```
class Meta(Persistent):
    """A small grab bag to store persistent meta information in
    the database.

    """

    unpacked = 0


class Archive(Persistent):

    # Either set to a duration to update every `age_limit` seconds or to None
    # to never update after the initial fetch.
    age_limit = None
    name = None
    products = None
    last_update = 0

    _cleanup = ()

    def __init__(self, name):
        self.name = name

    @property
    def upstream_filename(self):
        return 'nvdcve-2.0-{}.xml.gz'.format(self.name)

    @property
    def is_current(self):
        if self.age_limit is None:
            # We already have this data and don't want to update it.
            return True
        return (self.last_update + self.age_limit > time.time())

    def update(self, mirror):
        logger.info('Updating {}'.format(self.name))
        try:
            filename = self.download(mirror)
            self.parse(filename)
        except Exception:
            self.clean()
            raise
        self.last_update = time.time()
        return True

    def download(self, mirror):
        self._cleanup = []

        # Phase 1: download
        _, compressed = tempfile.mkstemp()
        self._cleanup.append(compressed)

        url = mirror + self.upstream_filename
        logger.debug("Downloading {}".format(url))
        r = requests.get(url, stream=True)
        r.raise_for_status()
        with open(compressed, 'wb') as fd:
            for chunk in r.iter_content(128*1024):
                fd.write(chunk)
```

http://tinyurl.com/vulnix

Vulnix

Ein vulnerability scanner für Nix/NixOS

Aufgabe

Gib mir eine Liste von Programmen aus, die **womöglich** von Sicherheitslücken betroffen sind

Nix auf einer Slide

- **Nix** = Interpreter und Programmiersprache, beschreibt Paketerstellung und Abhängigkeiten (lazy!)
- **NixOS** = wie Nix, zusätzlich auch Systemzustände
- **Output Paths** = Dateien und Ordner im /nix/store
- **Garbage Collector Roots** = Symlinks aus dem /nix/store nach /nix/var/nix/gcroots/$name
- **Derivations** = Paketerzeugungsdeklaration 😳

Kontext

Tool

- In Python implementiertes CLI Werkzeug
- läuft auf Nix (Linux, macOS) und NixOS
- Monitoring freundliche Ausgabe (z.B. Sensu)
- Whitelisting möglich
- https://github.com/flyingcircusio/vulnix

Tool

```
➜  vulnix git:(master)  bin/vulnix
Usage: vulnix {--system | PATH [...]}

vulnix is a tool that scan the NixOS store for packages with
known
security issues. There are two main modes of operation:

* Is my NixOS system installation affected?

Invoke:  vulnix --system


* Is my project affected?

Invoke after nix-build:  vulnix ./result
```

Tool

```
➜  vulnix git:(master)  bin/vulnix ./result
Found 5 advisories for busybox, bzip2, gcc, ... (and 2 more)

================================================================
busybox

CVEs:
    https://web.nvd.nist.gov/view/vuln/detail?vulnId=CVE-2016-6301
    https://web.nvd.nist.gov/view/vuln/detail?vulnId=CVE-2016-2147
    https://web.nvd.nist.gov/view/vuln/detail?vulnId=CVE-2016-2148
    https://web.nvd.nist.gov/view/vuln/detail?vulnId=CVE-2013-1813

================================================================
bzip2-1.0.6

CVEs:
    https://web.nvd.nist.gov/view/vuln/detail?vulnId=CVE-2016-3189
```

CVE/NVD

- Parsen der CVE-Archive (Common Vulnerabilities and Exposures)
- U.S. National Vulnerability Database (NVD)
  Datenbanken 🇺🇸
- XML: <entry id="CVE-$YEAR-$id">…</entry>
- CPE-Language: stellt u.a. Relationen zwischen Produkten, Versionen, Herstellern, Betriebssystemen her
- im Moment, die letzten fünf Jahre (geht bis 2002)

CVE/NVD

```
<entry id="CVE-2016-3189">
    <vuln:vulnerable-configuration id="http://nvd.nist.gov/">
      <cpe-lang:logical-test operator="OR" negate="false">
        <cpe-lang:fact-ref name="cpe:/a:bzip:bzip2:1.0.6"/>
      </cpe-lang:logical-test>
    </vuln:vulnerable-configuration>
    <vuln:vulnerable-software-list>
      <vuln:product>cpe:/a:bzip:bzip2:1.0.6</vuln:product>
 ...
    <vuln:summary>Use-after-free vulnerability in
bzip2recover in bzip2 1.0.6 allows remote attackers to cause
a denial of service (crash) via a crafted bzip2 file, related
to block ends set to before the start of the block.</
vuln:summary>
  </entry>
```

Nix Expression

```
{ stdenv, fetchurl, linkStatic ? false }:

let version = "1.0.6"; in

stdenv.mkDerivation {
  name = "bzip2-${version}";

...
}
```

- mkDerivation orchestriert den buildprozess
- nix-instantiate erzeugt drv file

Nix Expression

```
Derive([("out","/nix/store/hgv2y5a75hs52hh1riydwvq56fq470as-bzip2-1.0.6","","")],[("/nix/
store/l243x77fng2d362f6xma6mr1y1fpsnby-stdenv-linux-boot.drv",["out"]),("/nix/store/
rlbfc41y1hkdhympy67lbfg8r6nc93ab-bootstrap-tools.drv",["out"]),("/nix/store/
sli88bbqn7air359789 4l9dz46hkifd7-bzip2-1.0.6.tar.gz.drv",["out"])],["/nix/store/
4z4mw30jmvi1s8fdyx504bjhlvqm9zvp-builder.sh"],"x86_64-linux","/nix/store/
k0vqprjmxybr7clvfljk13zsdjwklcch-bootstrap-tools/bin/sh",["-e","/nix/store/
4z4mw30jmvi1s8fdyx504bjhlvqm9zvp-builder.sh"],[("buildInputs",""),("builder","/nix/store/
k0vqprjmxybr7clvfljk13zsdjwklcch-bootstrap-tools/bin/sh"),("linkStatic",""),("makeFlags",""),
("name","bzip2-1.0.6"),("nativeBuildInputs",""),("out","/nix/store/
hgv2y5a75hs52hh1riydwvq56fq470as-bzip2-1.0.6"),("patchPhase",""),
("preConfigure","substituteInPlace Makefile --replace '$(PREFIX)/man' '$(PREFIX)/share/
man'"),("propagatedBuildInputs",""),("propagatedNativeBuildInputs",""),("sharedLibrary","1"),
("src","/nix/store/g2n88bdva89wyzbh854l597z4c49l690-bzip2-1.0.6.tar.gz"),("stdenv","/nix/
store/sxlcjmaff6cja2rfjqb3f54qbhsd2m0f-stdenv-linux-boot"),("system","x86_64-linux")])
```

- Python kompatible Syntax? 🤫

Algorithmus

- Cachen der CVE-Archive
- Ermitteln aller abhängigen Derivations
- Namensmatching der CVE-Einträge mit den ermittelten Derivations **no magic!**

Whitelisting

- false positives (Namen sind tricky)
- für operations: work-in-progress marker
- yaml-file

Whitelisting

```yaml
# Default vulnix whitelist listing common exceptions. The entries given below
# should be common sense for every Nix installation.
-
    cve: CVE-2015-2503
    comment: |
         microsoft access, accidentally matching the 'access' derivation

         https://plan.flyingcircus.io/issues/18544

-
    vendor: microsoft
    product: access
```

Finale

- $ python3.x -m venv .
- $ bin/pip install -e . \[test]
- twitter: @dvhfm
- github: plumps