Requirements, installation and preliminary operations

In this tutorial, CONFETTY was set up on **Windows 11** using **Ubuntu in WSL2**.

1. Anonymous GitHub Repository

1. Download the repository from Anonymous GitHub:

   ```
   https://anonymous.4open.science/r/CONFETTY
   ```

2. Extract the ZIP into a folder.

---

2. MongoDB

1. Download MongoDB Community Edition:

   ```
   https://www.mongodb.com/try/download/community
   ```

   > ***Note:*** *During the installation, make sure to also install* ***MongoDB Compass****.*

2. Install MongoDB.
3. Open **MongoDB Compass** and create a new connection named `ChorChain`.

---

3. Apache Tomcat (Version 9)

1. Download Apache Tomcat 9:

   ```
   https://tomcat.apache.org/download-90.cgi
   ```

2. Extract the ZIP into a folder.

> ***Note:*** *This is required to connect your IDE with Tomcat and run the project on the Tomcat web server.*

---

4. IntelliJ IDEA

> ***Note:*** *In this setup, IntelliJ IDEA was used to start the web server, but you are welcome to use any IDE of your choice.*

1. Download IntelliJ IDEA:

   ```
   https://www.jetbrains.com/idea/download
   ```

2. Install IntelliJ IDEA.
3. In IntelliJ IDEA, open the `Process Manager` folder from the extracted GitHub repository.

4. Configure Tomcat in IntelliJ:
    1. Go to `File > Settings`.
    2. Navigate to `Build, Execution, Deployment > Application Servers`.
    3. Click the `+` icon and select **Tomcat Server**.
    4. Browse to your extracted `apache-tomcat` folder and select it.
    5. Click `Apply/OK` to save the configuration.
5. Create a **Run/Debug Configuration** for Tomcat:
    1. Open the dropdown in the top-right corner or go to `Run > Edit Configurations`.
    2. Click the `+` icon and choose `Tomcat Server > Local`.
    3. In the **Server** tab:
        - Change the `Open browser` URL to:

          ```
          http://localhost:8081/ChorChain/
          ```

        - Set the **HTTP port** to:

          ```
          8081
          ```

    4. In the **Deployment** tab:
        - Click the `+` button, choose `Artifact...` and select:

          ```
          ChorChain:war exploded
          ```

        - Set the **Application context** to:

          ```
          /ChorChain
          ```

    5. Click `Apply` and `OK`.
6. Run the server from the top-right corner of IntelliJ. A browser tab should automatically open.

---

5. MetaMask

1. Install the MetaMask browser extension:

   ```
   https://chromewebstore.google.com/detail/metamask/nkbihfbeogaeaoehlefnkodbefgpgkn
   ```

2. Create a new wallet.
3. In the web server tab (`http://localhost:8081/ChorChain/#!#signup`), register and log in using your first MetaMask address.

---

6. Ganache GUI

1. Download Ganache GUI:

   ```
   https://archive.trufflesuite.com/ganache/
   ```

2. Install Ganache GUI.
3. Create a **New Workspace** with the following options:
    - **Hostname (SERVER):**

      ```
      0.0.0.0
      ```

    - **Total accounts to generate (ACCOUNT & KEYS):**

      ```
      20
      ```

    - **Seed (ACCOUNTS & KEYS):**

      ```
      control pulse code indoor off imitate uncover lesson fragile isolate
      fault blast
      ```

    - **Gas limit (CHAIN):**

      ```
      35700000
      ```

    - **Output logs to file (ADVANCED):**

      Select the `Ganache_Temp_Logs` folder inside the `\CONFETTY\Evaluation Tool` directory.

4. Click `Start` in the top-right corner of Ganache.

> ***Info**: The first generated address is the default `ChorChain` and serves as the **Attribute Certifier**. The second, third, fourth, and fifth addresses are used as **Authorities**.*

---

7. Python

1. Download Python:

   ```
   https://www.python.org/downloads/
   ```

2. During the installation, make sure to check the box:

   ```
   Add python.exe to PATH
   ```

---

8. Node.js

1. Download the `.msi` Node.js installer :

   ```
   https://nodejs.org/en/download
   ```

2. Install Node.js.

---

9. Install Ubuntu in WSL2 (Windows 11)

1. Open PowerShell as **Administrator** and run:

   ```
   wsl --install
   ```

2. Start Ubuntu:

   ```
   wsl.exe -d Ubuntu
   ```

3. Choose a username and password.

4. In Ubuntu, run the following commands:

   1. Create a new folder named `Docker` and change directory to it:

      ```
      mkdir Docker
      cd Docker
      ```

   2. Download and install Docker, then pull the MARTSIA image:

      ```
      curl -fsSL https://get.docker.com -o get-docker.sh && \
      sudo sh get-docker.sh && \
      sudo chmod 666 /var/run/docker.sock && \
      docker pull apwbs/martsia:martsia_ethereum
      ```

   3. Deploy the smart contracts (Replace `$path` with the absolute path to your CONFETTY folder):

      ```
      docker run -it --name martsia_ethereum_container --network host \
      -v $path/Confidentiality\ Manager:/MARTSIA-KoB-API \
      006be17f8d0c \
      bash -c "cd /MARTSIA-KoB-API/sh_files && \
      sh deployment.sh; exec bash"
      ```

      You're all set! Now you can start testing CONFETTY.

Summary

Once you complete all the steps above, the CONFETTY environment should be set up and ready to use.

- MongoDB Compass is connected and configured.
- Apache Tomcat is running the web server at `http://localhost:8081/ChorChain/`.
- Ganache is simulating the blockchain with predefined accounts and keys.
- MetaMask is ready for interaction with the system.
- CONFETTY smart contracts are deployed in Docker via WSL2.

Test CONFETTY locally

Below, we provide a step-by-step guide to test CONFETTY locally. All tests were executed on **Windows 11** using **Ubuntu** in WSL2.

Please ensure that you have completed the steps in the **Requirements, Installation, and Preliminary Operations section** before proceeding with the tests.

Overview

The CONFETTY Evaluation Tool runs a series of tests that analyze gas costs and execution times for various process configurations. The tests include:

- **X-ray tests** for the number of writing participants, message size, and process size.
- **Synthesis tests** including parallel split, parallel split & join, exclusive split, and exclusive split & join.
- A test with three state-of-the-art processes.

Start IPFS

Open **a Ubuntu WSL2 console** and run the following command.

> ***Important:*** *Update the* `$path` *variable with the absolute path to your CONFETTY folder. Leave this console open and idle; do not close it.*

```
sudo chmod 666 /var/run/docker.sock && \
docker ps -aq | xargs docker rm -f && \
docker run -it --name martsia_ethereum_container --network host -v \
$path/Confidentiality\ Manager:/MARTSIA-KoB-API 006be17f8d0c \
bash -c "python3 -m pip install flask && \
python3 -m pip install Flask_cors && \
cd /MARTSIA-KoB-API/sh_files && \
sh db_and_IPFS.sh; exec bash"
```

Python Tests

The Python tests are executed from PowerShell. Inside the `\CONFETTY\Evaluation Tool` folder, the `test_Auto.py` script will:

- Generate the required JSON test inputs.
- Start the confidentiality API.
- Execute each test case 5 times.
- Create two Excel files in the `table_Output` folder:
  - `costs.xlsx` for gas cost measurements.
  - `timings.xlsx` for execution time measurements.

Test Cases

1. **Number of Writing Participants X-ray Test (2 to 10 participants):**

   ```
   python test_Auto.py -t1
   ```

2. **Message Size Dimension X-ray Test (x1 to x9):**

```
python test_Auto.py -t2
```

3. **Process Size Dimension X-ray Test (x1 to x10):**

```
python test_Auto.py -t3
```

4. **Parallel Split Synth Test (x1 to x10):**

```
python test_Auto.py -t4
```

5. **Parallel Split and Join Synth Test (x1 to x10):**

```
python test_Auto.py -t5
```

6. **Exclusive Split Synth Test (x1 to x10):**

```
python test_Auto.py -t6
```

7. **Exclusive Split and Join Synth Test (x1 to x10):**

```
python test_Auto.py -t7
```

8. **Three State-of-the-Art Processes Test:**

```
python test_Auto.py -t8
```

9. **Default Execution:**

   Without any `-t` input, the script will execute 5 times the default X-ray Test:

```
python test_Auto.py
```

10. **Execute All Tests Together:**

    Run the following command to execute all tests sequentially.
    Note: This process will take approximately **3 hours** (i7-13700h, 32 GB RAM).

```
python test_Auto.py; python test_Auto.py -t1; python test_Auto.py -t2; python test_Auto.py -t3; python test_Auto.py -t4; python test_Auto.py -t5; python test_Auto.py -t6; python test_Auto.py -t7; python test_Auto.py -t8
```

> ***Note:*** *After executing the scripts, PowerShell may crash.*

Troubleshooting

- **Ganache Performance Issues:**
  If you experience performance degradation in Ganache, ensure that logs are being directed to the specified file (`Ganache_Temp_Logs` , **Requirements, Installation, and Preliminary Operations section**).

- **Docker Container Issues:**
  Verify that the volume mapping for the CONFETTY folder is correct in the Docker command.
