

## User Manual for EtherAttack

- 1) Visit <https://github.com/Xenomii/EtherAttack> to view EtherAttack Project
- 2) Clone the EtherAttack project on GitHub through this link below:

<https://github.com/Xenomii/EtherAttack>

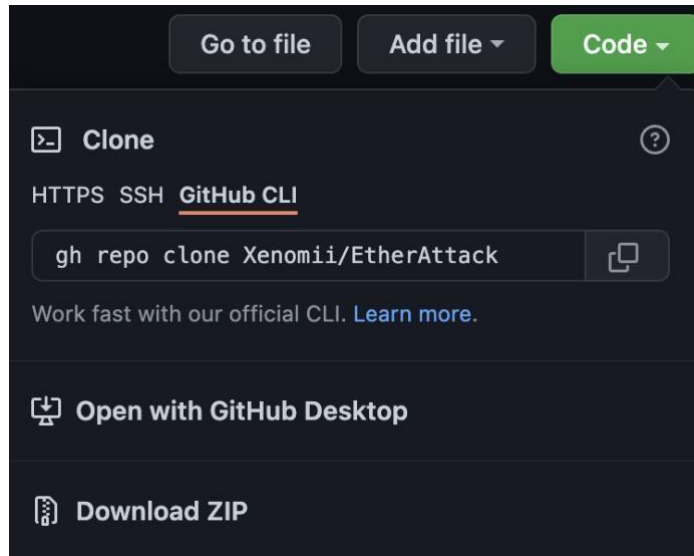


Figure 1 Clone from GitHub

- 3) Open terminal and type:

\$ git clone <https://github.com/Xenomii/EtherAttack>

```
Last login: Sun Apr  3 15:21:23 on ttys003
brilliantstar@Stellars-MacBook-Pro ~ % git clone https://github.com/Xenomii/EtherAttack
```

Figure 2 Git clone on Terminal

- 4) Press Enter to clone the project.

```
etherattack
Cloning into 'EtherAttack'...
remote: Enumerating objects: 479, done.
remote: Counting objects: 100% (479/479), done.
remote: Compressing objects: 100% (334/334), done.
remote: Total 479 (delta 268), reused 325 (delta 136), pack-reused 0
Receiving objects: 100% (479/479), 29.89 MiB | 7.05 MiB/s, done.
Resolving deltas: 100% (268/268), done.
```

*Figure 3 EtherAttack successfully cloned*

- 5) Enter the path of the where EtherAttack is located

Eg: cd Path\_Of\_EtherAttack

```
/Users/brilliantstar
brilliantstar@Stellars-MacBook-Pro ~ % cd /Users/brilliantstar/GitHub/EtherAttack
```

*Figure 4 locate EtherAttack*

- 6) Navigate to webapp

Eg: cd webapp

```
brilliantstar@Stellars-MacBook-Pro EtherAttack % cd webapp
```

*Figure 5 cd webapp*

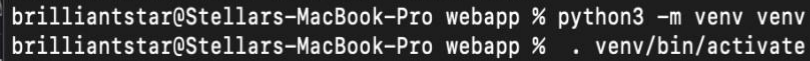
- 7) Install python virtual environment

sudo apt install python3-pip python3-venv

8) Create a python virtual environment by inputting the command:

```
python3 -m venv venv
```

```
. venv/bin/activate
```



```
brilliantstar@Stellars-MacBook-Pro webapp % python3 -m venv venv
brilliantstar@Stellars-MacBook-Pro webapp % . venv/bin/activate
```

*Figure 6 create virtual environment*

9) (.venv) will appear at the start of the directory terminal output.



```
[(venv) brilliantstar@Stellars-MacBook-Pro webapp %
```

*Figure 7 (.venv) appearing at the start of directory terminal*

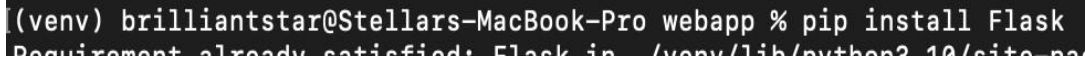
10) Install dependencies that is required for this project using pip:

```
pip install -r requirements.txt
```

```
pip install Flask
```



```
(venv) brilliantstar@Stellars-MacBook-Pro webapp % pip install -r requirements.txt
```



```
(venv) brilliantstar@Stellars-MacBook-Pro webapp % pip install Flask
Requirement already satisfied: Flask in /venv/lib/python2.10/site-packages
```

*Figure 8 pip install*

11) Install solidity compiler:

```
solc-select install 0.6.10
```

```
chmod +x SlitherScanner.sh
```



```
(venv) brilliantstar@Stellars-MacBook-Pro webapp %
(venv) brilliantstar@Stellars-MacBook-Pro webapp % solc-select install 0.6.10
```

*Figure 9 install solidity compiler*

12) Verify that the compiler is installed

```
solc-select versions
```

```
version '0.6.10' installed.  
[(venv) brilliantstar@Stellars-MacBook-Pro webapp % solc-select versions  
0.6.10 (current, set by /Users/brilliantstar/.solc-select/global-version)
```

*Figure 10 verify that compiler is well installed*

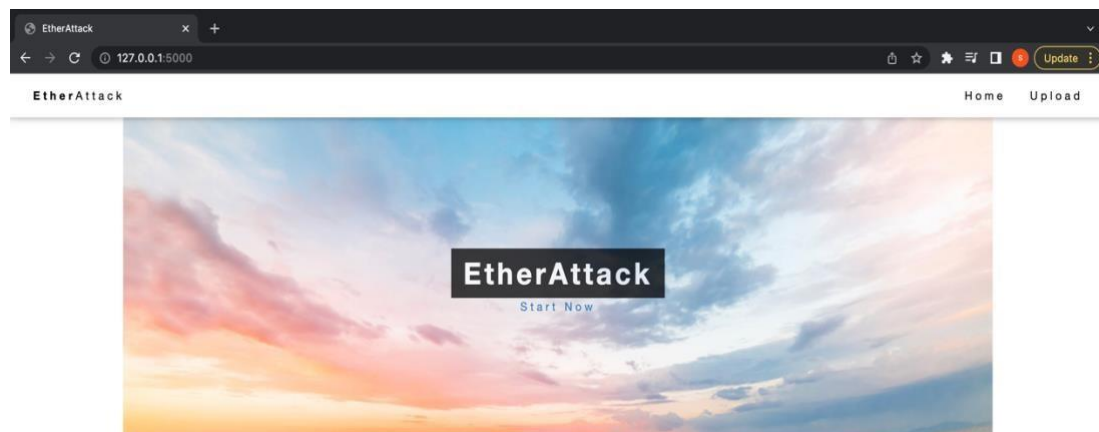
13) Run the application:

```
python3 app.py
```

```
[(venv) brilliantstar@Stellars-MacBook-Pro webapp % python app.py  
* Serving Flask app 'app' (lazy loading)  
* Environment: production  
  WARNING: This is a development server. Do not use it in a production deployment.  
  Use a production WSGI server instead.  
* Debug mode: on  
* Running on http://127.0.0.1:5000 (Press CTRL+C to quit)  
* Restarting with stat  
* Debugger is active!  
* Debugger PIN: 295-929-250
```

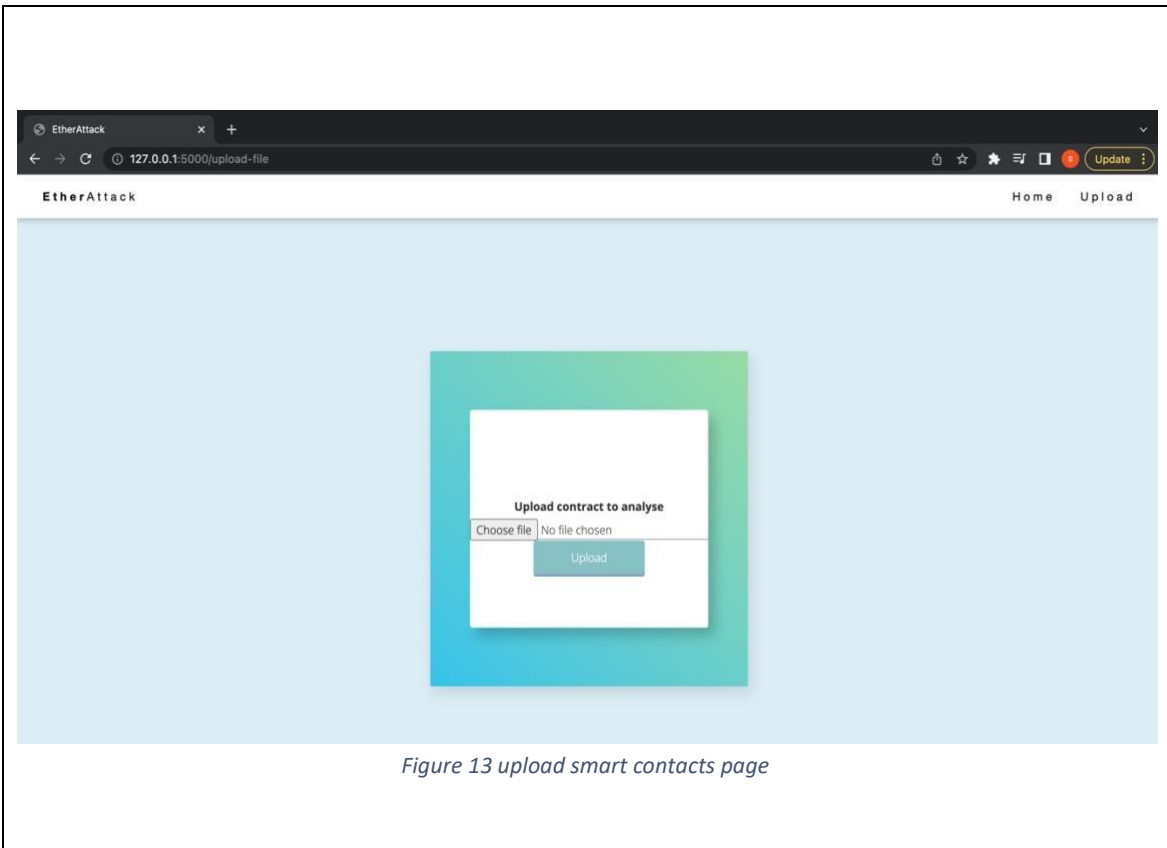
*Figure 11 run the application*

14) Visit <http://127.0.0.1:5000> . EtherAttack webpage would be shown

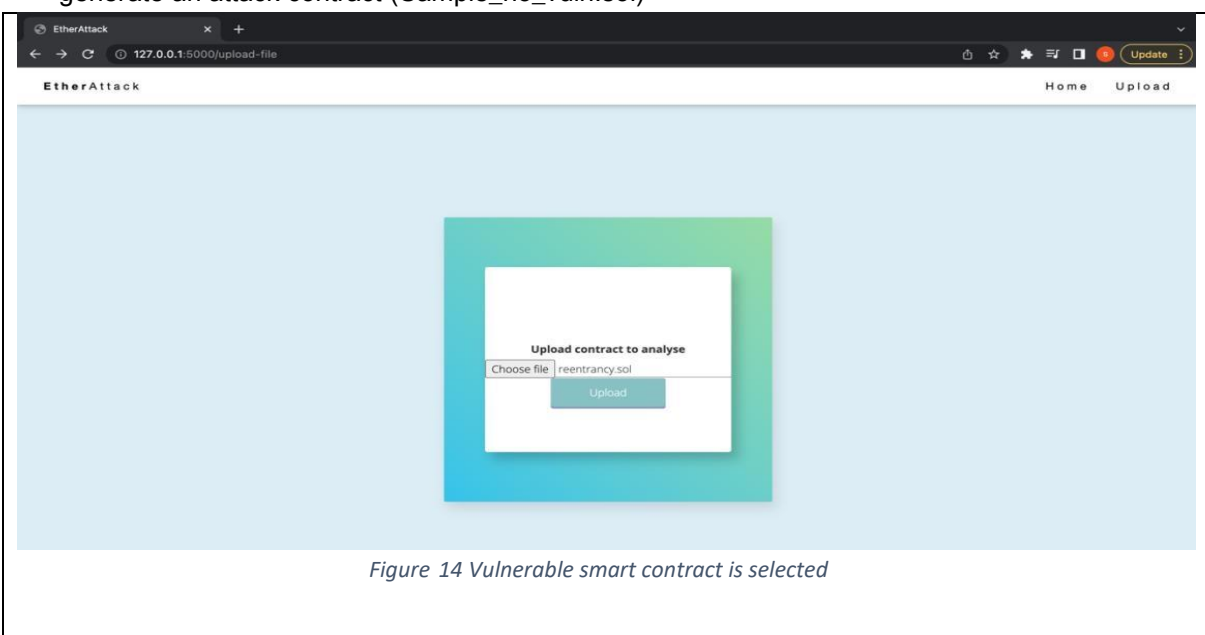


*Figure 12 EtherAttack main page*

- 15) Click on upload on the main page and you will be directed to the upload page to upload smart contracts.



- 16) Choose a smart contract to upload. For this example, a vulnerable smart contract has been chosen. Use any smart contracts in the Sample\_Contracts folder to upload. This folder contains vulnerable contracts (Phising\_Tx\_origin.sol & reentrancy.sol) and contracts that will not generate an attack contract (Sample\_no\_vuln.sol)



- 17) After clicking the upload button, it will redirect you to the analysis page. The analysis page will show detailed information about the smart contract and the attack contract. This includes Attack contract, Slither static analysis, variable data dependencies, summary of functions and contract summary.

The screenshot shows the EtherAttack web application interface. The browser address bar indicates the URL `127.0.0.1:5000/content.html/reentrancy.sol`. The application has a navigation bar with 'Home' and 'Upload' links. The main content area is divided into four panels:

- Original Contract**: Contains Solidity code for an `EtherStore` contract with a `balances` mapping and a `deposit()` function.
- Attack Contract**: Contains Solidity code for an `Attack` contract that inherits from `EtherStore` and implements a `withdraw()` function to exploit a reentrancy vulnerability.
- Slither Static Analysis**: Displays the results of a Slither static analysis, identifying a reentrancy issue in the `withdraw` function of the `EtherStore` contract.
- Variable Data Dependencies**: Shows a dependency graph for the `deposit()` function, illustrating the relationships between variables like `balances`, `_amount`, and `msg.value`.

Each panel includes a 'Download' button for saving the respective content.

Figure 15 example of the analysis page

18) Click the download button on the top of each section to download.

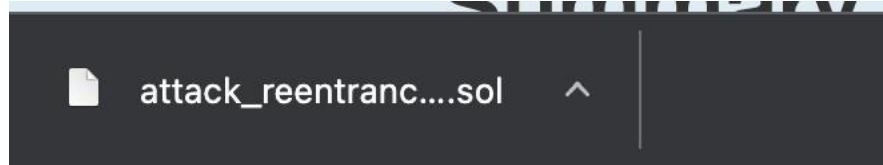


Figure 16 The downloaded file

19) If there is no vulnerable in the smart contract, a “no vulnerabilities” notice will be shown under the Attack Contract section in the analysis page.

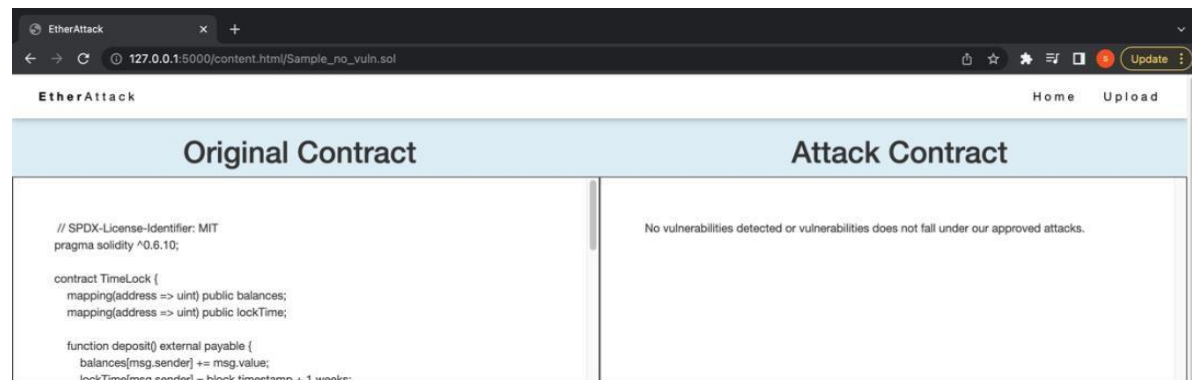


Figure 17 No vulnerable detected notice