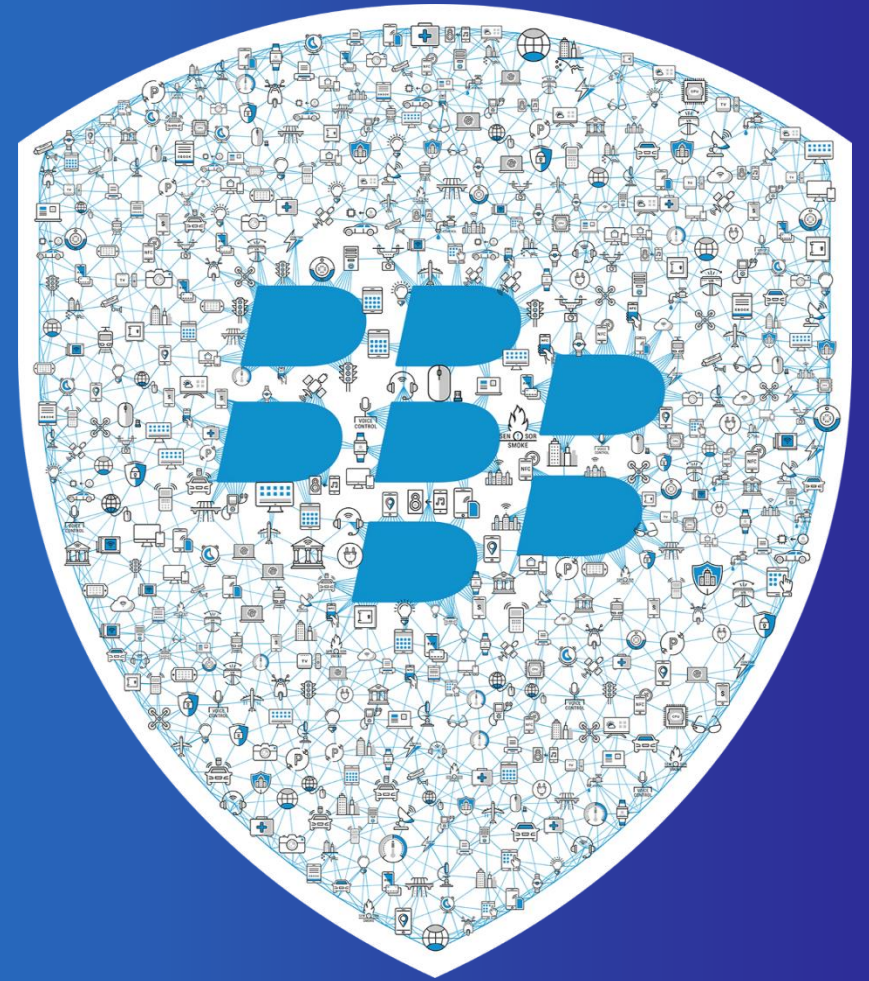


LET'S BUILD AN OSS VULNERABILITY MANAGEMENT PROGRAM!

June 2018



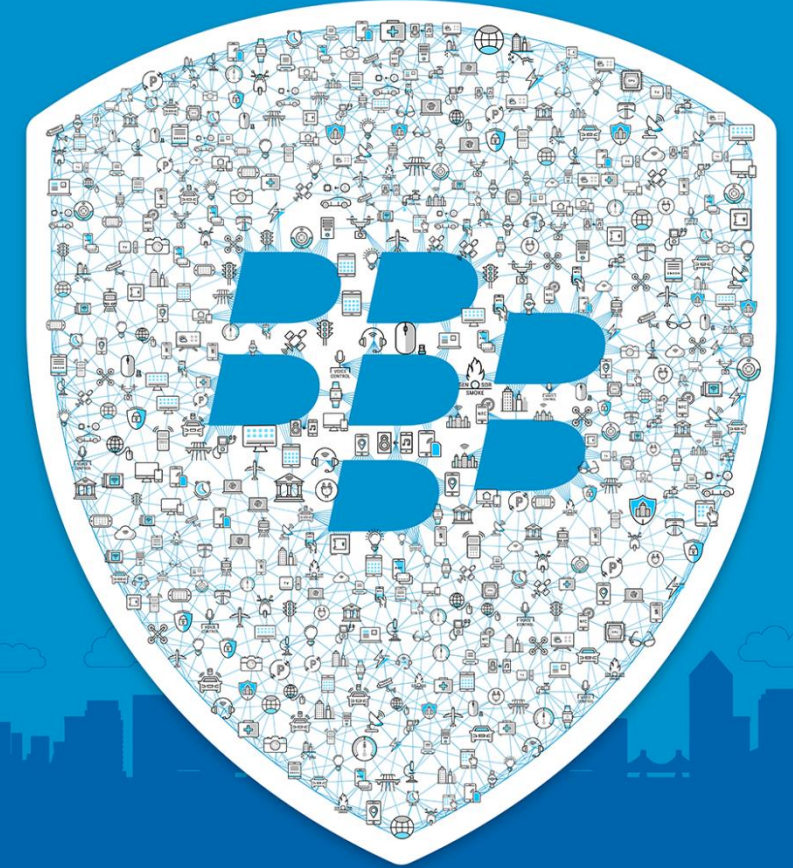
Agenda

- Introduction to Open Source Software
- Open Source Software Challenges and Risks
- Bill of Materials
- Sourcing Threat Intelligence
- Investigation and Remediation
- Maturing the Process



Who am I?

- 5 years at BlackBerry Product Security
- Lead Security Program Manager
 - Incident management
 - Vulnerability management
 - Risk management
 - Coordinated disclosure
- Mobile malware / Spyware investigator



BlackBerry Secure

Open source software libraries (OSS)

- A collection of common software functions publicly available to integrate into a product
- Source code available for anyone to contribute to or inspect
- “Open source makes sense when a software product reaches commodity status”
-Carl Grant
- Example libs:
 - OpenSSL
 - LibTiff
 - OpenJPEG
 - LibXML2



Open Source Initiative

<https://opensource.org/>

Disclaimer

- Nothing in this talk is meant to disparage Open Source
- Vulnerabilities happen whether it's open source, closed source, paid, free
- Maturity in using and maintaining Open Source installations



Enterprises being bitten by out of date OSS

- <https://googleprojectzero.blogspot.ca/2016/06/how-to-compromise-enterprise-endpoint.html>
- Popular AV products being shipped with components containing 7 year out of date OSS libraries
- Dozens of vulnerabilities with public exploits!



BlackBerry's open source usage (2017)

- 600+ unique OSS libraries used across 80+ products
- 8,000 OSS object references
- 20,000+ vulnerabilities investigated



Example: multiplatform client/server product

- Product contains Android, iOS, Mac, Windows, and Web app variants
- 100 unique open source libraries
- Over 300 vulnerabilities in 2017
- CVSS scores from 2.4 to 9.8 (avg 6.5)



Challenges with open source

- Steady stream of vulnerabilities being found and disclosed at unpredictable intervals
- OSS codebase is too large to perform proper security audits before integrating
- As an integrator of the library you have no part in the disclosure process



Is anyone at your company managing OSS?

- Do you know what OSS is being used in your products? Are you sure?
- Is someone in legal already working on licensing conformance?
- Is someone responsible for pulling in vulnerability patches?
- Many major companies still failing to address OSS vulnerabilities in a meaningful way



Building the BOM

- Bill of materials is a master list of all OSS used within a product
- The accuracy of the BOM will greatly impact your effectiveness for the rest of your efforts
- Where can we gather our data?
 - Manual inspection of code repository
 - Architecture documentation
 - Developer interviews
 - Project Tracker / Ticketing System
 - Legal team



Example BOM

Third Party Component	Version	Source Location	License	iOS	Android	Windows Desktop	MacOSX	Web
boost	1.65.1	http://www.boost.org/	Boost 1.0 License		x	x	x	
Django	1.9.2	https://github.com/django/django	DSF					x
libcurl	7.57.0	http://curl.haxx.se/	Curl License	x	x	x	x	
libvpx	1.6.0	https://chromium.googlesource.com/webm/libvpx	BSD-3	x	x	x	x	
OpenSSL	1.1.0g	http://www.openssl.org/	OpenSSL Combined License	x	x	x	x	x
sqlite3	3.21.0	https://www.sqlite.org/	Public Domain	x	x	x	x	

Sourcing threat intel

- Paired with the BOM you are ready to start making meaningful impact to reduce risk
- Free sources of threat intel:
 - <https://www.cvedetails.com/>
 - <https://nvd.nist.gov/>
 - <https://www.debian.org/security/>
 - <http://seclists.org>
 - <https://twitter.com/CVEnew>
 - Forums!
 - Google Alerts
 - Mailing Lists

Investigating your backlog

[Openssl](#) » [Openssl](#) : Vulnerability Statistics

[Vulnerabilities \(185\)](#) [CVSS Scores Report](#) [Browse all versions](#) [Possible matches for this product](#) [Related Metasploit Modules](#)

[Related OVAL Definitions](#) : [Vulnerabilities \(316\)](#) [Patches \(350\)](#) [Inventory Definitions \(1\)](#) [Compliance Definitions \(0\)](#)

[Vulnerability Feeds & Widgets](#)

Vulnerability Trends Over Time

Year	# of Vulnerabilities	DoS	Code Execution	Overflow	Memory Corruption	Sql Injection	XSS	Directory Traversal	Http Response Splitting	Bypass something	Gain Information	Gain Privileges	CSRF	File Inclusion	# of exploits
1999	1									1					
2000	1														
2001	1														
2002	4	2	3	2											
2003	8	5	1	3							2				
2004	3	3													
2005	4														
2006	5	3		1											
2007	4	1	2												
2008	2	2													
2009	12	7		2						1					
2010	12	4	3	1						2	1				
2011	4	2								1	1				
2012	16	10		2	2						1				1
2013	4	3													
2014	24	17	1	3						1	3				2
2015	34	24		4	4						2				
2016	34	24	2	9	5						8				
2017	12	2		2							3				
Total	185	109	12	29	11					6	21				3
% Of All		58.9	6.5	15.7	5.9	0.0	0.0	0.0	0.0	3.2	11.4	0.0	0.0	0.0	

Investigating your backlog

[Openssl](#) » [Openssl](#) : Security Vulnerabilities Published In 2016

2016 : [January](#) [February](#) [March](#) [April](#) [May](#) [June](#) [July](#) [August](#) [September](#) [October](#) [November](#) [December](#) CVSS Scores Greater Than: [0](#) [1](#) [2](#) [3](#) [4](#) [5](#) [6](#) [7](#) [8](#) [9](#)

Sort Results By : [CVE Number Descending](#) [CVE Number Ascending](#) [CVSS Score Descending](#) [Number Of Exploits Descending](#)

[Copy Results](#) [Download Results](#)

#	CVE ID	CWE ID	# of Exploits	Vulnerability Type(s)	Publish Date	Update Date	Score	Gained Access Level	Access	Complexity	Authentication	Conf.	Integ.	Avail.
1	CVE-2016-7052	476		DoS	2016-09-26	2018-01-18	5.0	None	Remote	Low	Not required	None	None	Partial
crypto/x509/x509_vfy.c in OpenSSL 1.0.2i allows remote attackers to cause a denial of service (NULL pointer dereference and application crash) by triggering a CRL operation.														
2	CVE-2016-6309	416		DoS Exec Code	2016-09-26	2018-01-18	10.0	None	Remote	Low	Not required	Complete	Complete	Complete
statem/statem.c in OpenSSL 1.1.0a does not consider memory-block movement after a realloc call, which allows remote attackers to cause a denial of service (use-after-free) or possibly execute arbitrary code via a crafted TLS session.														
3	CVE-2016-6308	399		DoS	2016-09-26	2018-01-18	7.1	None	Remote	Medium	Not required	None	None	Complete
statem/statem_dtls.c in the DTLS implementation in OpenSSL 1.1.0 before 1.1.0a allocates memory before checking for an excessive length, which might allow remote attackers to cause a denial of service (memory consumption) via crafted DTLS messages.														
4	CVE-2016-6307	400		DoS	2016-09-26	2018-01-18	4.3	None	Remote	Medium	Not required	None	None	Partial
The state-machine implementation in OpenSSL 1.1.0 before 1.1.0a allocates memory before checking for an excessive length, which might allow remote attackers to cause a denial of service (memory consumption) via crafted TLS messages, related to statem/statem.c and statem/statem_lib.c.														
5	CVE-2016-6306	125		DoS	2016-09-26	2018-01-18	4.3	None	Remote	Medium	Not required	None	None	Partial
The certificate parser in OpenSSL before 1.0.1u and 1.0.2 before 1.0.2i might allow remote attackers to cause a denial of service (out-of-bounds read) via crafted certificate operations, related to s3_clnt.c and s3_srvr.c.														
6	CVE-2016-6305	20		DoS	2016-09-26	2018-01-18	5.0	None	Remote	Low	Not required	None	None	Partial
The ssl3_read_bytes function in record/rec_layer_s3.c in OpenSSL 1.1.0 before 1.1.0a allows remote attackers to cause a denial of service (infinite loop) by triggering a zero-length record in an SSL_peek call.														
7	CVE-2016-6304	399		DoS	2016-09-26	2018-01-18	7.8	None	Remote	Low	Not required	None	None	Complete
Multiple memory leaks in t1_lib.c in OpenSSL before 1.0.1u, 1.0.2 before 1.0.2i, and 1.1.0 before 1.1.0a allow remote attackers to cause a denial of service (memory consumption) via large OCSP Status Request extensions.														
8	CVE-2016-6303	787		DoS Overflow	2016-09-16	2018-01-18	7.5	None	Remote	Low	Not required	Partial	Partial	Partial
Integer overflow in the MDC2_Update function in crypto/mdc2/mdc2dgst.c in OpenSSL before 1.1.0 allows remote attackers to cause a denial of service (out-of-bounds write and application crash) or possibly have unspecified other impact via unknown vectors.														
9	CVE-2016-6302	20		DoS	2016-09-16	2018-01-18	5.0	None	Remote	Low	Not required	None	None	Partial
The tls_decrypt_ticket function in ssl/t1_lib.c in OpenSSL before 1.1.0 does not consider the HMAC size during validation of the ticket length, which allows remote attackers to cause a denial of service via a ticket that is too short.														

CVE-2016-6309

Vulnerability Details : [CVE-2016-6309](#)

statem/statem.c in OpenSSL 1.1.0a does not consider memory-block movement after a realloc call, which allows remote attackers to cause a denial of service (use-after-free) or possibly execute arbitrary code via a crafted TLS session.

Publish Date : 2016-09-26 Last Update Date : 2018-01-18

[Collapse All](#) [Expand All](#) [Select](#) [Select&Copy](#) [▼ Scroll To](#) [▼ Comments](#) [▼ External Links](#)

[Search Twitter](#) [Search YouTube](#) [Search Google](#)

– CVSS Scores & Vulnerability Types

CVSS Score	10.0
Confidentiality Impact	Complete (There is total information disclosure, resulting in all system files being revealed.)
Integrity Impact	Complete (There is a total compromise of system integrity. There is a complete loss of system protection, resulting in the entire system being compromised.)
Availability Impact	Complete (There is a total shutdown of the affected resource. The attacker can render the resource completely unavailable.)
Access Complexity	Low (Specialized access conditions or extenuating circumstances do not exist. Very little knowledge or skill is required to exploit.)
Authentication	Not required (Authentication is not required to exploit the vulnerability.)
Gained Access	None
Vulnerability Type(s)	Denial Of Service Execute Code
CWE ID	416


CVE-2016-6309

– References For CVE-2016-6309

<https://www.tenable.com/security/tns-2016-20> CONFIRM

<http://www.oracle.com/technetwork/security-advisory/cpujan2018-3236628.html> CONFIRM

<https://www.tenable.com/security/tns-2016-16> CONFIRM

 <https://git.openssl.org/?p=openssl.git;a=commit;h=acacbfa7565c78d2273c0b2a2e5e803f44afefeb> CONFIRM

<http://www.oracle.com/technetwork/security-advisory/cpuoct2016-2881722.html> CONFIRM

<http://www.oracle.com/technetwork/security-advisory/cpujul2017-3236622.html> CONFIRM

<https://www.openssl.org/news/secadv/20160926.txt> CONFIRM

<https://bto.bluecoat.com/security-advisory/sa132> CONFIRM

<http://www.securityfocus.com/bid/93177>

BID 93177 OpenSSL CVE-2016-6309 Remote Code Execution Vulnerability *Release Date:*2017-04-24

<http://www.securitytracker.com/id/1036885>

SECTRAK 1036885

<http://kb.juniper.net/InfoCenter/index?page=content&id=JSA10759> CONFIRM

<http://www-01.ibm.com/support/docview.wss?uid=swg21995039> CONFIRM

CVE-2016-6309

Fix Use After Free for large message sizes

author Matt Caswell <matt@openssl.org>
Fri, 23 Sep 2016 10:58:11 -0500 (16:58 +0100)
committer Matt Caswell <matt@openssl.org>
Mon, 26 Sep 2016 03:05:30 -0500 (09:05 +0100)

The buffer to receive messages is initialised to 16k. If a message is received that is larger than that then the buffer is "realloc'd". This can cause the location of the underlying buffer to change. Anything that is referring to the old location will be referring to free'd data. In the recent commit [c1ef7c97](#) (master) and [4b390b6c](#) (1.1.0) the point in the code where the message buffer is grown was changed. However `s->init_msg` was not updated to point at the new location.

CVE-2016-6309

Reviewed-by: Emilia Käsper <emilia@openssl.org>
(cherry picked from commit [0d698f6696e114a6e47f8b75ff88ec81f9e30175](#))

```
diff --git a/ssl/statem/statem.c b/ssl/statem/statem.c
index 5faf6ae..caaf068 100644 (file)
--- a/ssl/statem/statem.c
+++ b/ssl/statem/statem.c
@@ -445,6 +445,21 @@ static void init_read_state_machine(SSL *s)
     st->read_state = READ_STATE_HEADER;
 }

+static int grow_init_buf(SSL *s, size_t size) {
+    size_t msg_offset = (char *)s->init_msg - s->init_buf->data;
+    if (!BUF_MEM_grow_clean(s->init_buf, (int)size))
+        return 0;
+    if (size < msg_offset)
+        return 0;
+    s->init_msg = s->init_buf->data + msg_offset;
+    return 1;
+}
+
+/*
+ * This function implements the sub-state machine when the message flow is in
+ * MSG_FLOW_READING. The valid sub-states and transitions are:
+ */
@@ -545,9 +560,8 @@ static SUB_STATE_RETURN read_state_machine(SSL *s)
     /* dtls_get_message already did this */
     if (!SSL_IS_DTLS(s)
         && s->s3->tmp.message_size > 0
         && !BUF_MEM_grow_clean(s->init_buf,
                                (int)s->s3->tmp.message_size
                                + SSL3_HM_HEADER_LENGTH)) {
+        && !grow_init_buf(s, s->s3->tmp.message_size
+                        + SSL3_HM_HEADER_LENGTH)) {
         ssl3_send_alert(s, SSL3_AL_FATAL, SSL_AD_INTERNAL_ERROR);
         SSLerr(SSL_F_READ_STATE_MACHINE, ERR_R_BUF_LIB);
         return SUB_STATE_ERROR;
```


CVE-2016-6309

```
/* Initialise the MSG_FLOW_READING sub-state machine
 */
static void init_read_state_machine(SSL *s)
{
    OSSL_STATEM *st = &s->statem;

    st->read_state = READ_STATE_HEADER;
}

/*
 * This function implements the sub-state machine when the message flow is in
 * MSG_FLOW_READING. The valid sub-states and transitions are:
```

Is that code actually used?

- Often whole libraries are integrated but only a subset of files are actually required
- Is there a code path? (is the code called)
- Is it compiled?
- Get rid of it if you're not using it!



Your mileage may vary

- A variety of different ways to utilize OSS libraries
 - Package managers (source or binary)
 - Forking from an upstream code repository
- A variety of different ways to fix OSS vulnerabilities
 - Full library upgrades
 - Point fixes (patches)
- The more out of date the library is, the more difficult it is to do a full library upgrade

Alert!

- Library upgrades take care of the heavy lift that is backlog maintenance
- OSS libraries require constant upkeep
- Be ready to be caught off guard



BOM – with threat intel

Third Party Component	Version	Latest Version Available	Source Location	License	iOS	Android	Windows Desktop	MacOSX	Web	Notes	Threat Intel
boost	1.65.1	1.67.0	http://www.boost.org/	Boost 1.0 License		x	x	x		Threat intel difficult to find. Low volume of CVEs	https://www.cvedetails.com/vendor/7685/Boost.html
Django	1.9.2	2.0.4 1.11	https://github.com/django/django	DSF					x	Frequent releases for CVE fixes. New codeline (2.x) available or stay on 1.x branch with 1.11 LTS release.	https://twitter.com/djangoproject
libcurl	7.57.0	7.59.0	http://curl.haxx.se/	Curl License	x	x	x	x		Pull information available at curl.haxx.se or we can sign up for debian/slackware mailing list and receive push notification of new lib updates	https://curl.haxx.se/docs/security.html http://www.slackware.com/security/ https://www.debian.org/security/
libvpx	1.6.0	1.7.0	https://chromium.googlesource.com/webm/libvpx	BSD-3	x	x	x	x		Push information available from debian	https://www.debian.org/security/
OpenSSL	1.1.0c	1.1.0h	http://www.openssl.org/	OpenSSL Combined License	x	x	x	x	x	Multiple threat intel sources, twitter is reliable.	https://twitter.com/OpenSSLannounce
sqlite3	3.15.0	3.23.1	https://www.sqlite.org/	Public Domain	x	x	x	x		No push feed from project itself	https://www.debian.org/security/

Python-Django



Follow

Django security releases issued: 1.9.3 and 1.8.10 - In accordance with our security release policy, the Django ... ow.ly/3bVR6y

10:02 AM - 1 Mar 2016

63 Retweets 26 Likes



Python-Django

CVE-2016-2512: Malicious redirect and possible XSS attack via user-supplied redirect URLs containing basic auth

Django relies on user input in some cases (e.g. `django.contrib.auth.views.login()` and `login()`) to redirect the user to an "on success" URL. The security check for these redirects (namely `django.utils.http.is_safe_url()`) considered some URLs with basic authentication credentials "safe" when they shouldn't be.

For example, a URL like `http://mysite.example.com\@attacker.com` would be considered safe if the request's host is `http://mysite.example.com`, but redirecting to this URL sends the user to `attacker.com`.

Also, if a developer relies on `is_safe_url()` to provide safe redirect targets and puts such a URL into a link, they could suffer from an XSS attack.

Thanks Mark Striemer for reporting the issue.

Python-Django

Resolution

Patches to resolve the issues have been applied to Django's master development branch and the 1.9 and 1.8 release branches. The patches may be obtained from the following changesets:

On the development master branch: * [is_safe_url\(\).patch](#)

On the 1.9 release branch: * [is_safe_url\(\).patch](#)

On the 1.8 release branch: * [is_safe_url\(\).patch](#)

The following new releases has been issued:

- Django 1.9.3 ([download Django 1.9.3](#) | [1.9.3 checksums](#))
- Django 1.8.10 ([download Django 1.8.10](#) | [1.8.10 checksums](#))

CVE-2016-2512

Fixed CVE-2016-2512 -- Prevented spoofing is_safe_url() with basic auth.

[Browse files](#)

This is a security fix.

🔑 master (#1) 📄 2.0.2 ... 1.10a1



mstriemer authored and timgraham committed on Feb 22, 2016

1 parent [f432916](#)

commit [c5544d289233f501917e25970c03ed444abbd4f0](#)

📄 Showing 4 changed files with 50 additions and 2 deletions.

Unified

Split

8 ■■■■ django/utils/http.py

View



@@ -290,8 +290,12 @@ def is_safe_url(url, host=None):

```
290      url = url.strip()
291      if not url:
292          return False
293      - # Chrome treats \ completely as /
294      - url = url.replace('\\', '/')
293      + # Chrome treats \ completely as / in paths but it could be part of some
294      + # basic auth credentials so we need to check both URLs.
295      + return _is_safe_url(url, host) and _is_safe_url(url.replace('\\', '/'), host)
296      +
297      +
298      +def _is_safe_url(url, host):
295      299      # Chrome considers any URL with more than two slashes to be absolute, but
296      300      # urlparse is not so flexible. Treat any url with three slashes as unsafe.
297      301      if url.startswith('///'):
```



CVE-2016-2512

```
282 def is_safe_url(url, host=None):
283     """
284     Return ``True`` if the url is a safe redirection (i.e. it doesn't point to
285     a different host and uses a safe scheme).
286
287     Always returns ``False`` on an empty url.
288     """
289     if url is not None:
290         url = url.strip()
291     if not url:
292         return False
293     # Chrome treats \ completely as /
294     url = url.replace('\\', '/')
295     # Chrome considers any URL with more than two slashes to be absolute, but
296     # urlparse is not so flexible. Treat any url with three slashes as unsafe.
297     if url.startswith('///'):
298         return False
299     url_info = urlparse(url)
300     # Forbid URLs like http:///example.com - with a scheme, but without a hostname.
301     # In that URL, example.com is not the hostname but, a path component. However,
302     # Chrome will still consider example.com to be the hostname, so we must not
303     # allow this syntax.
304     if not url_info.netloc and url_info.scheme:
305         return False
306     # Forbid URLs that start with control characters. Some browsers (like
307     # Chrome) ignore quite a few control characters at the start of a
308     # URL and might consider the URL as scheme relative.
309     if unicodedata.category(url[0])[0] == 'C':
310         return False
311     return ((not url_info.netloc or url_info.netloc == host) and
312            (not url_info.scheme or url_info.scheme in ['http', 'https']))
313
```


CVE-2016-2512 fix!

- Downloaded with package manager?
 - Upgrade with pip
- Forked?
 - Sync to upstream
 - Cherry-pick
- Manual code editing (last resort)



The long road

- A patch in your repo protects no one unless you ship an update
- Public exploits can appear at any time
- There is always going to be another vulnerability



Reduce your attack surface

- Eliminate duplicate libraries
- Only use the files you need
- Library reduction has non-security benefits too!



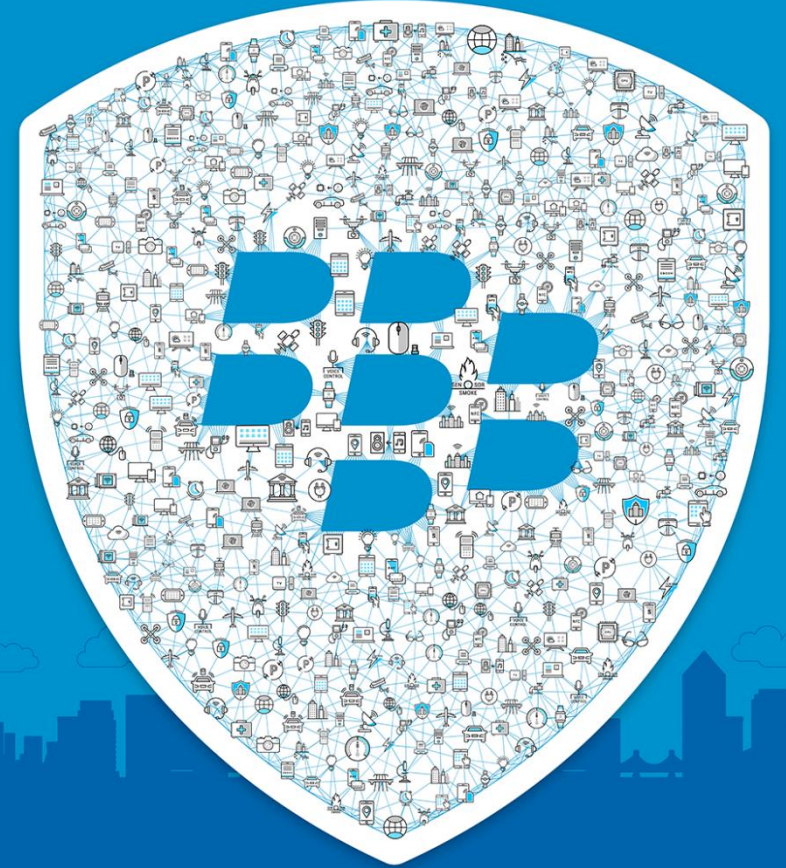
Let's spend some money!

- Enterprise-grade OSS scanning of your binary
- Vulnerability scanners can be useful for BOM building
- Paid sources of threat intelligence helps centralize intel sources
- Custom tooling helps tie everything together



Summary

- Build your bill of materials
- Assess your backlog
- Monitor & investigate alerts
- Contact
 - @tyler_townes
 - ttownes@blackberry.com



BlackBerry Secure