```
       _______  _________ _________     ________
      \      \ \_      ___ \\_    ___ \   /      ____/_______ ____  __ ________
      /   |   \/    \  \//    \  \/  /   \    __\_  __ \/  _ \|  |  \____ \
     /    |    \     \___\     \____ \    \_\  \  | \(  <_> )  |  /  |_> >
     \____|__  /\_______  /\_______  /  \_______  /__|   \____/|____/|    __/
             \/         \/         \/           \/                  |__|

                      https://www.nccgroup.com/research/

~.~.~.~.~.~.~.~.~.~.~.~.~.~.~.~.~.~.~.~.~.~.~.~.~.~.~.~.~.~.~.~.~.~.~.~.~.~.
 Vulnerability Summary
~.~.~.~.~.~.~.~.~.~.~.~.~.~.~.~.~.~.~.~.~.~.~.~.~.~.~.~.~.~.~.~.~.~.~.~.~.~.

Title             Xen HYPERVISOR_xen_version stack memory revelation
Release Date      6 March 2015
Reference         NCC00817
Discoverer        Aaron Adams
Vendor            Xen
Vendor Reference  XSA-122
Systems Affected  All
CVE Reference     CVE-2015-2045
Risk              Low
Status            Fixed

~.~.~.~.~.~.~.~.~.~.~.~.~.~.~.~.~.~.~.~.~.~.~.~.~.~.~.~.~.~.~.~.~.~.~.~.~.~.
 Resolution Timeline
~.~.~.~.~.~.~.~.~.~.~.~.~.~.~.~.~.~.~.~.~.~.~.~.~.~.~.~.~.~.~.~.~.~.~.~.~.~.

Discovered        11 February 2015
Reported          11 February 2015
Released          11 February 2015
Fixed              5 March 2015
Published          6 March 2015

~.~.~.~.~.~.~.~.~.~.~.~.~.~.~.~.~.~.~.~.~.~.~.~.~.~.~.~.~.~.~.~.~.~.~.~.~.~.
 Vulnerability Description
~.~.~.~.~.~.~.~.~.~.~.~.~.~.~.~.~.~.~.~.~.~.~.~.~.~.~.~.~.~.~.~.~.~.~.~.~.~.

A number of HYPERVISOR_xen_version sub-operations will leak uninitialized
stack data when returning data to a guest issuing the hypercall.

~.~.~.~.~.~.~.~.~.~.~.~.~.~.~.~.~.~.~.~.~.~.~.~.~.~.~.~.~.~.~.~.~.~.~.~.~.~.
 Technical Details
~.~.~.~.~.~.~.~.~.~.~.~.~.~.~.~.~.~.~.~.~.~.~.~.~.~.~.~.~.~.~.~.~.~.~.~.~.~.
The problem occurs due to a failure to initialize various structures to 0
and instead leaving them uninitialized and filling them with string data.
Any data beyond the strings NULL terminator will be passed back the guest
uninitalized. This could potentially leak data provided by other guests
that previously made hypercalls, or leak information about the hypervisor
such as return addresses and other pointers.

An example of this is in the XENVER_compile_info subcommand, which does the
following:

xen/common/kernel.c
--------------------------------------------------
249     case XENVER_compile_info:
250     {
251         struct xen_compile_info info;
252         safe_strcpy(info.compiler,       xen_compiler());
253         safe_strcpy(info.compile_by,     xen_compile_by());
254         safe_strcpy(info.compile_domain, xen_compile_domain());
255         safe_strcpy(info.compile_date,   xen_compile_date());
256         if ( copy_to_guest(arg, &info, 1) )
257             return -EFAULT;
258         return 0;
259     }
--------------------------------------------------

The xen_compile_info structure is defined as follows:

xen/include/public/version.h
--------------------------------------------------
 43 /* arg == xen_compile_info_t. */
 44 #define XENVER_compile_info 2
 45 struct xen_compile_info {
 46     char compiler[64];
```

```
 47     char compile_by[16];
 48     char compile_domain[32];
 49     char compile_date[32];
 50 };
 51 typedef struct xen_compile_info xen_compile_info_t;
--------------------------------------------------
```

In practice the strings that fill these arrays do not consume the entire array, so anything adjacent to the NULL terminator can be leaked.

The following is an example of leaking data using a kernel module on an unpatched Xen 4.4 guest:

```
* Compiler leak:
* ==================
* ffff880031f0d3c0: 67 63 63 20 28 55 62 75 6e 74 75 20 34 2e 38 2e  gcc
(Ubuntu 4.8.
* ffff880031f0d3d0: 32 2d 31 39 75 62 75 6e 74 75 31 29 20 34 2e 38
2-19ubuntu1) 4.8
* ffff880031f0d3e0: 2e 32 00 00 00 00 00 00 f8 7b 2b 80 d0 82 ff ff
.2.......{+.....
* ffff880031f0d3f0: 01 00 00 00 00 00 00 00 01 00 00 00 08 00 00 00
................
* ffff880031f0d400: 73 74 65 66 61 6e 2e 62 61 64 65 72 00 00 00 00
stefan.bader....
* ffff880031f0d410: 63 61 6e 6f 6e 69 63 61 6c 2e 63 6f 6d 00 00 00
canonical.com...
* ffff880031f0d420: 02 00 00 00 00 00 00 00 18 9d e8 6c 00 88 ff ff
...........l....
* ffff880031f0d430: 54 75 65 20 53 65 70 20 33 30 20 31 39 3a 31 39  Tue
Sep 30 19:19
* ffff880031f0d440: 3a 33 31 20 55 54 43 20 32 30 31 34 00 00 00 00  :31
UTC 2014....
* ==================
```

The data above leaks hypervisor addresses from the stack. An example from the compiler leak portion would be the address 0xffff82d0802b7bf8.

This problem affects the XENVER_compile_info, XENVER_extraversion, and XENVER_changeset sub-operations. This hypercall is available to both PV and HVM guests. Kernel privileges on a domain are required in order to exploit this vulnerability.

~.~.~.