

# Perimeter Management at Spotify



Automating the perimeter at large scale

# Spotify

Over 60 million monthly active users

Available in 58 markets

Over 30 million songs

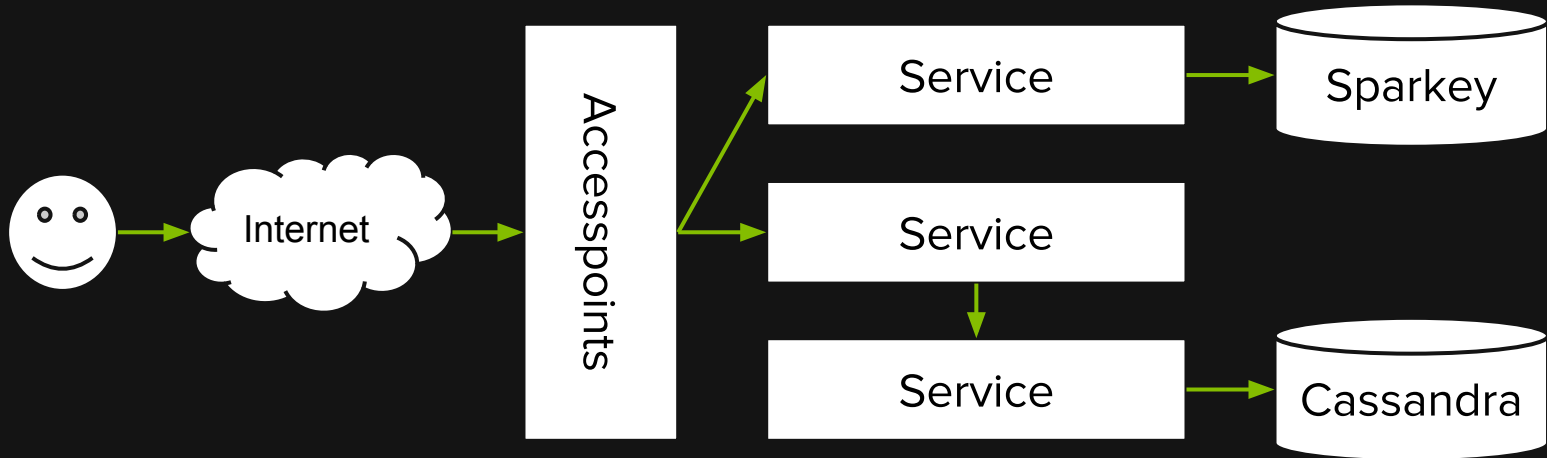
More than 1.5 billion playlists

Tech offices in Sweden and USA

# Architecture of Spotify

Mostly stateless microservices

Around 7k servers in 4 datacenters



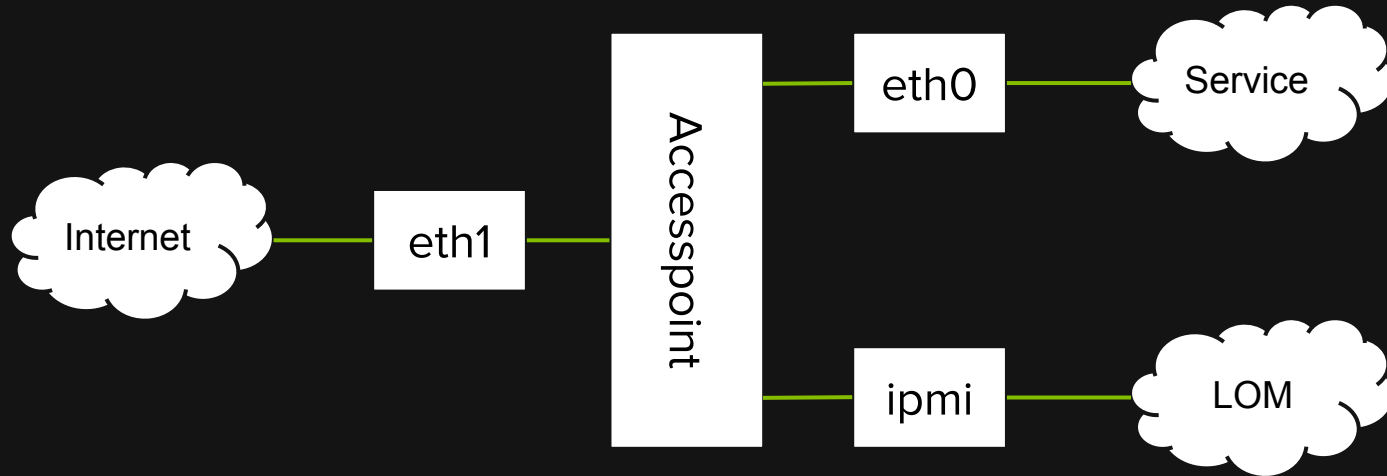
# Perimeter at Spotify



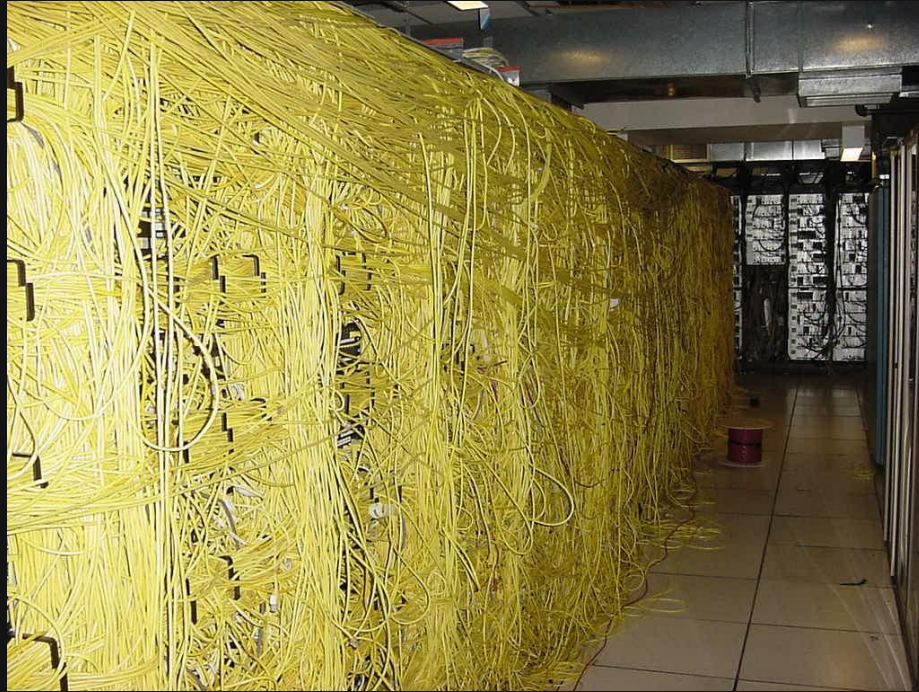
Seems legit.

# Typical perimeter host

Bare-metal commodity hardware



# Reality



# Reality

Manual perimeter management

Firewall exceptions everywhere

No idea what is exposed or not

No audit, false sense of security

Moving to the private ip ranges

# Perimeter requirements

Automated

Easy to use for developers

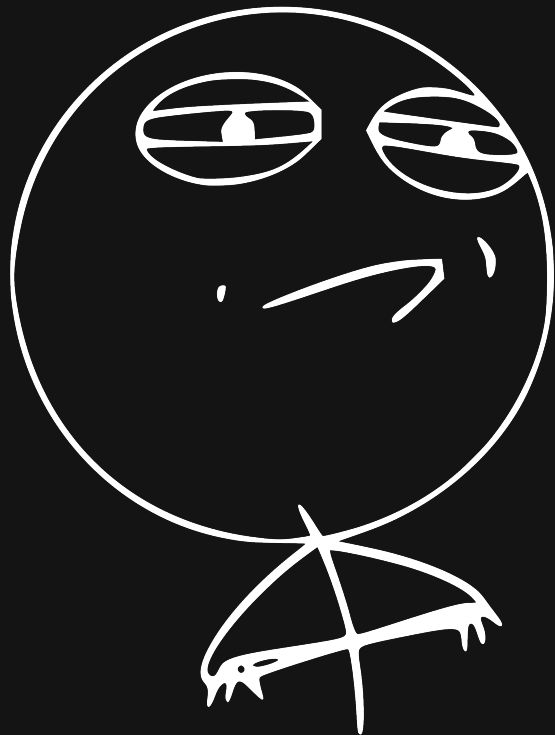
Secure by default

Highly available

Horizontally scalable

Everything must leave an audit trail





**6 months later...**

# Entry points

HTTP load-balancers

Internal SSO proxy

Outbound proxy

Inbound proxy

Accesspoints

# HTTP load balancers

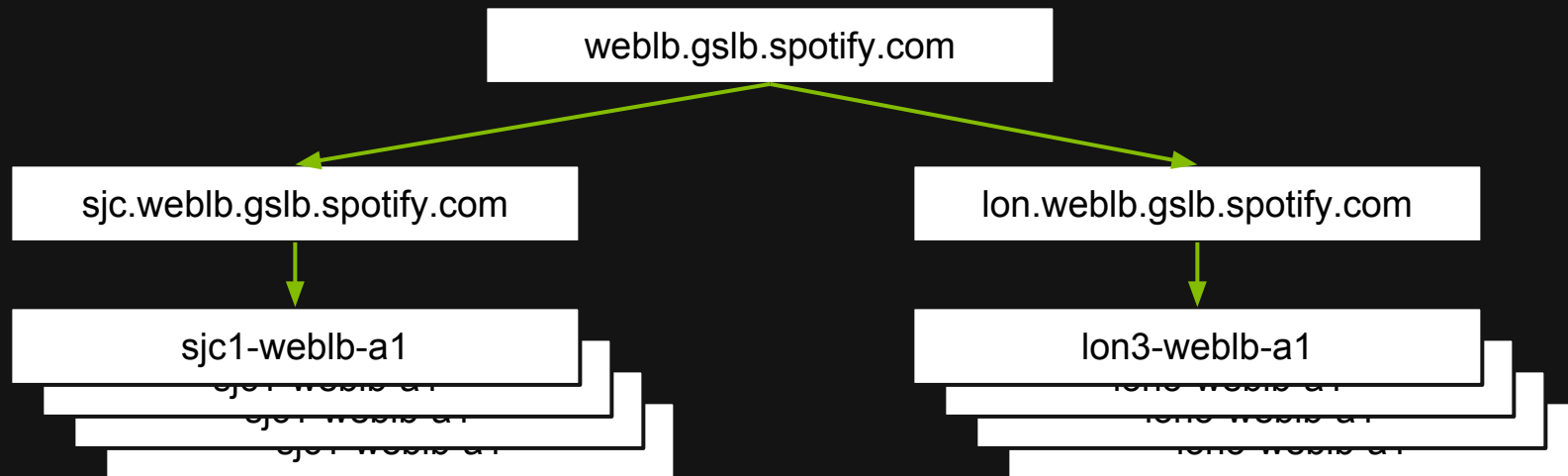
3 different pools for different purposes:

Generic websites

API-endpoints

Payment systems

# Generic weblb pool



Dyn<sup>SM</sup>

# HTTP load balancers

## [artistexplorer] Add artistexplorer.spotify.com to lon weblb

[Browse code](#)

Change-Id: Ib8a617251e0438207551dcf88f701cc45f5b26ea

 master



**fsahin** authored on Nov 23, 2014


1 parent [fc61621](#) commit [4a570ab3c095aebe18a0203b922d8d49f62e5eb8](#)

 Showing **1 changed file** with **4 additions** and **0 deletions**.


[Unified](#)[Split](#)

4  hiera-data/role/weblb/lon3/a.yaml

[View](#)

		@@ -105,3 +105,7 @@ weblb::frontends:
105	105	'clientweb.spotify.com':
106	106	service: 'clientwebdummy'
107	107	protocols: ['https']
	108	+ 'artistexplorer.spotify.com':
	109	+  site_aliases: ['artist-explorer.spotify.com']
	110	+  service: 'artistexplorer'
	111	+  protocols: ['https']

# HTTP load balancers



Home Projects Qualys.com Contact

You are here: [Home](#) > [Projects](#) > [SSL Server Test](#) > [artistexplorer.spotify.com](#)

## SSL Report: artistexplorer.spotify.com

Assessed on: Fri Feb 27 13:45:10 PST 2015 | [Clear cache](#) [Scan Another >>](#)

	Server	Domain(s)	Test time	Grade
1	<a href="#">194.132.198.165</a> lon3-weblb-a3.lon3.spotify.com Ready	artistexplorer.spotify.com	Fri Feb 27 13:39:01 PST 2015 Duration: 122.405 sec	A+
2	<a href="#">194.132.196.178</a> lon3-weblb-a6.lon3.spotify.com Ready	artistexplorer.spotify.com	Fri Feb 27 13:41:04 PST 2015 Duration: 123.75 sec	A+
3	<a href="#">194.132.196.212</a> lon3-weblb-a4.lon3.spotify.com Ready	artistexplorer.spotify.com	Fri Feb 27 13:43:07 PST 2015 Duration: 123.126 sec	A+

# HTTP load balancers



# Gotchas

Need to resolve upstream IPs by puppet

proxy\_next\_upstream timeout is tricky

Active health checks is a must

```
Syntax: proxy_ssl_verify on | off;
```

```
Default: proxy_ssl_verify off;
```

```
Context: http, server, location
```

```
This directive appeared in version 1.7.0.
```



# Internal SSO proxy



**Shibboleth®**

# Internal SSO proxy

Public and private pool

More than 200 websites

2-factor authentication



# Internal SSO proxy

## [babelfish] Shibboleth SSO proxy for admin UI

[Browse code](#)

🔒 master (#703)



poscar authored 29 days ago

1 parent [a634dc2](#) commit [421c6ecc401f645eb00653d79fad75940fa4419f](#)

Showing 3 changed files with 9 additions and 2 deletions.

Unified

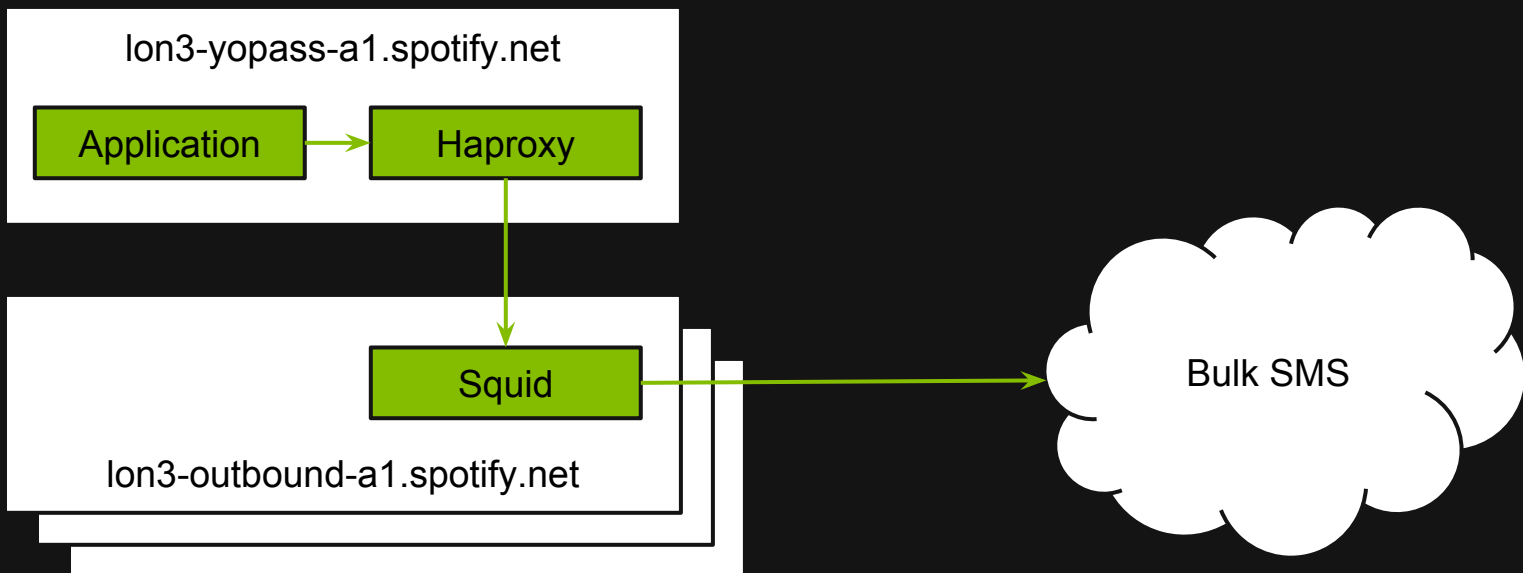
Split

6 ■ ■ ■ ■ ■ ■ hiera-data/role/ssoproxy/production.yaml

View

		@@ -108,6 +108,12 @@ sso_proxy::sso_sites_private:
108	108	- 'consultants'
109	109	backend_port: '8050'
110	110	whitelist_tech_offices: true
	111	+ 'babelfish.spotifyinternal.com':
	112	+  shibbolethgroups:
	113	+    - 'babelfish'
	114	+    - 'backend-gbg'
	115	+    - 'tunigo'
	116	+    service_name: 'babelfish'
111	117	'bamboo-payment.spotify.net':
112	118	shibbolethgroups:
113	119	- 'spotifyusers'

# Outbound proxy



# Outbound proxy

HTTP(S), FTP, TCP ACLs

HTTP traffic is allowed by default

SSL-Bumped traffic is allowed by default

Everything else must be ACL-whitelisted

Everything is logged to ElasticSearch

# Outbound proxy

## Adding outbound connection from yopass role to SMS provider bulksms

[Browse code](#)

Change-Id: Ie87cc9bd782dfccf874851aff0c4d277cee84369

 master

 **carla** authored on Feb 4

1 parent [55337d5](#) commit [ea96f1bf1008dbece1c8362468f8070c59d45eaa](#)

 Showing **2 changed files** with **3 additions** and **0 deletions**.

[Unified](#)[Split](#)

2  hiera-data/role/outbound.yaml

[View](#)

@@ -62,6 +62,8 @@ forward\_proxy::generic\_tcp\_access:

62 | 62 |       - .ap.spotify.com:1025-65535

63 | 63 |       jira::ubuntu:

64 | 64 |       - .atlassian.com:80

65 | 65 | +     yopass:

66 | 66 | +     - bulksms.vsms.net:5567

65 | 67 |

66 | 68 |     forward\_proxy::ftp\_access:

67 | 69 |     srv:



# HTTP proxy env mess

No way to exclude ip ranges via no\_proxy

No reliable way to set ENV vars system-wide

Various implementation bugs

Especially no\_proxy

Some local traffic goes through the outbound proxy

# Gotchas

X-Forwarded-For: unknown

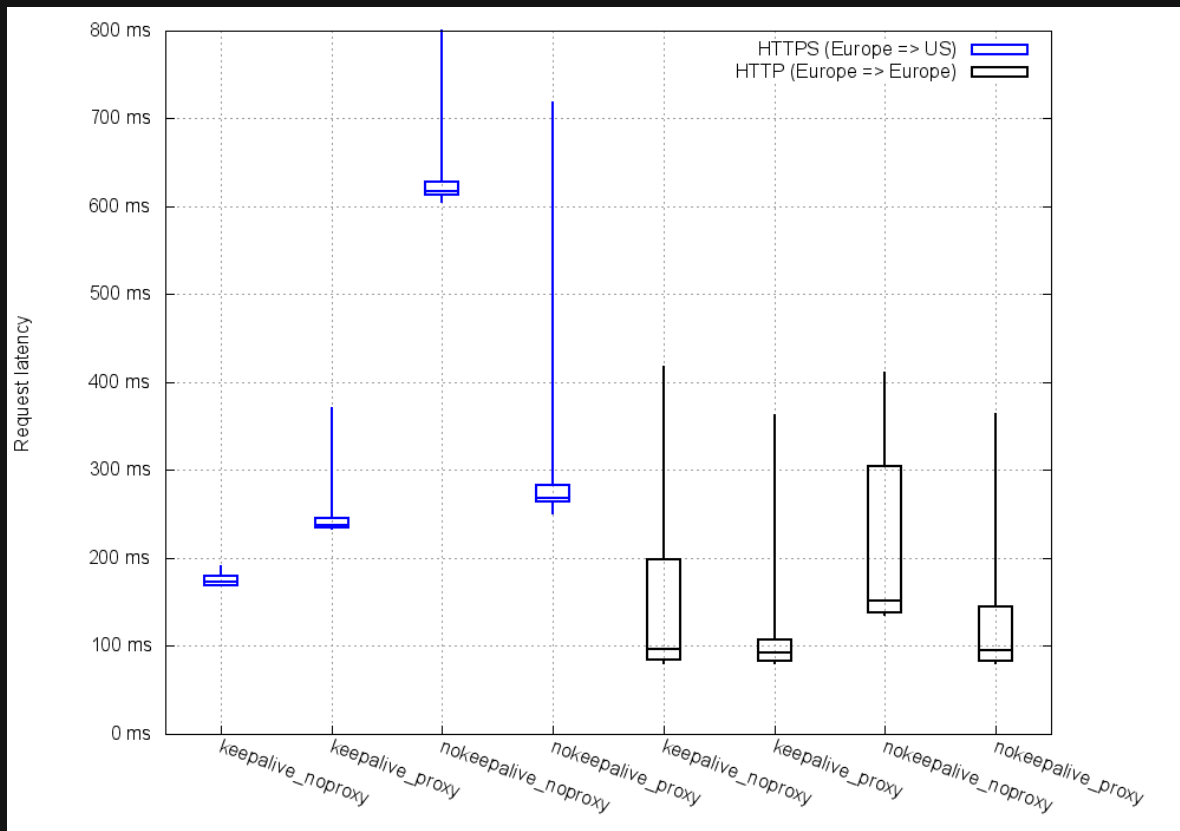
SSL-Bump: custom certificate deployment

Bundled CAs everywhere

Lower latency because of keep-alive



# Outbound latency overhead



# Incoming TCP proxy

Dynect + haproxy + ELK

Only used by our content team

Important to keep the fixed ip addresses

# Low-hanging fruits

2-factor authentication

Bug-bounty

Periodic network scans

# That's All Folks!

Alexey Lapitsky ([alexey@spotify.com](mailto:alexey@spotify.com))



# Accesspoint

Server written in C++

Using custom multiplexed protocol

With custom encryption

Horizontally scalable by design

10 years old codebase with more than 100k of LOC

# Zero-effort hardening

hardening-check /usr/sbin/accesspoint

Position Independent Executable: yes

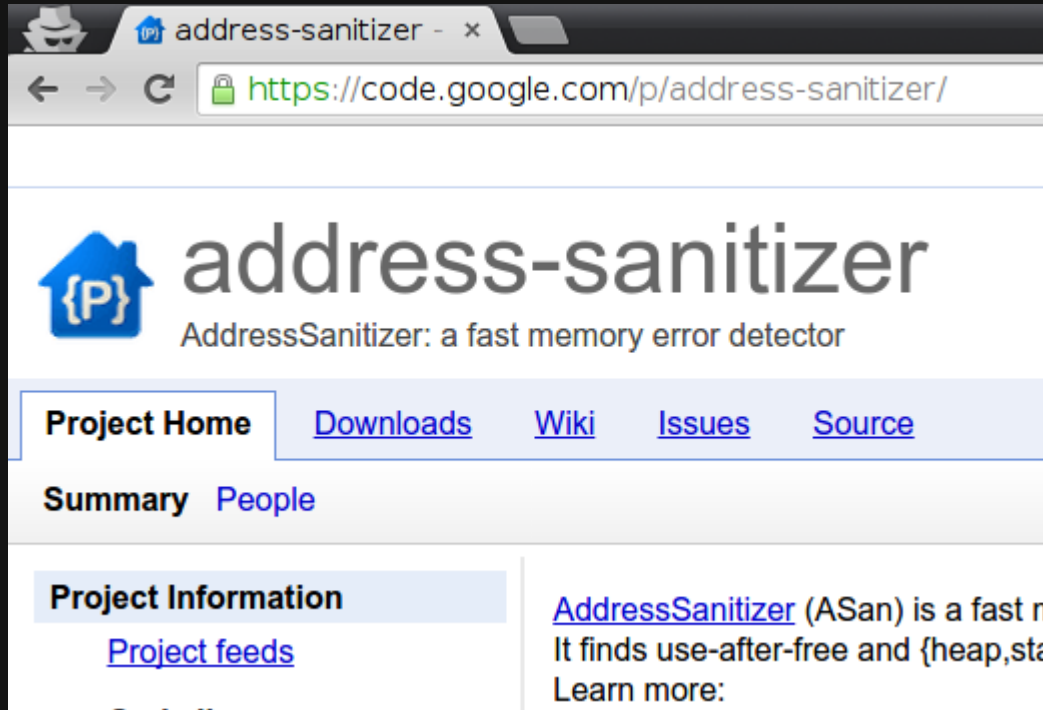
Stack protected: yes

Fortify Source functions: yes

Read-only relocations: yes

Immediate binding: yes


# ASAN



The image shows a screenshot of a web browser displaying the project page for AddressSanitizer on code.google.com. The browser's address bar shows the URL <https://code.google.com/p/address-sanitizer/>. The page features the project's logo, a blue house with a curly brace and 'P' inside, followed by the title "address-sanitizer" and the subtitle "AddressSanitizer: a fast memory error detector". A navigation menu includes links for "Project Home", "Downloads", "Wiki", "Issues", and "Source". Below the menu, there are sections for "Summary" and "People". The "Project Information" section is highlighted, containing a link to "Project feeds". To the right, a brief description of AddressSanitizer (ASan) is visible, stating it is a fast memory error detector that finds use-after-free and {heap,stack} overflow errors. A "Learn more:" link is also present.

address-sanitizer - x

<https://code.google.com/p/address-sanitizer/>

 address-sanitizer  
AddressSanitizer: a fast memory error detector

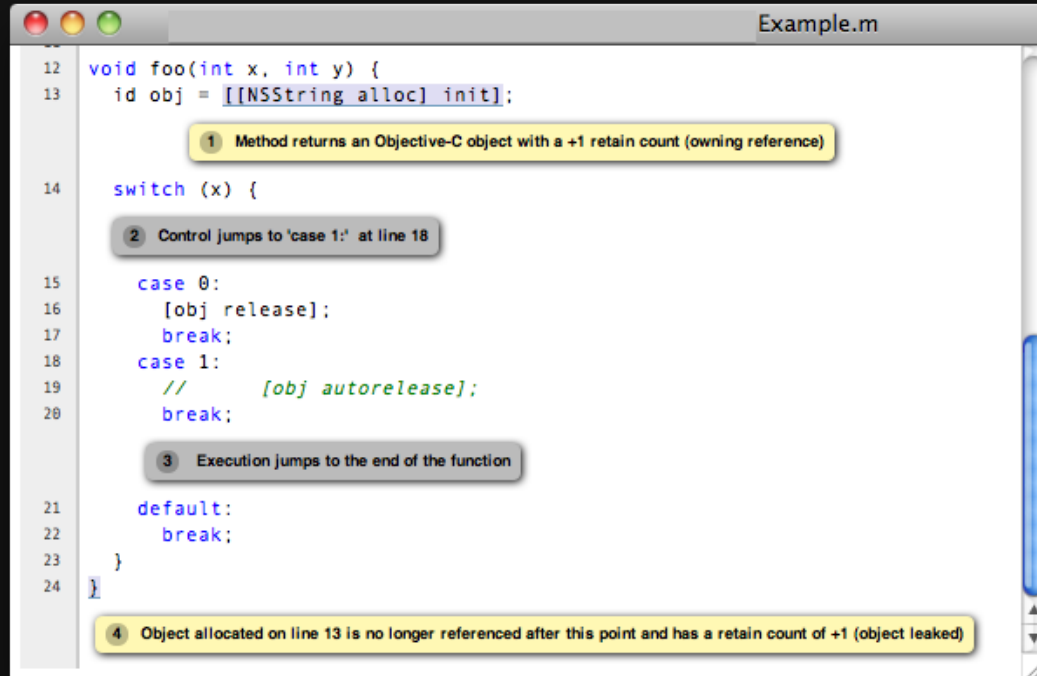
[Project Home](#) [Downloads](#) [Wiki](#) [Issues](#) [Source](#)

Summary [People](#)

**Project Information**  
[Project feeds](#)

[AddressSanitizer](#) (ASan) is a fast memory error detector. It finds use-after-free and {heap,stack} overflow errors. Learn more:

# Clang Static Analyzer



The screenshot shows a window titled "Example.m" containing the following C code:

```
12 void foo(int x, int y) {
13     id obj = [[NSString alloc] init];
14     switch (x) {
15         case 0:
16             [obj release];
17             break;
18         case 1:
19             // [obj autorelease];
20             break;
21     default:
22         break;
23     }
24 }
```

Four annotations are present:

- 1 Method returns an Objective-C object with a +1 retain count (owning reference)
- 2 Control jumps to 'case 1:' at line 18
- 3 Execution jumps to the end of the function
- 4 Object allocated on line 13 is no longer referenced after this point and has a retain count of +1 (object leaked)