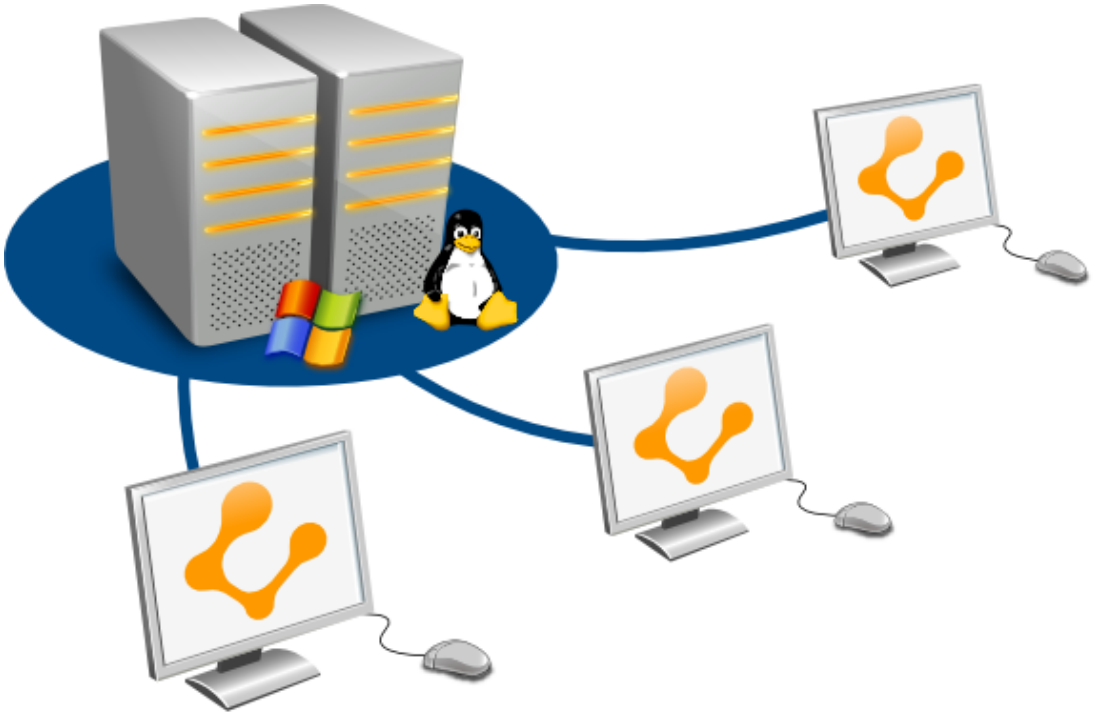
Ulteo Open Virtual Desktop v3.0

Protocol Description



Contents

1 List of Protocols used **3**

1.1 Hyper Text Transfert Protocol (HTTP) . . . 3

1.2 Remote Desktop Protocol (RDP) . . . 3

1.3 Secure Socket Layer (SSL) . . . 3

2 Server Communication **4**

2.1 Security . . . 4

2.2 HTTP return codes . . . 4

3 OVD Client session **5**

3.1 A simple session establishement process . . . 5

3.2 Session Manager client webservices . . . 6

3.2.1 applications.php . . . 6

3.2.2 auth.php . . . 7

3.2.3 icon.php . . . 7

3.2.4 logout.php . . . 7

3.2.5 mimetype-icon.php . . . 8

3.2.6 session_status.php . . . 8

3.2.7 userlist.php . . . 8

3.2.8 start.php . . . 9

4 Web Client external API **10**

The purpose of this documentation is to describe the protocols used by *Ulteo Open Virtual Desktop*.

1 List of Protocols used

1.1 Hyper Text Transfert Protocol (HTTP)

The base communication protocol used in Ulteo OVD to negociate session is HTTP over SSL. This protocol runs over TCP on the port 443.

Ulteo OVD use also HTTP for the communication between servers: Session Manager to Application Server and vice versa. For this usage, the TCP ports 1111 and 1112 (non standart service ports) are used.

Several product of Ulteo OVD like the Administration Console and Web Client are pure web-based and use HTTP because the client is supposed to be a web browser.

HTTP on Wikipedia

1.2 Remote Desktop Protocol (RDP)

RDP is the remote display protocol used by Microsoft Corp.® for their Terminal Services™ software.

RDP is used by Ulteo Open Virtual Desktop to display desktop and applications.

RDP use the TCP port 3389.

RDP on Wikipedia

1.3 Secure Socket Layer (SSL)

SSL is an cryptographic layer protocol that provide encription between server and client.

SSL is used by Ulteo OVD to tunnelize HTTP and/or RDP.

SSL on Wikipedia

2 Server Communication

Servers communicate by using HTTP webservices (hosted by Apache). The Session Manager listen on the TCP port 1111 and Application Servers are using port 1112.

The Session Manager identifies an Application Server using its Fully Qualified Domain Name (FQDN).

Applications Servers only answer to the Session Manager for which the address is stored in a configuration file.

2.1 Security

At the moment, the server are authenticated using the DNS resolution system.

When an Application Server sends a status, it sends an extra argument named *fqdn*. The Session Manager performs 2 authentication tests and 1 authorization.

- **FQDN resolution:** resolves the FQDN to get an IP address and tests if it matches the remote server IP ($SERVER['REMOTE_ADDR'] in PHP). Authentication directly depends on that match
- **reverse resolution:** resolves the server IP address and tests if it matches the FQDN argument. Authentication directly depends on that match. This test can be disabled in the administration console using *Disable FQDN check*
- **authorization:** tests the match of the FQDN with one of the *Authorized FQDN* defined in the administration console.

2.2 HTTP return codes

Webservices are using the standard HTTP return codes to know if the request succeeded.

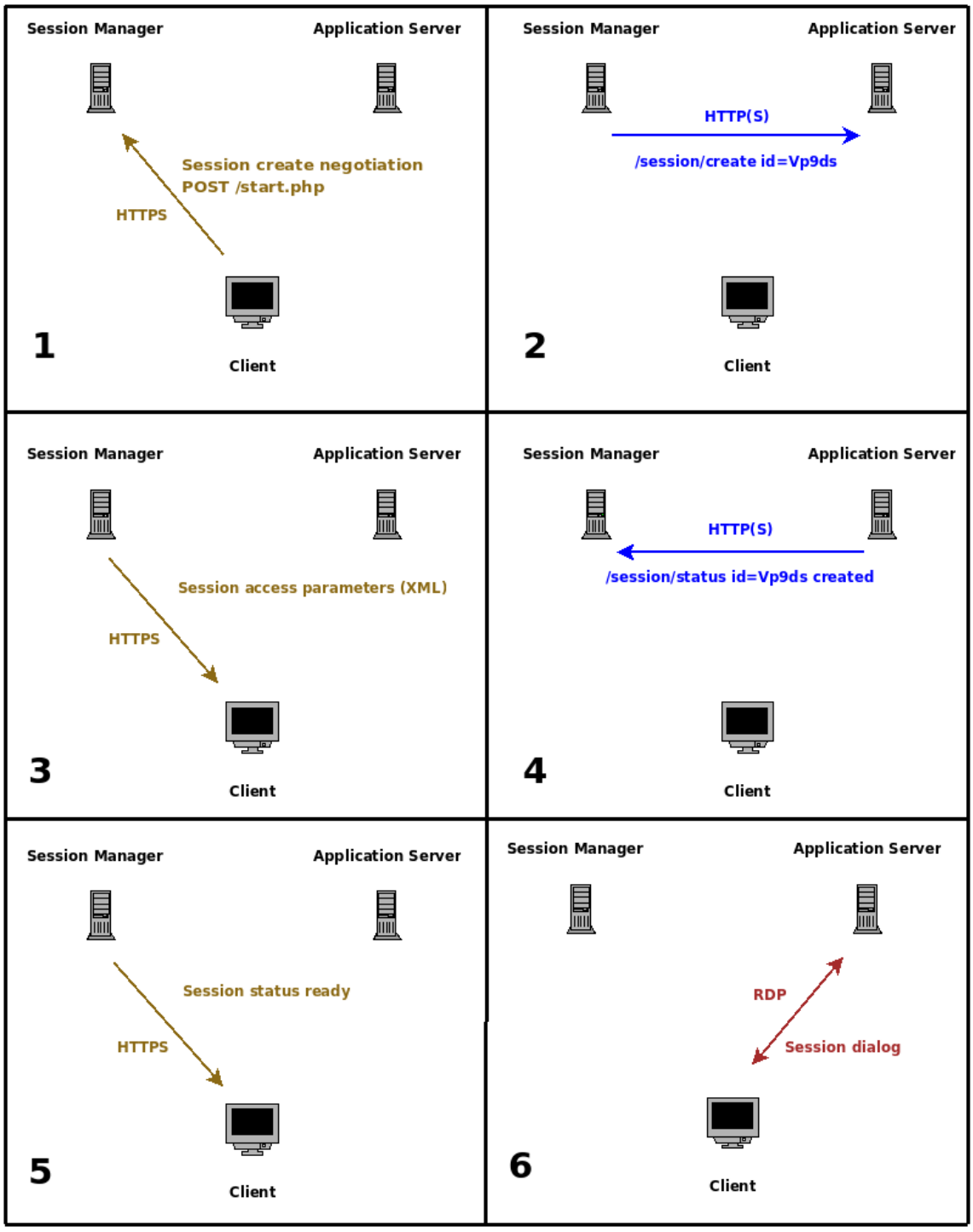
- **200 OK :** request succeeded
- **400 Bad Request :** request argument no valid
- **401 Unauthorized :** From SM to ApS: ApS detects that the remote address is not its SM. From ApS to SM: ApS is not registered yet or has invalid authentication.
- **500 Internal Server Error :** Either a bug was detected on the system or the system is broken.

3 OVD Client session

3.1 A simple session establishement process

The following schema describe the steps of the establishement of a Ulteo OVD session.

In this example, only one Application Server is used, the session do not use File Servers (internal or external) and the client connect directly without an OVD Gateway.



3.2 Session Manager client webservices

Session Manager client webservices are used by client softwares to negotiate and manage their session.

The base URL to access to these services is https://sm.ulteo.com/ovd.

Here is the list of availables webservices.

3.2.1 applications.php

- **method:** GET
- **arguments: none** (except authentication, see auth.php)
- **returns:** an applications list and details as XML content.

This webservice provide the available applications list and details for the current authenticated user. The authentication can be done either by the dedicated service (auth.php) or directly into this one.

XML format:

- **user** Root node
  - *login* Attribute
  - **application nodes**
- **application** Node
  - *id* Attribute
  - *name* Attribute
  - *description* Attribute
  - **mime nodes**
- *mime* Node
  - *type* Attribute

Here is an example:

```
<?xml version="1.0" encoding="utf-8"?>
<user login="mwilson">
    <application id="1850" name="AbiWord" description="Compose, edit ↩
       ,
and view documents">
        <mime type="application/docbook+xml"/>
        <mime type="application/msword"/>
        <mime type="application/rtf"/>
    </application>

    <application id="1857" name="Mousepad" description="Simple text  ↩
       editor">
        <mime type="text/plain"/>
    </application>
</user>
```

3.2.2 auth.php

- **method:** POST
- **arguments: XML input (optionnal)**
- **returns:** HTTP default return codes.

This is the dedicated authentication service. Authentication method depends of the enabled authentication module from the Administration console.

The XML input is only used for password and token authentication mechanisms.

XML format: the root node and the XML tree doesn't matter for this webservice. The system only check if a *user* node exist and test it.

- **user** node
  - *login* Attribute
  - *password* Attribute
  - *token* Attribute

If the password module is enabled, it uses the *login* and *password* attributes.

If the token module is enabled, it only uses the *token* attribute.

3.2.3 icon.php

- **method:** GET
- **arguments:**
  - id: an application identifier
- **returns:** the application icon as 32x32:32 image/png

This service provide an application icon from an application id.

The authentication can be done either by the dedicated service (auth.php) or directly into this one. It also possible to access to any icon without anonymoulsy by enable the *Public Webservices access* access from the Administration Console.

3.2.4 logout.php

- **method:** ANY
- **arguments:** None
- **returns:** XML

This webservice can be called to turn into de-authenticaded state.

3.2.5 mimetype-icon.php

- **method:** GET
- **arguments:**
  - id: a mime-type name
- **returns:** the application icon as 32x32:32 image/png

This service provide an mime-type icon building it on the fly from the most appropriate application supporting that mime-type.

3.2.6 session_status.php

- **method:** GET
- **arguments: None**
- **returns:** session status as XML content

XML format:

- **session** Root node
  - **id** Attribute
  - **status** Attribute

3.2.7 userlist.php

- **method:** GET
- **arguments: none**
- **returns:** a users list as XML content.

This webservice provide the available users list from the users base used by the Session Manager.

XML format:

- **users** Root node
  - **user nodes**
- **user** Node
  - *login* Attribute
  - *displayname* Attribute

The service only respond if the *Display users list* setting is enable from the Administration Console.

The service limits to 100 items in the list (can be changed from the Administration Console).

3.2.8 start.php

- **method:** POST
- **arguments: XML content**
- **returns:** session parameters as XML content

This service is the one used by a client to negotiate a session. In the input XML, the client describes the parameter wanted for the session. According with the Administration Console policy, the Session Manager returns either a session (with parameter and access) or an error message explaining why the session cannot be created.

The error returned message can be:

- *auth_failed*
- *in_maintenance*
- *internal_error*
- *invalid_user*
- *service_not_available*
- *unauthorized_session_mode*
- *user_with_active_session*

4 Web Client external API

The Ulteo OVD Web Client has an Javascript API to easily start an application or a desktop from another service.

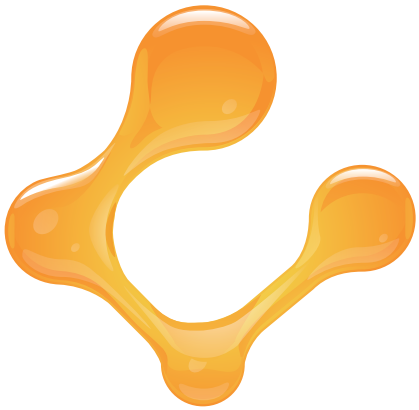
Here is a simple example HTML page which start a seamless Ulteo OVD application:

```
<html>
  <head>
    <!-- Load the Ulteo javascript files -->
    <script type="text/javascript"
      src="http://web.ulteo.com/media/script/lib/prototype/prototype ↩
          .js"
      charset="utf-8"></script>
    <script type="text/javascript"
      src="http://web.ulteo.com/media/script/external.js"
      charset="utf-8"></script>

    <script type="text/javascript">
function start(form_) {
  var ovd = new UlteoOVD_start_Application(
                'http://'+form_.server.value+'/ovd',
                form_.app.value
            );
        ovd.setAuthPassword(form_.login.value, form_.password.value) ↩
           ;
  ovd.start();
}
    </script>
  </head>
  <body>

<form action="#" onsubmit="start(this); return false;">
 Server:
 <input type="text" name="server" value="web.ulteo.com" />
 <br/>

 Login:
 <input type="text" name="login" value="" />
 <br/>

 Password:
 <input type="password" name="password" value="" />
 <br/>

 Application id:
 <input type="text" name="app" value="2" />
 <br/>

 <input type="submit" value="Launch" />
</form>
 </body>
</html>
```

Ulteo Open Virtual Desktop v3.0

Protocol Description

For further informations, check our website www.ulteo.com.



Ulteo