This project has received funding from the European Union’s Horizon 2020 research and innovation programme under the Grant Agreement No 871481.
Revision history (including peer reviewing & quality control)

<table>
<thead>
<tr>
<th>Version</th>
<th>Issue Date</th>
<th>% Complete</th>
<th>Changes</th>
<th>Contributor(s)</th>
</tr>
</thead>
<tbody>
<tr>
<td>v1.0</td>
<td>1/10/2020</td>
<td>40%</td>
<td>initial draft, D3.2 revision</td>
<td>Rosa Araujo, Evangelos Kotsifakos, Xavier Olivares (LST)</td>
</tr>
<tr>
<td>V2.0</td>
<td>1/11/2020</td>
<td>60%</td>
<td>draft restructuring and update</td>
<td>Rosa Araujo, Evangelos Kotsifakos, Xavier Olivares (LST)</td>
</tr>
<tr>
<td>V3.0</td>
<td>1/12/2020</td>
<td>70%</td>
<td>added information, comments from partners</td>
<td>Rosa Araujo, Evangelos Kotsifakos, Xavier Olivares (LST)</td>
</tr>
<tr>
<td>V4.0</td>
<td>10/12/2020</td>
<td>80%</td>
<td>completed first draft for review</td>
<td>Rosa Araujo, Evangelos Kotsifakos, Xavier Olivares (LST)</td>
</tr>
<tr>
<td>V5.0</td>
<td>14/12/2020</td>
<td>100%</td>
<td>incorporated reviewers’ comments/ ready for submission</td>
<td>Benjamin Heitmann (FhG), Evangelos Kotsifakos, Xavier Olivares Rosa Araujo (LST)</td>
</tr>
</tbody>
</table>

Disclaimer

The content of the publication herein is the sole responsibility of the publishers and it does not necessarily represent the views expressed by the European Commission or its services.

While the information contained in the documents is believed to be accurate, the authors(s) or any other participant in the TRUSTS consortium make no warranty of any kind with regard to this material including, but not limited to the implied warranties of merchantability and fitness for a particular purpose.

Neither the TRUSTS Consortium nor any of its members, their officers, employees or agents shall be responsible or liable in negligence or otherwise however in respect of any inaccuracy or omission herein.

Without derogating from the generality of the foregoing neither the TRUSTS Consortium nor any of its members, their officers, employees or agents shall be liable for any direct or indirect or consequential loss or damage caused by or arising from any information advice or inaccuracy or omission herein.
Copyright message

© TRUSTS, 2020-2022. This deliverable contains original unpublished work except where clearly indicated otherwise. Acknowledgement of previously published material and of the work of others has been made through appropriate citation, quotation or both. Reproduction is authorised provided the source is acknowledged.

Table of Contents

Document Summary Information ............................................................... 1
  Revision history (including peer reviewing & quality control) .............. 2
  Disclaimer .......................................................................................... 2
  Copyright message ............................................................................ 3
  Table of Contents .............................................................................. 3
  List of Figures .................................................................................. 4
  List of Tables .................................................................................... 4
  Glossary of terms and abbreviations used ........................................... 4
1 Executive Summary ............................................................................ 5
2 Introduction ....................................................................................... 5
  2.1 Mapping Projects’ Outputs .......................................................... 6
  2.2 Deliverable Overview and Report Structure .................................. 7
3 Environment ...................................................................................... 7
  3.1 Environment requirements ............................................................ 7
4 Technology selection .......................................................................... 8
5 Security and privacy .......................................................................... 10
6 Resources requirements ..................................................................... 11
7 Environment realization ..................................................................... 12
  7.1.1 Environment access ............................................................... 15
  7.1.2 Deployment ............................................................................. 18
  7.1.3 Infrastructure size limitations .................................................. 21
8 Conclusions and Next Actions ............................................................ 22
List of Figures

Figure 1: Containers and VM 9
Figure 2: The development environment overview 9
Figure 3: Google cloud administration dashboard 13
Figure 4: Google cloud interface details 14
Figure 5: Google cloud service monitoring 15
Figure 6: Google cloud Compute engine – VM instances 16
Figure 7: Google cloud Compute engine – VM instances details 17
Figure 8: Docker set-up 18
Figure 9: Gcloud UI main menu 19
Figure 10: GCloud VM instances list 20

List of Tables

Table 1: Adherence to TRUSTS GA Deliverable & Tasks Descriptions 6

Glossary of terms and abbreviations used

<table>
<thead>
<tr>
<th>Abbreviation / Term</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>API</td>
<td>Application Programming Interface</td>
</tr>
<tr>
<td>GA</td>
<td>Grant Agreement</td>
</tr>
<tr>
<td>GB</td>
<td>Gigabyte</td>
</tr>
<tr>
<td>HTTP</td>
<td>Hypertext Transfer Protocol</td>
</tr>
<tr>
<td>MB</td>
<td>Megabyte</td>
</tr>
<tr>
<td>OS</td>
<td>Operating System</td>
</tr>
<tr>
<td>VCS</td>
<td>Version Control System</td>
</tr>
<tr>
<td>VM</td>
<td>Virtual Machine</td>
</tr>
<tr>
<td>SDK</td>
<td>Software Development Kit</td>
</tr>
<tr>
<td>SSH</td>
<td>Secure Shell</td>
</tr>
</tbody>
</table>
1 Executive Summary

The objective of this deliverable is to provide a description of the infrastructure and guidelines on how to manage the environment to be used in the TRUSTS project. The scope of the TRUSTS infrastructure is to have a development environment ready from the beginning of the project for the technical partners to be able to run their developments. The platform provides a set of features that enable operators to develop applications with a high degree of privacy by design capabilities.

The current design is an update of the initial version drafted in Deliverable D3.1 and submitted in March 2020. The most important differences with respect to the previous deliverable are:

- The inclusion of potential new tools to be used for having a better control of the infrastructure as the project progresses (section 4)
- Detail of the required information for accessing the environment (section 5)
- User management and roles (section 7)
- Infrastructure size limitations (section 7.1.3)

This deliverable is related to the task T3.1 - Infrastructure set-up and technical operations [M1-M36]. The task's goal is to provide a stable and secure environment for developing and hosting the TRUSTS's components.

2 Introduction

This document details the implementation plan for resource allocation for TRUSTS infrastructure, which considers CPUs, GPUs, RAM, and storage requirements. To satisfy the requirements we are using Google Cloud which has a compute cluster with 3 compute nodes providing 6 CPUs, 21 GB RAM and 3 storage nodes. There are 3 snapshots scheduled to back up the main storage data node daily.

For this environment, LSTech is employing DevOps and all the state-of-the-art mechanisms to support the development of the components from the partners and meet the needs for the architecture and specifications defined in T2.4 “Architecture design and technical specifications” (D2.4 months 12 and 24).

The administration and security aspects are the main concern of this task. LSTech provides a DevOps environment that will allow the continuous integration, automated unit testing and collaboration, employing tools like Kubernetes\(^1\) and Docker\(^2\) for deployment, Jenkins\(^3\) for continuous integration, and Jupyter\(^4\) and GitHub\(^5\) for collaboration.

The task 3.1 “Infrastructure set-up and technical operations”, makes sure that components of the TRUSTS platform, that implement the APIs specified in T2.4, for the integration of external data-sets and platforms can be deployed, updated and maintained easily in production and test environments.

---

\(^1\) [https://kubernetes.io/](https://kubernetes.io/)

\(^2\) [https://www.docker.com/](https://www.docker.com/)

\(^3\) [https://jenkins.io/](https://jenkins.io/)

\(^4\) [https://jupyter.org/](https://jupyter.org/)

\(^5\) [https://github.com/](https://github.com/)
The aim of this task is to provide a site for collaboration and development of the TRUSTS platform components. The technical specifications will be continuously adapted until the end of the project, according to new requirements coming up from the outcome of task 2.4 and during the integration and deployment phases.

### 2.1 Mapping Projects’ Outputs

Purpose of this section is to map TRUSTS Grant Agreement commitments, both within the formal Deliverable and Task description, against the project’s respective outputs and work performed.

<table>
<thead>
<tr>
<th>Trusts Platform implementation</th>
<th>Respective Document Chapter(s)</th>
<th>Justification</th>
</tr>
</thead>
</table>
| **T3.1 Infrastructure set-up and technical operations** | A stable and secure environment for developing and hosting the project's components will be offered by LSTech. LSTech will employ DevOps and all the state-of-the-art mechanisms to support the architecture and specifications defined in T2.4. Setup, administration, and security will be the main concern for this task. LSTech will provide a dev-ops, continuous integration, automated unit testing and collaboration environment, employing tools like Kubernetes and Docker for deployment, Jenkins for continuous integration, and Jupyter and Github for collaboration. The preferred development methodology is Agile, supported by online collaboration tools for product development. The task makes sure that components of the TRUSTS platform, that implement the APIs specified in T2.4, for the integration of external data-sets and platforms can be deployed, updated and maintained easily in production and test environments | Chapters 3 - 7 | Section 3: Environment  
Section 4: Technology selection  
Section 5: Security and privacy  
Section 6: Resources requirements  
Section 7: Environment realization |

**TRUSTS Infrastructure Deliverable**

**D3.2 TRUSTS Infrastructure**

The deliverable provides an overview on the services of the infrastructure and guidelines on how to use it in the TRUSTS context. This is the second and last related deliverable regarding the development infrastructure.
2.2 Deliverable Overview and Report Structure

The structure for this technical deliverable is the following.

The document starts by a description of the environment requirements in section 3, while in section 4, a description of the selected technology is provided. Security and privacy issues are mentioned in section 5. In section 6 a description of the resources used is explained. Section 7 comprises the details of the environment realization, how to access it, how to deploy applications and also our size limitations. Finally, the last section is devoted to the conclusions.

3 Environment

3.1 Environment requirements

In this section we describe the basic requirements for the development environment of a cloud platform applicable for TRUSTS.

After analysing the Description of Action of the TRUSTS project we conclude that the development platform for TRUSTS should support the following general requirements:

✔ **Availability**
   The platform shall be available 24/7 in order for the partners to be able to develop, deploy and test their applications.

✔ **Continuous integration**
   The TRUSTS project follows a lean methodology, where elements of the architecture and infrastructure will evolve based on the continuous feedback from developers. In order to allow rapid yet safe evolution of systems, an automated build, test and deploy process will be provided. DevOps tools will allow the developers to build and test their applications directly on the cloud.

✔ **Security**
   The environment should be secure with restricted access and no connection to the external world. If it is necessary for applications to connect to external applications, this will be done through secure procedures.

✔ **Extensibility**
   The environment should be easily extensible. New applications should be easily integrated.

✔ **Interoperability**
   The environment should allow connections to external services when needed.

✔ **Administration and manageability**
   The environment should allow easy administration and management.
The infrastructure must be a dynamic, powerful, and robust centralised environment, able to provide TRUSTS members a system to access and share their different components without affecting each other, within their own workspace.

To this end, each technical partner will have at least one user to manage their workspace, with the proper access rights.

4 Technology selection

LSTech is providing a cloud-based environment for the infrastructure set-up and technical operations, using Google Cloud\textsuperscript{6}. Google infrastructure and servers are robust, and tools are provided to ensure data security with backup, monitoring and encryption also available.

The related security and privacy assurances can be overviewed here \url{https://cloud.google.com/security/overview/}.

Google is compliant with the European law and it has high security standards that are adequate for the TRUSTS platform.

For the development of the TRUSTS platform we are using a Dockerized environment. This will allow ease of implementation, extensibility, portability, and security.

A container\textsuperscript{7} is a standard unit of software that packages up code and all its dependencies, so the application runs quickly and reliably from one computing environment to another. A Docker container image is a lightweight, standalone, executable package of software that includes everything needed to run an application: code, runtime, system tools, system libraries and settings.

Container images become containers at runtime and in the case of Docker containers - images become containers when they run on Docker Engine. Available for both Linux and Windows-based applications, containerized software will always run the same, regardless of the infrastructure. Containers isolate software from its environment and ensure that it works uniformly despite differences for instance between development and staging.

Docker containers that run on Docker Engine:

- Standard: Docker created the industry standard for containers, so they could be portable anywhere
- Lightweight: Containers share the machine’s OS system kernel and therefore do not require an OS per application, driving higher server efficiencies and reducing server and licensing costs
- Secure: Applications are safer in containers and Docker provides the strongest default isolation capabilities in the industry.

Containers are an abstraction at the app layer that packages code and dependencies together. Multiple containers can run on the same machine and share the OS kernel with other containers, each running as

\textsuperscript{6} \url{https://cloud.google.com/}
\textsuperscript{7} \url{https://www.docker.com/resources/what-container}
isolated processes in user space. Containers take up less space than Virtual Machines (VM), (container images are typically tens of MBs in size), can handle more applications and require fewer VMs and Operating systems.

Each TRUSTS partner will create their own Docker application that will be hosted in a VM of the infrastructure. The following figure illustrates this setup.
According to the size progression of the project, new tools will be used in order to have better control of the infrastructure. Orchestration tools such as Ansible\(^8\) or Puppet\(^9\) will be set up as the number of Virtual Machines keep growing. Both technologies are open source and have a freemium version. Kubernetes will also be set up for the container management and deployment.

## 5 Security and privacy

The environment is accessible through only one secure point using SSH and HTTPS protocols and restricted to specific people related to the technical work packages and their applications.

So that the administrator of the site can grant access to the shared space, previously the developers have to fill in an excel file with the relevant information to manage their permissions and allocate the needed resources. The required information is the following:

- Partner name
- Module/component
- Contact person
- Email of the contact person
- Users to access The GCloud/ Gmail account
- Gitlab email
- Dependences
- Main technology/framework used
- Docker images of the software
- Docker compose or swarm
- Minimum requirements (CPU/RAM/disk)

On a daily basis the infrastructure is monitored to identify possible inconsistencies, conflicts or security issues based on system, application and access logs.

Applications and programming interfaces (APIs) will be designed, developed, deployed, and tested in accordance with leading industry standards and adhere to applicable legal, statutory, or regulatory compliance obligations.

Backups are taken periodically, once every week in the beginning of the project. As the project advances more frequent backups will be implemented. LSTech’s team will be responsible for restoring the environment in case of a failure within 48 hours.

All the servers will be hosted in EU countries and no data will be transferred to other countries outside the EU.

---

\(^8\) [https://www.ansible.com/](https://www.ansible.com/)

\(^9\) [https://puppet.com/](https://puppet.com/)
In case special agreements are needed between the partners to ensure secure access to the developed software and to disclosure of information or data, these will be drafted and signed according to the counselling of the project’s legal partners and we will implement the appropriate measures.

The technical partners may also provide additional special security and privacy requirements for their components.

6 Resources requirements

Initially, a shared Virtual Machine is to be used for all TRUSTS technical partners. Following the criterion of on demand service, when necessary a multi-VM configuration (Kubernetes) will be deployed.

The resources currently allocated are the following:

Three virtual machines: One generic with more RAM and two are on-demand for a particular partner. All of them serving as development environment:

The main VM has the following specifications:
- Machine type: Memory optimized
- Virtual CPUs: 2
- Ram: 13GB
- Disc size: 80G
- Image OS: Debian 10 - Buster

Services:
- Docker: 19.3.07
- Docker-compose: 1.25.4
- Docker network: Trustsnet

The 2 extra virtual machines have the following specifications:
- Machine type: Standard Medium performance
- Virtual CPUs: 2
- Ram: 4GB
- Disc size: 80G
- Image OS: Debian 10 - Buster

Services:
- Docker: 19.3.07
- Docker-compose: 1.25.4
- Docker network: Trustsnet
The resources are adequate to support the current developments and they will be updated properly according to the needs of the progression of the different components.

7 Environment realization

The infrastructure has been set-up on LSTech Google Cloud, and policies and procedures are being established and maintained in support of data security to include confidentiality, integrity, and availability across multiple system interfaces, jurisdictions, and business functions to prevent improper disclosure, alteration, or destruction.

The actions taken in order to put in place the security protocols and measures for this type of environment are:

- **Unique identification and authentication**: Internal corporate user account credential shall be restricted for ensuring appropriate identity, entitlement, and access management and in accordance with established policies and procedures:
  - One account per user on Google Cloud
  - One account per user on Gitlab.com

- **Version Control Systems**
  
  Git is a distributed Version Control System (VCS) where each local code repository maintains a complete copy of the history of changes to code.
  
  A TRUSTS group on gitlab.com has been registered.

- **Package Repositories**
  
  A different private repository for each component/application will be set up.

  The technical partners will be asked to provide a README.md file for describing application functionality and deployment instructions. It should be noted that each partner will deploy their application.

  If the partner’s code should be kept private, the developers will have available the use the Gitlab\(^\text{10}\) repository for their compiled application, instead of the source code.

  Dashboards and tools to manage the infrastructure are available by Google and in the following screenshots we provide examples of the management user interface. The dashboard is accessible only by LSTech administrators.

---

\(^{10}\) [http://Gitlab.com](http://Gitlab.com)
Figure 3: Google cloud administration dashboard
Figure 4: Google cloud interface details
7.1.1 Environment access

For accessing the infrastructure, one account per user on the Google Cloud has been set-up. For the development there is also one account per user on Gitlab.com. Each partner is responsible for their application’s deployment, on the shared Docker instance. Docker files should be used, and Docker-compose is to be preferably implemented.

Version Control System

LSTech provides for the developing environment:

- One TRUSTS group on gitlab.com.
- One different private repository for each component/application.
- README.md file for describing application functionality and deployment instructions.

If the code should remain private, partners are asked to use the Gitlab repository for their compiled application, instead of the source code.
Initially, a shared VM will be used for all partners. If and when needed a multi-VM configuration (probably with Kubernetes) will be deployed.

![Google cloud Compute engine – VM instances](image)

Figure 6: Google cloud Compute engine – VM instances
Figure 7: Google cloud Compute engine – VM instances details
7.1.2 Deployment

User management and roles:

The Google Cloud IAM is the component that allows users to have certain permissions when accessing the infrastructure. Currently, the roles assigned to all the partners are: Compute OS Login and Service Account User. These are GCloud premade roles that fit correctly to our current use case.

Compute Os Login role allows:

- Get and Log in into all the deployed VM instances
- Get the Resource manager projects
- Check the quotas and service usage benchmarks and costs.

Service Account User role allows:

- Check the IAM roles for the partners
- Get the Resource manager projects

A custom role will be created for the purpose of having a clearer control and management of the user roles, since there is some permission overlap between the current two roles. This way will be easier to modify the accesses in the big picture.

Instructions for TRUSTS technical partners:

- Users login to LSTech Google cloud.
- Pull their code from their Git repository (including the Docker-file and Docker-compose file), build the Docker image and deploy it.
- If a partner doesn’t want to share their code, they can share compiled versions of it.
GITLAB.COM project access and users

- The Group TRUSTS Platform has been created. https://gitlab.com/groups/trusts-platform
- Each project member will be invited, and they will create their code repository for version control

Deployment on Google Cloud

Setting up GCloud:

- Install Google Cloud SDK (Instructions available here: Windows, Linux, Macos).
- Open Google Cloud SDK Shell (or just the terminal on mac/linux) and run gcloud init.
  - Login with your Google credentials
  - If asked, click yes to generate the required ssh keys.
- Partners might need to close and re-open the terminal.

Connecting to GCloud:

From Gcloud UI:

Access the Google cloud portal: https://console.cloud.google.com/

Figure 9: GCloud UI main menu

To SSH into the general usage VM:
There are many ways to log in into the VM. The easiest way to do it is just pressing into the SSH and letting the browser open a local terminal.

It is also possible to generate SSH keys and log in from the Terminal.

**Instructions given to the developers:**

**From Terminal:**

Run the GCloud compute command to connect:

```
GCloud beta compute --project "trustseu" ssh --zone "europe-west1-b" "trusts-development"
```

**Deploying on GCloud:**

The Docker daemon is running and partners have been given access.

To check if it works, just run “Docker run hello-world”.

**Suggestion for deployment:**

The easiest way would be to have all the necessary files in your Git repo (including a Docker file and Docker-compose), and to pull it in the VM when there are changes. Partners can then easily run “Docker-compose restart” to apply the changes.

A Docker network has been created called “trusts net”.

To add partners’ containers to the TRUSTS net network (documentation).

Example Docker-compose.yml using trusts net network:

```yaml
version: '2'

services:

  nginx:
```

Figure 10: Gcloud VM instances list
container_name: nginx
image: nginx:latest
networks:
default:
  external:
    name: trustsnet

Contact and Help

Collaboration environment

A SLACK11 workplace has been selected and is available to all technical members and developers for easy and immediate communication.

https://trusts-dev.slack.com/

Access to the SLACK workplace and its channels is done through a related request to LSTech.

Technical Support

For supporting the project’s development and resolving technical issues about the development infrastructure, communication channels have been opened providing direct technical support. Details can be found in the online documentation:

https://docs.google.com/document/d/1PARgGNsFkml1hMQr6YEjN-WxdqKQfh5MozSkjffj2PM/edit?usp=sharing

7.1.3 Infrastructure size limitations

In terms of the type and amount of resources, we have used a tool provided by Google, Google Cloud Pricing Calculator, which allows us to estimate costs per resource in an approximate way, being able to add multiple variables in order to increase reliability and accuracy.

With the current budget, the maximum amount of resources that can be provided are:

Using the e2-standard-2 type of instance (2 vCPUs, 8GB Memory), we can provide 12 running simultaneously, 24/7. Each machine will have a hard drive and if requested, a static domain.

Depending on the geographical situation in which the machines are deployed, the price varies a lot, we have used Munich as an intermediate point, but there are 6 different cities within the range of possibilities in Western Europe. If the geographical situation is within North America, we would be able to extend the size to 15 VMs at the same cost. This is not an option since we prefer our servers to reside within the EU for privacy reasons.

11 https://slack.com/
8 Conclusions and Next Actions

With the purpose of offering the TRUSTS partners a site to facilitate the development of their applications, a training environment in Google Cloud has been set up. This document provides an update of the technical details for the environment realization for the TRUSTS project. The technologies and the resources available as well as instructions for accessing it and deploying applications are described and they are also available in an online shared document that is being updated on a regular basis.