ONEYE - A Cloud Operating System that implements a Desktop-as-a-Service

Anjan Prasad Hari (1322978), Istvan Mate Horvath (1322910), Ivan Serunkuma (1322525), and Ranjith Kumar Rama (1322172)

Referent: Prof. Dr. Christian Baun
Cloud Computing - WS2021
High Integrity Systems (M.Sc.)
Frankfurt University of Applied Sciences

Abstract. In this document, we present the results of the semester project for the course of Cloud Computing. The main focus of the document is the oneye project, which is an open-source Desktop-as-a-Service solution. We will give an overall idea about Desktop-as-a-Service (DaaS) services, describe the key functionalities of the oneye project and present a simple application which we have developed for oneye.

Keywords: Oneye Project · Desktop-As-A-Service · Cloud Computing.· PHP

1 Desktop-as-a-Service

Desktop-as-a-Service or simply Daas is a cloud computing service in which virtualized desktop and virtualized applications are hosted in the cloud. The back end of DaaS is hosted on servers of a third party and end users can access these virtual desktops using a dedicated, portal-like application or a simple web-browser. The necessary supporting infrastructures, like storage, networking and security data, are also placed in the cloud and hosted by the service provider. DaaS usually implemented based on a subscription-model, where organizations and tenants purchase virtual desktop which they can used for a given time (usually for a month).

There are plenty of advantages when we or our company use such a DaaS. Firstly, it provides great flexibility for the tenants and the users. Users and employees of a company can access their virtual desktop using any device with internet connection, either on a regular PC, tablet, smartphone or even smart-gadgets like smart-TV. The physical location of the user is irrelevant as well. DaaS also provides flexibility for the tenants. Companies can set up work environment for remote workers easily, they can scale up or down the amount of rented virtual desktops which is especially useful for companies working in seasonal cycles.

As every data is stored in the cloud, backup storage can be created easily. In case of some unforeseen event, data can be recovered using the backup storage. It is also possible to remotely access and recover errors in a single desktop in case the user is not familiar with Informatics. These features provides business continuity and secure productivity for the tenants.

The most important advantage for DaaS is security. In traditional environment sensitive data is stored on multiple individual machines residing on the edge. In this case however, the data is insecure as local desktops can be manipulated or physical machines can be stolen or damaged. Using DaaS, critical data would be stored in the cloud and we can only access it with proper authorization and authentication, providing more security for the data. Moreover, security updates and patches for security holes are easier to deploy in the cloud over the virtual desktops than patching machines and applications individually. DaaS is a good solution for organizations working with sensitive data in a distributed environment, for example the German State Ministry of Justice.

DaaS is a form of Virtual Desktop Infrastructure (VDI). In VDI virtualized desktops and applications are hosted not at a third party or service provider but at the own data center of the organization. Having a VDI is cost heavy at the deployment of the infrastructure. Beside the initial cost of the hardware and software components, the organization also require an experience IT team

who can install and maintain the infrastructure. In this case, the organization also have full control over the infrastructure and can manage it freely. In DaaS, all these work and costs are the responsibility of the service provider; but now the tenant has to deal with the subscription fee, which might be more costly in the long run. Moreover, as the computations of virtualized applications run in the cloud, the organization isn't required to have machines with high computing capabilities and strong hardware. Thus, the organization can save costs on the machines of the individual employees. In the end, organizations can choose if they want to spend more initially for a VDI and afterwards use the service basically for free or subscribe to a DaaS service at a service provider.

Nowadays there is a growing demand for DaaS. Companies and organization tend to use DaaS for its security, consistency and availability properties and its easy-to-use nature. The service providers provide a multi-tenant infrastructure, where a single application instance is delivered to all of the users/tenants. The different providers also have their own properties and specialities. The biggest DaaS providers are VMware Horizon Cloud with the best uptime, Amazon WorkSpaces with cost efficient solutions for smaller businesses, Microsoft Azure with a strong backup and failover system, and Citrix Virtual Apps and Desktops for more mobile workforce. [1] [2] [3]

2 Oneye

The oneye project is a takeover of another open-source cloud application, called eyeOS. The eyeOS was developed with the same concept and same tools as the oneye. EyeOS was first released in 2005 with version v0.6.0. Two years later, version 1.0 was released which included new code infrastructure and tool-kits for fast development of desktop applications for the eyeOS. In the next few years more versions of the eyeOS were released and the project won several awards. In 2011 the open-source project launched a commercial solution targeting business users. After the release of the commercial version, the development of the open-source version stopped. In 2014, the project was bought by Telefónica, a Spanish telecommunication company.

The oneye project was started in 2012. The community of eyeOS took over the structure and toolkits released in the eyeOS 1.0, and went on to develop their own cloud solution. The majority of the development was done between 2012 and 2015, but the core developers are active to this day. Oneye is a PHP based application with some usage of JavaScript. The oneye project does not require a portal-like application to access our desktop in the cloud, a web browser on any device can be used to reach the cloud. Although the eyeOS required an SQL database, the oneye project can function without one. [4]

2.1 Installation guide

As the oneye project is a PHP-application we are required to have a PHP 5 running web server. PHP 5 is an outdated version of PHP, therefore it is a bit more complicated to acquire. The easier way is to use an older version of XAMPP, which is a software bundle containing Apache HTTP server (which is also needed for oneye) and PHP. (It also contains MariaDB and Perl, but these applications are not used. The version of XAMPP is the same as the version of the PHP it includes. So, we should install XAMPP v5.x)

In the following, we present how to setup oneye on Linux machine. First, XAMPP must be installed:

Execute below commands as root user in terminal.

```
wget -O xampp-linux-x64-5.6.40-1-installer.run
// https://sourceforge.net/project/xampp/XAMPP%20Linux/5.6.40/
// xampp-linux-x64-5.6.40-1-installer.run
chmod 777 xampp-linux-x64-5.6.40-1-installer.run
./xampp-linux-x64-5.6.40-1-installer.run
```

After downloading and installing XAMPP, we can move on to set up oneye. We modify the ownership of the XAMPP directory, which will store the oneye files.

```
sudo chown -R <myuser>:<mygroup> /opt/lampp/htdocs
```

Download and extract the files of oneye.

Execute below commands as root user in terminal.

```
cd /opt/lampp/htdocs
wget https://github.com/oneye/oneye/releases/download/
// v0.9.6-preview/oneye_0.9.6.preview.zip
unzip oneye_0.9.6.preview.zip -d oneye
cd oneye
sudo chmod 777 ./installer ./package.eyepackage ./index.html
```

Now, oneye is up and running. To get started, the following URL should be visited, where the root password can be set up:

```
http://<server IP>/oneye/installer/
```

After setting up the root user for oneye, we can start using the oneye DaaS. From now on, we can use oneye on the `http://<server IP>/oneye/` url path, where the login screen will be shown.

If someone would like to run oneye on a Windows machine, that is also possible. The installation steps are basically the same as above. On Windows as well, XAMPP is an easy way to get PHP 5. After installing XAMPP, the http server should be started using the `xampp-control.exe` file. It is recommended to start the `.exe` file as Administrator, as the program will open ports to the outside network. It as also recommended that the installation path of XAMPP would contain no space, as it may cause trouble running the servers. After XAMPP had been started, a .zip file of oneye should be downloaded from GitHub and extracted into `<Path to XAMPP>/htdocs/oneye` (or any directory inside htdocs). Finally, we can access oneye from our browser, by visiting

```
http://localhost/oneye/installer/
```

Fig. 1 shows us the installation page of the oneye in web browser. As PHP 5.6 is now considered as an outdated version, it is desirable to be able to run oneye on newer PHP releases. PHP 5 applications generally cannot run in PHP 7.x or in the newest PHP 8.0. However, some member of the oneye community had finished adjusting the oneye application to PHP 8.0 on the very end of 2020. In order to run it, we need to install XAMPP 8.0 instead of XAMPP 5.6, and download the modified oneye from the following repository: `https:// github.com/namnhatpham1995/installer`. Please note, that this repository is not yet merged into the official oneye repository. Figure 2 shows the login page of the oneye in browser.

2.2 Architecture of oneye

Oneye is DaaS solution written in PHP. As a result, the architecture of oneye is based on PHP files and on their interactions. Different PHP files describe different parts of the virtual desktop. Even though there are several PHP files, a user would (mostly) only access one of them, the index.php file. The only responsibility of this file to include the required functions and settings of the system, determine what kind of hardware the sender of an incoming HTTP request uses, and forward the request to other index.php files based on the sender's hardware (PC, android or iOS smartphone). These PHP files analyse the request and sends commands to specific services/applications.

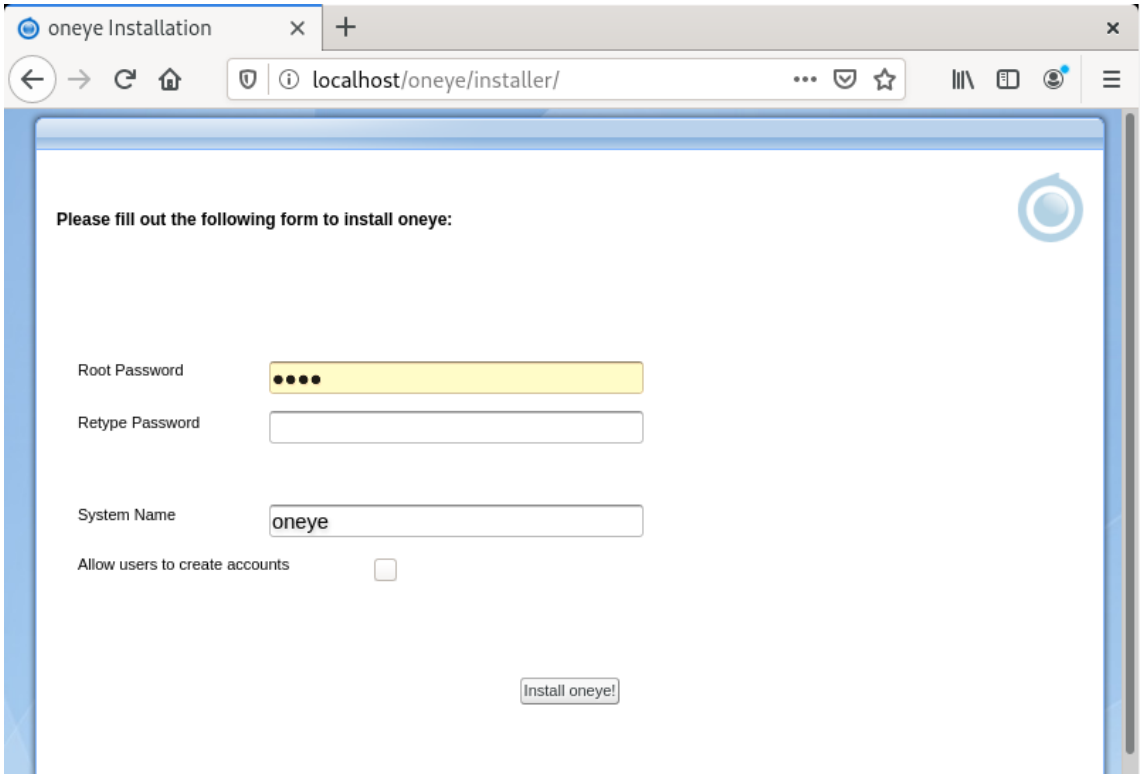

Fig. 1. Installation page of oneye.

The kernel of the OS of oneye is a microkernel. It provides a unified interface for services. Applications and index.php-s access services through the kernel, while the kernel locates and invokes the services. This way, critical/concurrent operations can be synchronized as well.
Services of oneye handles low-level tasks, so that application do not need to implement everything for themselves. These services can also check requests for security or concurrency, for example, an app cannot modify a file without the right permission. The services of oneye are:

- VFS: Virtual File System. This service handles read/write requests on files and directories.
- UM: User Management, register, delete, modify users. Also stores home directories.
- PROC: Process management. Launching, ending, listing processes.
- MMAP: Message Mapping. MMAP analyse and forwards the incoming messages from the users to the correct application (see later).
- eyeX: Creates a response for the user's requests. These responses are in XML format and describe how to change the desktop of the user on the client sides (see later).
- extern: This service allows client to download external files, usually visual files like .jgp and .css.

Applications are the actual softwares that the users use. These are office applications, e-mail clients, file storage apps or games. Applications have three main parts: metadata, user-interface, and event handling. For an application to work these aspects should be properly described (see later). External applications can be included in oneye too, although they are handled differently.
On Fig. 3 the general architecture is shown. There are additional parts of oneye, like logging and external applications, which are not represented.

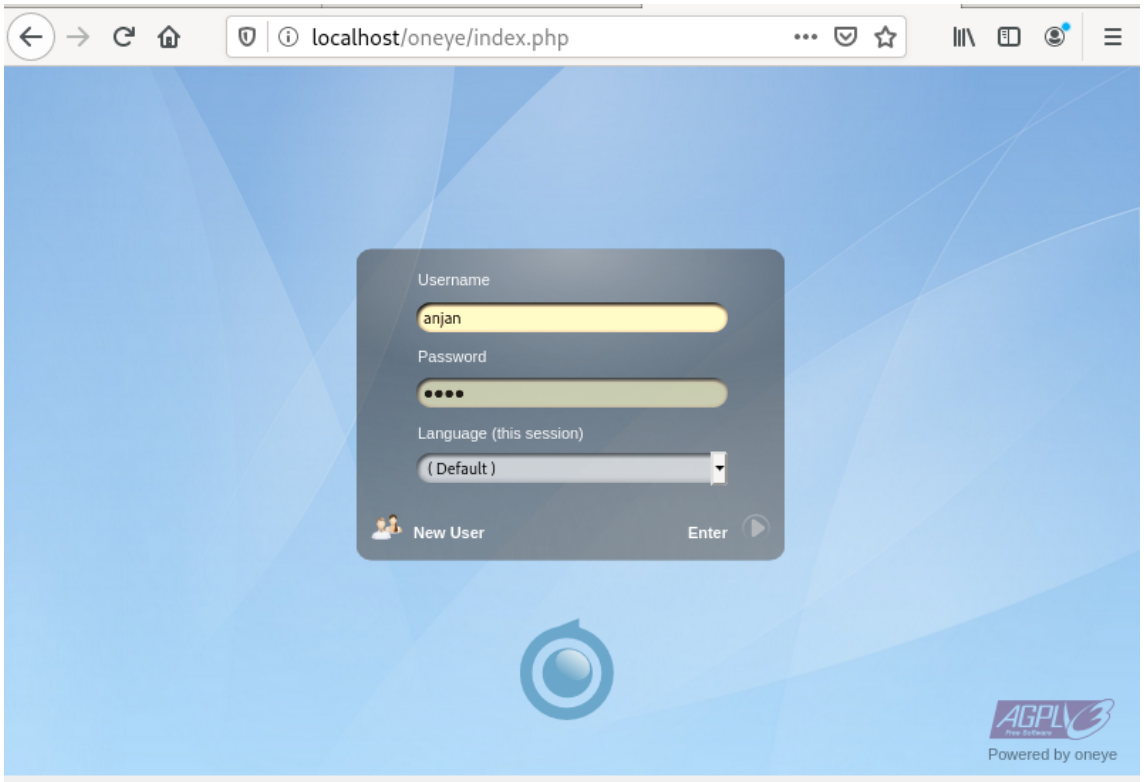

Fig. 2. Login page after installation.

3 Communication

This section provides an explanation of Oneye communication, which includes web server and PHP applications, between server-client and third party internet applications. This is an important aspect of oneye, as this is the only kind of communication between the user and the remote desktop, and all behaviour of the desktop must be encapsulated in HTTP messages.

3.1 Communication with general PHP applications

Generally, PHP applications reside behind a web server, which handles the communication. When a client machine wants to reach one of the PHP files, it will have to connect to the server application first and send a request for a certain resource. The web server analyzes the request, and if it determines that requested file is a PHP file, the web server will forward the PHP file to the PHP interpreter, which also resides in the server machine. (These two server-side software are bundled in XAMPP.) The PHP interpreter parses the file, executes it and returns the output to the web server. The web server now can send a response to the client, to which the PHP file's output is attached.

The clients does not communicate with the PHP application directly, they only requests the server to run the given PHP file. Each time a request was made, the PHP files will be parsed and executed. In a sense, PHP files behave like scripts. These "scripts" are run on request, and if the scripts return any output, it will be forwarded to the client. As a result, PHP files are not needed to be compiled before deploying them on the server, and they can be edited on the fly, without having to restart the server. [5]

3.2 Communication with Oneye

In the communication of Oneye, we were firstly interested in the ongoing communication between a Oneye server and a client. We had Oneye running on a machine with PHP 8 residing on 192.168.0.10

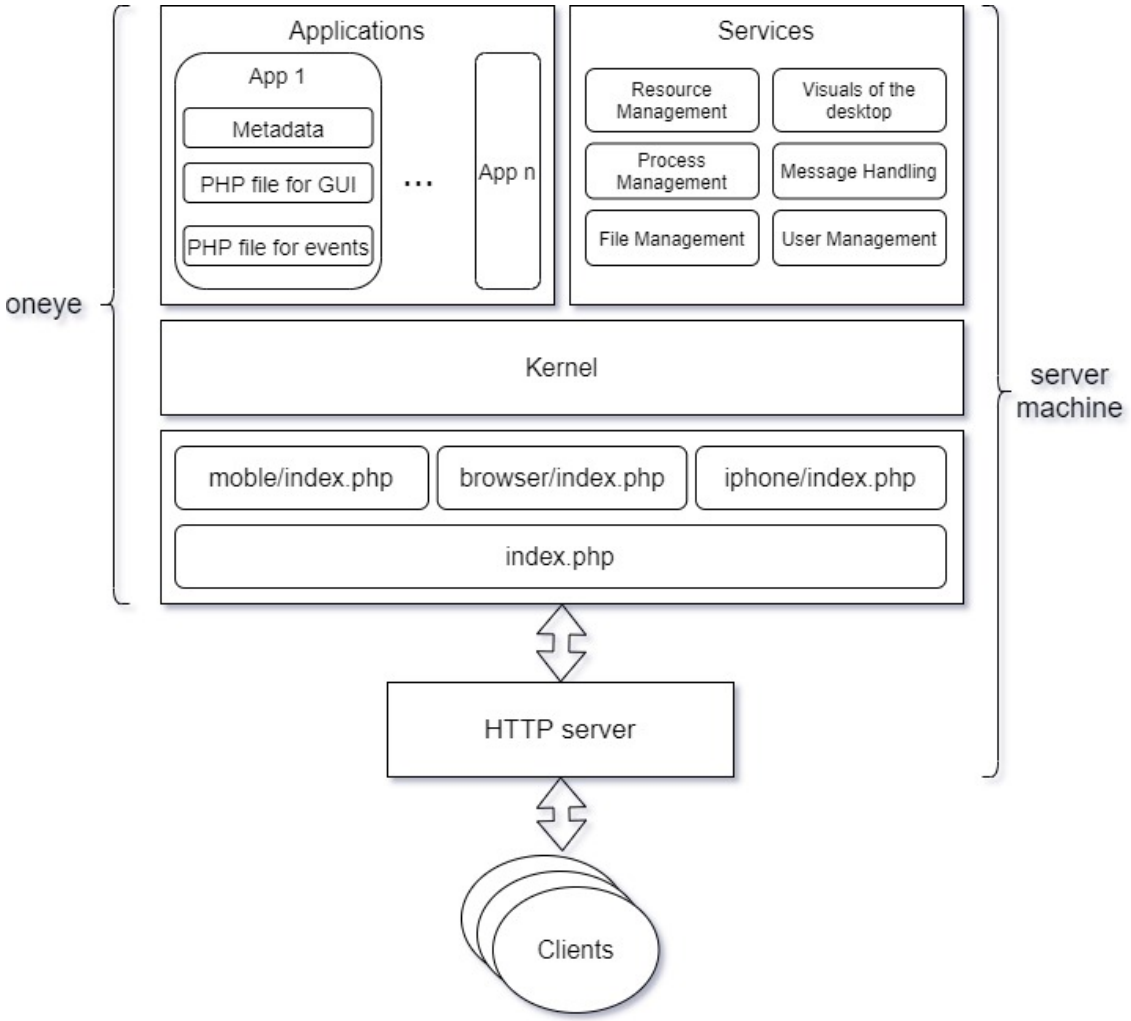

Fig. 3. Architecture of oneye

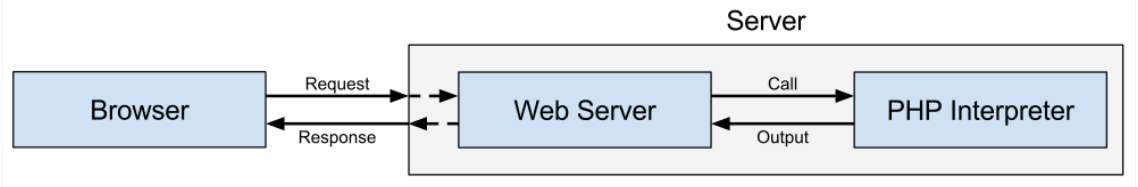

Fig. 4. General architecture of a PHP application. [5]

and we had an other machine on 192.168.0.29 using which we access Oneye. Wireshark, a packet logging program was running on the "server" machine to catch and examine every aspects of communication. Then, on the client machine we visited 192.168.0.10/oneye through the browser. A snippet of Wireshark packets is visible on Fig. 5

Source	Destination	Protocol	Length	Info
192.168.0.29	192.168.0.10	HTTP	490	GET /oneye/ HTTP/1.1
192.168.0.10	192.168.0.29	TCP	54	80 → 1057 [ACK] Seq=1 Ack=437
192.168.0.10	192.168.0.29	TCP	1514	80 → 1057 [ACK] Seq=1 Ack=437
192.168.0.10	192.168.0.29	TCP	1514	80 → 1057 [ACK] Seq=1461 Ack=4
192.168.0.10	192.168.0.29	TCP	1514	80 → 1057 [ACK] Seq=2921 Ack=4
192.168.0.10	192.168.0.29	HTTP	1228	HTTP/1.1 200 OK (text/html)

Fig. 5. Some caught packets inside Wireshark.

The first thing we noticed is that the communication does not use any encryption. That's because we did not set up any certificates or encryption keys on the Apache web-server. Naturally, in a commercial environment TLS is highly recommended, but in this case not having TLS actually helps us observing the sent packages. The web server returns the html output of the index.php based on the application sending the request (browser, mobile, or iphone?). Index.php describes what other JavaScript, css and png files are needed for the webpage. The client browser request them all as well, and present the login screen to the user.

Next, the user logs into their account and the client machine sends a HTTP POST with the following content:

```
params=
<eyeLogin_Textbox_1_User>root</eyeLogin_Textbox_1_User>
<eyeLogin_Textbox_1_Password>cm9vdA==</eyeLogin_Textbox_1_Password>
<eyeLogin_Select_1_Language>[auto]</eyeLogin_Select_1_Language>
```

We logged in using "root" username and "root" password. The password was sent after encoding it with base64. Unfortunately, this base64 string can be easily converted back to the original text. This presents a security hole by itself. Attackers can eavesdrop this message and get access to our account. A solution would be to use a hash function with the password as the function's input and only then use the base64 encoding. The oneye server can compare the incoming and the stored hash-values to authenticate the user.

After login, the client requests for the virtual desktop. The response is about 310 KB, and contains mostly raw JavaScripts which adjust the visual parts of the virtual desktop. Then the user can start working. When an application is being opened, the different visual aspects, pictures and icons will be sent to the client via HTTP GET. When something is being created, or modified HTTP POST will be sent to the server.

The responses of the server are not always in html language. They are XML-based, however they contain elements like <eyeMessage >and <action>. These responses are created by the eyeX service. They describe generic and JavaScript commands that the client should parse and execute. This is a clever solution, as the webpage does not have to be constantly rendered all over again from scratch, rather we make small modifications on the already shown desktop.

3.3 Communication with the Internet from oneye

The next thing we examined is the way oneye reaches third-party Internet applications. Oneye already has a built-in browser, which does not work unfortunately. In a GitHub issue, one of the developer explained, that nowadays browser vendors invented policies to tackle security issues in

web applications. These are understandable and valid reasons, but makes the oneye browser ”more and more useless”. An example for such a security measure is the X-Frame-Options, which can deny the browser to render the page in an ¡iframe¿ object. With X-Frame-Options sites can prevent click-jacking attacks, as the page cannot be rendered embedded into another site. Unfortunately, this is exactly what the oneye-browser tries to do. Most sites and web applications are not compatible with oneye.

Even though accessing the Internet with oneye is not an option (mostly), REST API call are still possible. To make an API call with oneye, the HelloWorld application has been modified. Whenever the app's button is being pushed, the script makes a pinging API call to the API of Binance (public cryptocurrency trading site). The response of the call is displayed in the HelloWord app. The code of the modified function:

```
function HelloWorld_on_HelloWorld_Button($params = '') {
// API call
$response =
  file_get_contents('https://api.binance.com/api/v3/time');
// Update the label of the app.
$GLOBALS['HelloWorld_Label']->setText($response );
}
```

This call was observed with Wireshark too. Firstly, the clients sends a request to the server, the oneye server executes the correct PHP file of the app, including making the API call. The communication of the API call between two servers was encoded with TLS. The response of the API call is parsed by the PHP script of oneye. Finally, the oneye server response to the client's call, describing how to update the HelloWorld app (Fig. 6). These communications were expected based on the general operation of PHP application described in Section 3.2.

```
<?xml version="1.0" encoding="UTF-8"?>
<eyeMessage><action>
        <task>rawjs</task>

<js>document.getElementById("10666_HelloWorld_Label").innerHTML=""<
/js>
</action><action>
        <task>rawjs</task>

<js>document.getElementById("10666_HelloWorld_Label").appendChild(d
ocument.createTextNode( "{\"serverTime\":1611477393343}" ));</js>
</action></eyeMessage>
```

Fig. 6. HTTP response after API call.

The important aspect of this communication is that the server made the actual API call. Even though the user was interested in the API call, who resides at 192.168.0.29, the API server saw that the source IP, the ”client”, was 192.168.0.10. This behaviour can give some sort of anonymity for the users of a DaaS application. The target server cannot be sure who the exact caller is, because it only sees the DaaS server's IP. DaaS applications can give privacy for their users, as users cannot be identified or tracked when they access web pages. Moreover, cookies are stored in the DaaS server, giving more anonymity for the users and making it easier to manage cookies.

In the case of Oneye, this privacy aspect of a DaaS is not present at the moment. We cannot access web pages with Oneye and REST calls are stateless by definition (no information is stored about the caller), so Oneye cannot provide this anonymity. Of course, this can be solved with a working browser inside Oneye.

4 Data Storage

Data and information in Oneye are stored in the cloud. A local Web server to handle the data delivery and content display from the local machine to the browser is a must.The local server will most likely be highly optimized for its task but would be capable of running locally installed Web applications. Using sessions, one can easily access data in different volumes at different locations with the use of the browser(Google chrome, Firefox). Oneye has its own virtual file system and does not require a database to work. The cloud backup storage makes recovery of data easy and possible. Through which the Users can easily upload, view, manage files and run applications on the web or their local machine often without knowing the difference. Applications or files are stored (i.e. word processor) from within the "desktop". files can be locally uploaded from the host operating system like Ubuntu or from the website. In addition, accessibility from anywhere with the use of the internet connection makes Oneye very mobile as people can use it everywhere.

5 Users and Session

Oneye allows the creation of multiple users accounts with password protection and these accounts can be logged in multiple times in the same and browser and at the same time log in with different users on the same and different browsers hence avoiding limitations with account access, the user accounts are also scalable since it allows multi-clusters and hybrid-cluster scenarios. Users can access applications and services on Kubernetes infrastructures, including on-premises solutions and cloud providers like AWS, Azure, Google Cloud, or Alibaba.

Groups on Oneye Every user on Oneye is a member of at-least one group i.e, public group. There exists a directory with the group name which contains the shared files between the group members [8].

Although User_manager(UM) virtual file system(VFS) provides few functions for group managing:
1. **um_getCurrentGroups** : returns a list of all existing groups.
2. **vfs_real_getDirContent_group**: returns the shared content of the folders of a group

There is one disadvantage regarding group management in Oneye, there is no level of abstraction for file sharing in Oneye group .

For example, User A can delete the shared files of User B in the group [8]

Working of session on Oneye The eyeSessions library has abstracted method to get the session from the array and if session variable is is not stored then, it creates an array to store session [8]. An example for storing and obtaining session

```
if(eyeSessions('checkVar',array(TABLENAME))==false){
eyeSessions('makeArrayVar',array(TABLENAME));
}
```

The use of eye Session library is helpful for making use of session on Oneye [8]

6 Security Issues and other Challenges

Oneye as a web operating system is not as robust as their desktop counterparts. But some people believe that it can provide just enough functionality to compete with more traditional software suites. If web Os providers can address issues like functionality gap and data security concerns, we might see a dramatic shift in computer network systems. A common concern about web operating systems is that they require users to trust a third party to potentially keep sensitive data secure. For many users, this is a leap of faith. Will the provider be able to fend off hackers? It's in the provider's best interests to employ advanced security measures to keep client data safe. Updates

typically must be performed by an administrator on the server side. In Oneye implementation and deployments, a user's account access to some applications and data can be restricted. Some delivered applications may not contain the full features set as those of their traditional desktop counterparts. Network latency or congestion can intentionally slow Oneye activities. Oneye's capabilities for business use are limited. The limitations are most obvious in the lack of collaborative features like shared calendars, task lists and mailboxes. Also, the lack of productivity and business applications is a challenge.

The functionality of existing OS applications are far too limited and current work-a rounds are unsatisfactory, due to limitations in the integration of applications and great effort of administration. Especially for larger organizations, the current solutions are not sufficient.

6.1 Oneye Improvement Solutions

To make the OS a business-class solution that could easily compete with commercial VDI's like Citrix or the (soon to emerge) Microsoft web Os we propose a project to add certain crucial functionality and management features to the existing one. The following are some of the improvements that can be made on the functionality of the applications.

1. Making office applications with more functionality, business users have a crucial demand for fully-featured office applications for word processing, spreadsheet and presentations. Support for templates and macro development is highly important.
2. Improvements in image editing functionality.
3. Groupware: To make OS suitable for business use, an Outlook-like groupware solution is an absolute must-have. Either improve and extend the existing applications for email, calendar and tasks or build tighter integration with open source groupware projects like FengOffice (OpenGoo) or SimpleGroupware.
4. The administration also like Implementing file/folder permission on a per-user basis, Develop a mechanism to publish/unpublish applications on a per-user / per-group basis, Implement a session timeout and session lock feature.

7 Applications in oneye

The applications are built using PHP language. Some PHP knowledge is required to understand the script. In order to run the application, we need a GNU/Linux environment, and the environment should have an Apache server, PHP version greater than 5, and PHP editor. [8]

7.1 MMAP message of Applications in Oneye

In Oneye applications are resided in server and processing of information takes place, client is the medium to interact with such applications on Oneye. All the communication between client and server (which process applications) happens in the form of message passing. For example, client is responsible for starting an application by interacting the UI in Oneye which sends the server the message by indicating that button has been clicked for particular application [8].
MMAP route every received messages from client to server for the required application to provide information [8] which is shown in Fig.7.
Using Wireshark, some of the commands from the user were caught as well.

```
POST /oneye/index.php?checknum=591345485083&msg=Command
HTTP/1.1
Host: 192.168.0.10
... [HTTP header] ...

params=
  <type>top</type>
  <place>left</place>
  <id>element-1</id>
```

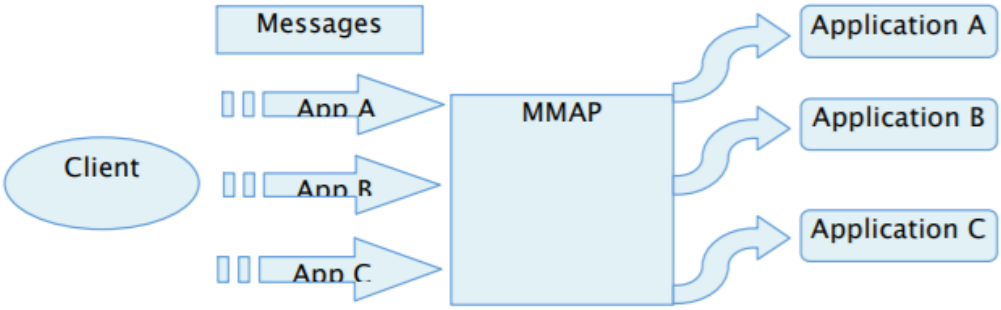

Fig. 7. MMAP message of Applications in Oneye [8]

```
<entry>entry-1</entry>
```

In second case, the response message to client is an XML, which contains the information to modify the interface of the client. For example, if client's click was to create a new window in the browser [8] as shown in Fig.8. This aspect of oneye was already mentioned in the secion about Communication as these XML-based commands were caught in Wireshark as well. A snippet of such command is shown on Fig. 9.

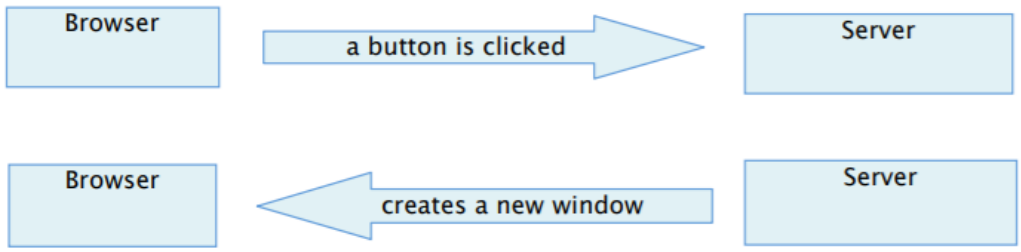

Fig. 8. Response for the client interaction [8]

The eyeX service which contains the method that encodes the XML response from MMAP and server. The eyeX service() operations uses the XML response to creating a window in the browser [8]. For example, to create message box in browser,

```
service('eyeX','messageBox',array('content'=>'Message accepted'));
```

7.2 Structure of an application folder

The directory and files of an application is as follows:
`opt/lampp/htdocs/oneye/system_6f547cf659/apps/application_name`
This directory contains the PHP code of respective applications. Directory contains 3 files:

1. app.eyecode - Initializing and ending code of an application
2. events.eyecode - Event reception code
3. info.xml - the details of the application as name of the application, which category it comes under, what is the version, description,author,license,type of application, and icon image.[8]

In order to run the application properly we need app.eyecode.

```
</action><action>
        <task>rawjs</task>
        <js>addLineToBar("95082_eyeFiles_Toolbar","",0);</js>
</action><action>
        <task>rawjs</task>
        <js>addItemToBar("95082_eyeFiles_Toolbar", "Home" , "i
</action><action>
        <task>createWidget</task>
        <position>
                <x>0</x>
                <y>0</y>
                <horiz>0</horiz>
                <vert>0</vert>
        </position>
        <checknum>205412124243</checknum>
        <name>95082_Home_Container_WidgetDrop</name>
        <father>95082_Home_Container</father>
        <widgetname>WidgetDrop</widgetname>
        <params>{"callback":"","cOrder":0,"signal":"","behavio
        <cent>0</cent>
</action><action>
```

Fig. 9. Some commands of oneye in XML format.

7.3 Initializing and ending of an application

We need app.eyecode to run the application properly, to do initialization and to end an application. The file has two functions:
ApplicationName_run: function called by the procedure when launching an application.
ApplicationName_end: function called by the procedure when terminating an application in case it exists.
A graphical application normally uses it to initialize the User Interface, but non-graphical and small applications concentrate all their code in this function [8].

7.4 Events

We need events.eyecode to make the application intractable. We can describe this as interaction by the user with the interface of our application. When an event takes place the information is shared with the server in the form of the message. The file has various function depending on the applications.

7.5 Application - PHP Version

Fig. 11 shows the application name phpVersion. This applcation shows the current PHP version used in oneye. Application folder path: /opt/lampp/htdocs/oneye/system_6f547cf659/apps/phpVersion

The phpVersion folder has 3 files:

1. app.eyecode - This file has two functions phpVersion_run to launch the application and phpVersion_end to terminate the application.
2. events.eyecode - This file has two fuctions phpVersion_on_Message waits for user to enter the message and responds to it, and phpVersion_on_Close to end the event.
3. info.xml - Fig. 10 shows the contains of the XML file which describes the application.

7.6 Application - Sticky Notes

Fig. 13 shows the application-notepad.This application lets us to enter a note and display it in a display box. Once we have completed the task using the "Reset" we can empty the display box. Application folder path:

```
<package>
        <name>phpVersion</name>
        <category>Utilities</category>
        <version>1.9</version>
        <description>Utility to show how to use the oneye Toolkit</description>
        <author>oneye Team</author>
        <license>AGPL</license>
        <type>Utility</type>
        <icon>index.php?theme=USERTHEME&amp;extern=icons/48x48/rename.png</icon>
</package>
```

Fig. 10. phpVersion application info.xml.

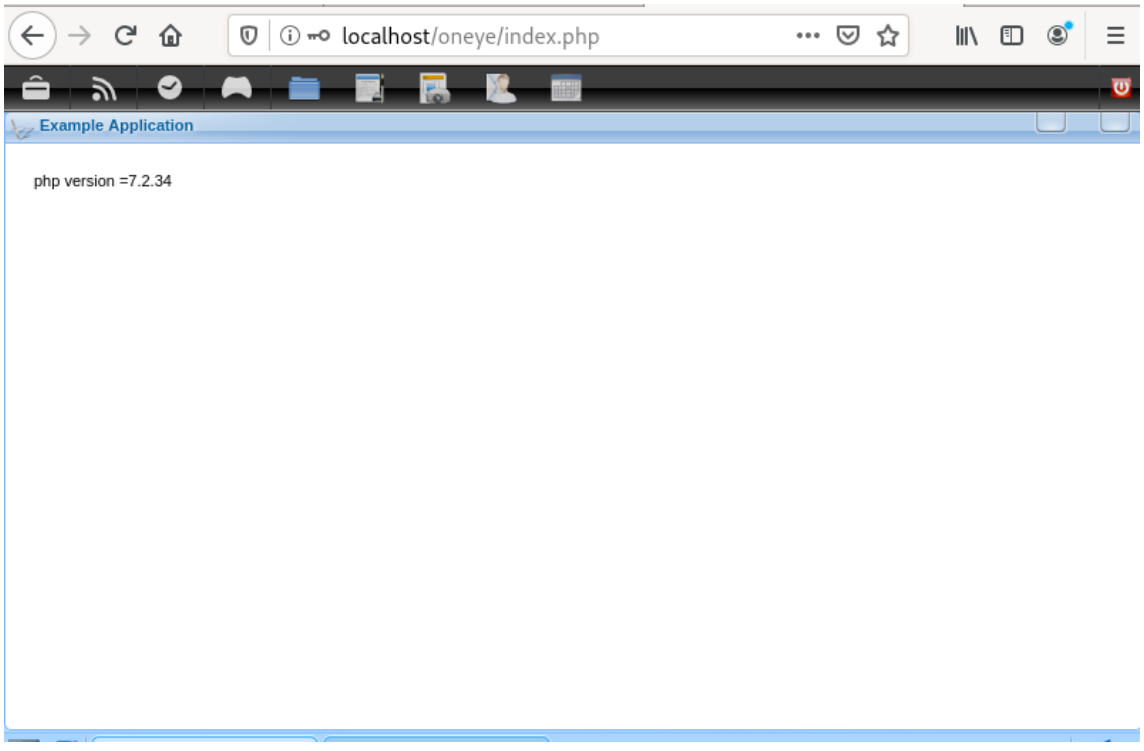

Fig. 11. Application to check the PHP vserion of oneye.

/opt/lampp/htdocs/oneye/system_6f547cf659/apps/sticky

The sticky folder has 3 files:

1. app.eyecode - Has two functions sticky_run to launch the application and sticky_end to terminate the application.
2. events.eyecode - Event file has four fuctions sticky_on_sticky_Button to grab the current text written by user from the text box and displays it on the display, and once the task is completed text box is emptied. sticky_on_delete_Button function will erase the contents in the display, it resets the display box. sticky_on_Message waits for user to enter the message and responds to it, phpVersion_on_Close to end the events.
3. info.xml - Fig. 12 shows the contains of the XML file which describes the application.

```
<package>
        <name>sticky</name>
        <category>Utilities</category>
        <version>1.9</version>
        <description>Utility to show how to use the oneye Toolkit</description>
        <author>oneye Team</author>
        <license>AGPL</license>
        <type>Utility</type>
        <icon>index.php?theme=USERTHEME&amp;extern=icons/48x48/rename.png</icon>
</package>
```

Fig. 12. Sticky Notes application info.xml.

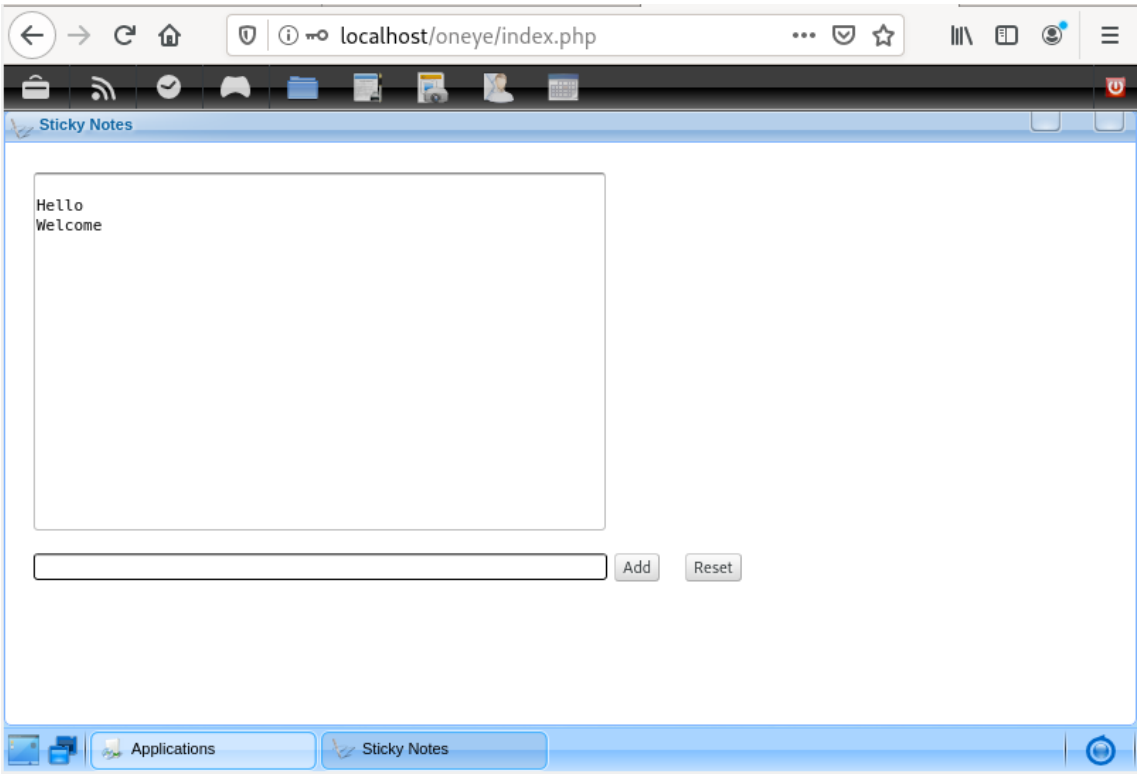

Fig. 13. Application - Sticky Notes.

8 Conclusion

Oneye is an open-source web desktop(DaaS) that is mainly written in PHP, XML and JavaScript. It lets you upload, store and access applications/files over the internet from Windows, Linux or Mac operating systems. These files can later be edited as you want using some of it's features. Oneye is developed for remote users and it's working requires only a web browser. Communications can be done between users through events. Oneye has many benefits to its users for instance being lightweight, cloud storage, platform-independence, security,accessibility, good compatibility and availability. With the increasing use of high-speed internet technologies, cloud computing has become more popular and is still growing.

References

1. What is Desktop as a Service (DaaS)?. (2021). Retrieved 14 January 2021, from https://www.citrix.com/glossary/what-is-desktop-as-a-service-daas.html
2. M. Dhall and Q. Tan, "A Profitable Hybrid Desktop as a Service Solution," 2019 IEEE 4th International Conference on Cloud Computing and Big Data Analysis (ICCCBDA), Chengdu, China, 2019, pp. 55-62, doi: 10.1109/ICCCBDA.2019.8725659.
3. Detwiler, B. (2021). Top desktop as a service (DaaS) providers: Amazon, Citrix, Microsoft, VMware, and more. Retrieved 12 January 2021, from https://www.techrepublic.com/article/top-desktop-as-a-service-daas-providers-amazon-citrix-microsoft-vmware-and-more/
4. Sahu, Shubham & Khare, Dr. (2019). Survey on web based operating system.
5. Alexander, C. (2021). How does a PHP application work? — The Man in the Arena. Retrieved 6 January 2021, from https://carlalexander.ca/php-application/
6. K. Garg, A. Agwarwal, M. Gaikwad, V. Inamadar and A. Rajpurohit, "XML based lucid web operating systems", 2012 IEEE International Conference on Engineering Education: Innovative Practices and Future Trends (AICERA), Kottayam
7. https://github.com/oneye/oneye
8. https://oneye-project.org/wp-content/uploads/2011/07/developer-manual.pdf