element, JavascriptNumber::ToVar(k,
scriptContext), pArr);

if (JavascriptConversion::ToBoolean(selected, scriptContext))
{
    // Try to fast path if the return object is an array
    if (newArr)
    {
        newArr->DirectSetItemAt(i, element);
    }
Array.species, etc.

- Very uncommon Symbol
  - ~150k pages on the internet
CVE-2019-8717

- IPComp is a standardized protocol for compressing IP
  - Rarely used, and broken in MacOS
- Use-after-free in MacOS implementation
- Fixed by removing support for IPComp
If I’ve learned one thing in life, it’s you can never go back.

Stephanie Zimbalist
⚠️ This is an experimental technology

Because this technology's specification has not stabilized, check the [compatibility table](#) for usage in various browsers. Also note that the syntax and behavior of an experimental technology is subject to change in future versions of browsers as the specification changes.
intent to remove DOMNodeInsertedIntoDocument, DOMNodeRemoved, DOMNodeRemovedFromDocument mutation events

21 posts by 14 authors

Ojan Vafai

DOMNodeInsertedIntoDocument, DOMNodeRemoved, DOMNodeRemovedFromDocument are used on 0.03%, 0.01%, 0.01% of documents respectively. They present an undue implementation burden, have been removed from the DOM4 spec and have general support from browser vendors to get rid of them. Also, MutationObservers have shipped for a while now and provide all the needed functionality from these events.

Unfortunately, this still leaves in DOMSubtreeModified, DOMNodeInserted and DOMCharacterDataModified, which are used on 0.47%, 1.02% and 0.23% of documents. Hopefully the usage numbers will drop over time and we can remove support for these as well.
Description of the security update for Office 2016: April 11, 2017

Microsoft has released an update for Microsoft Office that turns off the Encapsulated PostScript (EPS) Filter in Office by default, as a defense-in-depth measure. We are aware of limited targeted attacks that could leverage an unpatched vulnerability in the EPS filter, and we are taking this action to help reduce customer risk until the security update is released.

We strongly recommend against turning on the EPS filter at this time. However, customers who need to turn on the EPS filter can reference KB Article 2479871:

Disabling VBScript execution in Internet Explorer 11

By Microsoft Edge Team

VBScript is deprecated in Internet Explorer 11, and is not executed for webpages
- Base features on user need
- Track feature use in beta or production
- Be willing/able to disable features
IT IS NEVER TOO LATE TO DO THINGS DIFFERENTLY.
Old features

- Sometimes features fall slowly into disuse
- Lack of attention to unused code can be make it higher risk
CVE-2017-2988

● Dangling reference issue in Flash
● Result of a hack made to load macromedia.com in 2003
  ○ Deleting a MovieClip in onKillFocus
● Reported in 2017
CVE-2017-3558

- Memory corruption issue in Virtual Box allowing guest-to-host escalation
- Caused by old code not being fully removed
- Also fixed in upstream but not downstream
void ip_input(PNATState pData, struct mbuf *m){
    register struct ip *ip;
    [...] 
ip = mtod(m, struct ip *);
    [...] 
    {
        [...] 
        /*
           * XXX: TODO: this is most likely a leftover spooky action at
           * a distance from alias_dns.c host resolver code and can be
           * g/c'ed.
           */
        if (m->m_len != RT_N2H_U16(ip->ip_len))
            m->m_len = RT_N2H_U16(ip->ip_len);
    }
CVE-2019-8661

- Heap overflow in iMessage (Mac only) due to URL deserialization
- Occurred due to processing bookmark format from 2011
- Fixed by removing ability to parse that format
● Track feature use
  ○ Content stats are also possible
  ○ Compare usage to reported security issues
● Prune trees regularly
  ○ Track unmodified code
  ○ Refactor code if necessary
● Make sure all code has an owner
Code Sharing

- Using the same code for multiple purposes can expose it to new and unnecessary attack vectors
- Multiple copies of same code are difficult to maintain
CVE-2015-7894, etc

- 7 memory corruption issues in Samsung S6 Edge image processing
- Due to bugs in QJPG by QURAM
<table>
<thead>
<tr>
<th>Year</th>
<th>Year</th>
<th>History</th>
</tr>
</thead>
<tbody>
<tr>
<td>2008</td>
<td>History</td>
<td>Awarded Samsung Best Partnership for mobile solutions</td>
</tr>
<tr>
<td>2007</td>
<td>History</td>
<td>Established newly corporate identity using 1-master to Quram (meaning Quality Program)</td>
</tr>
<tr>
<td>2005</td>
<td>History</td>
<td>Contracted with Samsung Electronics for CODEC solutions</td>
</tr>
<tr>
<td>2002</td>
<td>History</td>
<td>Selected one of the best companies incubated by Ajou University</td>
</tr>
<tr>
<td></td>
<td></td>
<td>One of the best Technology Companies registered by Korean Government</td>
</tr>
<tr>
<td>2001</td>
<td>History</td>
<td>Corporation founded</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Business Partnership with Secutech, Inc (registered)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>New Excellent Technology development company designated by Korean Government</td>
</tr>
</tbody>
</table>
CVE-2015-7894, etc

- Old code
- Context issue
Android WebView Issues

- Several Android features contained their own version of WebView
- Bugs were sometimes fixed in one version but not another
- Moved to unified WebView
iMessage Issues

- Reported 7+ remotely exploitable issues in iMessage deserialization
- Bugs occurred because deserializing a class allows its subclasses to also be deserialized
- Useful in some uses of serialization, but not iMessage
- Added deserialization mode to prevent subclass deserialization
Prevention

- Make sure each attack surface only supports needed features
- Avoid multiple copies of the same library
  - Consider extending this to third parties if applicable
Third-party code

- Misuse
- Extra features
- Lack of updates
- Unexpected interactions
CVE-2016-4117 (666)

- Remote code execution in FireEye MPS
- Caused by JODE Java Decompiler executing Java classes
- JODE was never intended to be used on untrusted code
Linux Kernel Configuration and Android (feature reduction)

- CVE-2017-7308 is a memory corruption issue in Linux
- Requires CAP_NET_RAW on Android due to CONFIG_USER_NS not being defined
- Good design from both Android and Linux perspective
Project Zero

News and updates from the Project Zero team at Google

Tuesday, November 29, 2016

Breaking the Chain

Posted by James Forshaw, Wielder of Bolt Cutters.

Much as we’d like it to be true, it seems undeniable that we’ll never fix all security bugs just by looking for them. One of most productive ways to dealing with this fact is to implement exploit mitigations. Project Zero considers mitigation work just as important as finding vulnerabilities. Sometimes we can get our hands dirty, such as helping out Adobe and Microsoft in Flash mitigations. Sometimes we can only help indirectly via publishing our research and giving vendors an incentive to add their own mitigations.

Third-party mitigations are important and it’s mitigations developed for Chrome on Windows that help.
Lack of Updates

- Android libraries
  - WebView
  - Media
  - Qualcomm
  - Linux
Lack of Updates

A puppy isn't for Christmas...

A puppy is forever
Prevention

- Track third-party software
- Have an internal process for use
- Trim unnecessary features
- (Security) update frequently
Excessive SKUs and branching

- Every SKU and branch makes it harder to push security updates
- Can introduce bugs
- May patch incompletely
Too Many SKUS
Vendor 1

- New product shares code with old product
- Determined bug was in both products
- Forgot to merge fix into old product
- Forgot to merge different issue into all branches
  - Root cause: build failure
Vendor 2

- Releases ~365/SKUs per year with unclear support periods
- Could not fix bugs in 90 days
- Could not tell us fix date
- Could not tell us fix saturation
CVE 2019-2215

- Use-after-free in binder (Android kernel)
- Patched December 2017 without security advisory
- Fix not added to prebuilt kernel in most Android branches
- Discovered usage in the wild in October 2019
Prevention

- Avoid branching. Avoid SKUs.
- Make it difficult to branch
- Have a documented support period for every branch/SKU/product
  - Make downstream do it too
- Robustly test every branch/product
Sandboxing and Privilege Reduction

- Privilege reduction -- restricting the permissions of a high-risk process
- Sandboxing -- reducing privileges and using an intermediary program to perform privileged operations
Sandboxing and Privilege Reduction

- Makes existing vulnerabilities less severe
- Can require splitting functionality into separate processes
- Requires a clear risk analysis of all functionality in a piece of software
- Requires evaluating functionality versus current permissions on an ongoing basis
Stagefright Privilege Reduction

- Android reduced the privileges of media processing after many vulnerabilities in the component
- Required moving some functionality into a service
REDUCE
REUSE
RECYCLE
Conclusions

● Consider the security impact of features in design
● Track feature use, and remove old and unused features
● Carefully consider third-party code and keep it up to date
● Reduce SKUs and branches
● Have a clear support period for every product
● Sandbox and reduce privileges of necessary high-risk code
Conclusions

Only you can reduce attack surface
Questions

http://googleprojectzero.blogspot.com/
@natashenka
natashenka@google.com