Hardcoded temporary directory detected

`baselines/logger.py`: Potentially insecure use of a temporary file/directory

```
    debug("shouldn't appear")
    set_level(DEBUG)
    debug("should appear")
    dir = "/tmp/testlogging"
    if os.path.exists(dir):
        shutil.rmtree(dir)
    configure(dir=dir)
```

Description:

It is risky to use a hardcoded interim directory. The application can be manipulated to conduct file operations on the incorrect file or to use a malicious file instead of the anticipated temporary file. Use tempfile instead.

Malicious individuals can guess the file name and write to the temporary file's directory. They basically hijack the temporary file by generating a symlink with the file's name before the application creates the file itself. This enables a malicious user to submit harmful data or force the software to do activities that affect the attacker's chosen files.

To safely generate temporary files, use the `tempfile.TemporaryFile` function. Aside from creating temporary files safely, it generates random filenames that cannot be anticipated and automatically cleans up the file.

Examples:

Poor practice

```
with open('/tmp/abc', 'w') as f: # Insecure, Hard coded temporary directory used
    f.write('stuff')
```

Recommended

```
import tempfile

# Secure, temporary file is created using tempfile.TemporaryFile
# File will be deleted on close
with tempfile.TemporaryFile() as tmp:
    tmp.write('stuff')
```

References:

- OWASP Top 10 2021 Category A04 - Insecure Design
- CWE 377 - Insecure Temporary File
- Python tempfile