Institute of Information Systems Engineering
Distributed Systems Group
VU Internet of Things for Smart Systems (184.781)

Project Topic B – Blockchain-based Sensor Data Integration

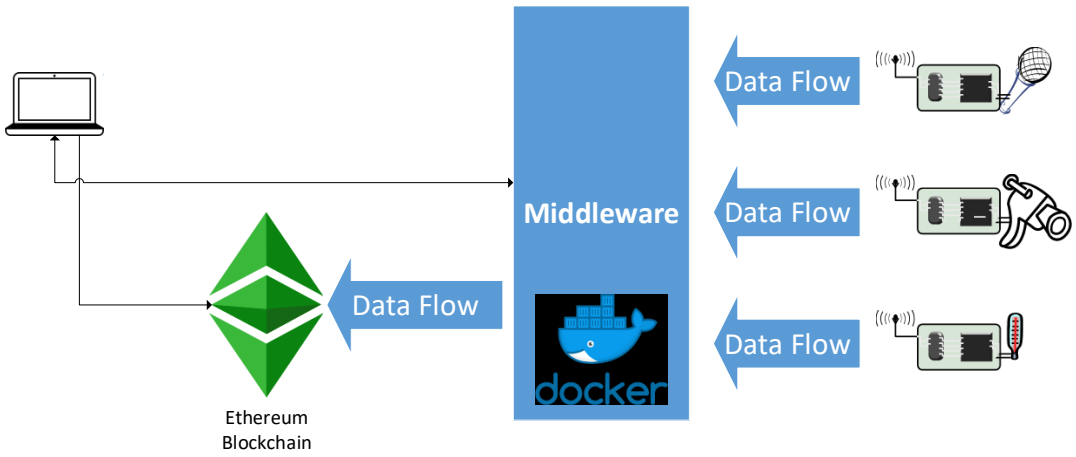

Figure 1: Example Scenario

Introduction

In Internet of Things (IoT) scenarios, e.g., smart factories or smart grids, huge amounts of data may be generated [1]. While not all of this data is interesting, it is important to identify relevant events and make them available for users. Once events have been identified, it is necessary to distribute them to end users using a trusted channel. Especially in distributed scenarios, where different organizations might be involved, it is necessary to send and store this data in an immutable way.

In the world of cryptocurrencies like Bitcoin, a similar problem arises, since transactions need to be stored in a permanent and unchangeable way [5]. For this, blockchains are applied. Within this project, it is the goal to make use of this basic idea and to apply blockchain technology in order to store IoT events [3, 4] and distribute these events to users.

Assume a basic setup as exemplified in Figure 1. As it can be seen, the data flow is from the devices to a Docker-based middleware, which filters the data, and only adds filtered data to the blockchain. The user (indicated by the laptop in the figure) does not access the single sensors. Instead, the user interface is only able to visualize data from the blockchain. In addition, the user can interact with the middleware in order to set parameters for filtering the data before it is forwarded to the blockchain. For instance, if the value from a temperature sensor falls below or exceeds a predefined parameter, this data item will be added to the blockchain, while all other data items are ignored.

As blockchain technology, Ethereum[1] is chosen. Like other second generation blockchains, Ethereum allows to implement *smart contracts*. In the project, a simple smart contract has to be implemented which pushes a notification to the user in case a critical event has been added to the blockchain. However, the notification does not contain the actual data: The user has to get this

[1] `https://www.ethereum.org/`

data from the blockchain.

As dataset, you have to make use of the 2017 release of the *UMass Smart* Dataset*[2] [2]. You may choose any dataset from the *Apartment dataset* or *Home dataset* for your project. The dataset can be used to emulate the generation of data items over time.

Outcomes

The expected outcomes of this project are two-fold: (1) the actual project solution, (2) two presentations of the results.

Project Solution

The project has to be hosted on a Git repository provided by the Distributed Systems Group – you will get instructions how to apply for such a repository at the lab kickoff meeting. Every member of your team is assigned a separate Git account. *We will check who has contributed to the source code*, so please make sure you use your own account when submitting code to the repository. Further, it is required to provide an easy-to-follow README that details how to deploy, start and test the solution. The best practice is to provide a README that describes "Plug-and-Play" instructions on how to use your solution.

For the submission (see below), you also have to create one or more Docker images, which contain your *complete* implemented solution, i.e., including all dependencies. You can find detailed instructions on how to prepare these Docker images on TUWEL. Please make sure that above-mentioned README covers full information on how to use the Docker images. Hint: If you are working on Topic A or Topic D, please make sure that all software running on the Raspberry Pis is also running in Docker containers. This will make it way easier to later on assess and grade your solution.

Presentations

There are two presentations. The first presentation is during the mid-term meetings, and covers at least the tasks of Stage 1 (see below), the second presentation is during the final meetings and contains all your results. Please note: Of course, you may already start to work on Stage 2 before the mid-term presentation. The actual dates for the meetings are announced on TISS and TUWEL.

Every member of your team is required to present either at the first *or* the second meeting. Each presentation needs to consist of a slides part and a demo of your implementation. Think carefully about how you are going to demonstrate your implementation, as this will be part of the grading. You have 10 minutes (strict) of time for your presentation at the mid-term meetings, and 15 minutes (also strict) of time for your presentation at the final meeting. We recommend to use about 5 minutes at each meeting for the slides part of your presentation, and the remaining 5 / 10 minutes, respectively, for the presentation of your implementation results. After each presentation, there will be a Q&A session of max. 5 minutes.

The past has shown that providing a nice use case story usually helps to present the project outcomes. While providing such a use case story is not an absolute must, it will surely help the audience to understand your work better.

Grading

A maximum of 60 points are awarded in total for the project. Of this, up to 30 points are awarded for the implementation (taking into account both quality and creativity of the solution as well as code quality), 20 points are awarded for the presentations (taking into account content,

[2] `http://traces.cs.umass.edu/index.php/Smart/Smart`

quality of slides, presentation skills, and discussion), and 10 points are awarded for individual accomplishments.

For the individual accomplishments, you can actually reach a maximum of 12 points (despite the fact that it officially counts for only 10 points). A maximum of 6 points can be earned by participating in the Q&A sessions, i.e., by asking meaningful questions at the presentation meetings. Another 6 points can be gained by the individual performance as part of the group work. Usually, this follows the overall performance of the group, unless we see that somebody has either performed exceptionally good or has not contributed to the group's accomplishments as required.

A strict policy is applied regarding plagiarism. Plagiarism in the source code will lead to 0 points for the particular student who has implemented this part of the code. If more than one group member plagiarizes, this may lead to further penalties, i.e., 0 points for the implementation for the whole group.

Deadline

The hard deadline for the project is January 22nd, 2021, 23:59. Please submit the presentations as a single ZIP file via TUWEL. The Docker images need to be uploaded to DockerHub and made available to the lecturer team. For this, please write a mail on how to access the containers to iot@dsg.tuwien.ac.at. The deadline for this is also January 22nd, 2021, 23:59. Late submissions will not be accepted.

Test Cloud Infrastructure

This year, the Google Cloud Platform is supporting the IoT lecture with an Education Grant, which you can use for Virtual Machines (VMs) and other cloud resources. The computational resources can be used for free, however, you need to own a Google account to use them. To access the resources, please visit the following URL (please note that this is a masking URL, i.e., you need to click it, simply copy/pasting it will *not* help): http://google.force.com/. The full URL is also provided in TUWel.

Please note that you need to request the resources until 2021-02-01. Afterwards, they will become void and you will not be able to access the budget.

The e-mail address you provide in this form does *not* have to be your Google account, but it *must* be an address within the TU Wien domain, i.e., @student.tuwien.ac.at, @tuwien.at or @tuwien.ac.at. By providing such an address, you can claim a coupon code, which you can then use to redeem a $25.00 voucher for the Google Cloud Platform. Detailed instructions are provided in this URL and subsequent e-mails.

You can redeem one coupon code per e-mail address at a time. The voucher of $25.00 is in most cases sufficient to conduct the implementation tasks of the lab exercises. However, if this is not the case, please send a mail to s.schulte@dsg.tuwien.ac.at. Per mail address, three vouchers can be used (one after another), summing up to $75.00. But unfortunately, Google does not allow to directly cash in more than one voucher per account. Hence, this has to be requested through the lecturer.

Note that exceeding this budget will lead to immediate shutdown of the project by Google, and we have no means of influencing this, or providing extensions. We therefore recommend you to monitor your resource usage, in order to avoid unnecessary spending, for instance, by VMs left running.

You are allowed to use the Google Cloud Resources only for the purpose of this course. Both Google and TU Wien are monitoring the use of the resources, and any misuse (hosting of illegal data, cryptocurrency mining, etc.) will be punished. In case of any questions regarding the organization of the Google Cloud Platform, please send an e-mail to s.schulte@dsg.tuwien.ac.at.

Please take care that you do not publish any credentials or (ssh) keys on the Internet. For your implementation, please use dedicated credential files and add these credential files and the keys to

the .gitignore file or put them on a system path outside of your Git repository. If you accidentally leak any credentials or ssh keys, inform us immediately.

In case of noncompliance, the person(s) responsible will fail the lab!

Stage 1

In the first stage, the focus should be on setting up the infrastructure.

Tasks:

1. Start by designing the architecture of the middleware.
2. Familiarize yourself with Ethereum and how to use it. For Stage 1, it is sufficient to set up a local Ethereum blockchain (e.g., Ganache[3]).
3. Build a *sensor data provider*, which emits data from the UMass Smart* Dataset over time. The sensor data provider emulates the generation of data items. It is advisable to make the submission speed configurable to apply different submission speeds during testing and actual performance tests of the system.
4. Integrate first data items into the blockchain. Prefiltering is not required as this stage.

Stage 2

In Stage 2, the solution from Stage 1 needs to be extended by more sophisticated functionalities. Most importantly, the user interface and data filtering functionalities need to be integrated.

Tasks:

1. Design a user interface for the end user. Through this user interface, data items from the blockchain will be provided to the user. Also, the user gets notifications (in case new data items have been added to the blockchain) through this user interface. Last but not least, the user should be able to set the parameters for data filtering in the user interface.
2. Realize filtering functionalities, i.e., parameters which are used as thresholds for adding data to the blockchain.
3. Implement the smart contract which notifies the user about new data items in the blockchain.
4. Deploy your smart contracts to one of the available (Ethereum-inspired) testnets. Well-known test networks are Ropsten[4], Kovan[5], and Rinkeby[6].
5. At this stage, your solution (middleware and user interface) needs to be deployable in the Google Cloud Platform (while it is sufficient to run it locally during Stage 1).

References

[1] Luigi Atzori, Antonio Iera, and Giacomo Morabito. The Internet of Things: A survey. *Computer Networks*, 54:2787–2805, 2010.

[3] `https://www.trufflesuite.com/ganache`
[4] `https://github.com/ethereum/ropsten`
[5] `https://kovan-testnet.github.io/website/`
[6] `https://www.rinkeby.io/`

[2] Sean Barker, Aditya Mishra, David Irwin, Emmanuel Cecchet, Prashant Shenoy, and Jeannie Albrecht. Smart*: An Open Data Set and Tools for Enabling Research in Sustainable Homes. In *2012 Workshop on Data Mining Applications in Sustainability (ACM SustKDD 2012)*. ACM, 2012.

[3] Konstantinos Christidis and Michael Devetsikiotis. Blockchains and Smart Contracts for the Internet of Things. *IEEE Access*, 4:2292–2303, 2016.

[4] Ana Reyna, Cristian Martin, Jaime Chen, Enrique Soler, and Manuel Diaz. On blockchain and its integration with IoT. Challenges and opportunities. *Future Generation Computer Systems*, 88:173–190, 2018.

[5] Florian Tschorsch and Björn Scheuermann. Bitcoin and Beyond: A Technical Survey on Decentralized Digital Currencies. *IEEE Communications Surveys and Tutorials*, 18(3):2084–2123, 2016.