Auditing hooks and security transparency for CPython

Steve Dower, Christian Heimes

EuroPython 2019, Basel, Switzerland

Why is SkelSec so sad?

skelsec

PRO

https://twitter.com/SkelSec

Block or report user

Overview | Repositories 50 | Projects 0 | Stars 19 | Followers 180 | Following 2

Pinned

pypykatz
Mimikatz implementation in pure Python
Python ★ 502 92

kerberoast
Kerberoast attack -pure python-
Python ★ 53 11

minikerberos
Kerberos manipulation library in pure Python
Python ★ 61 15

minidump
Python library to parse and read Microsoft minidump file format
Python ★ 30 5

CVE-2017-12542
Test and exploit for CVE-2017-12542
Python ★ 68 26

BitErrant
BitErrant
Objective-C ★ 50 11

We made SkelSec sad…

and that should make you all happy

Who are we?

Steve Dower

- CPython core developer
- Author of PEP 578
- @zooba

- (Also works at Microsoft)

Christian Heimes

- CPython core developer
- BDFL delegate for PEP 578
- @christianheimes

- (Also works at Red Hat)

Today's Agenda

- What are audit hooks, and why would I use them?
- Using audit hooks to improve security
- Integration on Windows-based systems
- Integration on Linux-based systems

Runtime Audit Hooks (PEP 578)

- One piece of a complete security solution
- Provides low-level insight into runtime behaviour
  - Opening files
  - Connecting sockets
  - Compiling strings
- By default, does nothing!
- Designed for security engineers to plug into

Python Security Engineer Checklist

Install security updates

Limit user accounts

Install security updates!

Use a firewall

Install security updates!!

Restrict package installation

Install security updates!!!

Think about maybe, possibly, using some Python audit hooks

Listening to audit hooks

```c
int hook(
    const char *event,
    PyObject *args,
    void *userData
) {
    printf("Saw %s\n", event);
    return 0;
}

PySys_AddAuditHook(hook, userData);
```

```python
import sys

def hook(event, args):
    print("Saw", event)

sys.addaudithook(hook)
```

Listening to audit hooks

C - PySys_AddAuditHook()

Pros:

- Faster
- Hard to bypass

Cons:

- More complex
- Requires custom Python

Python - sys.addaudithook()

Pros:

- Easy
- Convenient

Cons:

- Per-subinterpreter
- Slow

What events should you expect?

compile

builtins.input

import

os.system

exec

glob.glob

open

socket.__new__

docs.python.org/3.8/library/audit_events.html

What to do with an event?

- Nothing
- Log it
- Abort it
- Abort everything!

Correct answer: log it

If a tree falls in a forest… has it been logged?

When an intruder is trying to get in, or is already in, *you need to know*

Logging allows:

- Retrospective analysis
- Anomaly detection
- Incident response

Premature log filtering cripples your defence. Log everything.

Creating audit events

```
PySys_Audit("module.event",
            "isO", a, b, c);
```

```
import sys
sys.audit("module.event",
          a, b, c)
```

Tips:

- Prefer C call (PySys_Audit) when possible
- Prefix with your module (import) name
- Audit after validation, before execution

The io.open_code() function

- Code ≠ Data
- Your OS kernel already does this for binaries, but not via open()

```
import io
io.open_code("file.py")
```

Same as `open(…, "rb")` but can be hooked in C

```
PyFile_SetOpenCodeHook(callback, user_data);
```

Why would you hook io.open_code()?

- Validate file attributes
- Validate file contents
- Exclusive file access
- Return `BytesIO` instead of real file object

Careful implementation required:

- Cannot recurse (via PyImport_ImportModule)
- Callers assume they'll get a regular file object

What else do you need to do?

- Handle audit events
  - `compile`, `exec` – loading code not from files
  - `open` – loading other unexpected files
- Disable launch options (in audit hook)
  - `-c "…"` – code in arguments
  - `… | python3` – code from other shell commands
  - Force `-E` – ignore environment variables
- Use good access control rules
  - $TEMPDIR / %TEMP%
  - $HOME / %USERPROFILE%

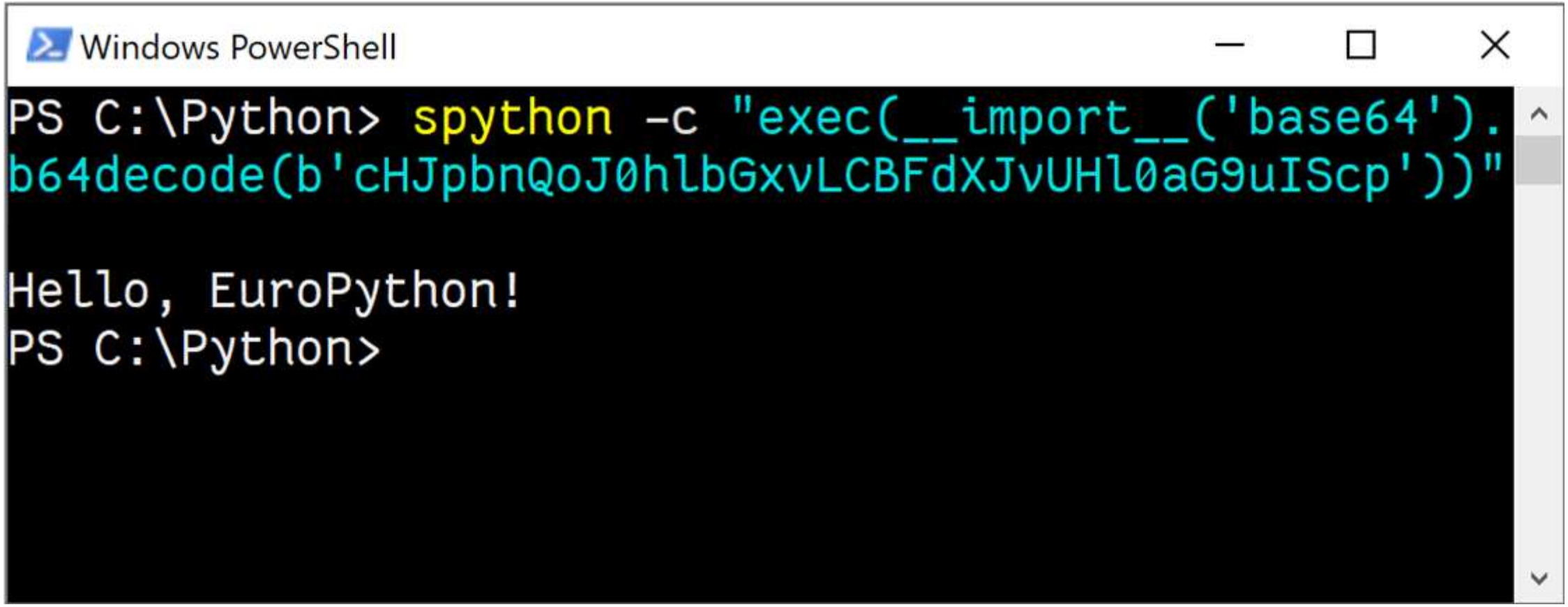
Integrating with Windows

Integrating with Windows

- Windows Event Log
- Catalog Signing
- Windows Defender Application Control

github.com/zooba/spython

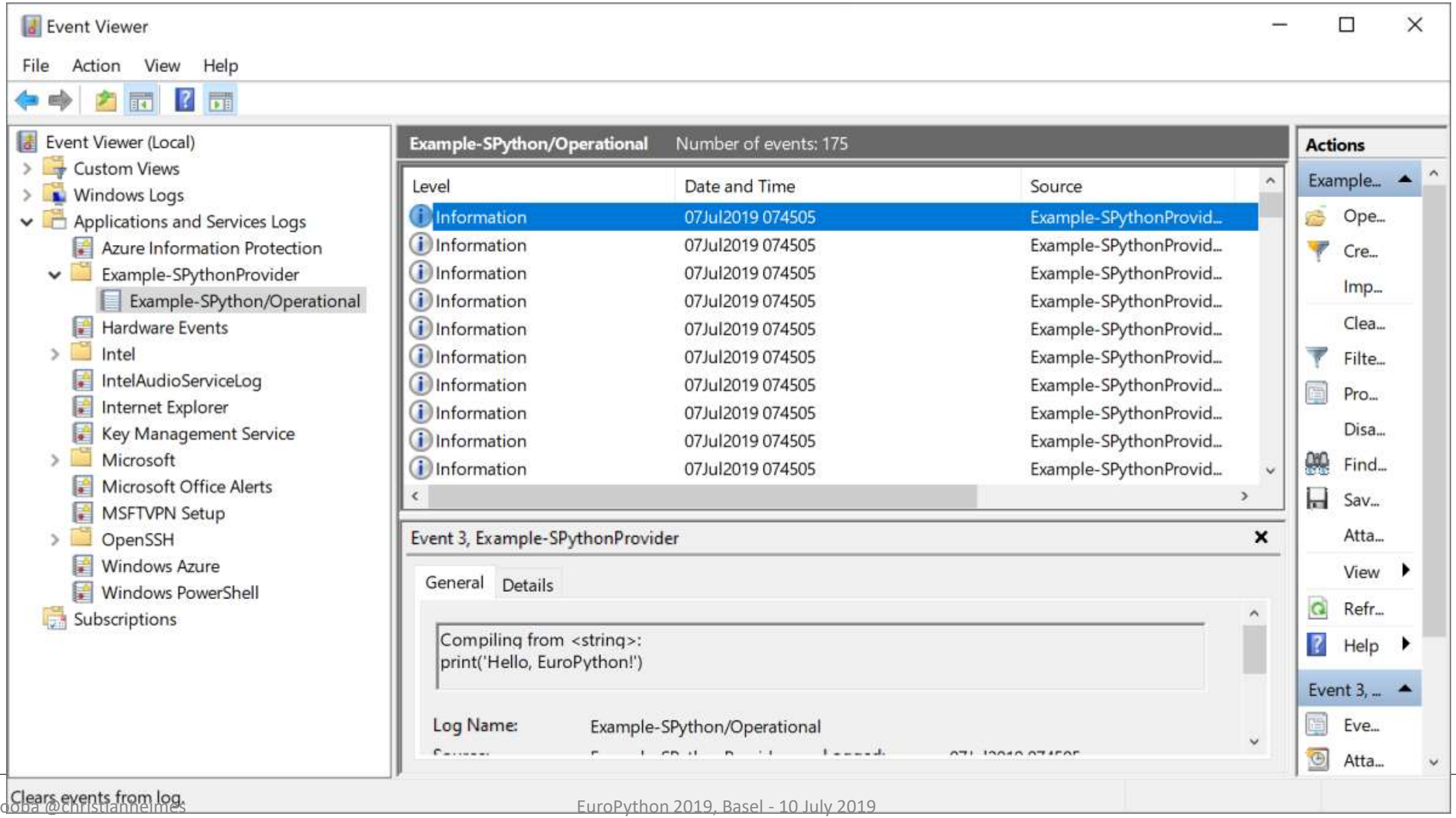
Windows Event Log

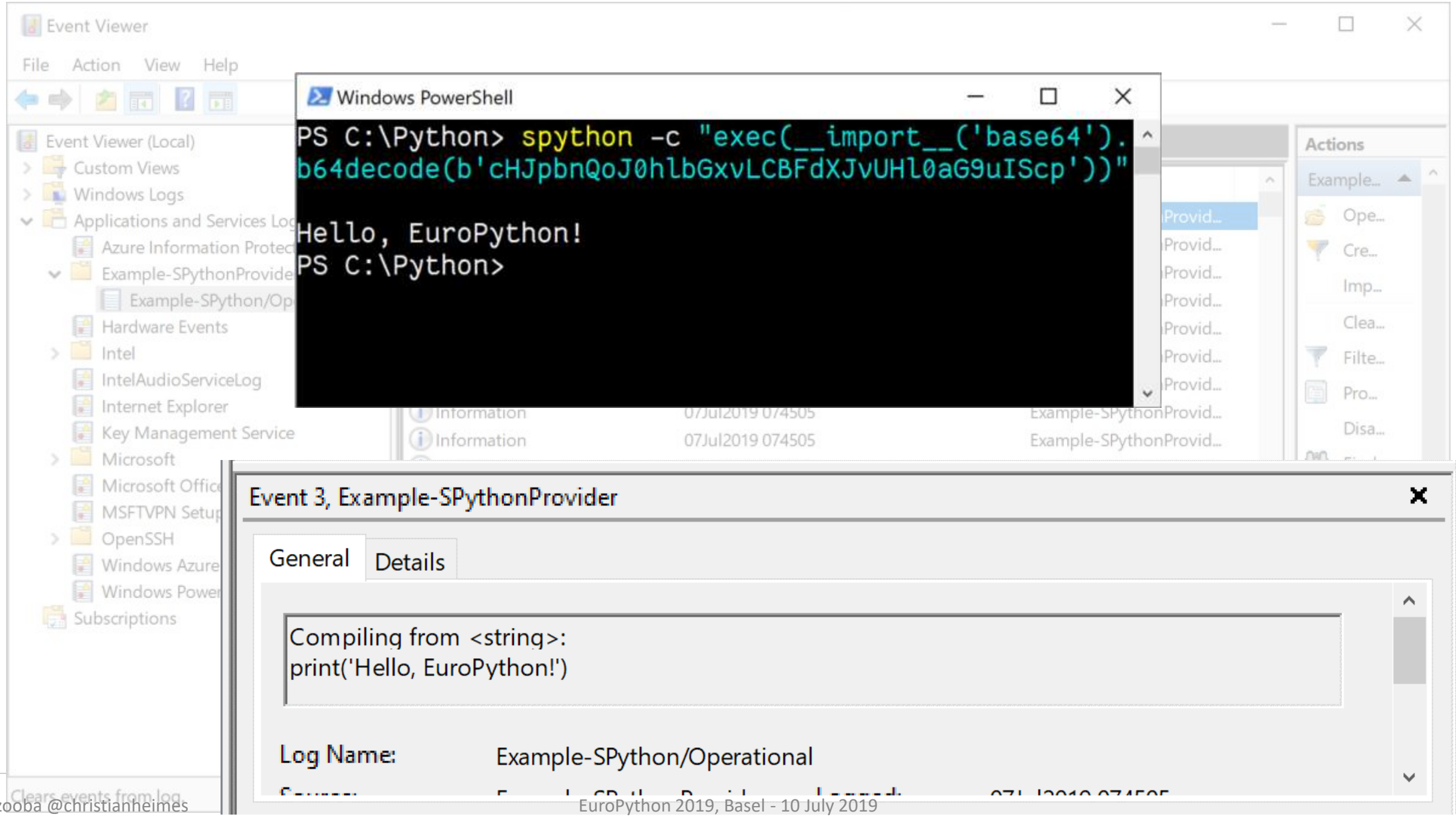

Windows PowerShell
PS C:\Python> spython -c "exec(__import__('base64').
b64decode(b'cHJpbnQoJ0hlbGxvLCBFdXJvUHl0aG9uIScp'))"
Hello, EuroPython!
PS C:\Python>

Windows Event Log features

- Event Log viewer
- Event forwarding
- Protected Event Logging
- Clearing/modifying logs adds a new event

```
static int
hook_compile(const char *event, PyObject *args)
{
    PyObject *code, *filename;
    const char *u8code = NULL, *u8filename = NULL;

    if (!EventEnabledIMPORT_COMPILE()) {
        return 0;
    }

    if (!PyArg_ParseTuple(args, "OO", &code, &filename)) {
        return -1;
    }

    u8code = PyUnicode_AsUTF8(code);
    u8filename = PyUnicode_AsUTF8(filename);

    EventWriteIMPORT_COMPILE(u8code, u8filename);

    return 0;
}
```


Event Viewer
File Action View Help
Event Viewer (Local)
Custom Views
Windows Logs
Applications and Services Logs
Azure Information Protection
Example-SPythonProvider
Example-SPython/Operational
Hardware Events
Intel
IntelAudioServiceLog
Internet Explorer
Key Management Service
Microsoft
Microsoft Office Alerts
MSFTVPN Setup
OpenSSH
Windows Azure
Windows PowerShell
Subscriptions
Example-SPython/Operational Number of events: 175
Level Date and Time Source
Information 07Jul2019 074505 Example-SPythonProvid...
Information 07Jul2019 074505 Example-SPythonProvid...
Information 07Jul2019 074505 Example-SPythonProvid...
Information 07Jul2019 074505 Example-SPythonProvid...
Information 07Jul2019 074505 Example-SPythonProvid...
Information 07Jul2019 074505 Example-SPythonProvid...
Information 07Jul2019 074505 Example-SPythonProvid...
Information 07Jul2019 074505 Example-SPythonProvid...
Information 07Jul2019 074505 Example-SPythonProvid...
Information 07Jul2019 074505 Example-SPythonProvid...
Event 3, Example-SPythonProvider
General Details
Compiling from <string>:
print('Hello, EuroPython!')
Log Name: Example-SPython/Operational
Actions
Example...
Ope...
Cre...
Imp...
Clea...
Filte...
Pro...
Disa...
Find...
Sav...
Atta...
View
Refr...
Help
Event 3, ...
Eve...
Atta...
Clears events from log.

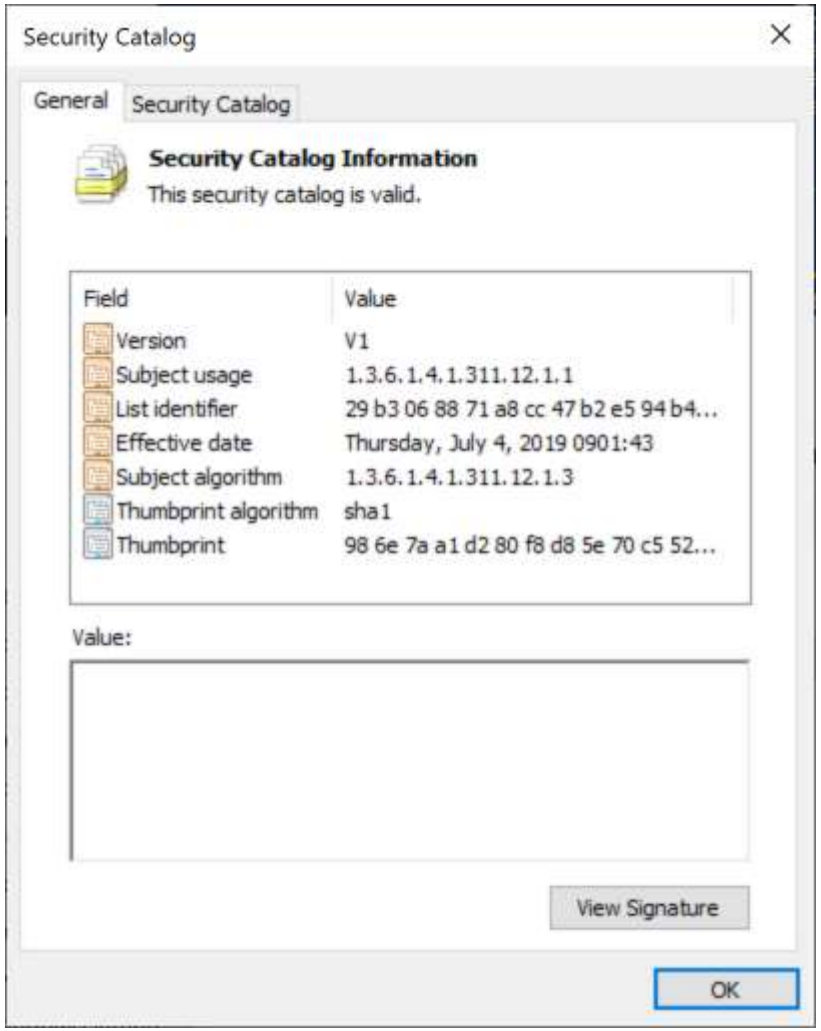

Event Viewer
File Action View Help
Event Viewer (Local)
Custom Views
Windows Logs
Applications and Services Log
Azure Information Protec
Example-SPythonProvide
Example-SPython/Op
Hardware Events
Intel
IntelAudioServiceLog
Internet Explorer
Key Management Service
Microsoft
Microsoft Offic
MSFTVPN Setup
OpenSSH
Windows Azure
Windows Power
Subscriptions
Clears events from log.
Windows PowerShell
PS C:\Python> spython -c "exec(__import__('base64').b64decode(b'cHJpbnQoJ0hlbGxvLCBFdXJvUHl0aG9uIScp'))"
Hello, EuroPython!
PS C:\Python>
Information 07Jul2019 074505 Example-SPythonProvid...
Information 07Jul2019 074505 Example-SPythonProvid...
Actions
Example...
Ope...
Cre...
Imp...
Clea...
Filte...
Pro...
Disa...
Event 3, Example-SPythonProvider
General
Details
Compiling from <string>:
print('Hello, EuroPython!')
Log Name: Example-SPython/Operational

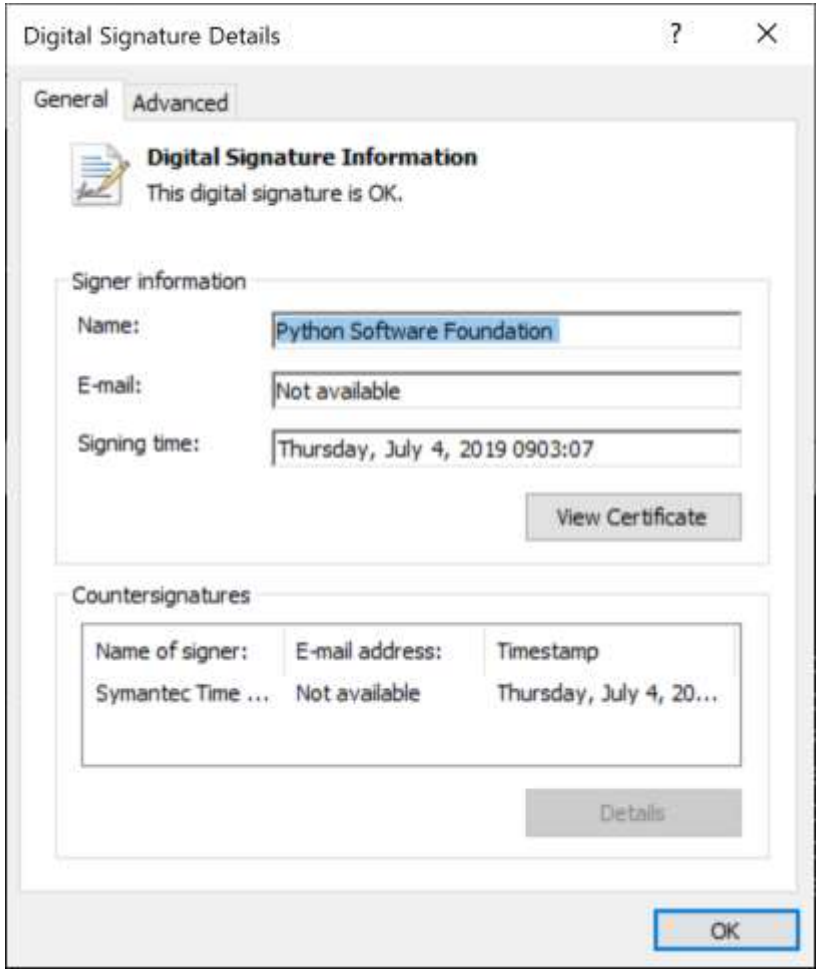
Signed Catalog Files

Code Signing

- Typical white-listing approach
- Attaches a signed hash of the file to the file
- *Catalog* signing signs a set of files
  - We can't sign .py files, so we use .cat

- Standard Python installers include a catalog file for all non-binaries

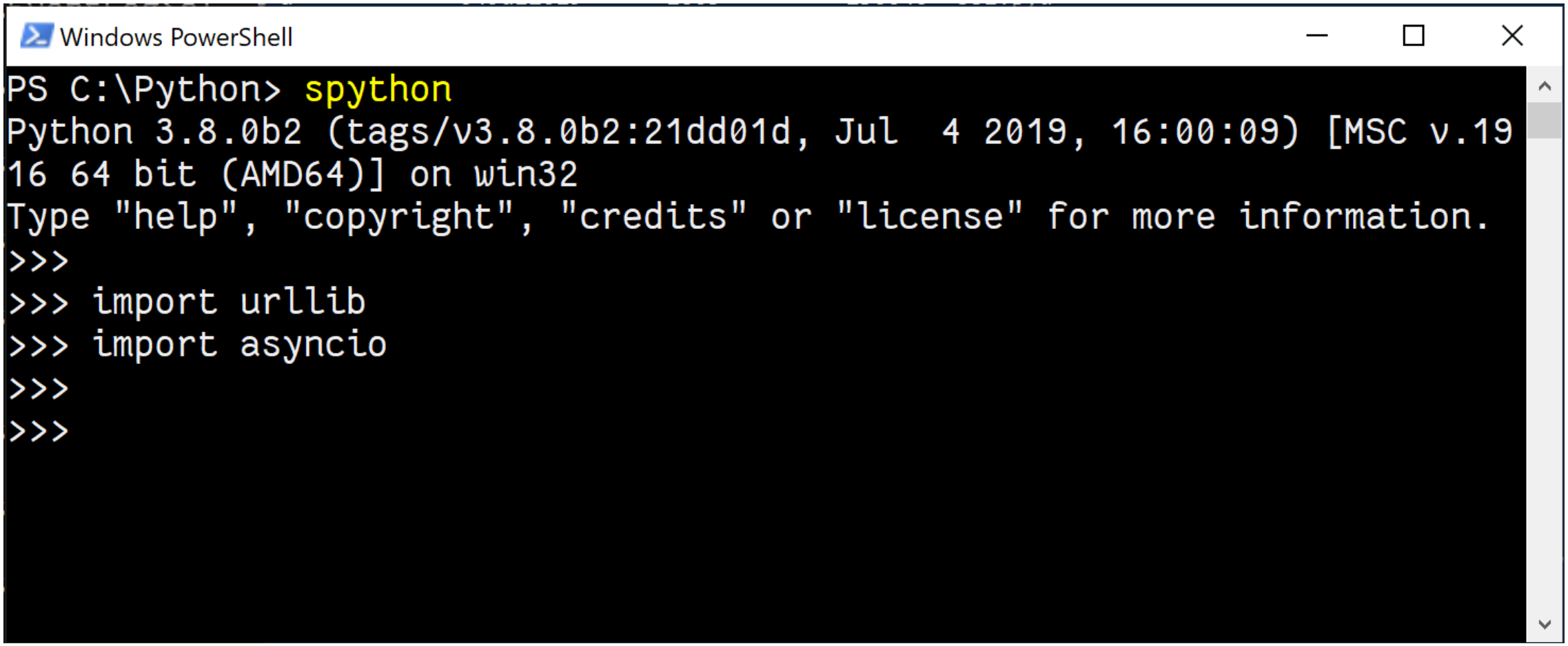

Security Catalog
General
Security Catalog
Security Catalog Information
This security catalog is valid.
Field
Value
Version
V1
Subject usage
1.3.6.1.4.1.311.12.1.1
List identifier
29 b3 06 88 71 a8 cc 47 b2 e5 94 b4...
Effective date
Thursday, July 4, 2019 0901:43
Subject algorithm
1.3.6.1.4.1.311.12.1.3
Thumbprint algorithm
sha1
Thumbprint
98 6e 7a a1 d2 80 f8 d8 5e 70 c5 52...
Value:
View Signature
OK

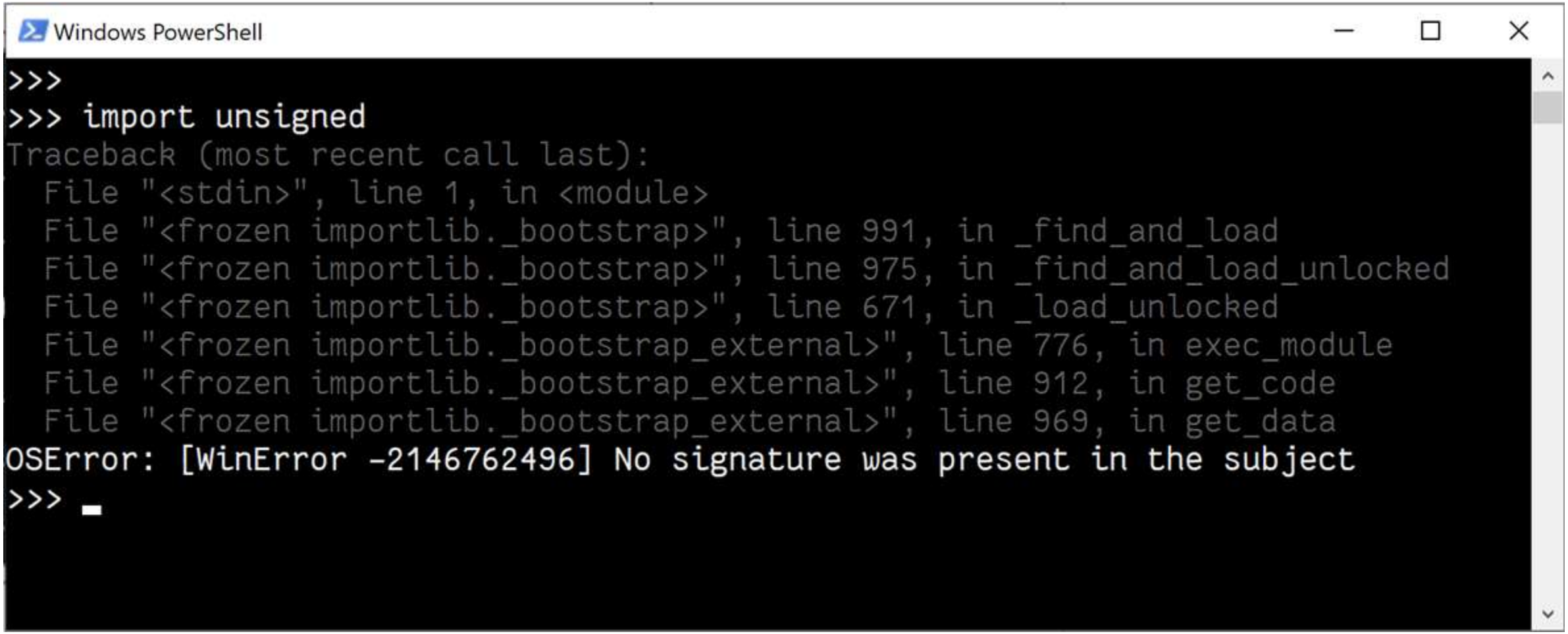

Digital Signature Details
General
Advanced
Digital Signature Information
This digital signature is OK.
Signer information
Name:
Python Software Foundation
E-mail:
Not available
Signing time:
Thursday, July 4, 2019 0903:07
View Certificate
Countersignatures
Name of signer:
E-mail address:
Timestamp
Symantec Time ...
Not available
Thursday, July 4, 20...
Details
OK

```c
static int
verify_trust(HANDLE hFile)
{
    static const GUID action = WINTRUST_ACTION_GENERIC_VERIFY_V2;
    BYTE hash[256];
    wchar_t memberTag[256];

    WINTRUST_CATALOG_INFO wci = {
        .cbStruct = sizeof(WINTRUST_CATALOG_INFO),
        .hMemberFile = hFile,
        .pbCalculatedFileHash = hash,
        .cbCalculatedFileHash = sizeof(hash),
        .pcwszCatalogFilePath = wszCatalog,
        .pcwszMemberTag = memberTag,
    };
    WINTRUST_DATA wd = {
        .cbStruct = sizeof(WINTRUST_DATA),
        .dwUIChoice = WTD_UI_NONE,
        .fdwRevocationChecks = WTD_REVOKE_WHOLECHAIN,
        .dwUnionChoice = WTD_CHOICE_CATALOG,
        .pCatalog = &wci
    };

    if (!CryptCATAdminCalcHashFromFileHandle(
        hFile, &wci.cbCalculatedFileHash, hash, 0)) {
        return -1;
    }

    for (DWORD i = 0; i < wci.cbCalculatedFileHash; ++i) {
        swprintf(&memberTag[i*2], 3, L"%02X", hash[i]);
    }

    HRESULT hr = WinVerifyTrust(NULL, (LPGUID)&action, &wd);
    if (FAILED(hr)) {
        PyErr_SetExcFromWindowsErr(PyExc_OSError);
        return -1;
    }
    return 0;
}
```

```
static int
verify_trust(HANDLE hFile)
{
    static const GUID action = WINTRUST_ACTION_GENERIC_VERIFY_V2;
    BYTE hash[256];
    wchar_t memberTag[256];

    WINTRUST_CATALOG_INFO wci = {
        .cbStruct = sizeof(WINTRUST_CATALOG_INFO),
        .hMemberFile = hFile,
        .pbCalculatedFileHash = hash,
        .cbCalculatedFileHash = sizeof(hash),
        .pcwszCatalogFi
        .pcwszMemberTag
    };
    WINTRUST_DATA wd =
        .cbStruct = sizeof(WINTRUST_DATA),
        .dwUIChoice = WTD_UI_NONE,
        .fdwRevocationChecks = WTD_REVOKE_WHOLECHAIN,
        .dwUnionChoice = WTD_CHOICE_CATALOG,
        .pCatalog = &wci
    };

    if (!CryptCATAdminCalcHashFromFileHandle(
        hFile, &wci.cbCalculatedFileHash, hash, 0)) {
        return -1;

                                        culatedFileHash; ++i) {
                                        , L"%02X", hash[i]);
    }

    HRESULT hr = WinVerifyTrust(NULL, &action, &wd);
    if (FAILED(hr)) {
        PyErr_SetExcFromWindowsErr(PyExc_OSError);
        return -1;
    }
    return 0;
}
```

```
WinVerifyTrust(NULL, &action, &wd)
```

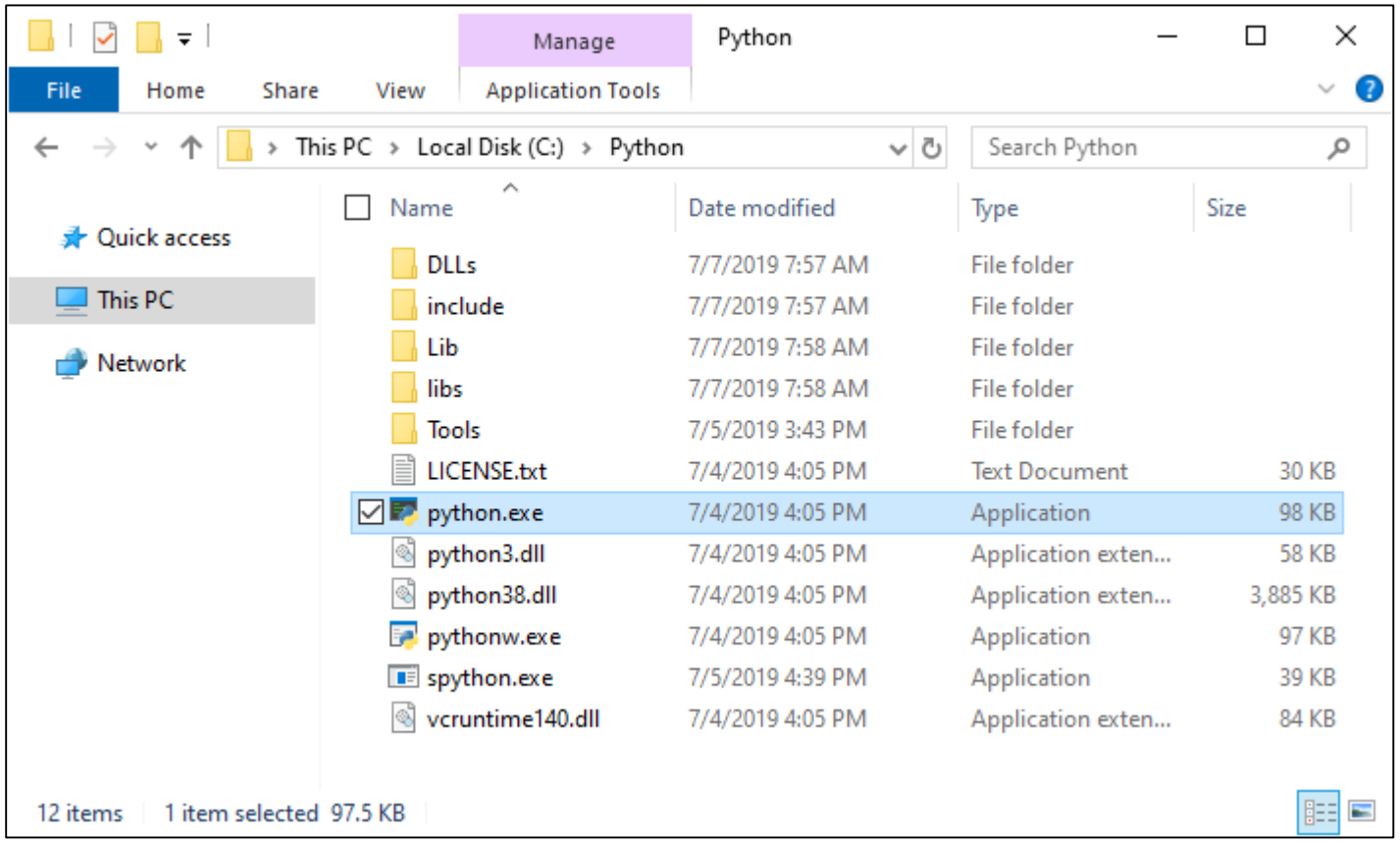

Windows PowerShell
PS C:\Python> spython
Python 3.8.0b2 (tags/v3.8.0b2:21dd01d, Jul 4 2019, 16:00:09) [MSC v.19
16 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license" for more information.
>>>
>>> import urllib
>>> import asyncio
>>>
>>>

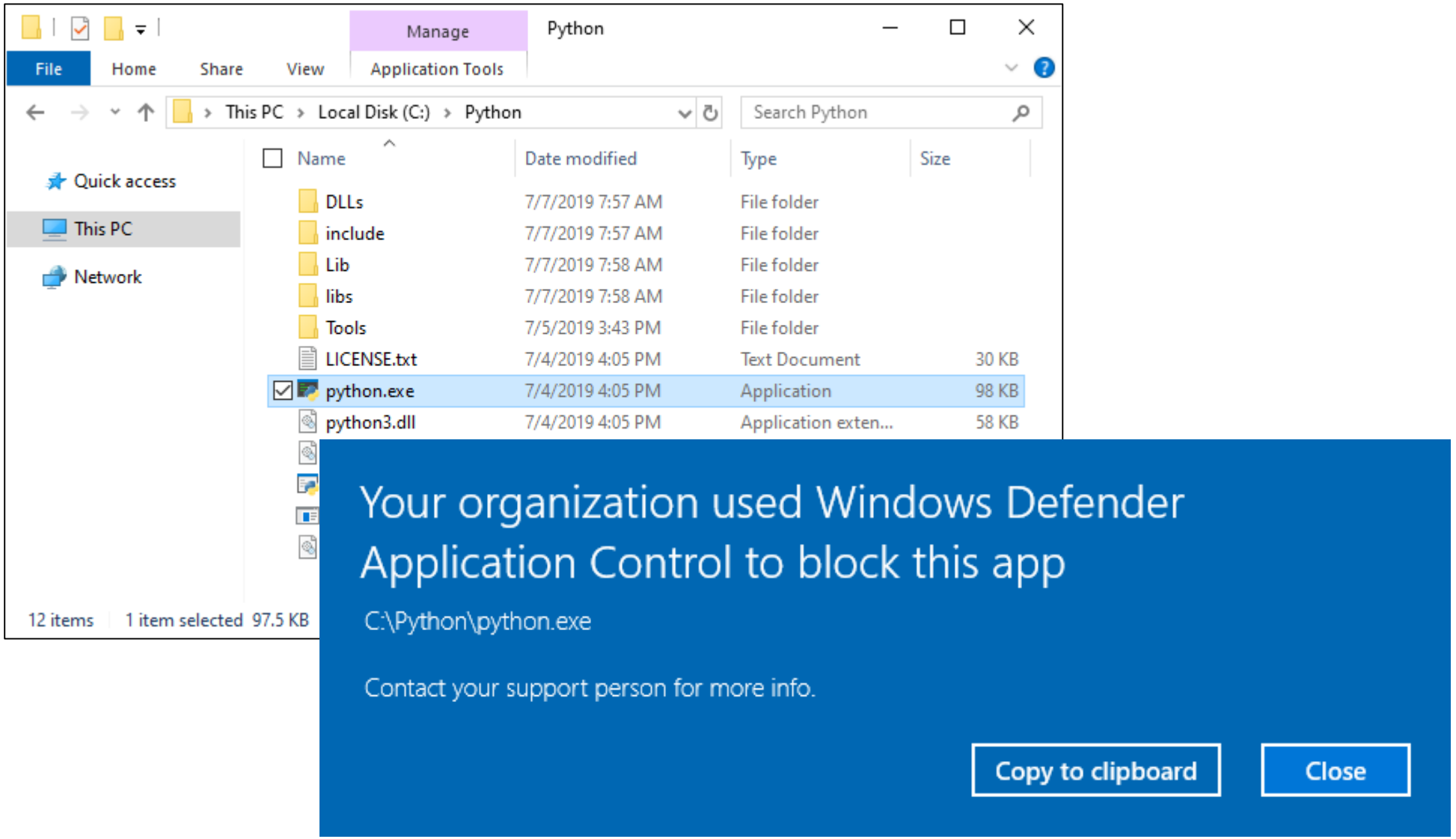

Windows PowerShell
>>>
>>> import unsigned
Traceback (most recent call last):
File "<stdin>", line 1, in <module>
File "<frozen importlib._bootstrap>", line 991, in _find_and_load
File "<frozen importlib._bootstrap>", line 975, in _find_and_load_unlocked
File "<frozen importlib._bootstrap>", line 671, in _load_unlocked
File "<frozen importlib._bootstrap_external>", line 776, in exec_module
File "<frozen importlib._bootstrap_external>", line 912, in get_code
File "<frozen importlib._bootstrap_external>", line 969, in get_data
OSError: [WinError -2146762496] No signature was present in the subject
>>>

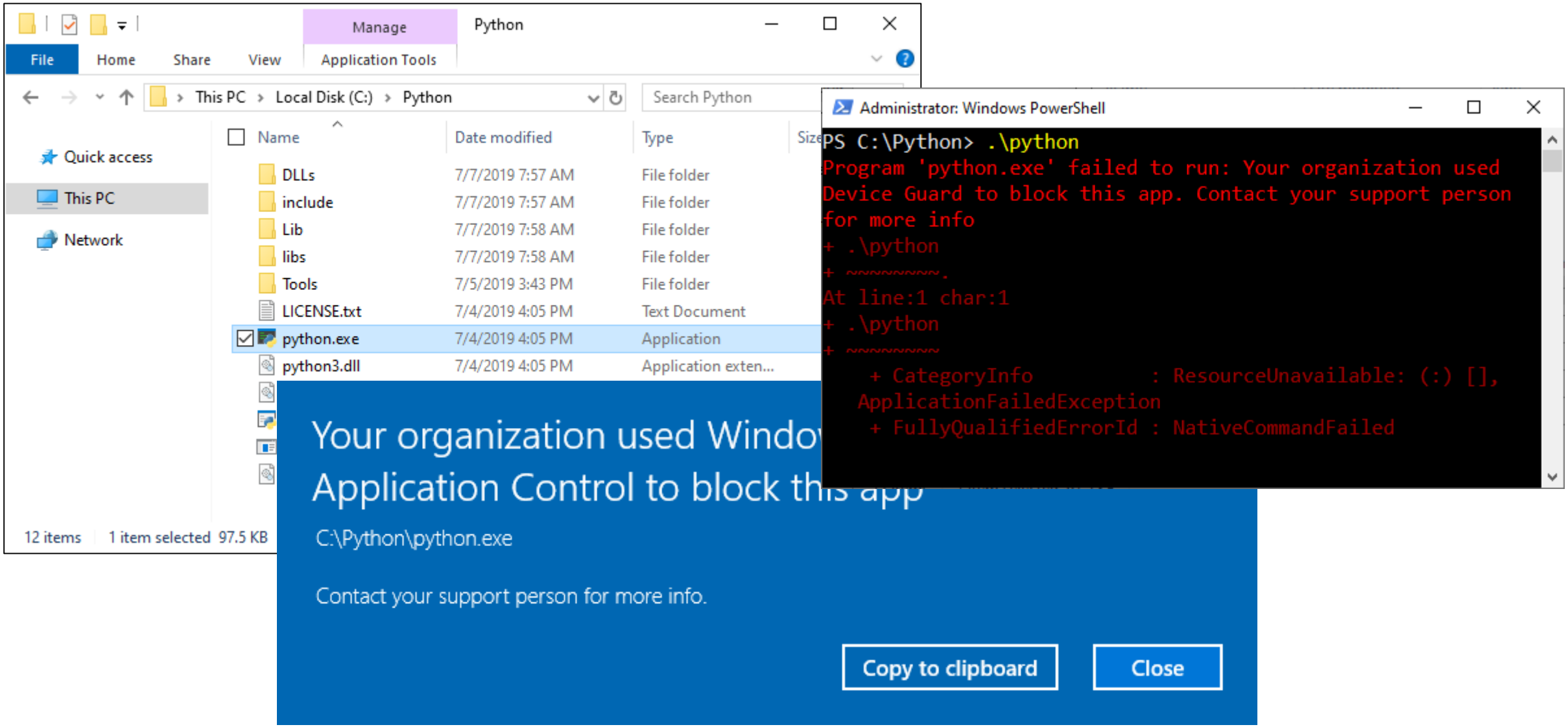
Windows Defender Application Control

Windows Defender Application Control

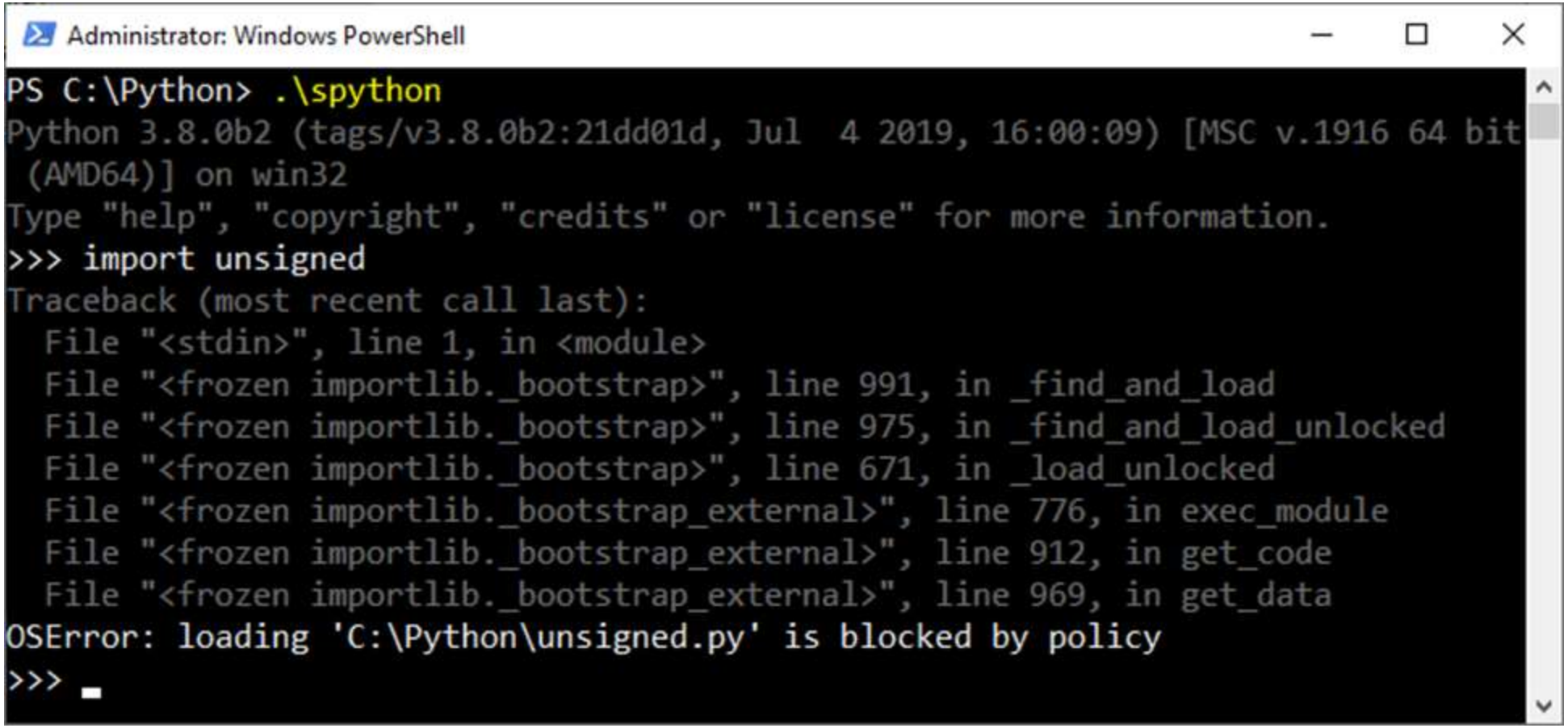
- Kernel enforced, configurable policy for allow/denying applications
- Use signatures, catalogs, file names, paths, etc.
- Integrated with event logging and detectors
- Good feedback for users

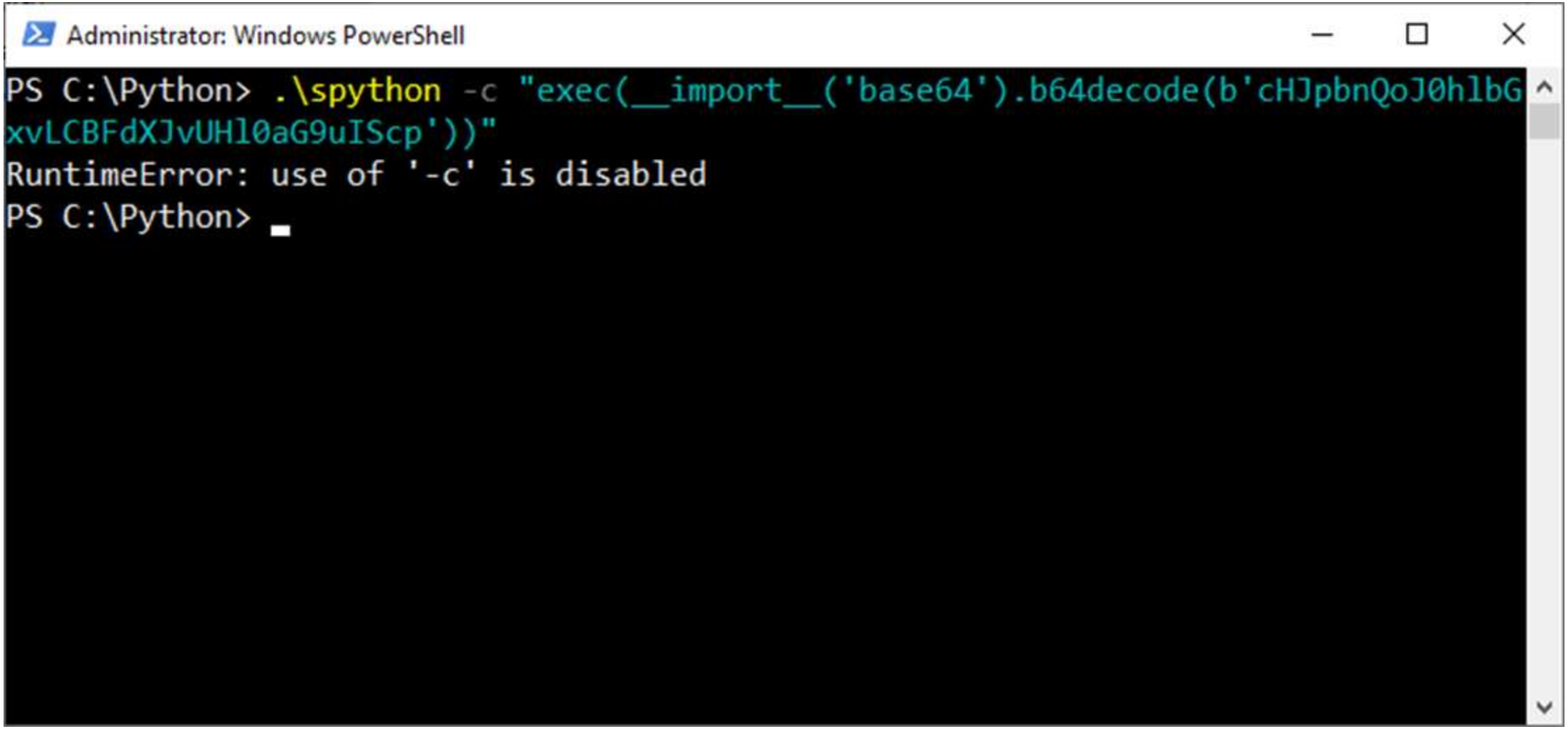

Manage
Python
File
Home
Share
View
Application Tools
This PC › Local Disk (C:) › Python
Search Python
Quick access
This PC
Network
Name
Date modified
Type
Size
DLLs 7/7/2019 7:57 AM File folder
include 7/7/2019 7:57 AM File folder
Lib 7/7/2019 7:58 AM File folder
libs 7/7/2019 7:58 AM File folder
Tools 7/5/2019 3:43 PM File folder
LICENSE.txt 7/4/2019 4:05 PM Text Document 30 KB
python.exe 7/4/2019 4:05 PM Application 98 KB
python3.dll 7/4/2019 4:05 PM Application exten... 58 KB
python38.dll 7/4/2019 4:05 PM Application exten... 3,885 KB
pythonw.exe 7/4/2019 4:05 PM Application 97 KB
spython.exe 7/5/2019 4:39 PM Application 39 KB
vcruntime140.dll 7/4/2019 4:05 PM Application exten... 84 KB
12 items
1 item selected 97.5 KB

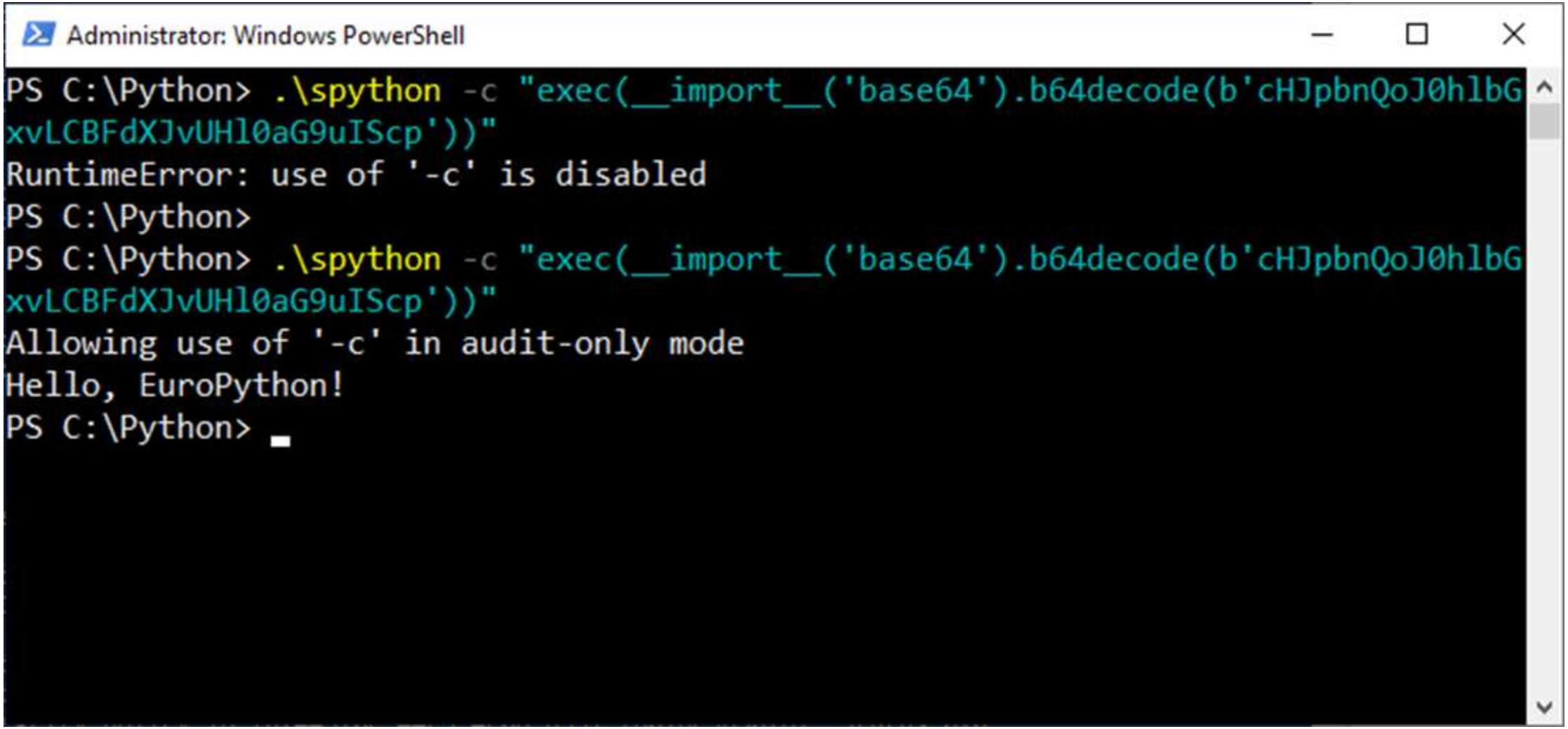

Manage
Python
File
Home
Share
View
Application Tools
This PC › Local Disk (C:) › Python
Search Python
Quick access
This PC
Network
Name
Date modified
Type
Size
DLLs 7/7/2019 7:57 AM File folder
include 7/7/2019 7:57 AM File folder
Lib 7/7/2019 7:58 AM File folder
libs 7/7/2019 7:58 AM File folder
Tools 7/5/2019 3:43 PM File folder
LICENSE.txt 7/4/2019 4:05 PM Text Document 30 KB
python.exe 7/4/2019 4:05 PM Application 98 KB
python3.dll 7/4/2019 4:05 PM Application exten... 58 KB
12 items
1 item selected 97.5 KB
Your organization used Windows Defender Application Control to block this app
C:\Python\python.exe
Contact your support person for more info.
Copy to clipboard
Close


Manage
Python
File
Home
Share
View
Application Tools
This PC › Local Disk (C:) › Python
Search Python
Quick access
This PC
Network
Name
Date modified
Type
Size
DLLs 7/7/2019 7:57 AM File folder
include 7/7/2019 7:57 AM File folder
Lib 7/7/2019 7:58 AM File folder
libs 7/7/2019 7:58 AM File folder
Tools 7/5/2019 3:43 PM File folder
LICENSE.txt 7/4/2019 4:05 PM Text Document
python.exe 7/4/2019 4:05 PM Application
python3.dll 7/4/2019 4:05 PM Application exten...
12 items
1 item selected 97.5 KB
Administrator: Windows PowerShell
PS C:\Python> .\python
Program 'python.exe' failed to run: Your organization used
Device Guard to block this app. Contact your support person
for more info
+ .\python
+ ~~~~~~~~~.
At line:1 char:1
+ .\python
+ ~~~~~~~~~
+ CategoryInfo : ResourceUnavailable: (:) [],
ApplicationFailedException
+ FullyQualifiedErrorId : NativeCommandFailed
Your organization used Windo
Application Control to block this app
C:\Python\python.exe
Contact your support person for more info.
Copy to clipboard
Close

```
- <Signers>
    - <Signer ID="ID_SIGNER_PSF" Name="Python Software Foundation">
          <CertRoot
              Value="CA1C82F3FD767430543909 8D87954FE2AFB4A06424BF492F273461462EEAD733"
              Type="TBS"/>
      </Signer>
  </Signers>
- <SigningScenarios>
    - <SigningScenario ID="ID_SIGNINGSCENARIO_WINDOWS" Value="12">
        - <ProductSigners>
            - <FileRulesRef>
                  <FileRuleRef RuleID="ID_ALLOW_SPYTHON"/>
              </FileRulesRef>
            - <AllowedSigners>
                - <AllowedSigner SignerId="ID_SIGNER_PSF">
                      <ExceptDenyRule DenyRuleID="ID_DENY_PYTHON"/>
                      <ExceptDenyRule DenyRuleID="ID_DENY_SQLITE3"/>
                      <ExceptDenyRule DenyRuleID="ID_DENY_CTYPES"/>
                      <ExceptDenyRule DenyRuleID="ID_DENY_LIBSSL"/>
                  </AllowedSigner>
              </AllowedSigners>
          </ProductSigners>
      </SigningScenario>
  </SigningScenarios>
```


Administrator: Windows PowerShell
PS C:\Python> .\spython
Python 3.8.0b2 (tags/v3.8.0b2:21dd01d, Jul 4 2019, 16:00:09) [MSC v.1916 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license" for more information.
>>> import unsigned
Traceback (most recent call last):
File "<stdin>", line 1, in <module>
File "<frozen importlib._bootstrap>", line 991, in _find_and_load
File "<frozen importlib._bootstrap>", line 975, in _find_and_load_unlocked
File "<frozen importlib._bootstrap>", line 671, in _load_unlocked
File "<frozen importlib._bootstrap_external>", line 776, in exec_module
File "<frozen importlib._bootstrap_external>", line 912, in get_code
File "<frozen importlib._bootstrap_external>", line 969, in get_data
OSError: loading 'C:\Python\unsigned.py' is blocked by policy
>>>


Administrator: Windows PowerShell
PS C:\Python> .\spython -c "exec(__import__('base64').b64decode(b'cHJpbnQoJ0hlbG
xvLCBFdXJvUHl0aG9uIScp'))"
RuntimeError: use of '-c' is disabled
PS C:\Python>


Administrator: Windows PowerShell
PS C:\Python> .\spython -c "exec(__import__('base64').b64decode(b'cHJpbnQoJ0hlbG
xvLCBFdXJvUHl0aG9uIScp'))"
RuntimeError: use of '-c' is disabled
PS C:\Python>
PS C:\Python> .\spython -c "exec(__import__('base64').b64decode(b'cHJpbnQoJ0hlbG
xvLCBFdXJvUHl0aG9uIScp'))"
Allowing use of '-c' in audit-only mode
Hello, EuroPython!
PS C:\Python>

Integrating with Linux

Integrating with Linux

- DTrace / SystemTap
- SysLog
- io.open_code()

Prerequisites

- Install security updates
- Run as unprivileged user or drop capabilities (container)
- Restrict write access
- Enforce security policy: AppArmor, SELinux, TOMOYO
- Configure system and central logging: syslog, rsyslog, journald

DTrace / SystemTap instrumentation

```
# audit(str event, void *tuple)
probe process("/usr/lib64/libpython3.8.*").mark("audit") {
    printf("%s\n", user_string($arg1))
}
```

```
$ sudo stap audit.stp  -c "python3.8 -c pass"
...
cpython.run_command
compile
exec
```

More on DTrace and SystemTap tomorrow at 10:30am from Christian

Logging

```
openlog(NULL, LOG_CONS|LOG_PERROR|LOG_PID, LOG_USER);

syslog(LOG_CRIT, "spython critical failure: %s", msg);
_exit(255);
```

Configure your container runtime to forward syslog!

io.open_code() on Linux

Simple proof-of-concept

- Resolved file must be a regular file
- Deny *noexec* filesystems (/proc, hardened /tmp)
- Hash file content with OpenSSL
- Use xattr (extended file attributes) to flag files and store hash

github.com/zooba/spython/tree/master/linux_xattr

Extended file attributes

- Custom name/values pairs on files and directories
- Namespaces: user, trusted, system, security
- Access permission to “user” namespace is controlled by DAC.
- Inspired by “Integrity Measurement Architecture” (IMA-appraisal)

```
$ getfattr -d /usr/lib64/python3.8/os.py
user.org.python.x-spython-hash="75454b1944227c1418473..."
```

Example

```
$ ./spython example.py
Fatal Python error: init_import_size: Failed to import the site module
Traceback (most recent call last):
   ...
ValueError: File hash mismatch: /usr/lib64/python3.8/os.py (expected: '75454b...',
got '31d6c3...')
```

```
$ sudo python3.8 ./mkxattr.py --verbose
Adding spython hash to '/usr/lib64/python3.8/os.py'
Adding spython hash to '/usr/lib64/python3.8/__pycache__/os.cpython-38.pyc'

$ ./spython example.py
OK
```

Example – mkxattr

```
XATTR_NAME = b"user.org.python.x-spython-hash"
for filename in LIST_OF_PY_FILES:
    hasher = hashlib.new("sha256")
    with open(filename, "rb") as f:
        hasher.update(f.read())
    hexdigest = hasher.hexdigest().encode("ascii")

    os.setxattr(filename, xattr_name, hexdigest)
```

Securing xattr

- Store hash in restricted xattr namespace
- Use signed hash
- Block syscall (container policy, seccomp)

```
prctl(PR_SET_NO_NEW_PRIVS, 1, 0, 0, 0);
scmp_filter_ctx *ctx = seccomp_init(SCMP_ACT_ALLOW);
// setxattr, fsetxattr, lsetxattr
seccomp_rule_add(ctx, SCMP_ACT_KILL_PROCESS,
                 SCMP_SYS(setxattr), 0);
seccomp_load(ctx);
```

Open issues and exploits

- `LD_PRELOAD`
- Open Container Image Format clobbers xattr in layers
  - github.com/opencontainers/image-spec/issues/503
- Modify code with `/proc/self/mem`
- `void *dlopen(const char *filename, int flags)`
  - github.com/nullbites/SnakeEater
- …

O_MAYEXEC

- GNU/Linux CLIP OS 4
- Articles and talks
  - Linux Security Summit Europe 2018
  - Kernel Recipes 2018
  - lwn.net/Articles/774676

Summary

- When your security is already good, audit hooks can make it better
- Hooks provide transparency, not security
- Enables use of OS technologies that was unavailable to Python
- Install your security updates!

Resources

- docs.python.org/3.8/library/sys.html#sys.audit
- www.python.org/dev/peps/pep-0578/
- github.com/zooba/spython

Steve Dower
@zooba

Christan Heimes
@christianheimes