Hunting for LoLs

A ML Living off the Land Classifier

whoami

Adobe
Security
Intelligence
Team

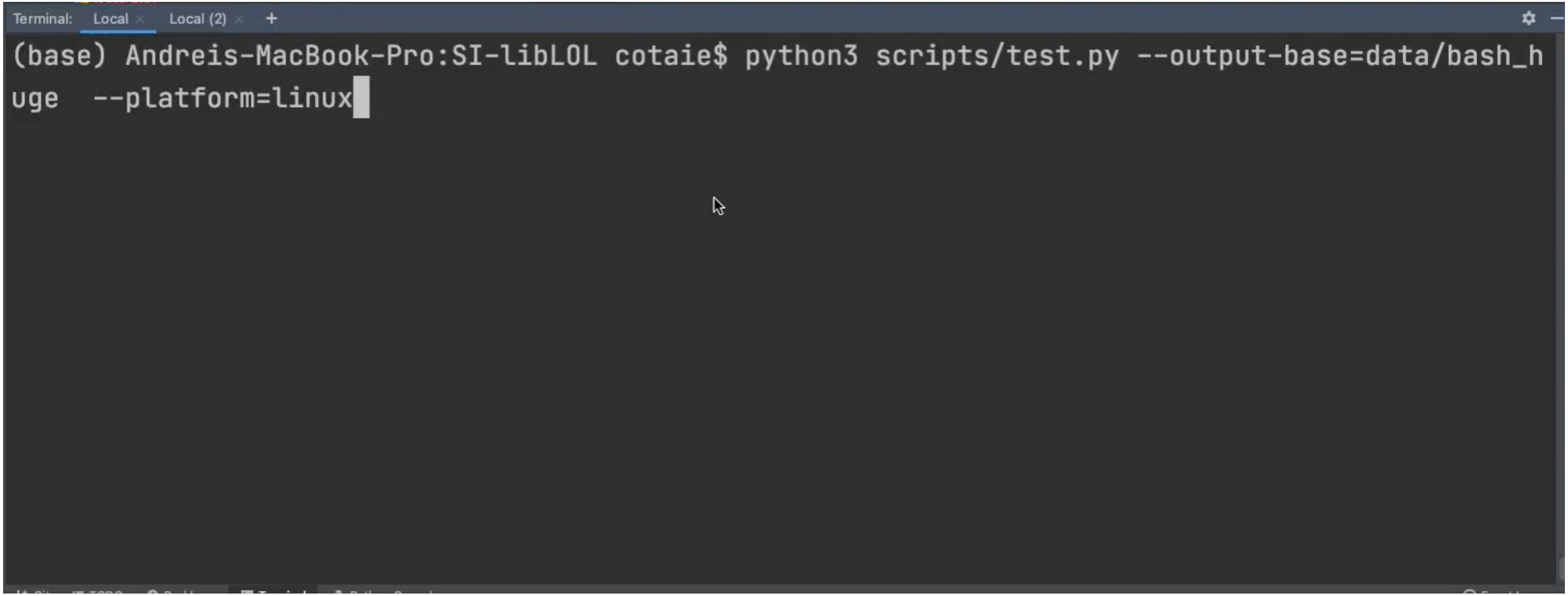
- Andrei Cotaie
  - Technical Lead - Security Engineer
  - cotaie@adobe.com
- Tiberiu Boros
  - Data Scientist / Machine Learning Engineer
  - boros@adobe.com
- Kumar Vikramjeet
  - Security Engineer
  - vikrakum@adobe.com
- Vivek Malik
  - Incident responder
  - vivmalik@adobe.com

What is Living off the Land?

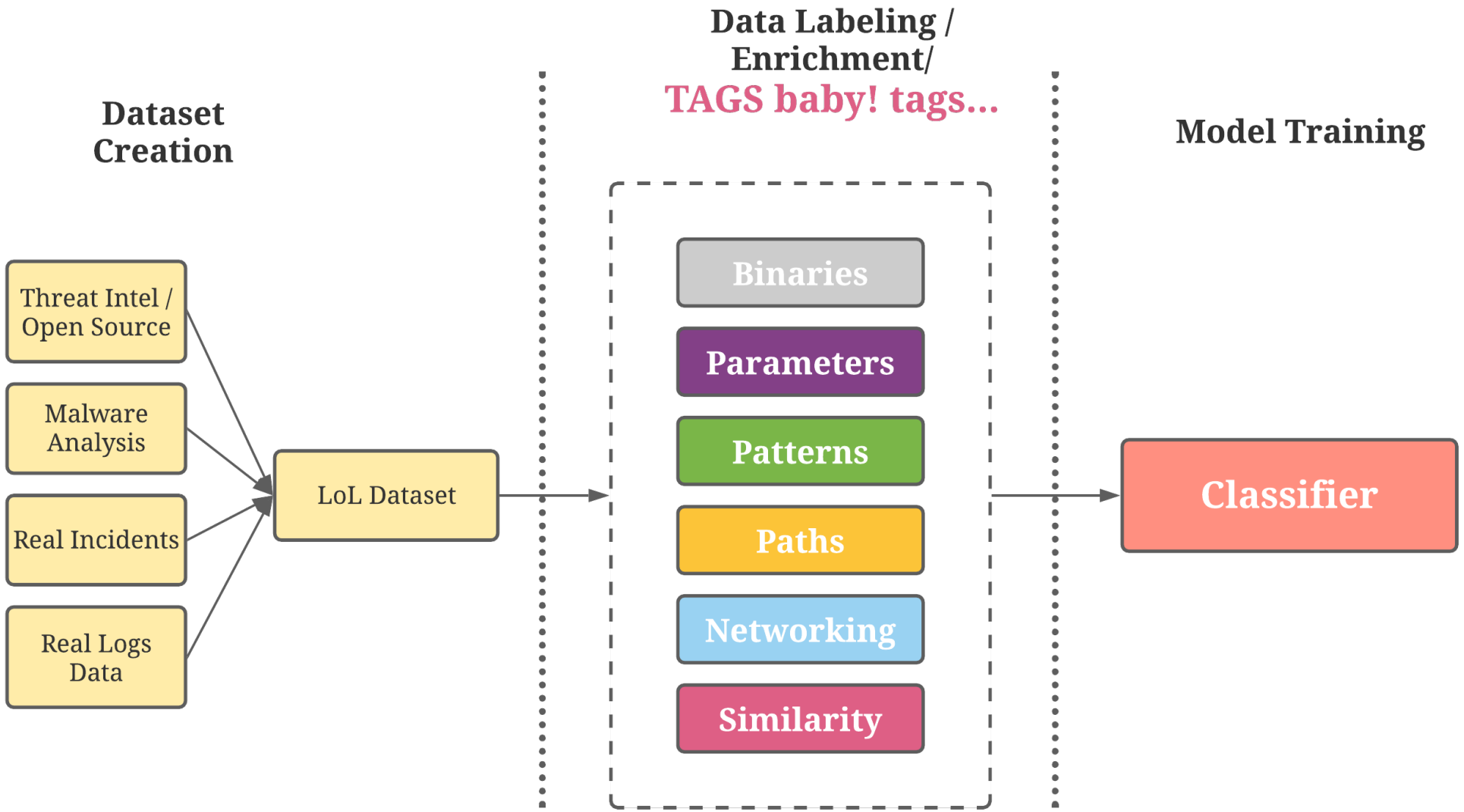
YO admin… is that you?

- Unix:
  - https://gtfobins.github.io/
- Windows:
  - https://lolbas-project.github.io/

What does libLOL/lolc do ?



It’s all about the tags!



Command/Binaries and Keywords Tags

```
>>> import shlex
>>> text = 'python3 -c __import__("pty").spawn("/bin/sh")'
>>> c_toks = list(shlex.shlex(text, punctuation_chars=True, posix=True))
>>> print(c_toks)

['python3', '-c', '__import__', '(', 'pty', ')', '.spawn', '(', '/bin/sh', ')']

>>> if "python3" in c_toks:
...     print("COMMAND_PYTHON3")
...

COMMAND_PYTHON3

>>> if "pty" in c_toks:
...     print("KEYWORD_PTY")
...

KEYWORD_PTY
```

The Paths we take…

'/dev/mem', '/dev/tcp', '/dev/udp',
'/dev/kmem', '/dev/null', '/inet/tcp',
'/bin/sh', '/inet/udp', '/etc/crontab',
'/var/spool/cron', '/bin/bash',
'/etc/passwd', '/etc/issue',
'/etc/shadow', '/proc/version',
'/etc/ssh/sshd_config',
'/etc/services',
'/etc/network/interfaces',
'/etc/sysconfig/network',
'/etc/resolv.conf', '/etc/fstab',
'/etc/group', '/etc/hostname',
'/etc/hosts', '/etc/inetd.conf',
'/boot', '.ssh/', '/etc/fstab',
'/etc/profile', '/bin/bash',
'/tmp/bash', '/etc/sudoers',
'/etc/vncpasswd',
'/proc/','/var/tmp/', '/dev/tty'

Networking with my data

```python
from ipaddress import ip_address
import re2 as re
_re_ipv4 = r"((25[0-5]|2[0-4][0-9]|" \
           r"[01]?[0-9][0-9]?)\.){3}" \
           r"(25[0-5]|2[0-4][0-9]|[01]?[0-9]" \
           r"[0-9]?)"
def _match_ips(text):
    try:
        ips = re.finditer(_re_ipv4, text)
        labels = {}
        for ip in ips:
            ipt = ip_address(ip.group())
            if ipt.is_link_local:
                resp = "link_local"
            elif ipt.is_loopback:
                resp = "loopback"
            elif ipt.is_multicast:
                resp = "multicast"
            elif ipt.is_reserved:
                resp = "reserved"
            elif ipt.is_unspecified:
                resp = "unspecified"
            elif ipt.is_private:
                resp = "private"
            else:
                resp = "public"

            resp = "IP_{0}".format(resp).upper()
            if resp not in labels:
                labels[resp] = 1
    except:
        pass
    return [label for label in labels]
```

```
>>> ssh -R 1234:127.0.0.1:4321 root@11.22.33.44
[Parallel(n_jobs=16)]: Using backend ThreadingBackend with 16 concurrent workers.
[Parallel(n_jobs=16)]: Done   9 tasks      | elapsed:    0.0s
[Parallel(n_jobs=16)]: Done  45 out of  50 | elapsed:    0.0s remaining:    0.0s
[Parallel(n_jobs=16)]: Done  50 out of  50 | elapsed:    0.0s finished
Extracted features: IP_LOOPBACK IP_PUBLIC COMMAND_SSH KEYWORD_-R ssh_R LOOKS_LIKE_KNOWN_LOL
Prediction: BAD
```

Regex – like it's your last string

```
(?P<comm>comm.*dev.*)
(?P<perl_shell>perl.*\-e.*sh)
(?P<tcpdump_eth>tcpdump.*eth)
(?P<ping_p>ping.*\-p)
(?P<openvt_shell>openvt.*sh.*\-c)
(?P<bin_sh_invokation>\#\!\/bin\/.{0,2}sh)
(?P<amazon>amazonaws\.com)
```

```
(?P<pastebin>pastebin\.com)
(?P<nc_impersonation>\-e.*\/bin\/\w{0,2}sh)
(?P<nmap_sU>nmap.*-sU)
(?P<find_bash>find.*bin\/.*sh)
(?P<exec_bin_sh>exec.*\/bin\/.*sh)
(?P<ssh_R>ssh.*\-R)
```

```
(?P<python_shell>python.*\-c.*pty.*sh)
```

```
(?P<socat_tcp_listen>socat.*tcp\-listen)
(?P<nmap_O>nmap.*-O)
(?P<sh_i>sh\s\-i)
(?P<sshuttle_r>sshuttle.*\-r)
(?P<ksh_tcp>ksh.*-c.*dev.*tcp)
(?P<sqlite3_shell>sqlite3.*shell.*sh)
(?P<perl_rever_shell>perl.*\-e.*socket.*\/bin\/.*sh)
```

```
(?P<sh_dev_tcp_i>sh.*\-i.*dev.*tcp)
(?P<php_listener>php.*\-S)
(?P<python_system_shell>python.*\-c.*system.*sh)
(?P<python__import>python.*\-c.*\_\_import)
(?P<smbclient_shell>smbclient.*\!.*sh)
(?P<lua_socket>lua.*socket)
```

Let the data speak

- Linux model (stable)
- Number of LOL command lines: 1609
- Number of benign command lines: 7.9M
- Reduced number of FPs
- Ratio: 0.02%
- ~500 unique tags
- 7.554.915 – total tags in the dataset

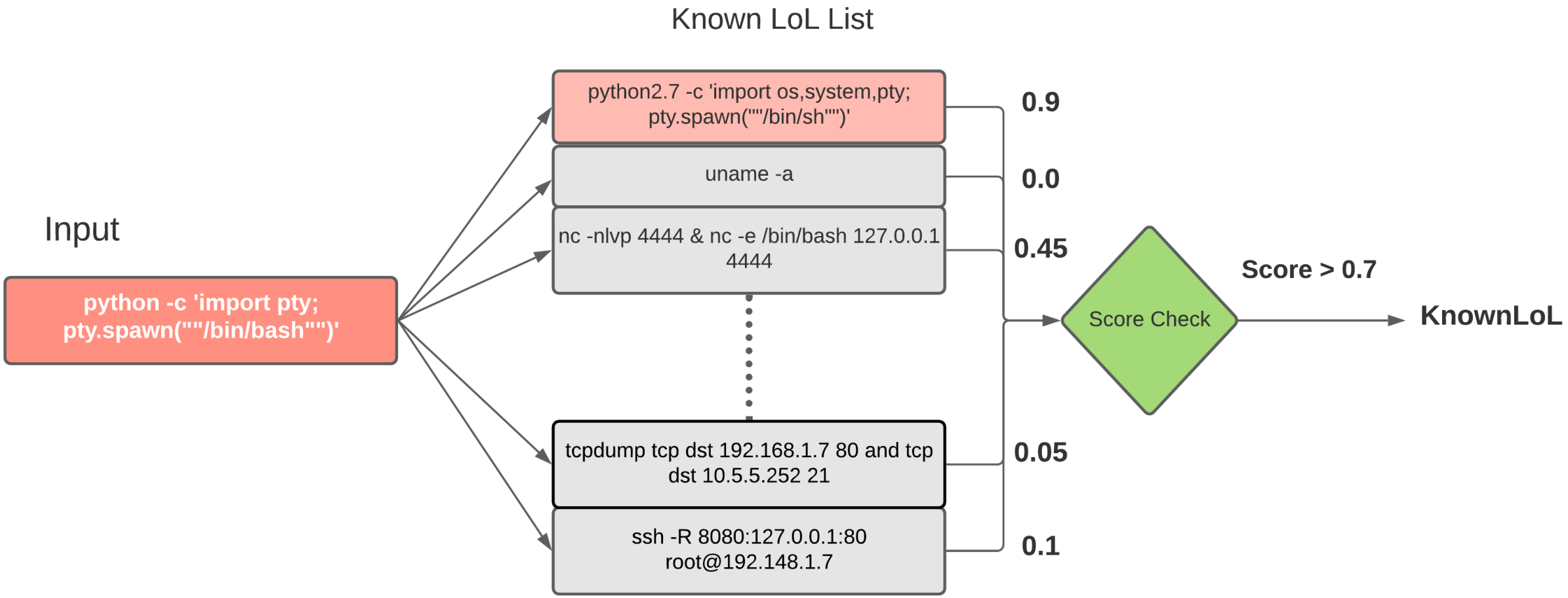
Into the FOREST. The Classifiers

Classifier	F-Score	Standard deviation	Training time
Random forest	0.95	0.013	18 minutes
SVM (intel patch)	0.95	0.027	3.5 hours
Logistic regression	0.93	0.014	1.2 hours

5-fold cross validation results

Shortcut: The looks_like_known_lol TAG

IF it **looks** like a **duck**, **swims** like a **duck**, and **quacks** like a **duck**, then it probably is a **duck**.



Install LoL

- GitHub: https://github.com/adobe/libLOL
  - git clone git@github.com:adobe/libLOL.git
  - cd libLOL
  - virtualenv –p `which python3` venv
  - source venv/bin/activate && pip install –r requirements
- PIP package: lolc
  - pip install lolc

Run LoL

Step 1 – Import packages:

```
from lol.api import LOLC, PlatformType
```

Step 2 – Create a new instance:

```
lolc=LOLC(PlatformType.LINUX) # allowed parameters are PlatformType.LINUX and
PlatformType.WINDOWS
```

Step 3 – Get your data:

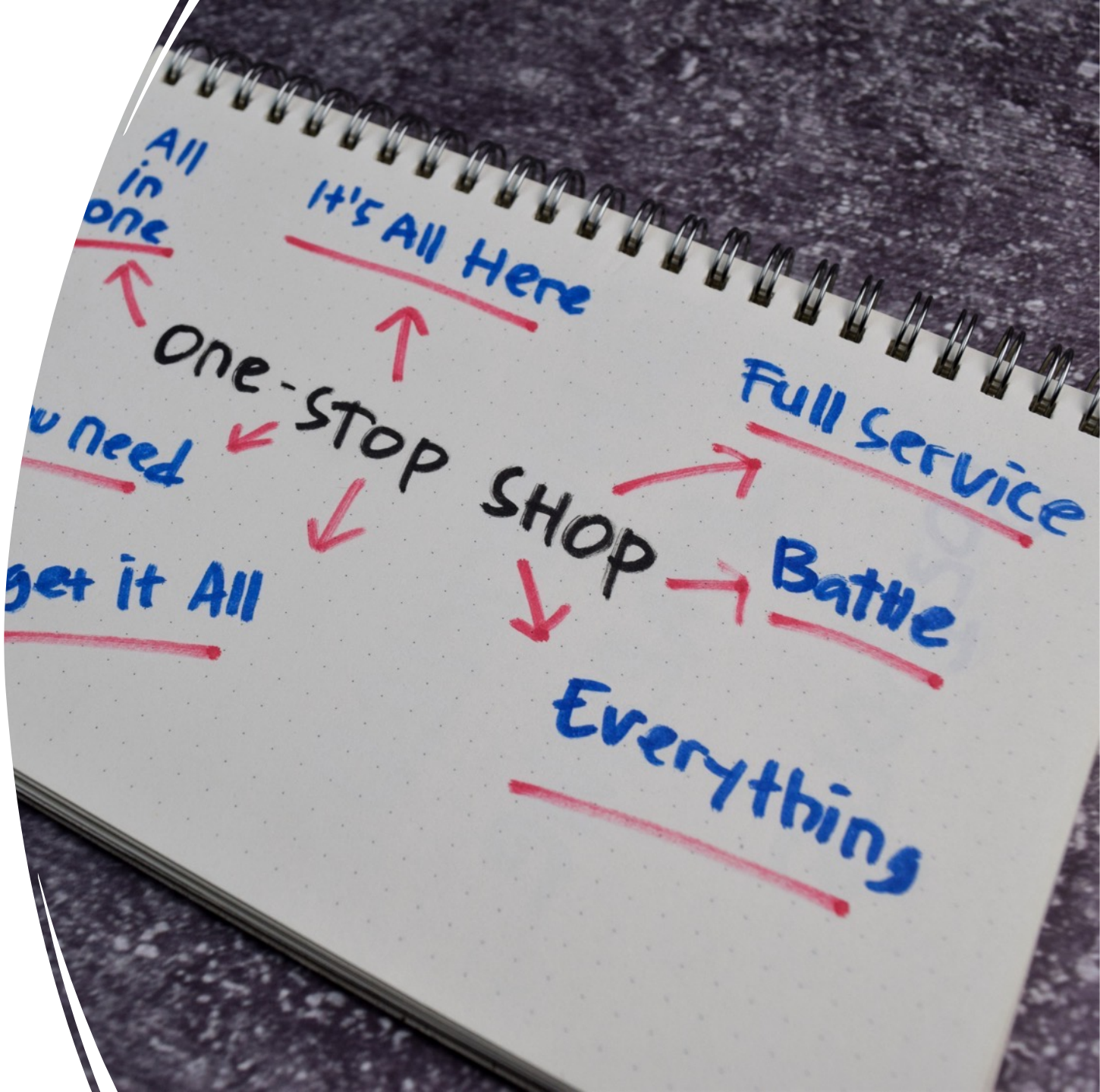
```
commands=[…,…]
```

Step 4 – Run the classifier:

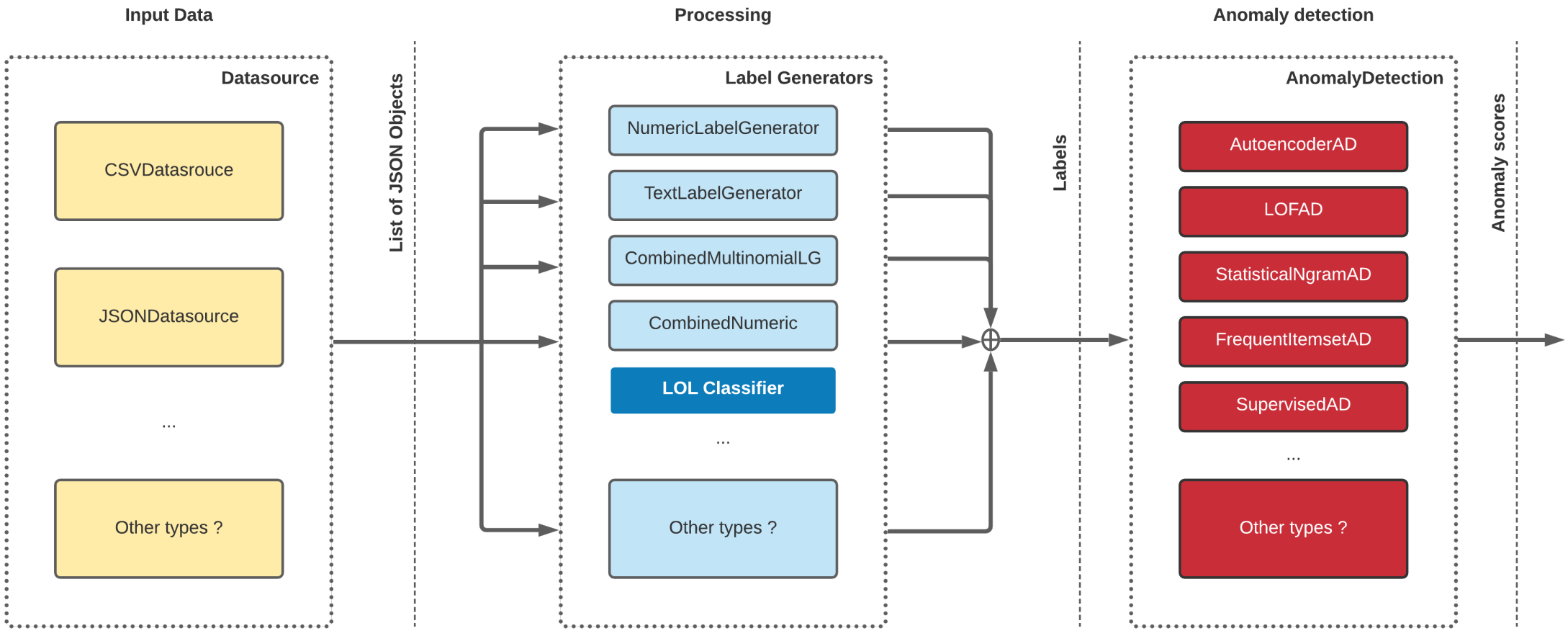
```
classification, tags = lolc(commands)
```

Demo time

What’s OSAS?!


All in one
It's All Here
One-Stop Shop
Full Service
Battle
Everything
Get it All

Adding OSAS in the mix !



https://github.com/adobe/OSAS

How to Get and Use Adobe's SI Living of the Land

GitHub: https://github.com/adobe/libLOL

Blog post:

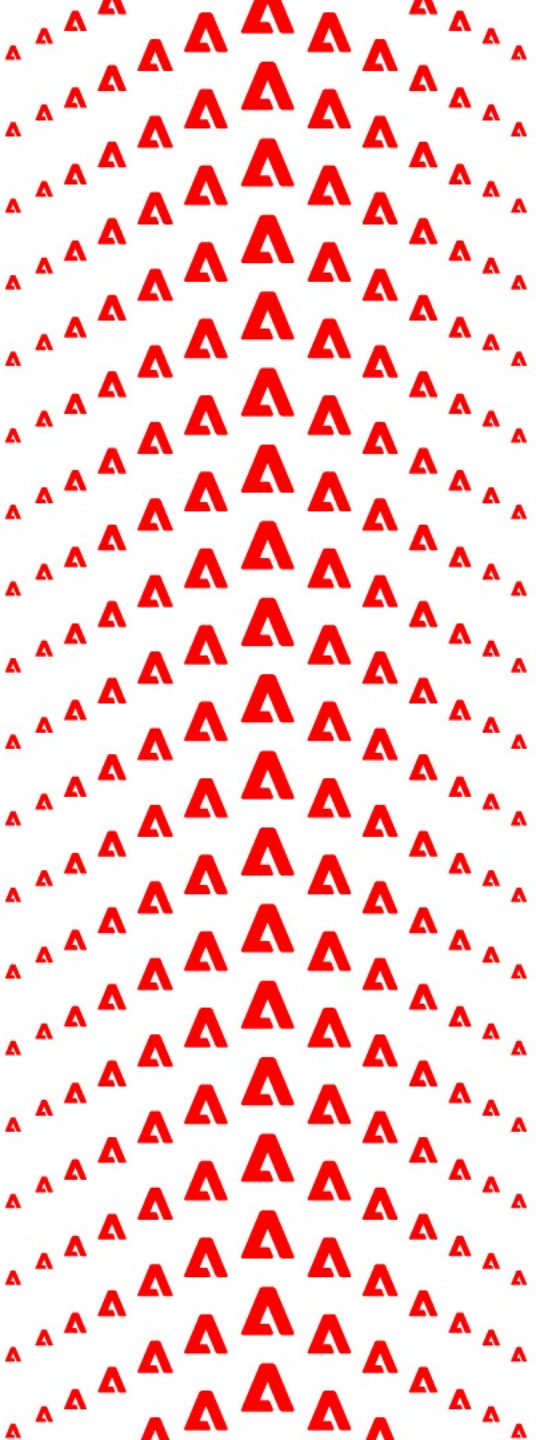
https://medium.com/adobetech/living-off-the-land-lotl-classifier-open-source-project-b167484c8187

PIP package:

https://pypi.org/project/lolc/

OSAS:

https://github.com/adobe/OSAS



Resources

Adobe Security Newsletter
adobe.com/go/securitynews

Twitter
@AdobeSecurity

Trust Center
trust.adobe.com

Open Source CCF v3.0
adobe.com/go/open-source-ccf

Security @ Adobe blog
medium.com/adobetech/tagged/security


Adobe