

# ACCESO REMOTO A LOS SISTEMAS INFORMÁTICOS DE LOS LABORATORIOS DOCENTES DEL DIS BASADO EN LA TECNOLOGÍA APACHE GUACAMOLE



Alumno: Javier Suárez Suárez

Tutores: Carmelo Rubén García Rodríguez y Francisco Alexis Quesada Arencibia

Grado en Ingeniería Informática (GII)

Junio 2022 – Universidad de Las Palmas de Gran Canaria (ULPGC)

## Resumen

Este proyecto tiene como objetivo la instalación del servicio Apache Guacamole para el acceso remoto a los equipos informáticos de los laboratorios docentes del Departamento de Informática y Sistemas (DIS) de la Universidad de Las Palmas de Gran Canaria (ULPGC). Con ello, se busca permitir que los estudiantes puedan realizar actividades prácticas docentes de forma telemática, requiriendo únicamente la instalación de un navegador para ello. Se trabajará con: Servicios Web (JavaScript, HTML, APACHE), Protocolos de acceso remoto (VNC y RDP) y Sistemas de autenticación de Usuarios (base de datos local, LDAP, RADIUS, CAS). Como medios, se dispondrá de equipos informáticos que actuarán como clientes del servicio y de un servidor que ofrecerá dicho servicio.

## Abstract

This project aims to install the Apache Guacamole service to get remote access to the University of Las Palmas of Gran Canaria's (ULPGC) Systems and Computing Department's (DIS) teaching labs' PCs. The goal is to allow students to carry out practical teaching activities telematically, requiring only the installation of a browser to do so. The technologies that will be used in this project include Web Services (JavaScript, HTML, APACHE), Remote access protocols (VNC and RDP) and Authentication Systems (local database, LDAP, RADIUS, CAS). As means to get this project done, the personal computers belonging to the teaching labs will act as clients of the service and a server will offer it.

# Índice

1.- Introducción .....	6
1.1.- Estado actual y objetivos iniciales .....	6
1.2.- Justificación de las competencias específicas cubiertas.....	7
1.2.1.- TI01 .....	7
1.2.2.- TI02 .....	7
1.2.3.- TI05 .....	8
1.2.4.- TI06 .....	8
1.3.- Aportaciones .....	9
1.3.1.- Entorno socioeconómico .....	9
1.3.2.- Entorno técnico .....	9
1.3.3.- Entorno científico.....	10
1.4.- Estructura de este documento.....	10
2.- Desarrollo.....	11
2.1.- Apache Guacamole .....	11
2.1.1.- El protocolo Guacamole.....	12
2.1.2.- El demonio guacd.....	12
2.1.3.- La aplicación web.....	13
2.2.- Fases del Proyecto.....	13
2.3.- Fase I: Estudio Previo/Análisis .....	15
2.3.1.- Análisis de requisitos basado en las necesidades del CPD del DIS .....	15
2.3.2.- Estudio de las tecnologías a utilizar .....	16
2.4.- Fase II: Diseño/Desarrollo/Implementación .....	23
2.4.1.- Diseño de la Infraestructura .....	23
2.4.2.- Instalación del Servicio .....	24
2.4.2.1.- Instalación del Servlet .....	25
2.4.2.2.- Instalación del servidor de Apache Guacamole .....	30
2.4.2.3.- Instalación del cliente de Apache Guacamole.....	33
2.4.3.- Configuración inicial del servicio .....	37
2.4.3.1.- Directorio GUACAMOLE_HOME (/etc/guacamole) .....	37
2.4.3.2.- Fichero guacamole.properties .....	38
2.4.3.3.- Fichero guacd.conf.....	39
2.4.3.4.- Autenticación por defecto (user-mapping.xml).....	41
2.4.3.5.- VNC (Virtual Network Computing).....	42
2.4.3.6.- RDP (Remote Desktop Protocol) .....	43

2.4.3.7.- Configuración inicial realizada .....	46
2.4.4.- Instalación del soporte para la base de datos y para el sistema de autenticación .....	49
2.4.4.1.- Instalación del soporte para la base de datos.....	49
2.4.4.2.- Configuración realizada en Guacamole para la Base de Datos .....	55
2.4.4.3.- Instalación del soporte para el sistema de autenticación CAS .....	57
2.4.4.4.- Configuración realizada en Guacamole para el sistema de autenticación CAS .....	58
2.4.5.- Instalación del software cliente en los equipos docentes .....	59
2.4.5.1.- Instalación de servidores VNC.....	60
2.4.5.2.- Habilitación de las conexiones RDP .....	61
2.5.- Fase III: Evaluación/Validación/Prueba.....	62
2.5.1.- Pruebas Iniciales. Instalación de KVM .....	62
2.5.2.- Pruebas Iniciales. Resultados obtenidos en las máquinas virtuales .....	64
2.5.2.1.- Resultados obtenidos a modo de ejemplo en la conexión VNC.....	65
2.5.3.- Prueba y análisis de rendimiento en equipos docentes basados en Windows y Linux .....	66
2.5.3.1.- Configuración y Resultados: Equipo con sistema operativo Linux .....	67
2.5.3.2.- Configuración y Resultados: Equipo con sistema operativo Windows.....	68
2.6.- Competencias Adicionales .....	70
3.- Conclusiones y Trabajos futuros .....	71
4.- Anexo: Manual de Usuario de Apache Guacamole .....	73
5.- Referencias .....	120

## Índice de figuras

<b>Figura 1:</b> Representación de la estructura de Apache Guacamole. [7].....	11
<b>Figura 2:</b> Fases del Proyecto.....	14
<b>Figura 3:</b> Logo de Apache Guacamole [3].....	16
<b>Figura 4:</b> Logo de CentOS7 [9].....	17
<b>Figura 5:</b> Logo de Apache Tomcat [11].....	17
<b>Figura 6:</b> Logo de VNC [14].....	18
<b>Figura 7:</b> Logo de RDP [17].....	18
<b>Figura 8:</b> Logo de Telnet [20].....	19
<b>Figura 9:</b> Logo de SSH [22].....	19
<b>Figura 10:</b> Logo de <i>MariaDB</i> [24].....	20
<b>Figura 11:</b> Logo de CAS [26].....	20
<b>Figura 12:</b> Logo de LDAP [30].....	21
<b>Figura 13:</b> Logo de RADIUS [33].....	21
<b>Figura 14:</b> Logo de KVM [36].....	22
<b>Figura 15:</b> Establecimiento de la versión de Java por defecto en el sistema.....	25
<b>Figura 16:</b> Fichero <i>/etc/environment</i> .....	26
<b>Figura 17:</b> Localización del fichero de Apache Tomcat. [41].....	26
<b>Figura 18:</b> Bloque comentado (Apache Tomcat).....	28
<b>Figura 19:</b> Creación de usuario administrador de las aplicaciones de Apache Tomcat.....	28
<b>Figura 20:</b> Acceso por vía del navegador a la interfaz web de Apache Tomcat.....	29
<b>Figura 21:</b> Solicitud de autenticación para usar la aplicación administrativa “ <i>Host Manager</i> ”.....	29
<b>Figura 22:</b> Página de la aplicación administrativa “ <i>Host Manager</i> ” de Apache Tomcat.....	29
<b>Figura 23:</b> Mensaje de información sobre la ausencia de dependencias para el soporte de VNC.....	32
<b>Figura 24:</b> Resultado de la ejecución del script <i>configure</i> .....	33
<b>Figura 25:</b> Ubicación del fichero de Apache Maven. [46].....	34
<b>Figura 26:</b> Fichero <i>/etc/profile.d/maven.sh</i> .....	34
<b>Figura 27:</b> Versión de Maven.....	35
<b>Figura 28:</b> Resultado de la construcción y compilación realizada por Apache Maven.....	35
<b>Figura 29:</b> Aplicación web de Apache Guacamole.....	36
<b>Figura 30:</b> Formato del fichero <i>user-mapping.xml</i> . [47].....	42
<b>Figura 31:</b> Directorio <i>GUACAMOLE_HOME</i> y subdirectorios <i>extensions/</i> y <i>lib/</i> .....	46
<b>Figura 32:</b> Contenido del fichero <i>guacd.conf</i> .....	47
<b>Figura 33:</b> Contenido del fichero <i>logback.xml</i> .....	47
<b>Figura 34:</b> Contenido del fichero <i>guacamole.properties</i> .....	47
<b>Figura 35:</b> Contenido del fichero <i>user-mapping.xml</i> .....	48
<b>Figura 36:</b> Ubicación de la extensión y la librería para el soporte de Base de Datos.....	51
<b>Figura 37:</b> Sitio web del conector JDBC. [50].....	51
<b>Figura 38:</b> Recomendación de la versión 10.6 de <i>MariaDB</i> .....	52
<b>Figura 39:</b> Ejecución del script de seguridad de <i>MariaDB</i> (I).....	53
<b>Figura 40:</b> Ejecución del script de seguridad de <i>MariaDB</i> (II).....	53
<b>Figura 41:</b> Establecimiento del juego de caracteres para el idioma Español en <i>MariaDB</i> .....	54
<b>Figura 42:</b> Creación de la base de datos y ejecución de scripts SQL.....	54
<b>Figura 43:</b> Creación del usuario para Apache Guacamole y concesión de permisos. [49].....	55
<b>Figura 44:</b> Propiedades para Base de Datos en el fichero <i>guacamole.properties</i> .....	56
<b>Figura 45:</b> Panel de usuarios de la interfaz web.....	56
<b>Figura 46:</b> Panel de conexiones de la interfaz web.....	56

<b>Figura 47:</b> Extensiones de Base de Datos y CAS ubicadas en <i>GUACAMOLE_HOME/extensions</i> . .....	57
<b>Figura 48:</b> Propiedades para CAS en el fichero <i>guacamole.properties</i> . .....	59
<b>Figura 49:</b> Contenido del fichero <i>/etc/yum.conf</i> . .....	62
<b>Figura 50:</b> Comprobación del estado de <i>libvirtd</i> y de la carga de módulos en el <i>kernel</i> . .....	63
<b>Figura 51:</b> Máquinas virtuales creadas. ....	64
<b>Figura 52:</b> Características de la conexión “Estación Linux”. .....	65
<b>Figura 53:</b> Características de la conexión “Estación Windows”. .....	65
<b>Figura 54:</b> Resultado de la conexión “Estación Linux”. .....	66
<b>Figura 55:</b> Conexiones del usuario. ....	67
<b>Figura 56:</b> Características de la conexión “Equipo Linux”. .....	67
<b>Figura 57:</b> Resultado de la conexión “Equipo Linux”. .....	68
<b>Figura 58:</b> Características de la conexión “Equipo Windows”. .....	69
<b>Figura 59:</b> Resultado de la conexión “Equipo Windows”. .....	69
<b>Figura 60:</b> Formato del fichero <i>user-mapping.xml</i> . [47] (Manual de Usuario) .....	98
<b>Figura 61:</b> Creación de la base de datos y ejecución de consultas. [49] (Manual de Usuario) .....	102
<b>Figura 62:</b> Creación del usuario para Apache Guacamole y concesión de permisos. [49] (Manual de Usuario) .....	103

# 1.- Introducción

## 1.1.- Estado actual y objetivos iniciales

Al Departamento de Informática y Sistemas (DIS) [1] pertenecen las áreas de conocimiento de Arquitectura y Tecnología de Computadores, Ciencia de la Computación e Inteligencia Artificial y Lenguajes y Sistemas Informáticos [2]. Por tanto, es su responsabilidad el desarrollo de la docencia de estas tres áreas de conocimiento. Actualmente tiene docencia adscrita en 14 titulaciones oficiales de grado, master y doctorado. Para el desarrollo de la docencia de contenidos prácticos y las actividades de evaluación dispone de 11 laboratorios docentes, proporcionando en torno a los 300 puestos de trabajo. Además, su centro de proceso de datos dispone de una infraestructura que permite el despliegue de puestos de trabajos virtuales. Por ello, es de interés la implementación de medidas para permitir la docencia telemática de calidad, debido a la situación de pandemia por el COVID-19 como la que estamos sufriendo o para que las titulaciones en las que imparte docencia puedan ampliar su oferta formativa en la modalidad docencia no presencial. En el año 2020, se realizaron diversas pruebas para implementar tecnologías orientadas a permitir dicha docencia. Entre estas pruebas, se realizaron aquellas destinadas a la instalación y configuración de la tecnología Apache Guacamole [3], servicio web basado en HTML5 (*HyperText Markup Language*) que permite el acceso remoto a equipos informáticos. Estas pruebas no obtuvieron los resultados esperados y, por dicha razón, se descartó implementar esta tecnología.

Sin embargo, Apache Guacamole es una tecnología que ofrece grandes posibilidades, pues permite que, por vía de la web, se haga uso de esta, proporcionando al usuario un sitio web donde este se autentica y, tras este proceso, ve aquellas máquinas a las que puede conectarse remotamente.

Por esta razón, surge la idea de este proyecto, cuyo fin es instalar un servicio que permita el acceso remoto a los equipos informáticos presentes en los laboratorios docentes del DIS para la realización de actividades prácticas. Para ello, se han abordado una serie de hitos:

- H1. Diseño de la infraestructura básica en la que instalar y ejecutar el servicio Apache Guacamole.
- H2. Instalación del servicio Apache Guacamole.
- H3. Establecimiento de los requerimientos para la instalación del software cliente en los equipos utilizados en las actividades prácticas.
- H4. Instalación del software cliente en los equipos utilizados en las actividades prácticas.
- H5. Pruebas y evaluación del rendimiento del sistema.

## 1.2.- Justificación de las competencias específicas cubiertas

La realización de este proyecto cubre una serie de competencias específicas correspondientes a la intensificación en tecnologías de la información contemplada en el plan de estudios 40 del Grado en Ingeniería Informática [4]. Dichas competencias cubiertas se corresponden con aquellas con los códigos TI01, TI02, TI05 y TI07 y su justificación se expone a continuación.

### 1.2.1.- TI01

*“Capacidad para comprender el entorno de una organización y sus necesidades en el ámbito de las tecnologías de la información y las comunicaciones.”*

La justificación de esta competencia radica en el análisis realizado de los requerimientos del Centro de Procesamiento de Datos (CPD) perteneciente al Departamento de Informática y Sistemas (DIS), con el objetivo de conocer su estructura y cómo integrar la tecnología Apache Guacamole en él.

### 1.2.2.- TI02

*“Capacidad para seleccionar, diseñar, desplegar, integrar, evaluar, construir, gestionar, explotar y mantener las tecnologías de hardware, software y redes, dentro de los parámetros de coste y calidad adecuados.”*

La justificación de esta competencia se basa en el proceso de 4 fases (Análisis de requerimientos, Diseño e instalación, Pruebas de rendimiento y Documentación) llevado a cabo para la integración de la nueva tecnología con la ya existente. Este proceso ha estado regido bajo los límites hardware y software existentes en el entorno de la organización y ha culminado con la instalación de un nuevo servicio que hace uso de software libre y de código abierto.

### 1.2.3.- TI05

*“Capacidad para seleccionar, desplegar, integrar y gestionar sistemas de información que satisfagan las necesidades de la organización, con los criterios de coste y calidad identificados.”*

La justificación de esta competencia consiste en que se ha integrado la tecnología Apache Guacamole en el sistema de información del DIS para que los usuarios que hagan uso de ella puedan autenticarse y ver sus máquinas asociadas, estableciendo aquellas conexiones que deseen y, adicionalmente, ofreciendo posibilidades de gestión web para usuarios con privilegios de administrador. Esta aplicación web, al igual que el resto de la tecnología, es de software libre y de código abierto.

### 1.2.4.- TI06

*“Capacidad de concebir sistemas, aplicaciones y servicios basados en tecnologías de red, incluyendo Internet, web, comercio electrónico, multimedia, servicios interactivos y computación móvil.”*

La justificación de esta competencia se corresponde con el nuevo servicio de acceso remoto instalado en un servidor ubicado en el CPD del DIS, existente pero inoperativo hasta la realización de este proyecto, que opera en Internet para recibir y gestionar solicitudes de usuarios por vía de la web.

## 1.3.- Aportaciones

### 1.3.1.- Entorno socioeconómico

El desarrollo de este TFT proporciona una nueva herramienta muy útil a nivel socioeconómico. Al ser una tecnología que posibilita el acceso remoto a equipos informáticos, abre nuevas posibilidades en el ámbito del teletrabajo y el uso de tecnologías de forma no presencial. Esto se traduce en un gran beneficio para las organizaciones en general y, concretamente, aquellas ubicadas en las Islas Canarias, pues permite que empleados que se hallen fuera de las islas puedan realizar su jornada laboral sin necesidad de estar presencialmente en la oficina. Esto supone una gran ventaja, pues las organizaciones pueden ahorrar en costes y poseen planes de contingencia para situaciones en las que el trabajo presencial no es posible, como por ejemplo en años recientes debido a la pandemia por el virus SARS-CoV-2 (también denominado COVID-19). [5]

Adicionalmente, la tecnología estudiada e implementada en este TFT, a diferencia de otras de la misma índole, requiere recursos mínimos por parte de los usuarios que hagan uso de esta. Concretamente, los equipos informáticos con los que se quiera hacer uso de esta tecnología solo requieren de acceso a Internet y de un navegador web. Si el usuario posee autorización, podrá conectarse de forma efectiva a aquellas máquinas a las que tenga acceso. Esto es una gran ventaja, pues hace que el uso de esta tecnología no requiera de altos costes que puedan ser inasumibles para muchas personas.

### 1.3.2.- Entorno técnico

La tecnología estudiada e implementada en este Trabajo de Fin de Título supone, a nivel técnico, reducciones en los costes de las infraestructuras. Esta tecnología requiere, a nivel hardware, de conexiones en red con equipos informáticos físicos o virtuales que estén configurados correctamente para recibir las conexiones remotas de los usuarios y de la disposición de un servidor que ofrezca el nuevo servicio que, si así se desea, puede satisfacer también los requerimientos a nivel software para el correcto funcionamiento de dicha tecnología. Estos requerimientos a nivel software consisten en sistemas de autenticación de usuarios y almacenamiento de conexiones (Bases de Datos, CAS - *Central Authentication Service*, entre otros).

Otros requerimientos que suelen existir para implementar otras tecnologías de la misma índole, como la disposición de VPNs (*Virtual Private Network*), no son necesarios, pues las conexiones de los usuarios desde la web se centralizan y gestionan desde el servidor mencionado.

### 1.3.3.- Entorno científico

El desarrollo de ese Trabajo de Fin de Título aporta también beneficios a nivel científico. A pesar de no ser un avance científico como tal, sí supone una herramienta de apoyo para la realización de dichos avances. Gracias a esta tecnología, se posibilita la investigación por medios remotos, lo cual es una gran ventaja para numerosos investigadores que se encuentran repartidos en diferentes lugares del planeta, pues hacer uso de esta tecnología les dota de flexibilidad para las tareas que realizan a diario, como por ejemplo recopilaciones de datos, puestas en común con compañeros, entre otras.

### 1.4.- Estructura de este documento

Este documento se estructura en diversas secciones cuya descripción se expone a continuación:

➤ 1. Introducción

Esta sección, que finaliza con este apartado, tiene la finalidad de mostrar el estado actual y los objetivos iniciales de este TFT, así como justificar las competencias específicas que cubre e indicar las aportaciones que genera a nivel socioeconómico, técnico y científico.

➤ 2. Desarrollo

Esta sección posee la finalidad de mostrar detalladamente todas las fases llevadas a cabo para la realización del proyecto, explicando el trabajo realizado en cada una de ellas y concluyendo con las competencias adicionales adquiridas.

➤ 3. Conclusiones y Trabajos futuros

Esta sección tiene como objetivo presentar los resultados obtenidos tras la realización de este proyecto, describiendo el grado de consecución de los objetivos y comentando las posibles extensiones de este.

➤ 4. Anexo: Manual de Usuario de Apache Guacamole

Esta sección incluye el Manual de Usuario desarrollado conjuntamente a este proyecto.

➤ 5. Fuentes de Información

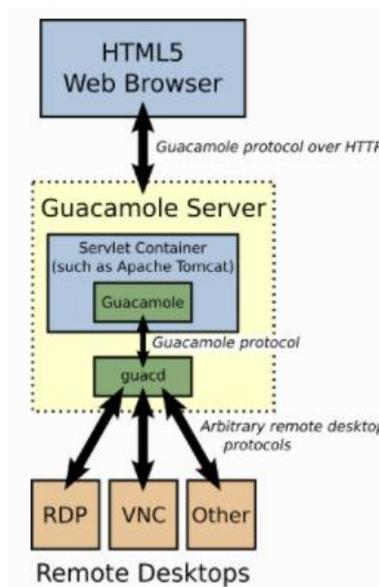
Esta última sección contiene todas aquellas fuentes de información utilizadas para la realización de este proyecto.

## 2.- Desarrollo

En esta sección del documento se expondrá el trabajo realizado en este TFT. En primer lugar, se introducirá la tecnología implementada para posteriormente presentar las fases del proyecto y profundizar en cada una de ellas, detallando los procedimientos llevados a cabo para la implementación del nuevo servicio. Finalmente, se presentarán las competencias adicionales adquiridas en la realización de este proyecto, que incluyen tecnologías secundarias estudiadas que se encuentran ampliadas en el Manual de Usuario ubicado en la sección “Anexo: Manual de Usuario de Apache Guacamole” de este documento.

### 2.1.- Apache Guacamole

Apache Guacamole [6] es un servicio web basado en HTML5 que permite el acceso remoto a sistemas informáticos, haciendo uso de protocolos de escritorio remoto (como VNC - *Virtual Network Computing*) o RDP - *Remote Desktop Protocol*, entre otros). Las razones para usar Apache Guacamole incluyen el acceso remoto a través de cualquier dispositivo, la posesión de un sistema informático en la “nube”, el fácil acceso a varios sistemas informáticos y la capacidad de integrar el servicio en cualquier infraestructura existente. Apache Guacamole [7] está compuesto por una aplicación web y varios componentes de bajo nivel. El esquema de su estructura es el siguiente:



**Figura 1:** Representación de la estructura de Apache Guacamole. [7]

Los usuarios se conectan a un servidor de Apache Guacamole con su navegador. El cliente Guacamole, programado en el lenguaje de programación JavaScript, es servido a los usuarios por parte de un servidor web integrado en el servidor de Guacamole. Una vez cargado, el cliente se conecta de nuevo hacia el servidor a través de HTTP (*Hypertext Transfer Protocol*) usando el protocolo Guacamole.

Tanto el protocolo Guacamole como el demonio *guacd* proporcionan independencia de protocolos evitando que ni el cliente Guacamole ni la aplicación web deban saber qué protocolo de escritorio remoto se está realmente usando.

### 2.1.1.- El protocolo Guacamole

La aplicación web de Apache Guacamole no comprende ningún protocolo de escritorio remoto al completo. En su lugar, solo trabaja con el protocolo Guacamole, un protocolo para el renderizado remoto y transporte de eventos. A diferencia de los protocolos de escritorio remoto, los principios de diseño del protocolo Guacamole indican la no intención de implementar las características de un entorno de escritorio específico.

El protocolo Guacamole implementa un conjunto de protocolos de escritorio remoto existentes. Añadir soporte para un determinado protocolo de escritorio remoto conlleva la creación de una capa intermedia que “traduce” la comunicación entre el protocolo Guacamole y el protocolo de escritorio remoto. Esta capa intermedia es *guacd*.

### 2.1.2.- El demonio guacd

*guacd* es el núcleo de Apache Guacamole que dinámicamente carga el soporte para protocolos de escritorio remoto, conectándolos a escritorios remotos, basándose en instrucciones recibidas desde la aplicación web.

*guacd* es un proceso catalogado como *demonio del sistema* [8] que se instala junto a Apache Guacamole y se ejecuta en segundo plano, escuchando conexiones TCP (*Transmission Control Protocol*) procedentes de la aplicación web. No comprende ningún protocolo de escritorio remoto concreto, pero implementa suficientes funciones pertenecientes al protocolo Guacamole para determinar qué soporte de protocolo necesita cargar y con qué argumentos. Una vez el protocolo se carga, este se ejecuta independientemente de *guacd* y posee el control completo de la comunicación entre sí mismo y la aplicación web hasta que el cliente finaliza.

*guacd* y todos los protocolos de escritorio remoto dependen de una librería común, *libguac*, que facilita y hace más abstracta la comunicación a través del protocolo Guacamole.

### 2.1.3.- La aplicación web

La aplicación web incluida en Apache Guacamole es donde el usuario realiza su interacción con el servicio. Esta aplicación solo implementa una interfaz web y una capa de autenticación de usuarios, sin tener en cuenta ningún protocolo de escritorio remoto concreto. La aplicación web que proporciona Apache Guacamole está escrita en el lenguaje de programación Java, pero no es necesario hacer uso de esta, pues Guacamole posee una API (*Application Programming Interface*) con el objetivo de ser integrado en infraestructuras existentes.

A modo de historia, Guacamole comenzó siendo un cliente Telnet programado en JavaScript denominado *RealMint*. Tras ello, evolucionó hacia un cliente VNC programado en el mismo lenguaje y, posteriormente, se convirtió en una *Gateway* de escritorio remoto de HTML5 (*Hypertext Markup Language*) que puede ser usada como *gateway* central o implementada mediante su API.

## 2.2.- Fases del Proyecto

El desarrollo de este TFT fue dividido en un total de cuatro fases correspondientes al **Estudio Previo/Análisis**, **Diseño/Desarrollo/Implementación**, **Evaluación/Validación/Prueba** y **Documentación/Presentación**.

Dichas fases fueron asociadas a una estimación en horas que, en conjunto, cumplieran con las 300 horas estipuladas para los TFT, tal y como se indica en los requerimientos de la Escuela de Ingeniería Informática. Sin embargo, las fases de Estudio Previo/Análisis y Diseño/Desarrollo/Implementación se han visto ligeramente modificadas en su estimación temporal, debido a que se requirió de un mayor número de horas para la primera fase y uno menor para la segunda, al partir esta última de la información adquirida durante la fase anterior. En la siguiente tabla, se pueden observar las fases junto a las tareas realizadas en cada una de ellas y la estimación en horas establecida, así como la duración real.

Fases	Duración estimada (en horas)	Tareas	Duración real (en horas)
<b>Estudio previo/Análisis</b>	25	<b>Tarea 1.1:</b> Análisis de requisitos basado en las necesidades del CPD del Dpto. de Informática y Sistemas	75
		<b>Tarea 1.2:</b> Estudio de las tecnologías a utilizar	
<b>Diseño/Desarrollo/Implementación</b>	200	<b>Tarea 2.1:</b> Diseño de la infraestructura	150
		<b>Tarea 2.2:</b> Instalación y configuración del servicio.	
		<b>Tarea 2.3:</b> Instalación y configuración del sistema de autenticación	
		<b>Tarea 2.4:</b> Instalación software cliente en los equipos docentes	
<b>Evaluación/Validación/Prueba</b>	50	<b>Tarea 3:</b> Prueba y análisis de rendimiento en equipos docentes basados en Windows y Linux	50
<b>Documentación/Presentación</b>	25	<b>Tarea 4.1:</b> Desarrollo de la documentación de la memoria del trabajo de fin de título.	25
		<b>Tarea 4.2:</b> Desarrollo de los manuales de usuario.	

**Figura 2:** Fases del Proyecto.

A continuación, se describirán las diferentes tareas realizadas en el proyecto dividiéndolas por fases tal y como se muestra en la tabla. Se excluirá detallar la fase final de Documentación/Presentación debido a que consiste en desarrollar esta memoria de TFT junto al Manual de usuario ubicado en la sección “*Anexo: Manual de Usuario de Apache Guacamole*”.

## 2.3.- Fase I: Estudio Previo/Análisis

La primera fase del proyecto consistió en el análisis de requisitos basado en las necesidades del Centro de Procesamiento de Datos (CPD) del DIS y en el estudio de las tecnologías a utilizar. Para ello, se acudió a dicho CPD y se observaron sus características, estableciendo, en consenso con el personal responsable y los tutores, la forma de instalar la nueva tecnología e integrarla con la ya presente.

Simultáneamente, se acudió a la documentación oficial de Apache Guacamole (indicada en [3]) para el estudio de la tecnología, incluyendo la forma de instalarla y configurarla y cómo hacer uso de las extensiones que posee para integrarse con sistemas de autenticación presentes en el mercado.

### 2.3.1.- Análisis de requisitos basado en las necesidades del CPD del DIS

Como primer paso para realizar este proyecto, se realizó un análisis de requisitos, junto al personal responsable y los tutores, basado en las necesidades del CPD del DIS.

De este análisis se obtuvo, en consenso, la forma de instalar la nueva tecnología e integrarla con la ya presente. Para ello, se realizó una reunión con los tutores en la que se debatieron diversas formas de instalar dicha tecnología.

La primera forma consistía en habilitar una máquina virtual con recursos suficientes en la que, con privilegios de administrador y vía acceso remoto, se realizaría el despliegue del servidor de Apache Guacamole. Adicionalmente, se proporcionarían, a modo de pruebas de funcionamiento, diversas máquinas virtuales que emularían las diferentes configuraciones de los equipos ubicados en los laboratorios docentes. Todas estas máquinas se ejecutarían en un equipo con bastantes recursos que posee el DIS para el que no se iba a poseer privilegios de administración debido a que dicho equipo ofrece soporte para servicios de la Escuela de Ingeniería Informática en explotación.

Sin embargo, se decidió finalmente hacer uso de una máquina física, denominada Guacamole, que fue utilizada en el año 2020 para realizar pruebas de instalación de la tecnología que no obtuvieron los resultados esperados. Esta máquina poseía una instalación previa que fue eliminada y se partió de una nueva a partir de la versión “*Minimal*” de la distribución Linux denominada CentOS7 (*Community ENTerprise Operating System 7*).

Con el objetivo de evaluar inicialmente el funcionamiento del servicio, se utilizó la tecnología KVM (*Kernel-based Virtual Machine*) para instalar en el propio equipo máquinas virtuales que se utilizaron para simular los equipos ubicados en los laboratorios docentes.

Posteriormente, se procedió a configurar los propios equipos docentes para evaluar el funcionamiento completo del nuevo servicio.

### 2.3.2.- Estudio de las tecnologías a utilizar

Para el desarrollo de este TFT se realizó, junto al análisis de requisitos basado en las necesidades del CPD del DIS, un estudio de las tecnologías a utilizar para conseguir instalar, configurar e integrar el nuevo servicio. Dichas tecnologías estudiadas se detallan a continuación.

#### Apache Guacamole



**Figura 3:** Logo de Apache Guacamole [3]

Apache Guacamole [6] es un servicio web basado en HTML5 (*HyperText Markup Language*) que permite el acceso remoto a equipos informáticos. Está dividido en dos partes (cliente y servidor) que, instaladas y configuradas de forma adecuada, operan en conjunto para ofrecer el servicio.

Adicionalmente, posee diversas extensiones para ser configurado utilizando sistemas de autenticación presentes en el mercado, como Bases de Datos (*MariaDB, MySQL, PostgreSQL, SQL Server*), LDAP (*Lightweight Directory Access Protocol*), RADIUS (*Remote Authentication Dial in User Service*) o CAS (*Central Authentication Service*), entre otros.

## Distribución CentOS7

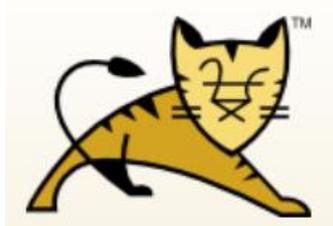


**Figura 4:** Logo de CentOS7 [9]

CentOS7 (*Community ENTERprise Operating System 7*) [9, 10] es una distribución Linux de código abierto derivada a partir de la distribución *Red Hat Enterprise Linux* (RHEL). Es una plataforma estable, gestionable, predecible y reproducible que trata de ser compatible funcionalmente con RHEL. Su objetivo es proporcionar una plataforma base para aquellas comunidades que hagan uso de software libre.

Se ha decidido escoger esta distribución como sistema operativo donde instalar la nueva tecnología debido a su estabilidad y popularidad. Además, se ha seleccionado su versión 7 y no una posterior debido a que en el CPD se emplea actualmente esta edición.

## Apache Tomcat



**Figura 5:** Logo de Apache Tomcat [11]

Apache Tomcat [11] es una implementación vía software libre de diversas especificaciones incluidas en la plataforma Jakarta EE [12]. Esta plataforma es una evolución de otra, denominada Java EE [13]. La versión 10 de Tomcat implementa especificaciones desarrolladas a partir de Jakarta EE, mientras que versiones anteriores lo hacen a partir de Java EE.

Esta tecnología software proporciona soporte para aplicaciones web de gran escala pertenecientes a numerosas organizaciones e industrias.

Se ha decidido usar esta tecnología para instalar el *Servlet* necesario para la parte cliente de Apache Guacamole, debido a las menciones que se le realiza en la documentación oficial del servicio. Además, se ha escogido su versión 9 debido a que su versión 10, de 20 de enero de 2022, requiere de un proceso de migración de aplicaciones con motivo del cambio de plataforma Java EE a Jakarta EE que, debido a su reciente publicación, se ha decidido no implementar para evitar posibles inconvenientes de estabilidad.

## VNC (*Virtual Network Computing*)



**Figura 6:** Logo de VNC [14]

VNC [15] es un software libre consistente en una estructura cliente-servidor y un protocolo de comunicación que permite el acceso y la compartición de escritorios remotos. No impone restricciones en cuanto al tipo de sistema operativo de cliente y servidor.

Se ha utilizado este software y, concretamente, su implementación denominada *TigerVNC* [16], como protocolo de escritorio remoto para el que Apache Guacamole ofrece soporte con el objetivo de permitir las conexiones remotas a equipos con sistema operativo Linux (aunque este software puede utilizarse para otros sistemas operativos del mercado).

## Protocolo RDP (*Remote Desktop Protocol*)



**Figura 7:** Logo de RDP [17]

RDP [18, 19] es un protocolo desarrollado por Microsoft que permite que el escritorio de un equipo informático sea controlado de forma remota. Existen numerosas implementaciones de este protocolo para permitir su utilización en diversos sistemas operativos. Para los sistemas Windows 10/11 Pro/Enterprise, entre otras versiones, se dispone, de forma nativa, de opciones para habilitarlo.

Se ha utilizado este protocolo con el objetivo de permitir las conexiones remotas a equipos con sistema operativo Windows (aunque, como se ha mencionado, este software puede utilizarse para otros sistemas operativos del mercado).

## Protocolo Telnet

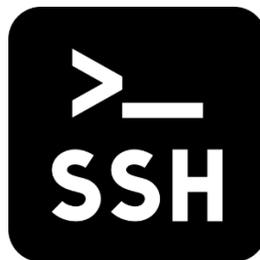


**Figura 8:** Logo de Telnet [20]

Telnet [21] es un protocolo de acceso remoto destinado a la administración telemática únicamente por terminal. Posee una serie de desventajas que hacen que su uso no sea recomendado en sistemas modernos. Estas desventajas incluyen vulnerabilidades en los dominios usados por el protocolo, así como la ausencia de esquemas de autenticación y de cifrado de conexiones.

Este protocolo no se ha implementado en este TFT a causa de sus desventajas, pero sí fue estudiado, debido a su sencilla implementación e integración con Apache Guacamole, con el objetivo de reflejar la forma de configurarlo por si se requiere, en líneas futuras de este proyecto, su implementación para alguna función concreta, aunque no se recomienda hacerlo. El estudio realizado puede encontrarse en el Manual de Usuario ubicado en la sección “*Anexo: Manual de Usuario de Apache Guacamole*”.

## Protocolo SSH (*Secure Shell*)



**Figura 9:** Logo de SSH [22]

SSH [23] es un protocolo de acceso remoto destinado a la administración remota principalmente por un terminal. Se creó como un reemplazo seguro para el protocolo Telnet debido a que este último no realiza cifrado de conexiones.

Al igual que Telnet, este protocolo no se ha implementado en este TFT, pero sí fue estudiado, debido a su sencilla implementación e integración con Apache Guacamole, con el objetivo de reflejar la forma de configurarlo, por si es requerido en líneas futuras de este proyecto. El estudio realizado puede encontrarse en el Manual de Usuario ubicado en la sección “*Anexo: Manual de Usuario de Apache Guacamole*”.

## Base de Datos (*MariaDB*)



**Figura 10:** Logo de *MariaDB* [24]

Apache Guacamole ofrece extensiones para el uso de Bases de Datos junto a la tecnología. Actualmente, ofrece soporte para las bases de datos *MariaDB*, *MySQL*, *PostgreSQL*, *SQL Server*.

*MariaDB* [25] es uno de los servidores de Base de Datos más populares del mundo. Fue creado por desarrolladores originales de *MySQL* y es una tecnología de código abierto. Algunos de los clientes que la utilizan son Wikipedia, WordPress y Google.

Se ha decidido utilizar *MariaDB* debido a su facilidad de uso, popularidad, rapidez, escalabilidad y robustez. El fin de utilizar esta Base de Datos ha sido como medio para almacenar la información relacionada con las conexiones que se asocian a los usuarios que se autentican en Apache Guacamole.

## CAS (*Central Authentication Service*)



**Figura 11:** Logo de CAS [26]

CAS [27] es una solución software destinada a la autenticación de usuarios en base al Inicio de Sesión Único (SSO) [28] que consiste en habilitar a un usuario el acceso a varias aplicaciones con una sola instancia de identificación.

Se ha decidido utilizar esta tecnología como medio de autenticación de los usuarios que se conectan a Apache Guacamole. Para ello, se ha hecho uso del servidor CAS que posee la ULPGC (indicado en [29]).

## LDAP (*Lightweight Directory Access Protocol*)



**Figura 12:** Logo de LDAP [30]

LDAP [31] es un protocolo basado en estándares que se ubica sobre los protocolos TCP (*Transfer Control Protocol*) e IP (*Internet Protocol*) y permite a los usuarios realizar diversas operaciones en un servidor denominado “*Directorio*”, como almacenar y obtener datos, buscar datos en base a unos criterios o realizar procesos de autenticación, entre otros. La implementación de código abierto más conocida es *OpenLDAP*. [32]

Este protocolo no se ha implementado en este proyecto, pero sí fue estudiado como alternativa de sistema de autenticación a CAS, con el objetivo de reflejar la forma de configurarlo por si es requerido en líneas futuras de este TFT. El estudio realizado puede encontrarse en el Manual de Usuario ubicado en la sección “*Anexo: Manual de Usuario de Apache Guacamole*”.

## RADIUS (*Remote Authentication Dial in User Service*)



**Figura 13:** Logo de RADIUS [33]

RADIUS [34] es un protocolo de autenticación, autorización y contabilidad (AAA por sus siglas en inglés). Posee soporte para diversos protocolos de autenticación como PPP (*Point-to-Point Protocol*), PAP (*Password Authentication Protocol*), CHAP (*Challenge Handshake Authentication Protocol*), entre otros. La implementación de código abierto más conocida es *FreeRADIUS*. [35]

Este protocolo, al igual que LDAP, no se ha implementado en este proyecto, pero sí fue estudiado como alternativa de sistema de autenticación a CAS, con el objetivo de reflejar la forma de configurarlo por si es requerido en líneas futuras. El estudio realizado puede encontrarse en el Manual de Usuario ubicado en la sección “*Anexo: Manual de Usuario de Apache Guacamole*”.

## KVM (*Kernel-based Virtual Machine*)



**Figura 14:** Logo de KVM [36]

KVM [36] es una solución de virtualización completa para Linux con hardware x86 que haga uso de las extensiones de virtualización (Intel VT o AMD-V). Consiste en un módulo de *kernel* cargable denominado *kvm.ko* que proporciona la infraestructura de virtualización y en un módulo de procesador específico, denominados *kvm-intel.ko* y *kvm-amd.ko*. Gracias a esta tecnología, se pueden ejecutar máquinas virtuales usando imágenes de los sistemas operativos Windows y Linux que poseen hardware virtualizado (tarjetas de red, discos, entre otros).

Se ha utilizado esta tecnología como medio de evaluación inicial del funcionamiento de Apache Guacamole, creando diversas máquinas virtuales con sistemas operativos Windows y Linux que han sido configuradas para emular las configuraciones de los equipos informáticos ubicados en los laboratorios docentes de la Escuela de Ingeniería Informática. Tras ello, se configuró Apache Guacamole para evaluar si el acceso remoto a las máquinas funcionaba correctamente. El desarrollo de la fase de evaluación del funcionamiento se puede observar en la sección “*Fase III: Evaluación/Validación/Prueba*” de este documento.

## 2.4.- Fase II: Diseño/Desarrollo/Implementación

La segunda fase del proyecto consistió en el diseño de la Infraestructura necesaria para la posterior instalación y configuración del nuevo servicio y de sus extensiones, así como del software cliente en los equipos docentes. Para ello, se acudió en primer lugar al CPD del DIS para planificar adecuadamente cómo integrar Apache Guacamole con la infraestructura existente. Posteriormente, se instaló y configuró el nuevo servicio haciendo uso de la documentación oficial de Apache Guacamole, realizando el mismo proceso para la instalación del software cliente en los equipos docentes.

### 2.4.1.- Diseño de la Infraestructura

Con el objetivo de instalar el nuevo servicio, se procedió en primer lugar a establecer como sería la infraestructura física donde se ubicaría la máquina que soportaría dicho servicio. Para ello, se acudió a los resultados obtenidos en el Análisis de Requisitos basado en las necesidades del CPD (indicado en la sección “2.3.1-Análisis de requisitos basado en las necesidades del CPD del DIS”) y se adoptó la decisión de hacer uso de una máquina física, denominada Guacamole, que fue utilizada en el año 2020 para realizar pruebas de instalación de la tecnología que no obtuvo los resultados esperados.

Esta máquina, que poseía una instalación previa, fue reinstalada a partir de la versión “*Minimal*” de la distribución Linux denominada CentOS7 (*Community ENTERprise Operating System 7*). Tras poseer la versión “*Minimal*” operativa, se instaló un entorno gráfico, a través del grupo de paquetes denominado “*Escritorio GNOME*”, y se instaló la tecnología *NoMachine* [37] que, junto a un certificado VPN (*Virtual Private Network*) obtenido gracias al personal del departamento, permitió el acceso remoto al equipo para realizar las tareas de instalación del nuevo servicio.

Esta tecnología (*NoMachine*) se diferencia de Apache Guacamole en que requiere que las conexiones se realicen desde la misma red que el equipo al que se quiere acceder, de ahí la necesidad del certificado VPN. Apache Guacamole permite las conexiones remotas desde cualquier red y sin necesidad de instalar ningún software (más allá de un navegador) por parte de los usuarios.

Posteriormente, se revisaron las características del sistema (CPU - *Central Processing Unit*, Memoria RAM - *Random Access Memory* y sistemas de almacenamiento, entre otros) ajustando aquellas que fueran necesarias para el correcto funcionamiento de la nueva tecnología.

A continuación, se analizaron las características de la red y se decidió dejar establecidos los parámetros por defecto obtenidos a través del servidor DHCP (*Dynamic Host Configuration Protocol*) ubicado en el departamento. Esta decisión se tomó debido a que, temporalmente, mientras se instalaba, configuraba y evaluaba el funcionamiento del servicio, el equipo poseería una dirección IP (*Internet Protocol*) en una red privada para posteriormente, cuando se estableciera el servicio en explotación, modificarla a una dirección pública, por lo que realizar una configuración manual de la red era innecesario.

Además, debido a los mismos motivos, se decidió cambiar el nombre del host (haciendo uso de la orden *hostnamectl*), pasando de “*localhost*” a “*guacamole.dis.ulpgc.es*” y se decidió realizar toda la instalación del servicio usando este FQDN (*Fully Qualified Domain Name*) teniendo en cuenta que, al poner el nuevo servicio en producción, se agregaría un nuevo registro al sistema DNS (*Domain Name System*) del departamento que asociaría dicho FQDN con la IP pública que se le proporcionó al equipo.

Por otro lado, se instaló la tecnología KVM con el objetivo de realizar pruebas iniciales de funcionamiento del servicio (ver “*2.5-Fase III: Evaluación/Prueba*” para más información acerca de la instalación de esta tecnología y los resultados de las pruebas realizadas).

Finalmente, tras establecer esta configuración inicial del equipo, se procedió a instalar el nuevo servicio.

## 2.4.2.- Instalación del Servicio

La instalación de Apache Guacamole requirió realizar, a su vez, tres instalaciones correspondientes a un *Servlet* para aplicaciones desarrolladas en el lenguaje de programación Java, debido a que la aplicación web de la denominada parte “*cliente*” de Apache Guacamole está escrita en este lenguaje; la parte denominada “*servidor*” de la tecnología y la mencionada parte “*cliente*” de la misma.

En primer lugar, se llevó a cabo la instalación del *Servlet* para el cuál se seleccionó Apache Tomcat [11] debido a que la documentación de Guacamole realiza varias menciones al mismo. Además, se escogió la versión 9 de esta tecnología debido a que, como se comentó en la sección “*2.3.2-Estudio de las tecnologías a utilizar*”, su versión 10, de 20 de enero de 2022, requiere de un proceso de migración de aplicaciones debido al cambio de plataforma Java EE a Jakarta EE que, debido a su reciente publicación, se ha decidido no implementar para evitar posibles inconvenientes de estabilidad.

Tras instalar el *Servlet* mencionado, se construyó y compiló desde el código fuente la parte servidor de Apache Guacamole y, finalmente, se instaló la parte cliente siguiendo el mismo procedimiento.

### 2.4.2.1.- Instalación del *Servlet*

A continuación, se describirán los pasos que se llevaron a cabo para instalar el *Servlet Apache Tomcat v9* en el sistema. Para la realización de dichos pasos, se hizo uso de los recursos indicados en [38,39,40].

1. En primer lugar, se actualizó el sistema y se instaló el JDK (*Java Development Kit*) de Java con la versión 11, debido a que la hallada en el sistema era la 8 y, aunque con esta versión la tecnología funciona correctamente, de cara a líneas futuras del proyecto, es recomendable la versión 11 para hacer uso de algunas tecnologías que puedan requerirla. Tras instalarla, fue configurada para que fuera la versión por defecto del sistema:

```
-yum update
-yum install java-11-openjdk-devel
-alternatives --config java
-java -version
```

```
[root@guacamole ~]# alternatives --config java
Hay 3 programas que proporcionan 'java'.
-----
Selección  Comando
-----
  1          java-1.7.0-openjdk.x86_64 (/usr/lib/jvm/java-1.7.0-openjdk-1.7.0.261-2.6.22.2.el7_8.x86_64/jre/bin/java)
  *+ 2       java-1.8.0-openjdk.x86_64 (/usr/lib/jvm/java-1.8.0-openjdk-1.8.0.332.b09-1.el7_9.x86_64/jre/bin/java)
  3          java-11-openjdk.x86_64 (/usr/lib/jvm/java-11-openjdk-11.0.15.0.9-2.el7_9.x86_64/bin/java)
Presione Intro para mantener la selección actual[+], o escriba el número de la selección: 3
[root@guacamole ~]# java -version
openjdk version "11.0.15" 2022-04-19 LTS
OpenJDK Runtime Environment 18.9 (build 11.0.15+9-LTS)
OpenJDK 64-Bit Server VM 18.9 (build 11.0.15+9-LTS, mixed mode, sharing)
[root@guacamole ~]#
```

**Figura 15:** Establecimiento de la versión de Java por defecto en el sistema.

2. A continuación, se añadió como variable de entorno la variable *\$JAVA\_HOME*, para indicar la ruta del directorio *home* de Java (ruta mostrada en segundo lugar). Tras ello, se aplicó esta variable de entorno con la orden *source*:

```
-gedit /etc/environment
    JAVA_HOME=/etc/alternatives/jre
-source /etc/environment
```

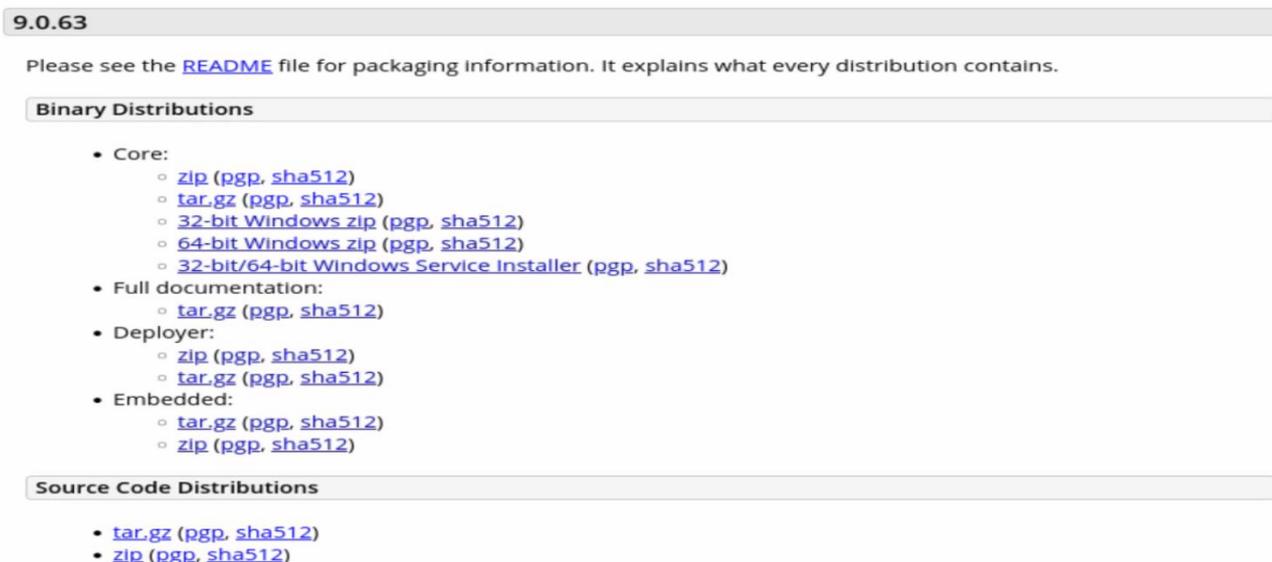


**Figura 16:** Fichero */etc/environment*.

3. El siguiente paso fue añadir al usuario *tomcat* al sistema para operar el servicio, creando un grupo con el mismo nombre, estableciéndole como directorio de usuario */opt/tomcat*, copiando el esqueleto de su directorio a partir de */dev/null* y con el *shell /bin/false*:

```
-useradd -m -U -k /dev/null -d /opt/tomcat -s /bin/false tomcat
```

4. Tras ello, se descargó Apache Tomcat desde el sitio web indicado en [41] seleccionando la opción *tar.gz* del subapartado “*Core*” de la versión 9 correspondiente.



**Figura 17:** Localización del fichero de Apache Tomcat. [41]

5. Una vez descargado el fichero empaquetado, se desempaquetó y se ubicó en el directorio del usuario *tomcat (/opt/tomcat)*:

```
-tar xf apache-tomcat-9.0.63.tar.gz -C /opt/tomcat/
```

6. Además, se creó un enlace simbólico del directorio extraído para poseer un nombre uniforme del directorio independientemente de la versión del software:

```
-ln -s /opt/tomcat/apache-tomcat-9.0.63/ /opt/tomcat/apache-tomcat
```

7. El siguiente paso fue establecer como propietario al usuario *tomcat* del directorio extraído y de todos los ficheros contenidos en el:

```
-chown -R tomcat: /opt/tomcat/apache-tomcat/
```

8. Tras esto, se añadió el servicio *tomcat9* a *Systemd*, creando para ello un fichero (*tomcat9.service*), con la siguiente orden, cuyo contenido se muestra a continuación de dicha orden:

```
-gedit /etc/systemd/system/tomcat9.service
```

```
[Unit]
```

```
Description=Tomcat 9.0 servlet container para CentOS 7
```

```
After=network.target
```

```
[Service]
```

```
Type=forking
```

```
User=tomcat
```

```
Group=tomcat
```

```
Environment="JAVA_OPTS=-Djava.security.egd=file:///dev/urandom"
```

```
Environment="CATALINA_BASE=/opt/tomcat/apache-tomcat"
```

```
Environment="CATALINA_HOME=/opt/tomcat/apache-tomcat"
```

```
Environment="CATALINA_PID=/opt/tomcat/apache-tomcat/temp/tomcat.pid"
```

```
Environment="CATALINA_OPTS=-Xms512M -Xmx1024M -server -XX:+UseParallelGC"
```

```
ExecStart=/opt/tomcat/apache-tomcat/bin/startup.sh
```

```
ExecStop=/opt/tomcat/apache-tomcat/bin/shutdown.sh
```

```
[Install]
```

```
WantedBy=multi-user.target
```

9. Posteriormente, se inició y habilitó en el arranque el servicio *tomcat9* y se añadió una regla al cortafuegos para permitir la comunicación por el puerto 8080, debido a que es el que usa por defecto el *Servlet*:

```
-systemctl start tomcat9
```

```
-systemctl enable tomcat9
```

```
-firewall-cmd --permanent --add-port=8080/tcp
```

```
-firewall-cmd --reload
```

10. A continuación, se comentó el bloque de la siguiente figura en los archivos XML (*eXtensible Markup Language*) indicados en las órdenes, con el objetivo de permitir el acceso desde la red a la administración del *Servlet*:

```
-gedit /opt/tomcat/apache-tomcat/webapps/manager/META-INF/context.xml
-gedit /opt/tomcat/apache-tomcat/webapps/host-manager/META-INF/context.xml
```

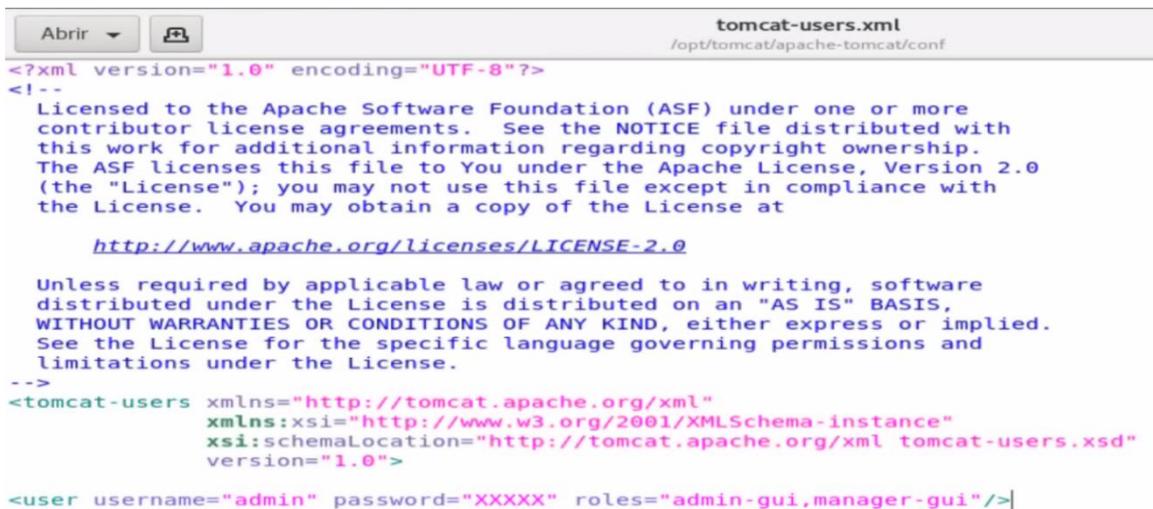
```
<Context antiResourceLocking="false" privileged="true" >
  <CookieProcessor className="org.apache.tomcat.util.http.Rfc6265CookieProcessor"
    sameSiteCookies="strict" />

  <!--<Valve className="org.apache.catalina.valves.RemoteAddrValve"
    allow="127\.\d+\.\d+\.\d+|::1|0:0:0:0:0:0:0:1" />
  -->
  <Manager sessionAttributeValueClassNameFilter="java\.lang\.(?:Boolean|Integer|Long|Number|string)|
org\.apache\.catalina\.filters\.CsrfPreventionFilter\$LruCache(?:\$1)?|java\.util\.(?:Linked)?HashMap"/>
</Context>
```

**Figura 18:** Bloque comentado (Apache Tomcat).

11. Adicionalmente, se añadió la siguiente línea al fichero XML de usuarios de Apache Tomcat (*tomcat-users.xml*), estableciendo el nombre de usuario y la contraseña, para crear un usuario con roles suficientes para gestionar las aplicaciones web administrativas que se incluyen en la instalación del *Servlet*:

```
-gedit /opt/tomcat/apache-tomcat/conf/tomcat-users.xml
<user username="nombre" password="XXXXXXXX" roles="admin-gui,manager-gui"/>
```



```
tomcat-users.xml
/opt/tomcat/apache-tomcat/conf

<?xml version="1.0" encoding="UTF-8"?>
<!--
Licensed to the Apache Software Foundation (ASF) under one or more
contributor license agreements. See the NOTICE file distributed with
this work for additional information regarding copyright ownership.
The ASF licenses this file to You under the Apache License, Version 2.0
(the "License"); you may not use this file except in compliance with
the License. You may obtain a copy of the License at

    http://www.apache.org/licenses/LICENSE-2.0

Unless required by applicable law or agreed to in writing, software
distributed under the License is distributed on an "AS IS" BASIS,
WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
See the License for the specific language governing permissions and
limitations under the License.
-->
<tomcat-users xmlns="http://tomcat.apache.org/xml"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:schemaLocation="http://tomcat.apache.org/xml tomcat-users.xsd"
  version="1.0">

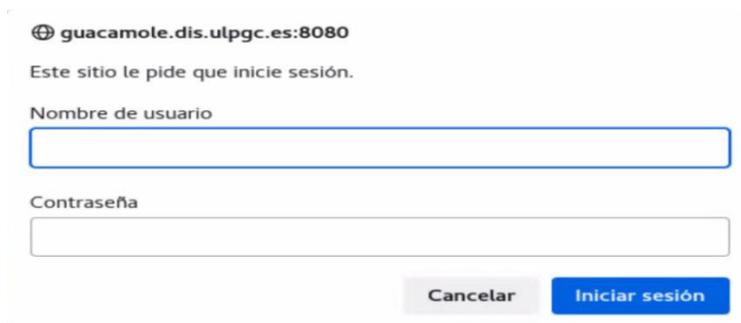
  <user username="admin" password="XXXXX" roles="admin-gui,manager-gui"/>
```

**Figura 19:** Creación de usuario administrador de las aplicaciones de Apache Tomcat.

12. Finalmente, se pudo acceder, por vía del navegador, haciendo uso del FQDN (*Full Qualified Domain Name*) establecido en la sección “4.2.1.-Diseño de la Infraestructura”, así como de la dirección IP de la máquina, a *http://FQDN:8080/* y *http://IP:8080/* respectivamente, donde se observó la página principal de Apache Tomcat, en donde se hallaron las aplicaciones administrativas cuya autenticación se estableció en el paso anterior. En las siguientes figuras se puede observar dicha página principal de Apache Tomcat, así como la solicitud de credenciales que se realiza al acceder a una de las aplicaciones administrativas (“*Host Manager*”) y la página que se muestra tras la correcta autenticación.



**Figura 20:** Acceso por vía del navegador a la interfaz web de Apache Tomcat.



**Figura 21:** Solicitud de autenticación para usar la aplicación administrativa “*Host Manager*”.



**Figura 22:** Página de la aplicación administrativa “*Host Manager*” de Apache Tomcat.

### 2.4.2.2.- Instalación del servidor de Apache Guacamole

Tras instalar el *Servlet Apache Tomcat*, se procedió a construir y compilar desde el código fuente la denominada parte “*servidor*” de Apache Guacamole (*guacamole-server*) haciendo uso de la guía indicada en [42].

1. En primer lugar, se actualizó el sistema:

-yum update

2. Una vez hecho lo anterior, se instalaron una serie de dependencias, algunas necesarias y otras opcionales, para permitir el correcto funcionamiento de Apache Guacamole. Las dependencias, divididas en requeridas y opcionales, son las siguientes:

#### Requeridas

**-Cairo (paquete cairo-devel):** Lo utiliza la API (*Application Programming Interface*) en C de Apache Guacamole, *libguac*, para el renderizado de gráficos.

**-libjpeg-turbo (paquete libjpeg-turbo-devel):** Lo utiliza *libguac* para proporcionar soporte para JPEG (*Joint Photographic Experts Group*).

**-libpng (paquete libpng-devel):** Lo utiliza *libguac* para crear imágenes PNG (*Portable Network Graphics*), el tipo de imagen principal usado por el protocolo Guacamole.

**-libtool (paquete libtool):** Se utiliza durante el proceso de construcción. Crea librerías compiladas que son necesarias para Apache Guacamole.

**-libuuid (paquete libuuid-devel):** Lo utiliza *libguac* para asignar identificadores internos y únicos a cada usuario y conexión. Estos identificadores son la base para el soporte de conexiones compartidas.

#### Opcionales

**-FFmpeg (paquete ffmpeg-devel; SI se instaló):** Lo emplea la utilidad *guacenc* para codificar retransmisiones de video cuando traduce las grabaciones de las sesiones de Apache Guacamole. La mencionada utilidad no será construida sin esta dependencia.

**-FreeRDP (paquete freerdp-devel; SI se instaló):** Se requiere la versión 2.0.0 o más reciente de esta dependencia para proporcionar soporte para RDP (*Remote Desktop Protocol*).

**-Pango (paquete pango-devel; SI se instaló):** Apache Guacamole hace uso de esta librería de disposición de textos para renderizar texto para los protocolos que requieren de terminal.

**-libssh2 (paquete libssh2-devel; SI se instaló):** Se requiere para proporcionar soporte a los protocolos SSH (*Secure Shell*) y SFTP (*Secure File Transfer Protocol*).

**-libtelnet (paquete libtelnet-devel; SI se instaló):** Se requiere para proporcionar soporte al protocolo Telnet.

**-libVNCServer (paquete libvncserver-devel; SI se instaló):** Proporciona la librería *libvncclient*, requerido para el soporte de VNC (*Virtual Network Computing*).

**-libwebsockets (paquete libwebsockets-devel; SI se instaló):** Se requiere para proporcionar soporte para Kubernetes.

**-PulseAudio (paquete pulseaudio-libs-server; SI se instaló):** proporciona la librería *libpulse*, utilizada por el soporte para VNC de Guacamole para proporcionar audio experimental.

**-OpenSSL (paquete openssl-devel; SI se instaló):** Se requiere para proporcionar soporte para SSL (*Secure Sockets Layer*) y TLS (*Transport Layer Security*), esquemas de cifrado usados normalmente para el tráfico web. También es necesaria para proporcionar soporte SSH.

**-libvorbis (paquete libvorbis-devel; SI se instaló):** Se requiere para proporcionar soporte para *Ogg Vorbis*, un estándar gratuito para la compresión de sonido.

**-libwebp (paquete libwebp-devel; SI se instaló):** Se requiere para proporcionar soporte para crear imágenes *WebP*.

Para instalar las dependencias, se ejecutó la siguiente orden:

```
-yum install cairo-devel libjpeg-turbo-devel libpng-devel libtool libuuid-devel
freerdp-devel pango-devel libssh2-devel libvncserver-devel pulseaudio-libs-devel
openssl-devel libvorbis-devel libwebp-devel
```

Como se puede observar, en la orden no se encuentran las dependencias *ffmpeg-devel*, *libwebsockets-devel* y *libtelnet-devel*. Esto es debido a que los repositorios base de CentOS7 no poseen paquetes para ellas o requieren de un fichero con extensión *.rpm* determinado para su instalación. Por ello, se instalaron a continuación los repositorios EPEL (*Extra Packages for Enterprise Linux*) y se instaló de manera local un fichero con extensión *.rpm*, con el objetivo de permitir la correcta instalación de estas dependencias. Para todo lo comentado, se ejecutaron las siguientes órdenes:

```
-yum install epel-release
-yum install libtelnet-devel
-yum install libwebsockets-devel
-yum localinstall --nogpgcheck https://download1.rpmfusion.org/free/el/rpmfusion-free-release-7.noarch.rpm
-yum install ffmpeg-devel
```

Además, se instaló la siguiente dependencia para que posteriormente un script incluido en el paquete *guacamole-server* detectara correctamente que podía proporcionar soporte para VNC (*Virtual Network Computing*), ya que, en caso contrario, no lo proporcionaba:

```
-yum install libgcrypt-devel
```

```
-----
libvncserver appears to be built against
libgcrypt, but the libgcrypt headers
could not be found. VNC will be disabled.
-----
```

**Figura 23:** Mensaje de información sobre la ausencia de dependencias para el soporte de VNC.

3. Una vez instaladas las dependencias, se descargó desde [43] el fichero *guacamole-server-1.4.0.tar.gz*. Se desempaquetó con la siguiente orden y se accedió al directorio extraído:

```
-tar -xzf guacamole-server-1.4.0.tar.gz
-cd guacamole-server-1.4.0/
```

4. En este directorio se ejecutó el script mencionado, denominado *configure*, añadiendo el directorio de inicio, tal y como se muestra en la siguiente orden. Este script comprobó las dependencias instaladas en el sistema e indicó al final de la ejecución qué funcionalidades podía implementar la instalación de Apache Guacamole. Se comprobó que las dependencias obligatorias y opcionales instaladas aparecían con un “yes” a su lado. La única dependencia que aparecía con un “no” se indicaba de esta forma debido a que es *wsock32*, una librería del sistema operativo Windows:

```
./configure --with-init-dir=/etc/init.d
```

```

-----
guacamole-server version 1.4.0
-----

Library status:

freerdp2 ..... yes
pango ..... yes
libavcodec ..... yes
libavformat ..... yes
libavutil ..... yes
libssh2 ..... yes
libssl ..... yes
libswscale ..... yes
libtelnet ..... yes
libVNCServer ..... yes
libvorbis ..... yes
libpulse ..... yes
libwebsockets ..... yes
libwebp ..... yes
wsock32 ..... no

Protocol support:

Kubernetes .... yes
RDP ..... yes
SSH ..... yes
Telnet ..... yes
VNC ..... yes

Services / tools:

guacd ..... yes
guacenc ..... yes
guaclog ..... yes

FreeRDP plugins: /usr/lib64/freerdp2
Init scripts: /etc/init.d

```

**Figura 24:** Resultado de la ejecución del script *configure*.

5. Por último, se ejecutaron las tres órdenes siguientes para construir y compilar *guacamole-server* y para actualizar la memoria caché del sistema con las nuevas librerías:

```

-make
-make install
-ldconfig

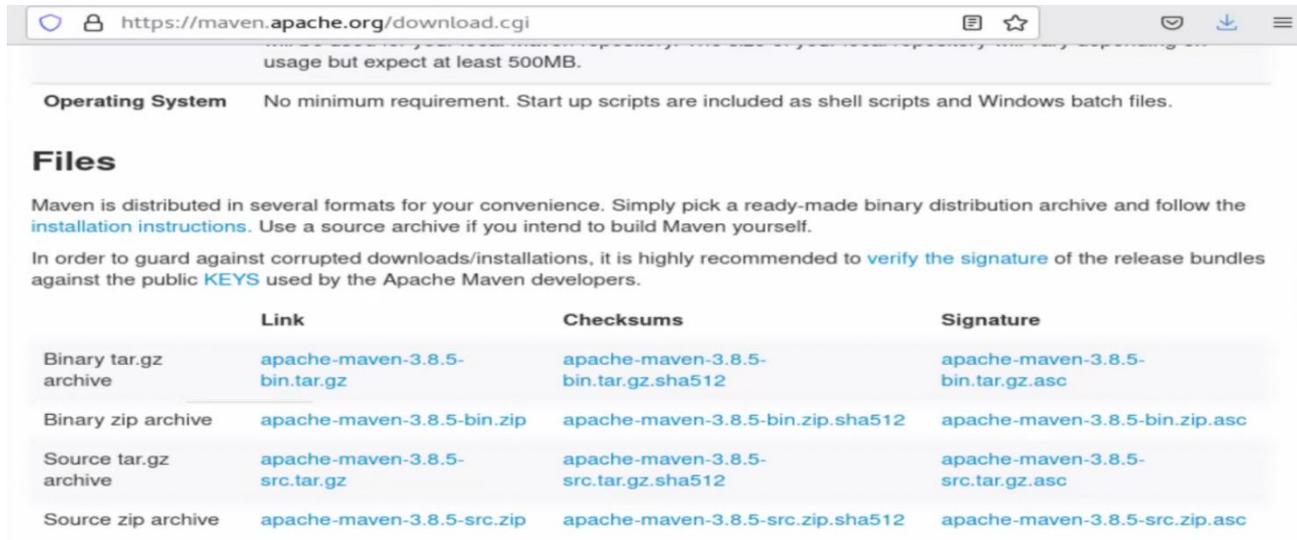
```

### 2.4.2.3.- Instalación del cliente de Apache Guacamole

Para finalizar la instalación del servicio Apache Guacamole, se realizó la instalación de su denominada parte “cliente” (*guacamole-client*) a partir de la misma guía usada para la parte servidor. Se siguió el mismo procedimiento de construcción y compilación, para el que se necesitó Apache Maven [44]. A pesar de que existe otra forma de realizar la instalación de la parte cliente, se decidió seguir la indicada debido a posibles líneas futuras del proyecto pues, sistemas de autenticación como RADIUS (*Remote Authentication Dial in User Service*), no poseen una extensión disponible en el sitio web de Guacamole debido a restricciones de licencias y, por ello, para su implementación se necesita construir la parte cliente incluyéndola en el proceso.

Los pasos que se realizaron para esta metodología de instalación (construyendo y compilando el código fuente) son los siguientes [42, 45]:

1. En primer lugar se descargó Apache Maven desde el sitio web indicado en [46]. El archivo se denomina *apache-maven-3.8.5-bin.tar.gz*. Se realizó de esta forma pues, si se instalaba desde línea de comandos, la versión era antigua (procede de los repositorios de CentOS7) y podía ocasionar conflictos con algunos *plugins* que se instalaron posteriormente en la construcción.



**Figura 25:** Ubicación del fichero de Apache Maven. [46]

2. Una vez descargado el fichero, se desempaquetó en el directorio */opt* y se creó un enlace simbólico para omitir la versión del software en el propio nombre del fichero:

```
-tar xf apache-maven-3.8.5-bin.tar.gz -C /opt
-ln -s /opt/apache-maven-3.8.5/ /opt/maven
```

3. Tras esto, se estableció las variables de entorno necesarias para Apache Maven creando un fichero denominado *maven.sh* en la ruta indicada en la siguiente orden y añadiendo en él lo que se muestra a continuación de dicha orden:

```
-gedit /etc/profile.d/maven.sh
export JAVA_HOME=/etc/alternatives/jre
export M2_HOME=/opt/maven
export MAVEN_HOME=/opt/maven
export PATH=${M2_HOME}/bin:${PATH}
```



**Figura 26:** Fichero */etc/profile.d/maven.sh*.

4. A continuación, se proporcionaron permisos al fichero para que fuera ejecutable, se cargaron las variables de entorno con la orden *source* y se comprobó la versión de Apache Maven:

```
-chmod +x /etc/profile.d/maven.sh
-source /etc/profile.d/maven.sh
-mvn -version
```

```
[root@guacamole Descargas]# gedit /etc/profile.d/maven.sh
[root@guacamole Descargas]# chmod +x /etc/profile.d/maven.sh
[root@guacamole Descargas]# source /etc/profile.d/maven.sh
[root@guacamole Descargas]# mvn -version
Apache Maven 3.8.5 (3599d3414f046de2324203b78ddcf9b5e4388aa0)
Maven home: /opt/maven
Java version: 11.0.15, vendor: Red Hat, Inc., runtime: /usr/lib/jvm/java-11-openjdk-11.0.15.0.9-2.el7_9.x86_64
Default locale: es_ES, platform encoding: UTF-8
OS name: "linux", version: "3.10.0-1160.62.1.el7.x86_64", arch: "amd64", family: "unix"
[root@guacamole Descargas]# █
```

**Figura 27:** Versión de Maven.

5. Tras instalar Apache Maven, se descargó desde [43] el fichero *guacamole-client-1.4.0.tar.gz*. Se desempaquetó con la siguiente orden y se accedió al directorio extraído:

```
-tar -xzf guacamole-client-1.4.0.tar.gz
-cd guacamole-client-1.4.0/
```

6. El siguiente paso fue ejecutar la siguiente orden para que Apache Maven construyera y empaquetara todos los componentes, produciendo un fichero con extensión *.war* que contenía la aplicación web completa. El argumento de la orden (*-Plgpl-extensions*) sirvió para crear la extensión correspondiente al sistema de autenticación RADIUS, por si se deseara hacer uso de este de cara a líneas futuras de este proyecto:

```
-mvn clean package -Plgpl-extensions
```

```
[INFO] -----
[INFO] Reactor Summary for guacamole-client 1.4.0:
[INFO]
[INFO] guacamole-client ..... SUCCESS [ 57.415 s]
[INFO] guacamole-common ..... SUCCESS [ 39.084 s]
[INFO] guacamole-ext ..... SUCCESS [ 20.127 s]
[INFO] guacamole-common-js ..... SUCCESS [01:01 min]
[INFO] guacamole ..... SUCCESS [01:37 min]
[INFO] extensions ..... SUCCESS [ 0.253 s]
[INFO] guacamole-auth-duo ..... SUCCESS [ 12.207 s]
[INFO] guacamole-auth-header ..... SUCCESS [ 6.572 s]
[INFO] guacamole-auth-jdbc ..... SUCCESS [ 0.196 s]
[INFO] guacamole-auth-jdbc-base ..... SUCCESS [ 12.914 s]
[INFO] guacamole-auth-jdbc-mysql ..... SUCCESS [ 10.583 s]
[INFO] guacamole-auth-jdbc-postgresql ..... SUCCESS [ 12.066 s]
[INFO] guacamole-auth-jdbc-sqlserver ..... SUCCESS [ 10.028 s]
[INFO] guacamole-auth-jdbc-dist ..... SUCCESS [ 10.957 s]
[INFO] guacamole-auth-json ..... SUCCESS [ 17.622 s]
[INFO] guacamole-auth-ldap ..... SUCCESS [ 19.539 s]
[INFO] guacamole-auth-quickconnect ..... SUCCESS [ 12.493 s]
[INFO] guacamole-auth-sso ..... SUCCESS [ 0.181 s]
[INFO] guacamole-auth-sso-base ..... SUCCESS [ 6.233 s]
[INFO] guacamole-auth-sso-cas ..... SUCCESS [ 30.294 s]
[INFO] guacamole-auth-sso-openid ..... SUCCESS [ 10.047 s]
[INFO] guacamole-auth-sso-saml ..... SUCCESS [ 14.756 s]
[INFO] guacamole-auth-sso-dist ..... SUCCESS [ 17.406 s]
[INFO] guacamole-auth-totp ..... SUCCESS [ 13.309 s]
[INFO] guacamole-auth-radius ..... SUCCESS [ 27.237 s]
[INFO] guacamole-example ..... SUCCESS [ 8.313 s]
[INFO] guacamole-playback-example ..... SUCCESS [ 3.188 s]
[INFO] -----
[INFO] BUILD SUCCESS
[INFO] -----
[INFO] Total time: 08:52 min
[INFO] Finished at: 2022-05-18T12:25:57+01:00
[INFO] -----
[root@guacamole guacamole-client-1.4.0]# █
```

**Figura 28:** Resultado de la construcción y compilación realizada por Apache Maven.

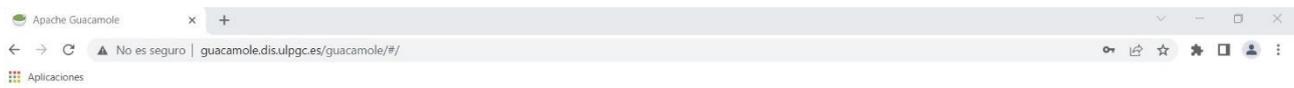
7. El fichero con extensión `.war` que contiene la aplicación web completa que se obtuvo se encontraba ubicado en el subdirectorio `guacamole/target` del directorio donde se ejecutó la orden anterior. Lo que se realizó a continuación fue copiar este archivo renombrándolo, para omitir el nombre de la versión, al directorio del *Servlet* destinado al almacenamiento de las aplicaciones web (*/webapps*):

```
-cp guacamole-1.4.0.war /opt/tomcat/apache-tomcat/webapps/guacamole.war
```

8. Tras esto, se reinició el servicio `tomcat9` y se inició el servicio `guacd` haciendo uso de las órdenes siguientes:

```
-systemctl restart tomcat9  
-/etc/init.d/guacd start
```

9. Finalmente, se comprobó la disponibilidad de la nueva aplicación web accediendo al *Servlet* por vía del navegador y, específicamente, a la ruta indicada en este destinada a la aplicación web de Apache Guacamole.



**Figura 29:** Aplicación web de Apache Guacamole.

### 2.4.3.- Configuración inicial del servicio

Una vez completada la instalación del servicio, se procedió a realizar una serie de tareas de configuración de usuarios y conexiones para permitir cumplir, de forma básica, con el objetivo de la tecnología: proporcionar acceso remoto a sistemas informáticos para los usuarios.

Para ello, se describirán a continuación, haciendo uso del sitio web de Apache Guacamole y su página dedicada a configurar el servicio [47], todos aquellos elementos cuya existencia y razón de ser es necesario conocer para configurar correctamente Apache Guacamole. Concretamente, se hablará de su directorio *GUACAMOLE\_HOME*, de sus ficheros de configuración *guacamole.properties* y *guacd.conf*, del módulo que posee Guacamole para la autenticación por defecto y de cómo se configuran conexiones y usuarios mínimamente para permitir hacer uso del soporte para los protocolos VNC (*Virtual Network Computing*) y RDP (*Remote Desktop Protocol*). Posteriormente, se detallará la configuración realizada en este proyecto haciendo uso de estos elementos.

#### 2.4.3.1.- Directorio *GUACAMOLE\_HOME* (/etc/guacamole)

*GUACAMOLE\_HOME* es el nombre del directorio de configuración de Apache Guacamole y se encuentra ubicado en el directorio */etc/guacamole* por defecto. Todos los ficheros de configuración, extensiones y otros componentes residen en este directorio. La estructura de este directorio consiste en los siguientes ficheros opcionales:

**-guacamole.properties:** Es el fichero principal de configuración de Apache Guacamole. Las propiedades establecidas en este fichero indican como Apache Guacamole se conecta a *guacd* y configuran el comportamiento de las extensiones de autenticación instaladas.

**-guacd.conf:** Es el fichero de configuración para el demonio *guacd* de Apache Guacamole.

**-logback.xml:** Apache Guacamole usa un sistema de registro de actividades del sistema (*logs*) denominado *Logback*. Por defecto, Apache Guacamole muestra los mensajes de log en la consola, pero se puede cambiar este comportamiento proporcionando este fichero.

**-extensions/:** Directorio donde se ubican todas las extensiones instaladas para Apache Guacamole. Se cargarán automáticamente desde aquí todos los ficheros *.jar* presentes en el arranque del servicio.

**-lib/:** Directorio de búsqueda de aquellas librerías requeridas por las extensiones de Apache Guacamole. Se marcarán como disponibles para todas las extensiones todos los ficheros con extensión *.jar*.

La localización del directorio *GUACAMOLE\_HOME* se puede sobrescribir siguiendo tres métodos:

- 1.Creando un directorio denominado *.guacamole* en el directorio *home* del usuario que ejecuta el *Servlet*.
- 2.Especificando en la variable de entorno *GUACAMOLE\_HOME* la ruta completa de un directorio alternativo.
- 3.Especificando en la propiedad del sistema *guacamole.home* la ruta completa de un directorio alternativo.

### 2.4.3.2.- Fichero *guacamole.properties*

*guacamole.properties* es un fichero de configuración opcional usado por la aplicación web de Apache Guacamole. Se usa para configurar Apache Guacamole en situaciones cuando las opciones por defecto son insuficientes o para proporcionar información de configuración adicional para las extensiones. Posee una serie de propiedades:

**-api-session-timeout:** Indica la cantidad de tiempo en minutos máxima permitida para que las sesiones de Apache Guacamole (en concreto sus *tokens* de autenticación) sean válidas a pesar de la inactividad. Si se omite, las sesiones expirarán tras 60 minutos de inactividad.

**-api-max-request-size:** Indica el número máximo de bytes a aceptar en el cuerpo de cualquier solicitud HTTP (*Hypertext Transfer Protocol*). Si se indica 0, significa que no se debe aplicar ningún límite a los bytes recibidos. Si se omite, las solicitudes se limitarán a 2MB por defecto. Este límite no se aplica a las subidas de ficheros.

**-allowed-languages:** Indica, mediante una lista de abreviaturas de idioma separadas por coma, los idiomas a elegir en la interfaz de Apache Guacamole.

**-enable-environment-properties:** Si se establece esta propiedad a *"true"*, Apache Guacamole evaluará primero su entorno para obtener el valor de cualquier propiedad de configuración dada, anteponiendo esta operación a usar valores especificados en el fichero *guacamole.properties* o a usar los valores por defecto.

**-extension-priority:** Indica, mediante una lista de nombres de extensiones separada por comas, qué extensiones deben ser cargadas en un orden específico.

**-guacd-hostname:** La interfaz del anfitrión (*host*) por la que el demonio proxy (*guacd*) escuchará. Si se omite, esta interfaz será *localhost*.

**-guacd-port:** El puerto por el que el demonio proxy (*guacd*) escuchará. Si se omite, este puerto será el 4822.

**-guacd-ssl:** Si se establece esta propiedad a “*true*”, Apache Guacamole requerirá de cifrado SSL(*Secure Sockets Layer*)/TLS(*Transport Layer Security*). Se necesita de un certificado SSL y una clave privada. Si se habilita esta opción, *guacd* debe ser configurado mediante la línea de comandos para usar SSL. Por defecto, la comunicación entre la aplicación web y *guacd* es no cifrada.

**-skip-if-unavailable:** Indica, mediante una lista de identificadores de proveedores de sistemas de autenticación separada por comas, los sistemas de autenticación permitidos para fallar internamente sin abortar el proceso de autenticación. Por defecto, Apache Guacamole aborta el proceso de autenticación si alguno de los sistemas destinados a este proceso falla.

Una configuración típica de este fichero que indica explícitamente los valores para *guacd-hostname* y *guacd-port* sería la siguiente:

```
# Hostname and port of guacamole proxy
guacd-hostname: localhost
guacd-port:      4822
```

#### 2.4.3.3.- Fichero *guacd.conf*

*guacd.conf* es el fichero de configuración para el demonio proxy *guacd* de Apache Guacamole y se ubica en el directorio */etc/guacamole*. El formato de este fichero es el siguiente:

```
# guacd configuration file
#
[daemon]
pid_file = /var/run/guacd.pid
log_level = info
[server]
bind_host = localhost
bind_port = 4822
#
# The following parameters are valid only if
# guacd was built with SSL support.
#
[ssl]
server_certificate = /etc/ssl/certs/guacd.crt
server_key = /etc/ssl/private/guacd.key
```

Cada sección y propiedad se define de la siguiente manera:

#### [daemon] section

**-pid\_file:** Es el nombre del fichero donde el PID (*Process Identifier*) del proceso principal de *guacd* debe ser escrito. Se suele usar para los scripts de inicio, que requieren monitorizar el estado de *guacd*. Si se especifica este parámetro, el usuario que ejecuta *guacd* debe poseer suficientes permisos para crear o modificar el archivo especificado, o el arranque del servicio fallará.

**-log\_level:** El máximo nivel al que *guacd* registrará mensajes de log. Si se omite, el nivel por defecto será *info*. Los valores válidos son: *trace*, *debug*, *info*, *warning* y *error*.

#### [server] section

**-bind\_host:** La interfaz del anfitrión (*host*) a la que *guacd* debe ligarse cuando escucha para conexiones entrantes. Si se omite, *guacd* se ligará a *localhost* y solo las conexiones hacia el propio servidor que sustenta *guacd* tendrán éxito.

**-bind\_port:** El puerto al que *guacd* debe ligarse cuando escucha para conexiones entrantes. Si se omite, el puerto será el 4822.

#### [ssl] section

**-server\_certificate:** El nombre del fichero que contiene el certificado SSL para el cifrado del protocolo Guacamole. Si se especifica, el cifrado SSL se habilitará y la aplicación web de Apache Guacamole requerirá ser configurada para usar SSL en el archivo *guacamole.properties*.

**-server\_key:** El nombre del fichero que contiene la clave privada para el cifrado SSL del protocolo Guacamole. Si se especifica, el cifrado SSL se habilitará y la aplicación web de Apache Guacamole requerirá ser configurada para usar SSL en el archivo *guacamole.properties*.

La configuración de *guacd* puede ser alterada adicionalmente a través de la línea de comandos, teniendo preferencia las configuraciones realizadas por esta vía sobre las establecidas en el fichero de configuración. Para obtener más detalles de las opciones de la orden destinada a tal fin, denominada igualmente *guacd*, ejecutar la orden *man guacd*.

#### 2.4.3.4.- Autenticación por defecto (*user-mapping.xml*)

Por defecto, el módulo de autenticación de Apache Guacamole consiste en una asociación entre nombres de usuario y configuraciones establecidas. Este módulo lee los nombres de usuario y contraseñas a partir de un fichero XML (*Extensible Markup Language*) denominado *user-mapping.xml* y ubicado en *GUACAMOLE\_HOME*. El módulo siempre está habilitado, pero solo lee a partir del fichero XML si este existe. Además, posee la prioridad más baja en actuación con respecto a otros sistemas de autenticación que puedan estar implementados.

Cada usuario se especifica con una etiqueta **<authorize>**. Esta etiqueta contiene todas las conexiones autorizadas para cada usuario concreto, estando cada una representada con la etiqueta **<connection>**. Cada conexión posee un protocolo para el que Apache Guacamole ofrece soporte y una serie de parámetros relacionados con este protocolo. El protocolo se denota con la etiqueta **<protocol>** y sus parámetros con la etiqueta **<param>**.

En la etiqueta **<authorize>** se especifica el nombre de usuario y su contraseña. Dicha contraseña puede ser representada en texto plano o mediante su representación *hash* obtenida tras aplicar el algoritmo MD5 (*Message-Digest 5*). Si se desea añadir un nuevo usuario, bastará con añadir un nuevo bloque con esta etiqueta. Los cambios, una vez guardados, se aplicarán inmediatamente debido a que Apache Guacamole vuelve a leer el fichero sin necesidad de reiniciar el *Servlet*.

Para añadir conexiones a un usuario, los pasos a seguir dependen de dos situaciones. En primer lugar, si el usuario sólo poseerá una conexión, se necesita únicamente especificarla con las etiquetas **<protocol>** y **<param>**. Si el usuario poseerá más de una conexión, se requiere hacer uso de la etiqueta **<connection>** para cada una. Un ejemplo del formato de este fichero es el siguiente:

```

<user-mapping>
  <!-- Per-user authentication and config information -->
  <authorize username="USERNAME" password="PASSWORD">
    <protocol>vnc</protocol>
    <param name="hostname">localhost</param>
    <param name="port">5900</param>
    <param name="password">VNCPASS</param>
  </authorize>

  <!-- Another user, but using md5 to hash the password
  (example below uses the md5 hash of "PASSWORD") -->
  <authorize
    username="USERNAME2"
    password="319f4d26e3c536b5dd871bb2c52e3178"
    encoding="md5">

    <!-- First authorized connection -->
    <connection name="localhost">
      <protocol>vnc</protocol>
      <param name="hostname">localhost</param>
      <param name="port">5901</param>
      <param name="password">VNCPASS</param>
    </connection>

    <!-- Second authorized connection -->
    <connection name="otherhost">
      <protocol>vnc</protocol>
      <param name="hostname">otherhost</param>
      <param name="port">5900</param>
      <param name="password">VNCPASS</param>
    </connection>

  </authorize>
</user-mapping>
    
```

**Figura 30:** Formato del fichero *user-mapping.xml*. [47]

#### 2.4.3.5.- VNC (Virtual Network Computing)

VNC es el primer protocolo para el que Guacamole tuvo soporte [47]. No es tan rápido como RDP, pero varios servidores VNC son adecuados y, además, VNC es más rápido sobre Guacamole que operando sobre sí mismo debido al bajo uso de ancho de banda. El soporte para VNC se proporciona en la librería *libguac-client-vnc* si las dependencias opcionales para este protocolo son instaladas.

Como parámetros de red, VNC posee los siguientes:

**-hostname:** El nombre o dirección IP del sistema servidor al que Apache Guacamole debe conectarse.

**-port:** El puerto en el que el sistema servidor escucha para conexiones de este protocolo, usualmente 5900 o 5900 + un número que indica la conexión (*display number*). Por ejemplo, si el *display number* es 1 (representado como :1), el puerto será 5901.

**-autoretry:** El número de veces que se debe reintentar la conexión antes de abandonarla y retornar un error. Si es una conexión inversa, posibles en VNC si se configura de la forma correspondiente, es el número de intentos permitidos del proceso de conexión.

Con respecto a la autenticación, el estándar VNC define solo la autenticación basada en contraseña. Existen otros mecanismos, pero estos no son estándar. Guacamole proporciona soporte para la autenticación basada en contraseña (**password**) y la autenticación basada en nombre de usuario y contraseña (**username** y **password**).

VNC posee también una serie de parámetros destinados a proporcionar y configurar distintas funcionalidades como son las *display settings* (**color-depth**, **swap-red-blue**, **cursor**, **encodings**, **read-only**, **force-lossless**), las opciones de posibles repetidores VNC que puedan existir (**dest-host**, **dest-port**), las opciones de las conexiones inversas (**reverse-connect**, **listen-timeout**), el soporte de audio con *PulseAudio* (**enable-audio**, **audio-servername**) y la codificación del portapapeles (**clipboard-encoding**).

Una configuración simple, usando el sistema de autenticación por defecto que ofrece Apache Guacamole (*user-mapping.xml*) es:

```
<connection name="Nombre único">
  <protocol>vnc</protocol>
  <param name="hostname"> <<IP | Nombre Host>> </param>
  <param name="port">5901</param>
</connection>
```

Esta conexión, de nombre “*Nombre único*” usará el protocolo VNC para conectarse al anfitrión proporcionado por su dirección IP o nombre de *host* por su puerto 5901.

Para obtener detalles de cómo se instaló un servidor VNC en los equipos para permitir los accesos remotos haciendo uso de este protocolo, ver “2.4.5.-*Instalación de software cliente en los equipos docentes*”.

#### 2.4.3.6.- RDP (*Remote Desktop Protocol*)

RDP es un protocolo más complejo y tiende a ser más rápido que VNC debido al uso de las operaciones de *caching*. El soporte para RDP [47] se proporciona en la librería *libguac-client-rdp* si las dependencias opcionales para este protocolo son instaladas.

Como parámetros de red, RDP posee los siguientes:

**-hostname:** El nombre o dirección IP del sistema servidor al que Apache Guacamole debe conectarse.

**-port:** El puerto en el que el sistema servidor escucha para conexiones de este protocolo. Si no se indica, se establecerá por defecto el puerto 3389 o el puerto por defecto de *VMConnect*, 2179, dependiendo del modo de seguridad seleccionado.

Con respecto a la autenticación, RDP la proporciona a través del uso de un nombre de usuario, una contraseña y un dominio opcional. Además, todas sus conexiones están cifradas. La mayoría de los servidores RDP proporcionan una opción de inicio de sesión de manera gráfica si los parámetros “usuario”, “contraseña” y “dominio” son omitidos.

Una excepción a esto es el uso de NLA (*Network Level Authentication*) [48], que realiza todo el proceso de autenticación externamente a la sesión de escritorio, con la ausencia de una interfaz gráfica. Los servidores de esta índole son gestionados por Guacamole de dos formas.

La primera forma es proporcionar el nombre de usuario y la contraseña internamente en la configuración de la conexión, ya sea haciendo uso de valores estáticos o transfiriendo credenciales de Guacamole con *tokens* y autenticación LDAP (*Lightweight Directory Access Protocol*). Alternativamente, si las credenciales no se establecen en la configuración de la conexión, Guacamole intentará preguntar al usuario por las credenciales de forma interactiva si las versiones de *guacd* y el cliente Guacamole lo permiten. Si ninguno de los componentes lo permite y las credenciales no están configuradas, la autenticación NLA fallará.

Los parámetros de autenticación son los siguientes:

**-username:** Nombre de usuario para la autenticación, si existe. Es un parámetro opcional.

**-password:** Contraseña para la autenticación, si existe. Es un parámetro opcional.

**-domain:** Dominio para la autenticación, si existe. Es un parámetro opcional.

**-security:** El modo de seguridad a usar en la conexión RDP. Este modo indica como se cifrarán los datos y qué tipo de autenticación, si existe, se llevará a cabo. Por defecto, el modo de seguridad se selecciona en base a un proceso de negociación que determina cual soportan tanto cliente como servidor. Los valores posibles para este parámetro son:

*any:* Selecciona el modo de seguridad automáticamente en base a los protocolos de seguridad soportados por el cliente y el servidor. Esta es la opción por defecto.

*nla*: Este modo usa cifrado TLS (*Transport Layer Security*) y requiere que se envíen con antelación el nombre de usuario y la contraseña. A diferencia del modo RDP, la fase de autenticación se realiza antes de que comience la sesión de escritorio remoto, evitando la necesidad de que el servidor Windows asigne un número significativo de recursos para usuarios que puede que no sean autorizados.

*nla-ext*: Este modo hace uso de NLA extendido. Es idéntico a NLA con la excepción de que se requiere que el servidor envíe hacia el cliente un *Early User Authorization Result* inmediatamente después de la operación NLA *handshake*.

*tls*: La autenticación y el cifrado se implementa mediante TLS (*Transport Layer Security*), también referido como RDSTLS. Este modo se utiliza normalmente en una configuración con balanceo de carga en la que el servidor RDP inicial puede redireccionar la conexión a otro servidor RDP.

*vmconnect*: El proceso de negociación que selecciona el modo de seguridad en base a los protocolos de seguridad soportados por el cliente y el servidor se limita a protocolos soportados por *Hyper-VMConnect*.

*rdp*: Es una opción para el cifrado *RDP Legacy*, usado para servidores Windows antiguos o cuando se desea una pantalla de inicio de sesión de Windows. Las nuevas versiones de Windows poseen este modo deshabilitado por defecto y solo aceptarán NLA a excepción de que se configure explícitamente de otra forma.

**-ignore-cert**: Si se establece a “*true*” el certificado devuelto por el servidor será ignorado, incluso si este no puede ser validado. Esta opción es útil si se confía completamente en el servidor y en la conexión, así como que se conoce que el certificado no puede ser validado.

**-disable-auth**: Si se establece a “*true*”, la autenticación llevada a cabo mientras se produce la conexión se deshabilitará. Cualquier otra autenticación forzada por el servidor remoto seguirá habilitada.

RDP posee también una serie de parámetros para funcionalidades adicionales como son la normalización del portapapeles (**normalize-clipboard**), las opciones de sesión (**client-name, console, initial-program, server-layout, timezone**), las opciones de *Display* (**color-depth, width, height, dpi, resize-method, force-lossless**), la redirección de dispositivos (**disable-audio, enable-audio-input, enable-touch, enable-printing, printer-name, enable-drive, disable-download, disable-upload, drive-name, drive-path, create-drive-path, console-audio, static-channels**), entre otros. Todos estos parámetros se pueden observar en la sección “*RDP*” de [47].

Una configuración simple, usando el sistema de autenticación por defecto que ofrece Apache Guacamole (*user-mapping.xml*), es:

```
<connection name="Nombre único">
  <protocol>rdp</protocol>
  <param name="hostname"> <<IP | Nombre Host>> </param>
  <param name="port">3389</param>
</connection>
```

Esta conexión, de nombre “*Nombre único*” usará el protocolo RDP para conectarse al anfitrión, dado por su dirección IP o nombre de *host*, por su puerto 3389.

Para obtener detalles de cómo se habilitaron las conexiones RDP en los equipos para permitir los accesos remotos haciendo uso de este protocolo, ver “2.4.5.-*Instalación de software cliente en los equipos docentes*”.

#### 2.4.3.7.- Configuración inicial realizada

Una vez descritos todos los elementos cuyas características es necesario conocer para la correcta configuración de Apache Guacamole, se procede a continuación a detallar la configuración inicial realizada en este TFT.

Esta configuración se corresponde con aquella realizada una vez instalado el servicio, haciendo uso del módulo de autenticación por defecto (la configuración para el sistema de autenticación y para la base de datos se describirá en el apartado “2.4.4-*Instalación del soporte para la base de datos y para el sistema de autenticación*”).

En primer lugar, se procedió a crear el directorio */etc/guacamole* en el sistema haciendo uso de la orden *mkdir /etc/guacamole*. En este directorio se crearon, a su vez, los subdirectorios *extensions/* y *lib/* que contienen, respectivamente, las extensiones y sus librerías requeridas. Adicionalmente, se estableció una variable de entorno, tal y como se hizo en los pasos para la instalación del servicio, con el objetivo de crear la variable *GUACAMOLE\_HOME* que apunta a la ruta */etc/guacamole*.

```
[root@guacamole ~]# ls /etc/guacamole
extensions  lib
[root@guacamole ~]# echo $GUACAMOLE_HOME
/etc/guacamole
[root@guacamole ~]# █
```

**Figura 31:** Directorio *GUACAMOLE\_HOME* y subdirectorios *extensions/* y *lib/*.

Posteriormente, se creó el fichero *guacd.conf*, destinado a configurar el demonio *guacd*, con el contenido mostrado en la siguiente figura:

```
[root@guacamole ~]# cat /etc/guacamole/guacd.conf
#
# guacd configuration file
#

[daemon]

pid_file = /var/run/guacd.pid
log_level = info

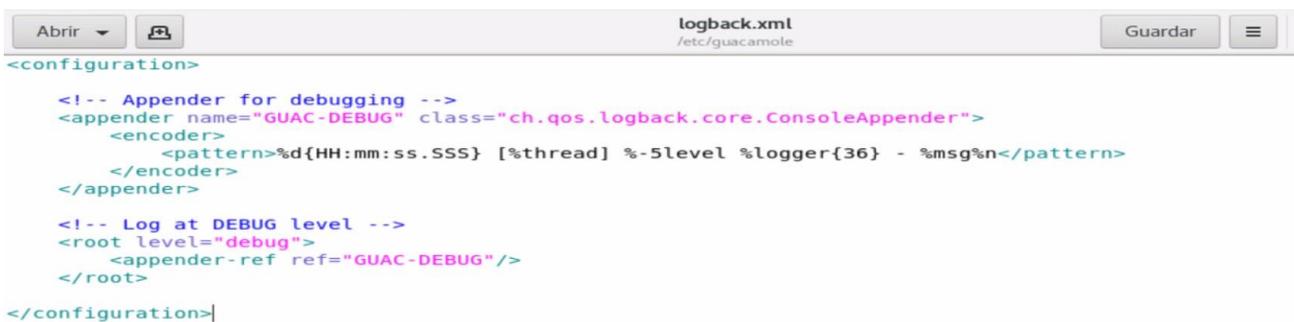
[server]

bind_host = guacamole.dis.ulpgc.es
bind_port = 4822
[root@guacamole ~]#
```

**Figura 32:** Contenido del fichero *guacd.conf*.

Este fichero permite configurar el demonio *guacd* para que el servicio Apache Guacamole se asocie a la interfaz especificada por el nombre de dominio establecido y al puerto 4822 que usa *guacd* por defecto. Además, indica cuál es su fichero de proceso correspondiente y el nivel de *log* a realizar.

Tras esto, se procedió a crear el fichero *logback.xml* con el contenido [47] de la siguiente figura para poder mostrar posibles mensajes de error de la tecnología y así actuar en consecuencia para corregirlos:



```
logback.xml
/etc/guacamole

<configuration>

  <!-- Appender for debugging -->
  <appender name="GUAC-DEBUG" class="ch.qos.logback.core.ConsoleAppender">
    <encoder>
      <pattern>%d{HH:mm:ss.SSS} [%thread] %-5level %logger{36} - %msg%n</pattern>
    </encoder>
  </appender>

  <!-- Log at DEBUG level -->
  <root level="debug">
    <appender-ref ref="GUAC-DEBUG"/>
  </root>

</configuration>
```

**Figura 33:** Contenido del fichero *logback.xml*.

Una vez concluido lo anterior, se creó el fichero *guacamole.properties* con el contenido de la siguiente figura:

```
[root@guacamole ~]# cat /etc/guacamole/guacamole.properties
# Hostname and port of guacamole proxy
guacd-hostname: guacamole.dis.ulpgc.es
guacd-port: 4822
```

**Figura 34:** Contenido del fichero *guacamole.properties*.

Este fichero permite configurar mínimamente Apache Guacamole para que se asocie a la interfaz especificada por el nombre de dominio establecido y al puerto 4822 que usa *guacd* por defecto. Además, configurado de esta forma, Guacamole hace uso del módulo de autenticación de usuarios por defecto.

El paso siguiente fue instalar máquinas virtuales haciendo uso de KVM (ver “2.5.1-Pruebas Iniciales. Instalación de KVM”) para realizar pruebas iniciales e instalar el software cliente (instalación de servidor VNC y habilitación de RDP) en estas (ver “2.4.5-Instalación del software cliente en los equipos docentes”).

A continuación, se creó el fichero *user-mapping.xml* para hacer uso del módulo de autenticación de usuarios por defecto. El contenido de este fichero es el siguiente:

```
[root@guacamole ~]# cat /etc/guacamole/user-mapping.xml
<user-mapping>

  <authorize username="usuario" password="XXX">

    <connection name="EstacionLinux">
      <protocol>vnc</protocol>
      <param name="hostname">192.168.122.114</param>
      <param name="port">5901</param>
    </connection>

    <connection name="EstacionWindows">
      <protocol>rdp</protocol>
      <param name="hostname">192.168.122.239</param>
      <param name="port">3389</param>
      <param name="security">nla</param>
      <param name="ignore-cert">>true</param>
    </connection>

  </authorize>
</user-mapping>
[root@guacamole ~]# □
```

**Figura 35:** Contenido del fichero *user-mapping.xml*.

En este fichero, se creó un usuario al que se le asignó dos conexiones, una por vía del protocolo VNC a la máquina virtual con sistema operativo Linux y otra por vía del protocolo RDP a la máquina virtual con sistema operativo Windows. Para ello, se establecieron las etiquetas de protocolo y dirección IP de las máquinas junto a parámetros específicos de cada conexión en base a dichos protocolos.

Para la conexión VNC, se estableció cuál será el puerto para conexiones con este protocolo teniendo en cuenta el *display number* (ver “2.4.5-Instalación del software cliente en los equipos docentes”).

Con respecto a la conexión RDP, se estableció el puerto 3389 que usa por defecto este protocolo, así como el nivel de seguridad NLA por defecto que viene configurado; y se estableció también que ignorara los certificados del servidor para evitar posibles errores derivados de que este requiriera conexiones cifradas.

Finalmente, se reinició el demonio *guacd* y el *Servlet* ejecutando las órdenes */etc/init.d/guacd restart* y *systemctl restart tomcat9.service* respectivamente, y se procedió a realizar las pruebas iniciales indicadas en “2.5.2-Pruebas Iniciales. Resultados obtenidos en las máquinas virtuales”.

## 2.4.4.- Instalación del soporte para la base de datos y para el sistema de autenticación

La configuración inicial realizada tras instalar el servicio utilizaba el módulo de autenticación de usuarios que posee Apache Guacamole, consistente en un archivo XML (*Extensible Markup Language*) que asociaba usuarios y conexiones a las que tenían permiso acceder.

Sin embargo, hacer uso del sistema de autenticación por defecto no es lo adecuado, pues consiste en un archivo que carece de medidas de seguridad y añadir cada usuario con sus conexiones en él puede convertirse en un trabajo inviable. Por ese motivo, se decidió hacer uso de dos extensiones que proporciona Apache Guacamole en su sitio web para hacer uso de Bases de Datos con el objetivo de almacenar la información de las conexiones de los usuarios y de un sistema CAS (*Central Authentication Service*) para autenticar dichos usuarios. En esta sección de la memoria se detallarán los procedimientos seguidos para habilitar dichas extensiones, así como la configuración de Apache Guacamole para operar correctamente con ellas.

### 2.4.4.1.- Instalación del soporte para la base de datos

Apache Guacamole permite usar Bases de Datos como sistema de autenticación [49], existiendo soporte para *MariaDB*, *MySQL*, *PostgreSQL* y *SQL Server*. Este soporte se proporciona a través de las extensiones disponibles en su sitio web. Usar una base de datos como sistema de autenticación proporciona características adicionales, como la posibilidad de usar grupos de conexiones y la existencia de una interfaz administrativa basada en la aplicación web de Guacamole. Los cambios realizados a usuarios y conexiones se aplican inmediatamente, sin la necesidad de que los usuarios reinicien su sesión para ver las nuevas conexiones.

Mientras que la mayoría de las extensiones de autenticación funcionan independientemente, la autenticación por base de datos puede actuar en un rol subordinado, permitiendo que los usuarios y grupos de usuarios pertenecientes a otras extensiones de autenticación (entre otras, la extensión CAS utilizada) sean asociados con conexiones internas a la base de datos. Los usuarios y grupos de usuarios son considerados idénticos a aquellos que residan en la base de datos si poseen los mismos nombres y el resultado de la autenticación de otra extensión será de confianza si es exitoso. Por tanto, un usuario con una cuenta registrada en varios sistemas de autenticación podrá observar datos de cada sistema cuando inicie sesión.

Para hacer uso de la extensión de autenticación por base de datos, se requiere de:

- Alguna de las bases de datos mencionadas anteriormente.
- Suficientes permisos para crear nuevas bases de datos, para crear nuevos usuarios y para concederles permisos de usuario.
- Acceso red a la base de datos desde el servidor Guacamole.

Para este TFT, se decidió hacer uso de *MariaDB*, debido a su facilidad de uso, popularidad, rapidez, escalabilidad y robustez. Se utilizó para almacenar la información de las conexiones de los usuarios, así como para autentificarlos en primera instancia, hasta que posteriormente esta autenticación se modificó para que se realizara bajo el sistema CAS de la ULPGC. Para implementar este sistema de autenticación por base de datos usando *MariaDB* se siguieron los siguientes pasos:

1. En primer lugar, se descargó desde el sitio web donde se descarga la parte cliente y servidor de Apache Guacamole (indicado en [43]) el fichero *guacamole-auth-jdbc-1.4.0.tar.gz* que se corresponde con la extensión correspondiente de Guacamole. Este fichero contiene, de forma empaquetada, diferentes directorios correspondientes a cada Base de Datos para la que Guacamole proporciona soporte. Cada directorio posee un subdirectorio denominado **schema/** y un fichero con extensión *.jar* (es la propia extensión de Guacamole). Se desempaquetó el fichero *guacamole-auth-jdbc-1.4.0.tar.gz* de la siguiente forma:

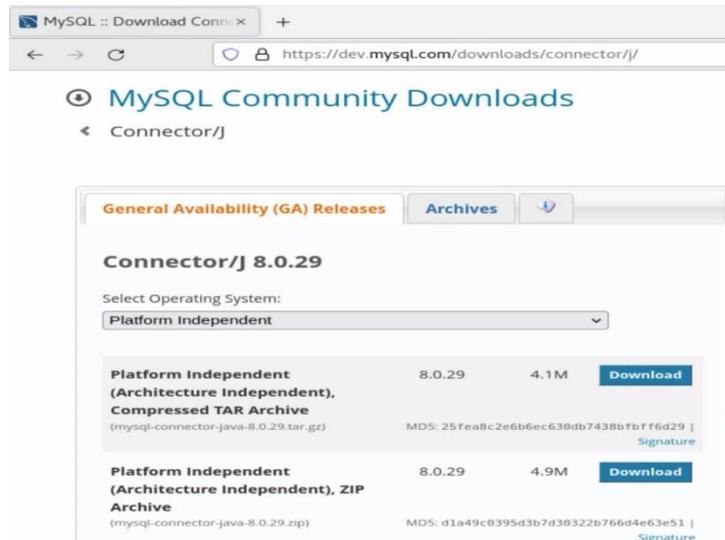
```
-tar -xzf guacamole-auth-jdbc-1.4.0.tar.gz
```

En el caso de *MariaDB*, el directorio **schema/** era *mysql/schema/*, y contenía scripts SQL para inicializar la base de datos, así como un directorio “*upgrade*” destinado a la actualización para distintas versiones de Guacamole. El fichero *.jar* correspondiente era *mysql/guacamole-auth-jdbc-mysql-1.4.0.jar* y fue ubicado en **GUACAMOLE\_HOME/extensions**.

2. Además, se requería de un *driver* JDBC (*Java Database Connectivity*) para que la aplicación web de Guacamole, programada en lenguaje Java, pudiera conectarse a través de este mecanismo a la base de datos. Este driver se obtuvo del sitio web indicado en [50], seleccionando la opción “**Plataforma independiente**”, debido a que CentOS7 no se halla en las opciones del desplegable, y descargando el fichero con extensión *.tar.gz* presente. Posteriormente, se desempaquetó este fichero y se ubicó el archivo con extensión *.jar* en **GUACAMOLE\_HOME/lib**. En la siguiente figura, se puede observar que el JDBC descargado es *mysql-connector-java-8.0.29.jar* y no el correspondiente a *MariaDB*. Esto es debido a que, tras realizar varias pruebas, no se detectaba el JDBC de *MariaDB* por parte de Guacamole, a diferencia del correspondiente a *MySQL*, que funciona debido a que *MariaDB* es una bifurcación de *MySQL* y, por ello, poseen compatibilidades entre sí.

```
[root@guacamole BD]# ls /etc/guacamole/extensions/
guacamole-auth-jdbc-mysql-1.4.0.jar
[root@guacamole BD]# ls /etc/guacamole/lib/
mysql-connector-java-8.0.29.jar
[root@guacamole BD]# █
```

**Figura 36:** Ubicación de la extensión y la librería para el soporte de Base de Datos.



**Figura 37:** Sitio web del conector JDBC. [50]

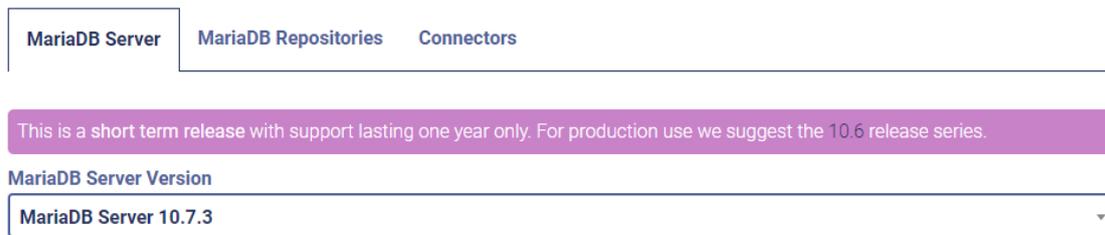
3. A continuación, se creó la base de datos en la que el módulo de autenticación por base de datos de Apache Guacamole almacena los datos de conexiones, así como un usuario requerido para acceder a los datos y manipularlos. Para ello, se instaló previamente *MariaDB* en CentOS7 [51] de la siguiente forma.

3.1. Para comenzar, se actualizó el sistema.

```
-yum update
```

3.2. Tras esto, se añadió un nuevo repositorio para *yum* con el objetivo de instalar la versión 10.6.7, recomendada en [24] por poseer soporte a largo plazo (ver figura a continuación), de *MariaDB*:

```
-gedit /etc/yum.repos.d/mariadb-10.6.repo
[mariadb]
name = MariaDB 10.6 para CentOS 7
baseurl = http://yum.mariadb.org/10.6.7/centos7-amd64
gpgkey=https://yum.mariadb.org/RPM-GPG-KEY-MariaDB
gpgcheck=1
```



**Figura 38:** Recomendación de la versión 10.6 de *MariaDB*.

3.3. Posteriormente, se guardaron los cambios del fichero y se actualizaron las listas de paquetes.

```
-yum update
```

3.4. A continuación, se instaló el servidor de *MariaDB*.

```
-yum install MariaDB-server
```

3.5. Una vez instalado, se inició y habilitó en el arranque del sistema el nuevo servicio.

```
-systemctl start mariadb  
-systemctl enable mariadb
```

3.6. El siguiente paso fue ejecutar el script de seguridad *mariadb-secure-installation*. Se respondió a las preguntas de la forma siguiente: no se modificó la contraseña del usuario *root*, se denegó la autenticación por *socket*, se eliminaron usuarios anónimos, se configuró el acceso del usuario *root* como exclusivamente local, se eliminaron las bases de datos de test y se aplicaron todos los cambios de forma inmediata.

```
-mariadb-secure-installation
```

```
[root@guacamole BD]# mariadb-secure-installation

NOTE: RUNNING ALL PARTS OF THIS SCRIPT IS RECOMMENDED FOR ALL MariaDB
SERVERS IN PRODUCTION USE! PLEASE READ EACH STEP CAREFULLY!

In order to log into MariaDB to secure it, we'll need the current
password for the root user. If you've just installed MariaDB, and
haven't set the root password yet, you should just press enter here.

Enter current password for root (enter for none):
OK, successfully used password, moving on...

Setting the root password or using the unix socket ensures that nobody
can log into the MariaDB root user without the proper authorisation.

You already have your root account protected, so you can safely answer 'n'.

Switch to unix_socket authentication [Y/n] n
... skipping.

You already have your root account protected, so you can safely answer 'n'.

Change the root password? [Y/n] n
... skipping.

By default, a MariaDB installation has an anonymous user, allowing anyone
to log into MariaDB without having to have a user account created for
them. This is intended only for testing, and to make the installation
go a bit smoother. You should remove them before moving into a
production environment.

Remove anonymous users? [Y/n] Y
```

**Figura 39:** Ejecución del script de seguridad de *MariaDB* (I).

```
Normally, root should only be allowed to connect from 'localhost'. This
ensures that someone cannot guess at the root password from the network.

Disallow root login remotely? [Y/n] Y
... Success!

By default, MariaDB comes with a database named 'test' that anyone can
access. This is also intended only for testing, and should be removed
before moving into a production environment.

Remove test database and access to it? [Y/n] Y
- Dropping test database...
... Success!
- Removing privileges on test database...
... Success!

Reloading the privilege tables will ensure that all changes made so far
will take effect immediately.

Reload privilege tables now? [Y/n] Y
... Success!

Cleaning up...

All done! If you've completed all of the above steps, your MariaDB
installation should now be secure.

Thanks for using MariaDB!
[root@guacamole BD]#
```

**Figura 40:** Ejecución del script de seguridad de *MariaDB* (II).

3.7. Tras esto, se estableció el juego de caracteres a la codificación para el idioma español, *utf8mb4*, accediendo al fichero indicado en la siguiente orden y añadiendo la línea mostrada a continuación en la sección *[mysqld]*.

```
-gedit /etc/my.cnf.d/server.cnf
-character_set_server=utf8mb4
```

```
#
# These groups are read by MariaDB server.
# Use it for options that only the server (but not clients) should see
#
# See the examples of server my.cnf files in /usr/share/mysql/
#

# this is read by the standalone daemon and embedded servers
[server]

# this is only for the mysqld standalone daemon
[mysqld]
character set server=utf8mb4
```

**Figura 41:** Establecimiento del juego de caracteres para el idioma Español en *MariaDB*.

3.8. Por último, se reinició el servicio.

```
-systemctl restart mariadb
```

4. Una vez instalado y configurado el nuevo servicio, el siguiente paso fue iniciar sesión en el monitor MySQL como usuario *root* y crear la base de datos de Guacamole. Posteriormente, se ejecutaron los scripts SQL (*Structured Query Language*) *001-create-schema.sql* y *002-create-admin-user.sql* incluidos en el directorio **schema/**. La forma de realizar estos pasos se muestra en la siguiente figura donde se crea una base de datos denominada *guacamole\_db* y se ejecutan los scripts mencionados.

```
[root@guacamole BD]# mysql -u root -p
Enter password:
Welcome to the MariaDB monitor.  Commands end with ; or \g.
Your MariaDB connection id is 3
Server version: 10.6.7-MariaDB MariaDB Server

Copyright (c) 2000, 2018, Oracle, MariaDB Corporation Ab and others.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

MariaDB [(none)]> CREATE DATABASE guacamole_db;
Query OK, 1 row affected (0.002 sec)

MariaDB [(none)]> quit
Bye
[root@guacamole BD]# cd guacamole-auth-jdbc-1.4.0/mysql/
[root@guacamole mysql]# ls schema/
001-create-schema.sql 002-create-admin-user.sql upgrade
[root@guacamole mysql]# cat schema/*.sql | mysql -u root -p guacamole_db
Enter password:
[root@guacamole mysql]# █
```

**Figura 42:** Creación de la base de datos y ejecución de scripts SQL.

5. Finalmente, para que Guacamole fuera capaz de ejecutar consultas en la base de datos, se creó un nuevo usuario y se le concedió suficientes privilegios para gestionar los contenidos de todas las tablas de la base de datos. Este usuario necesitó los permisos *SELECT*, *UPDATE*, *INSERT* y *DELETE* en todas las tablas de Guacamole.

En la siguiente figura, se puede observar un ejemplo obtenido de [49] de las diferentes consultas SQL ejecutadas donde “*guacamole\_user*” y “*some\_password*” se corresponden con las credenciales del usuario creado.

```

$ mysql -u root -p
Enter password:
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 233
Server version: 5.5.29-0ubuntu0.12.10.1 (Ubuntu)

Copyright (c) 2000, 2012, Oracle and/or its affiliates. All rights reserved.

Oracle is a registered trademark of Oracle Corporation and/or its
affiliates. Other names may be trademarks of their respective
owners.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

mysql> CREATE USER 'guacamole_user'@'localhost' IDENTIFIED BY 'some_password';
Query OK, 0 rows affected (0.00 sec)

mysql> GRANT SELECT,INSERT,UPDATE,DELETE ON guacamole_db.* TO 'guacamole_user'@'localhost';
Query OK, 0 rows affected (0.00 sec)

mysql> FLUSH PRIVILEGES;
Query OK, 0 rows affected (0.02 sec)

mysql> quit
Bye
$

```

**Figura 43:** Creación del usuario para Apache Guacamole y concesión de permisos. [49]

#### 2.4.4.2.- Configuración realizada en Guacamole para la Base de Datos

Para configurar Apache Guacamole para usar la extensión de Base de Datos, se añadieron una serie de propiedades al fichero *guacamole.properties*. Estas propiedades son específicas de cada base de datos y las que son absolutamente requeridas son las siguientes:

**-mysql-hostname:** Nombre de host o dirección IP del servidor que sustenta la base de datos.

**-mysql-database:** Nombre de la base de datos creada (*guacamole\_db* en el ejemplo).

**-mysql-username:** Nombre del usuario creado (*guacamole\_user* en el ejemplo).

**-mysql-password:** Contraseña del usuario creado (*some\_password* en el ejemplo).

Una configuración mínima de *guacamole.properties* para realizar conexiones a una base de datos ubicada de manera local es la siguiente:

```

# MySQL properties
mysql-hostname: localhost
mysql-database: guacamole_db
mysql-username: guacamole_user
mysql-password: some_password

```

Existen parámetros opcionales que permiten controlar como Guacamole se conecta al servidor de base de datos (**mysql-port**, **mysql-driver**, **mysql-server-timezone**, **mysql-ssl-mode**, **mysql-ssl-trust-store**, **mysql-ssl-trust-password**, **mysql-ssl-client-store**, **mysql-ssl-client-password** ) que se pueden ver en detalle en la guía usada en este apartado [49] en su sección “*Configuring Guacamole for database authentication*”.

Las propiedades del fichero *guacamole.properties* que se establecieron fueron las siguientes:

```
[root@guacamole ~]# cat /etc/guacamole/guacamole.properties
# Hostname and port of guacamole proxy
guacd-hostname: guacamole.dis.ulpgc.es
guacd-port: 4822

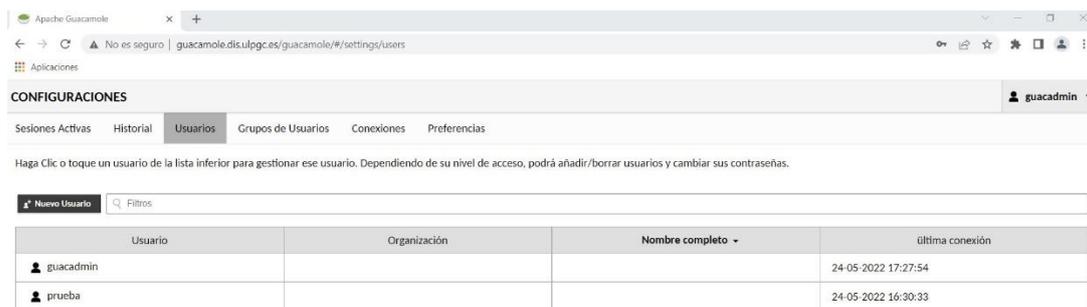
# MySQL properties
mysql-hostname: localhost
mysql-database: guacamole_db
mysql-username: guacamole_user
mysql-password: XXX
```

**Figura 44:** Propiedades para Base de Datos en el fichero *guacamole.properties*.

Tras esto, se reinició el *Servlet* para que Guacamole cargara las nuevas extensiones.

Los script SQL ejecutados crearon un usuario por defecto denominado “*guacadmin*” con una contraseña por defecto. Este usuario se corresponde con el administrador principal y posee todos los permisos para gestionar usuarios y conexiones, entre otras opciones. Tras verificar el correcto inicio de sesión, se modificó su contraseña.

A partir de aquí, se dispuso de una interfaz web dónde se pudo gestionar adecuadamente usuarios y conexiones haciendo uso de la guía “*Administration*” [52] del sitio web de Apache Guacamole.



**Figura 45:** Panel de usuarios de la interfaz web.



**Figura 46:** Panel de conexiones de la interfaz web.

### 2.4.4.3.- Instalación del soporte para el sistema de autenticación CAS

Una vez instalado el servicio y la extensión para Base de Datos, y tras realizar la configuración adecuada, se procedió a instalar el soporte que ofrece Guacamole para el sistema de autenticación CAS (*Central Authentication Service*) con el objetivo de autenticar los usuarios haciendo uso del sistema CAS de la ULPGC [29].

CAS [27] es un proveedor de Inicio de Sesión Único (SSO) [28] que consiste en habilitar a un usuario el acceso a varias aplicaciones con una sola instancia de identificación. Esto permite que múltiples aplicaciones y servicios se autenticuen con el servidor y este envíe las solicitudes de autenticación a un proveedor de autenticación *back-end*.

La extensión permite a Guacamole redireccionar hacia un servidor CAS para servicios de autenticación de usuarios. Este módulo fue ubicado sobre la extensión correspondiente a la Base de Datos debido a que solo proporciona autenticación de usuarios.

Para instalar esta extensión, se siguieron los siguientes pasos [53]:

1. En primer lugar, se descargó desde [43] el fichero *guacamole-auth-sso-1.4.0.tar.gz* que contiene, de forma empaquetada, una serie de directorios con ficheros con extensión *.jar* correspondientes a CAS y otras extensiones SSO. Se desempaquetó el fichero *guacamole-auth-sso-1.4.0.tar.gz* de la siguiente forma:

```
-tar -xzf guacamole-auth-sso-1.4.0.tar.gz
```

2. Tras esto, se ubicó el fichero *guacamole-auth-sso-cas-1.4.0.jar* hallado en el subdirectorio *cas/* en **GUACAMOLE\_HOME/extensions**.

```
[root@guacamole CAS]# ls /etc/guacamole/extensions/  
guacamole-auth-jdbc-mysql-1.4.0.jar  guacamole-auth-sso-cas-1.4.0.jar  
[root@guacamole CAS]#
```

**Figura 47:** Extensiones de Base de Datos y CAS ubicadas en *GUACAMOLE\_HOME/extensions*.

3. Posteriormente, se configuró Guacamole tal y como se indica a continuación y, para finalizar, se reinició el *Servlet* ejecutando la orden *systemctl restart tomcat9.service*.

#### 2.4.4.4.- Configuración realizada en Guacamole para el sistema de autenticación CAS

Para la configuración de Guacamole y la extensión CAS [53], se requirió especificar dos propiedades que describen al servidor CAS y al servidor Guacamole. Esto es debido a que dichas propiedades indican como Guacamole se conecta al servidor CAS y como el servidor CAS debe redirigir a los usuarios hacia Guacamole una vez su identidad ha sido autenticada:

**-cas-authorization-endpoint:** La URL (*Uniform Resource Locator*) del servidor de autenticación CAS.

**-cas-redirect-uri:** El URI (*Uniform Resource Identifier*) para redireccionar tras una autenticación exitosa. Normalmente, se corresponde con la URL completa de la instalación Guacamole.

Existen otras propiedades disponibles para controlar como se procesan los tokens de CAS, así como para configurar si se hace uso de *CAS ClearPass* [54] y como gestionar los grupos de usuarios. Una configuración mínima de Guacamole para hacer uso de CAS es la siguiente:

```
# CAS properties
cas-authorization-endpoint: <<dirección proveedor de autenticación>>
cas-redirect-uri: <<dirección servidor Guacamole>>
```

Guacamole carga las extensiones de autenticación en orden de prioridad y evalúa los intentos de autenticación en el mismo orden. Esto tiene implicaciones en el comportamiento del proceso de inicio de sesión de Guacamole cuando una extensión SSO está presente.

Si dicha extensión tiene prioridad, los usuarios que no han sido autenticados serán redirigidos automáticamente al proveedor de autenticación configurado. No verán la pantalla de inicio de sesión de Guacamole.

Si, por el contrario, la extensión no tiene prioridad, los usuarios que no han sido autenticados verán la pantalla de inicio de sesión de Guacamole y se mostrará un enlace hacia el proveedor de autenticación configurado para que, opcionalmente, los usuarios se autentiquen haciendo uso de dicho proveedor.

Para la configuración realizada, se decidió usar el sistema CAS como medio de autenticación de usuarios, por lo que la extensión correspondiente requería de prioridad. Por ello, se hizo uso de una propiedad descrita en “2.4.3-Configuración inicial del Servicio”, denominada **extension-priority**, con el objetivo de proporcionar prioridad a la extensión CAS sobre la correspondiente a la Base de Datos.

Adicionalmente, se comprobó, junto al personal responsable, cómo funcionaba el servidor CAS de la ULPGC, observando que solo aceptaba que servicios ubicados en equipos de la organización hicieran uso de dicho servidor. Además, solo aceptaba que los reenvíos de peticiones una vez autenticado el usuario se realizaran a destinos por su puerto 80. Por ello, se añadió una regla al cortafuegos para realizar la redirección de peticiones del puerto 80 hacia el 8080 que utiliza el *Servlet*. Las órdenes ejecutadas para este fin fueron las siguientes:

```
-firewall-cmd --permanent --add-forward-port=port=80:proto=tcp:toport=8080  
-firewall-cmd --reload
```

Finalmente, se modificó el fichero *guacamole.properties* para que incluyera, además de lo ya establecido, la propiedad **extension-priority** y las propiedades correspondientes al sistema de autenticación CAS. Esta configuración se puede ver reflejada en la siguiente figura:

```
[root@guacamole ~]# cat /etc/guacamole/guacamole.properties  
# Hostname and port of guacamole proxy  
guacd-hostname: guacamole.dis.ulpgc.es  
guacd-port: 4822  
extension-priority: cas  
  
# MySQL properties  
mysql-hostname: localhost  
mysql-database: guacamole_db  
mysql-username: guacamole_user  
mysql-password: XXX  
  
# CAS properties  
cas-authorization-endpoint: https://identificate.ulpgc.es/cas  
cas-redirect-uri: http://guacamole.dis.ulpgc.es/guacamole  
[root@guacamole ~]#
```

**Figura 48:** Propiedades para CAS en el fichero *guacamole.properties*.

#### 2.4.5.- Instalación del software cliente en los equipos docentes

Para el correcto funcionamiento de las conexiones remotas haciendo uso de Apache Guacamole, se requirió de instalar software en los equipos docentes. Debido a que se configuró Apache Guacamole para permitir las conexiones remotas por vía de los protocolos VNC y RDP, se necesitó instalar servidores VNC en los equipos, así como habilitar las conexiones RDP en los mismos. Por ello, se explicará a continuación como se llevó a cabo dicha instalación de servidores VNC y habilitación de conexiones RDP.

### 2.4.5.1.- Instalación de servidores VNC

Para la instalación de servidores VNC que permitieran responder a las solicitudes del cliente VNC (incluido en el soporte de Apache Guacamole para este protocolo), se acudió a la guía de Apache Guacamole utilizada para la configuración del servicio, indicada en [47], y, tras observar las diferentes recomendaciones de servidores de esta índole presentes en el mercado expuestas en dicha guía, se seleccionó, debido al buen rendimiento experimentado, *TigerVNC* [16].

Para la instalación de esta implementación de VNC se siguió el siguiente procedimiento para cada equipo [55]:

1. En primer lugar, se actualizó el sistema y se instaló el servidor *TigerVNC*:

```
-yum update
-yum install tigervnc-server
```

2. Tras ello, se copió el archivo del servicio para *Systemd* desde la ubicación donde se guarda al instalar el servidor en el paso anterior hacia el directorio de *Systemd*:

```
-cp /usr/lib/systemd/system/vncserver@.service /etc/systemd/system/vncserver@.service
```

3. Una vez hecho lo anterior, se editó el fichero copiado y se modificó donde indica “*USER*” a continuación, añadiendo el nombre de usuario que se utilizó en las conexiones remotas para cada equipo:

```
[Unit]
Description=Remote desktop service (VNC)
After=syslog.target network.target

[Service]
Type=simple
# Clean any existing files in /tmp/.X11-unix environment
ExecStartPre=/bin/sh -c '/usr/bin/vncserver -kill %i > /dev/null 2>&1 || :'
ExecStart=/usr/bin/vncserver_wrapper USER %i
ExecStop=/bin/sh -c '/usr/bin/vncserver -kill %i > /dev/null 2>&1 || :'

[Install]
WantedBy=multi-user.target
```

4. El siguiente paso fue establecer la contraseña con la que el usuario indicado en el paso anterior inicia sesión en el servidor. Para ello, se cambió la sesión actual en el sistema a dicho usuario (primera orden a continuación) y se estableció la contraseña con la segunda orden mostrada:

```
-su usuario  
-vncpasswd
```

5. Tras esto, se recargaron los servicios “*demonio*” del sistema:

```
-systemctl daemon-reload
```

6. A continuación, se añadió al cortafuegos el puerto que se usa en la conexión remota, teniendo en cuenta el factor del *display number*. El *display number* era :1 y, por tanto, el puerto fue el 5901. Una vez hecho esto, se recargó el cortafuegos:

```
-firewall-cmd --permanent --zone=public --add-port=5901/tcp  
-firewall-cmd --reload
```

7. Por último, se inició el servicio indicando tras el “@” el *display number* a utilizar:

```
-systemctl start vncserver@:1
```

#### 2.4.5.2.- Habilitación de las conexiones RDP

Con el objetivo de permitir las conexiones remotas por vía del protocolo RDP, se comprobó si los equipos con sistema operativo *Windows 10* disponían de las versiones *Windows 10 Pro* o *Windows 10 Enterprise* debido a que son las que permiten la habilitación de conexiones remotas a través de dicho protocolo de forma nativa, al incorporar un servidor RDP integrado. Se observó que los equipos disponían de la versión *Windows 10 Enterprise* por lo que no fue necesario instalar ningún servidor RDP de terceros. Para la habilitación de este tipo de conexiones, se accedió a *Configuración-Sistema-Escritorio Remoto* en cada sistema y se habilitó la opción deslizando el botón presente para ello. Además, se dejaron establecidas las opciones de configuración por defecto como la autenticación NLA. Por otro lado, en aquellos equipos que lo necesitaban, se configuró el cortafuegos para permitir las conexiones remotas por el puerto 3389 correspondiente a este protocolo.

## 2.5.- Fase III: Evaluación/Validación/Prueba

Una vez instalado y configurado el servicio junto al sistema de autenticación de usuarios y junto a la base de datos que almacena la información relacionada con las conexiones de dichos usuarios, se procedió a realizar pruebas de funcionamiento de la nueva tecnología. Para ello, se comenzó con pruebas iniciales para las que se instalaron dos máquinas virtuales que actuaron como equipos a los que se permitió el acceso remoto. Como tecnología de virtualización, se seleccionó KVM debido a la experiencia con esta tecnología adquirida en una asignatura del plan de estudios. Tras instalar la tecnología de virtualización y crear las máquinas virtuales, se realizaron diversas pruebas para evaluar el funcionamiento de Apache Guacamole.

Una vez realizadas estas pruebas, se procedió a configurar los equipos docentes siguiendo el mismo procedimiento que para las máquinas virtuales y se realizaron pruebas de funcionamiento para comprobar la correcta funcionalidad de Apache Guacamole.

Las pruebas iniciales, incluyendo la instalación de KVM, y las pruebas finales con los resultados obtenidos se describirán a continuación.

### 2.5.1.- Pruebas Iniciales. Instalación de KVM

KVM es una tecnología de virtualización completa para Linux con hardware x86 que haga uso de las extensiones de virtualización (Intel VT o AMD-V), tal y como se comentó en la sección de Estudio de Tecnologías (ver “2.3.2-Estudio de las tecnologías a utilizar”). Los pasos que se siguieron para instalar KVM fueron los siguientes. [56]

1. En primer lugar, se agregó al fichero `/etc/yum.conf` la directiva “`group_package_types= default, mandatory, optional`”. Con ello, se configuró la orden `yum` para instalar los 3 tipos de paquetes (por defecto, obligatorios, opcionales).

```
[root@guacamole ~]# cat /etc/yum.conf
[main]
cachedir=/var/cache/yum/$basearch/$releasever
keepcache=0
debuglevel=2
logfile=/var/log/yum.log
exactarch=1
obsoletes=1
gpgcheck=1
plugins=1
installonly_limit=5
bugtracker_url=http://bugs.centos.org/set_project.php?project_id=23&ref=http://bugs.centos.org/bug_report_page.php?category=yum
distroverpkg=centos-release
group_package_types=default,mandatory,optional

# This is the default, if you make this bigger yum won't see if the metadata
# is newer on the remote and so you'll "gain" the bandwidth of not having to
# download the new metadata and "pay" for it by yum not having correct
# information.
# It is esp. important, to have correct metadata, for distributions like
# Fedora which don't keep old packages around. If you don't like this checking
# interrupting your command line usage, it's much better to have something
# manually check the metadata once an hour (yum-updatesd will do this).
# metadata_expire=90m

# PUT YOUR REPOS HERE OR IN separate files named file.repo
# in /etc/yum.repos.d
[root@guacamole ~]# █
```

**Figura 49:** Contenido del fichero `/etc/yum.conf`.

2. A continuación, se instalaron con la orden “*yum groupinstall “<<nombre de grupo >>”*” una serie de grupos de paquetes que contienen todas aquellas dependencias que requiere KVM. Dichos grupos de paquetes se muestran a continuación junto a las órdenes de instalación:

```
-yum groupinstall "Virtualization Hypervisor"
-yum groupinstall "Virtualization Client"
-yum groupinstall "Virtualization Platform"
-yum groupinstall "Virtualization Tools"
```

3. El siguiente paso fue reiniciar el sistema con la orden “*reboot*” y, una vez iniciado, comprobar con la orden “*systemctl status libvirtd*” que el demonio correspondiente a la librería de virtualización sobre la que se apoyan los dos gestores de KVM (*virt-manager* y *virsh*), estaba activa e iniciada. Además, se comprobó con la orden “*lsmod | grep kvm*” que los módulos “*kvm*” y “*kvm\_intel*” se encontraban cargados en el núcleo del sistema operativo del host.

```
[root@guacamole ~]# lsmod | grep kvm
kvm_intel      188740  0
kvm            637515  1 kvm_intel
irqbypass     13503   1 kvm
[root@guacamole ~]# systemctl status libvirtd
● libvirtd.service - Virtualization daemon
   Loaded: loaded (/usr/lib/systemd/system/libvirtd.service; enabled; vendor preset: enabled)
   Active: active (running) since mar 2022-05-17 13:18:27 WEST; 1 weeks 0 days ago
     Docs: man:libvirtd(8)
           https://libvirt.org
  Main PID: 1141 (libvirtd)
    Tasks: 20 (limit: 32768)
   CGroup: /system.slice/libvirtd.service
           └─1141 /usr/sbin/libvirtd
             └─1673 /usr/sbin/dnsmasq --conf-file=/var/lib/libvirt/dnsmasq/default.conf --leasefile-ro --dhcp-script=/usr/libexec/libvirt_l...
             └─1677 /usr/sbin/dnsmasq --conf-file=/var/lib/libvirt/dnsmasq/default.conf --leasefile-ro --dhcp-script=/usr/libexec/libvirt_l...
```

**Figura 50:** Comprobación del estado de *libvirtd* y de la carga de módulos en el *kernel*.

Una vez concluido el paso anterior, KVM se encontraba completamente instalado en el sistema. A continuación, se crearon las máquinas virtuales a utilizar para evaluar el funcionamiento de Apache Guacamole. Para ello, se crearon, por vía del gestor *virt-manager*, dos máquinas virtuales, una con sistema operativo Linux distribución CentOS7 y otra con sistema operativo Windows. Las características de dichas máquinas virtuales son las siguientes:

### **Máquina Linux CentOS7: “EstacionLinux”**

**-CPU:** 1

**-RAM:** 1 GB

**-Almacenamiento:** 10 GB

**-Red “default”.** Red NAT (*Network Address Translation*) por defecto que proporciona KVM configurada vía DHCP.

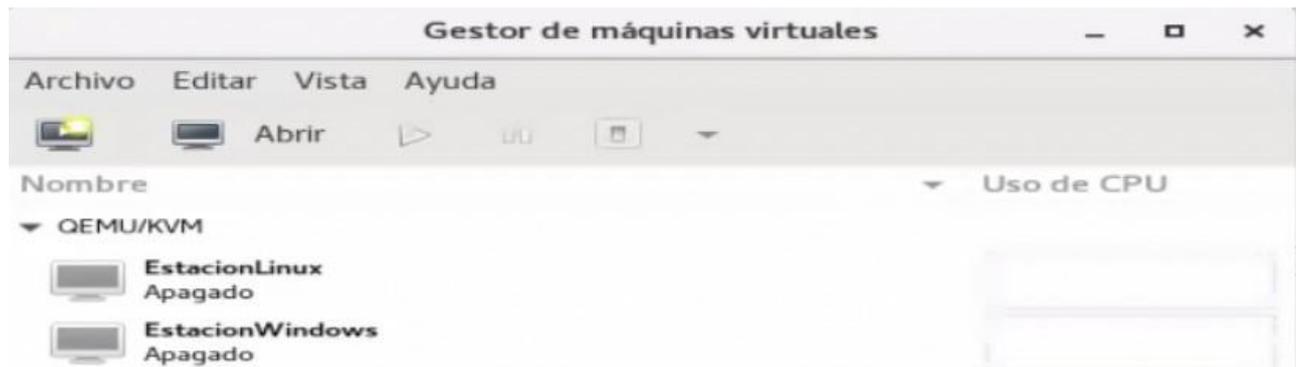
## Máquina Windows: “EstacionWindows”

-CPU: 2

-RAM: 4 GB

-Almacenamiento: 40 GB

-Red “*default*”. Red NAT (*Network Address Translation*) por defecto que proporciona KVM configurada vía DHCP.



**Figura 51:** Máquinas virtuales creadas.

### 2.5.2.- Pruebas Iniciales. Resultados obtenidos en las máquinas virtuales

Tras realizar la instalación de KVM y crear máquinas virtuales con los dos sistemas operativos que poseen los equipos docentes (Windows y Linux), se procedió a crear un usuario en Guacamole, denominado “prueba”, al que se le asignó dos conexiones, una para cada máquina virtual. La primera conexión hizo uso del protocolo VNC para conectarse a la máquina con sistema operativo Linux y la segunda conexión hizo uso del protocolo RDP para conectarse a la máquina con sistema operativo Windows. En las siguientes figuras, se puede observar las características de las conexiones (parámetros de red) del usuario creado a través de la interfaz web de Guacamole.

**PARÁMETROS**

**Red**

Nombre de Host:

Puerto:

**Autenticación**

Usuario:

Contraseña:

**Figura 52:** Características de la conexión “Estación Linux”.

**PARÁMETROS**

**Red**

Nombre de Host:

Puerto:

**Autenticación**

Usuario:

Contraseña:

Dominio:

Modo seguridad:

Desactivar autenticación:

Ignorar certificado del servidor:

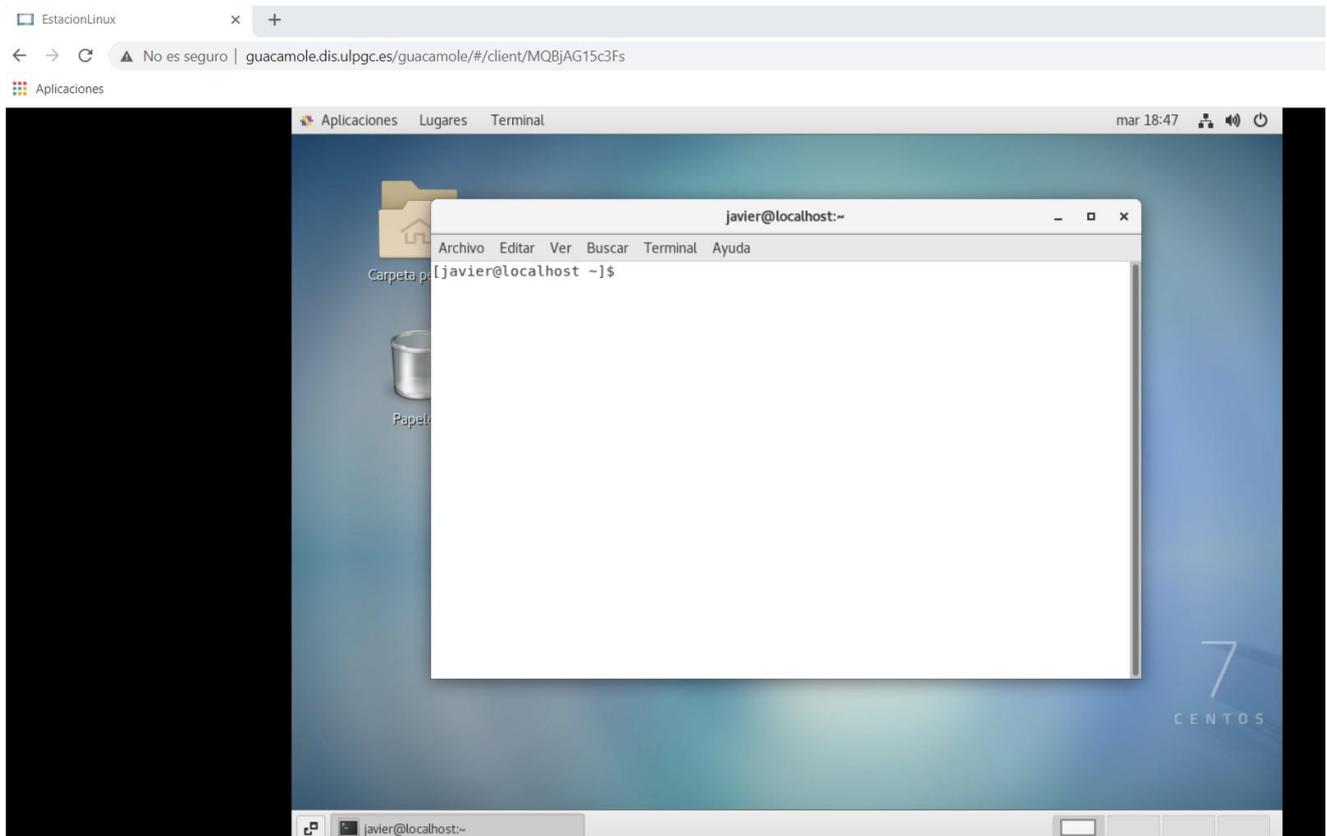
**Figura 53:** Características de la conexión “Estación Windows”.

Para ambas conexiones se crearon dos usuarios del sistema para los cuales no se proporcionó sus credenciales en la configuración de las conexiones para que, tras el acceso a Guacamole y selección de la conexión, el usuario tuviera que autenticarse de nuevo, debido a razones de seguridad. Adicionalmente, para la conexión con la máquina con sistema operativo Windows, se estableció el modo de seguridad NLA y que ignorara el certificado del servidor para evitar posibles errores derivados de que este requiriera conexiones cifradas.

A continuación, se mostrará, a modo de ejemplo, los resultados obtenidos para la conexión VNC.

### 2.5.2.1.- Resultados obtenidos a modo de ejemplo en la conexión VNC

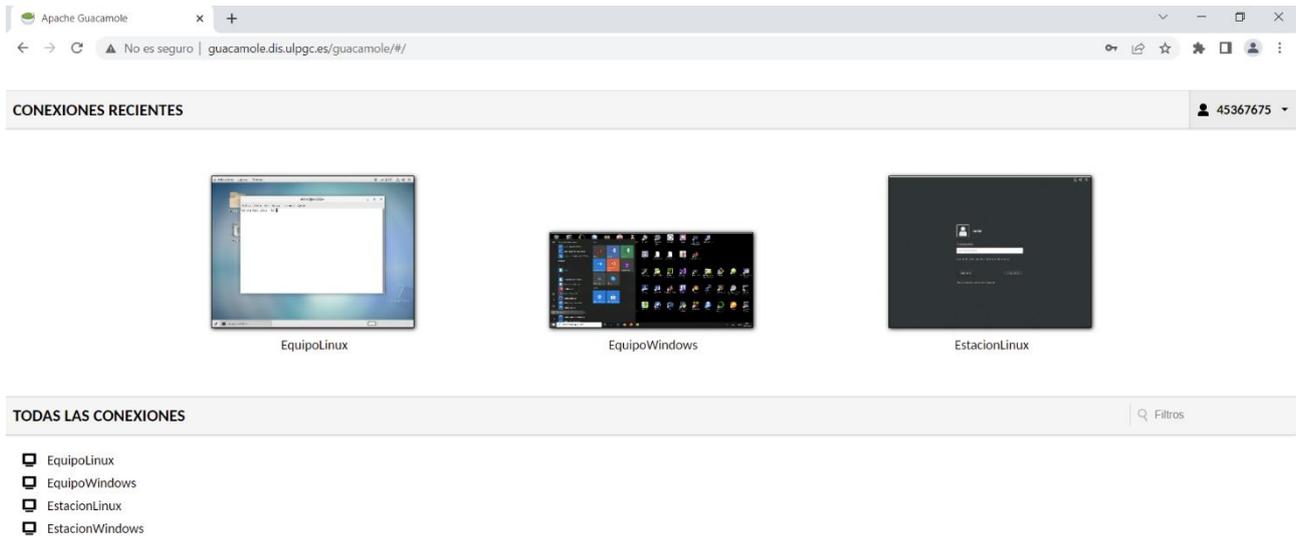
En la siguiente figura, se puede observar la correcta conexión realizada a la máquina virtual con sistema operativo Linux haciendo uso del protocolo VNC. Para ello, se instaló en dicha máquina el servidor *TigerVNC* siguiendo los pasos descritos en “2.4.5.1-Instalación de servidores VNC”. Para el usuario del sistema a utilizar en esta conexión VNC, descrito en dichos pasos, se creó un usuario denominado “javier” y se le estableció una contraseña para poder autenticarse en el servidor.



**Figura 54:** Resultado de la conexión “Estación Linux”.

### 2.5.3.- Prueba y análisis de rendimiento en equipos docentes basados en Windows y Linux

Una vez completadas las pruebas del servicio Apache Guacamole sobre máquinas virtuales implementadas bajo la tecnología KVM, se procedió a configurar conexiones para equipos físicos pertenecientes al Departamento de Informática y Sistemas. Para ello, se habilitaron dos equipos físicos, uno con sistema operativo *Windows 10 Enterprise* y otro con sistema operativo Linux distribución *CentOS7*. Sobre estos equipos, se instaló el software cliente necesario para permitir el funcionamiento de Guacamole (servidor VNC en equipo con *CentOS7* y habilitación de conexiones RDP en equipo con *Windows 10 Enterprise*) y se crearon, a través de la interfaz web de Guacamole, conexiones hacia estos equipos. El procedimiento realizado para probar las conexiones fue acudir a <http://guacamole.dis.ulpgc.es/guacamole> y realizar el proceso de autenticación bajo el sistema CAS. Una vez completado este proceso, el servidor CAS redirigió al usuario hacia la aplicación web donde pudo observar las conexiones a las que poseía acceso (debido a configurar este acceso previamente). La página que observó se puede ver representada en la siguiente figura.



**Figura 55:** Conexiones del usuario.

Como se puede observar, disponía de las 4 conexiones configuradas correspondientes a las máquinas virtuales y a los equipos físicos. A partir de ahí, el usuario ya podía seleccionar la conexión deseada y, tras autenticarse como el usuario del sistema creado para cada equipo, adquirir el acceso remoto a dichos equipos. La configuración realizada y los resultados obtenidos para cada equipo se detallarán a continuación.

### 2.5.3.1.- Configuración y Resultados: Equipo con sistema operativo Linux

Para permitir la conexión con el equipo con sistema operativo Linux distribución CentOS7, se instaló el servidor *TigerVNC* siguiendo los pasos de “2.4.5.1-Instalación de servidores VNC” y, tras ello, se creó una nueva conexión a través de la interfaz web de Apache Guacamole para permitir el acceso remoto a este equipo por vía del protocolo VNC. En la siguiente figura, se pueden observar los detalles (parámetros de red) de esta conexión:

**PARÁMETROS**

**Red**

Nombre de Host:

Puerto:

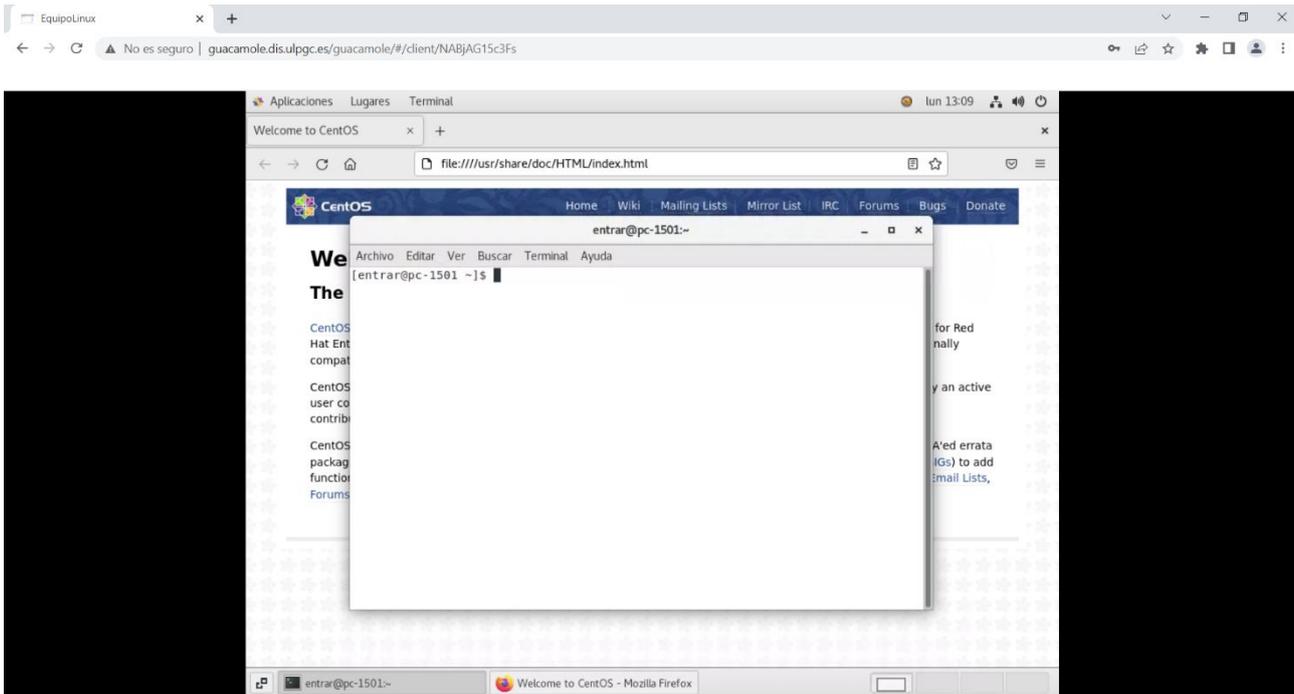
**Autenticación**

Usuario:

Contraseña:

**Figura 56:** Características de la conexión “Equipo Linux”.

Como se puede observar, se especificó en la conexión la dirección IP del equipo, así como el puerto 5901, debido a que es el que usa la conexión VNC con *display number* 1. Adicionalmente, tal y como se realizó para las máquinas virtuales, se decidió no establecer un usuario en la propia conexión para forzar la doble autenticación, debido a cuestiones de seguridad. Para dicho usuario, se hizo uso de uno por defecto denominado “*entrar*” con el que los alumnos realizan sus tareas, estableciéndole una contraseña para poder autenticarse en el servidor. El resultado de la conexión VNC se puede observar a continuación.



**Figura 57:** Resultado de la conexión “*Equipo Linux*”.

### 2.5.3.2.- Configuración y Resultados: Equipo con sistema operativo Windows

Para permitir la conexión con el equipo con sistema operativo *Windows 10 Enterprise*, se habilitaron las conexiones por vía del protocolo RDP siguiendo lo indicado en “2.4.5.2-*Habilitación de las conexiones RDP*” y, tras ello, se creó una nueva conexión a través de la interfaz web de Apache Guacamole para permitir el acceso remoto a este equipo haciendo uso de dicho protocolo. En la siguiente figura, se pueden observar los detalles (parámetros de red) de esta conexión:

### PARÁMETROS

**Red**

Nombre de Host:

Puerto:

**Autenticación**

Usuario:

Contraseña:

Dominio:

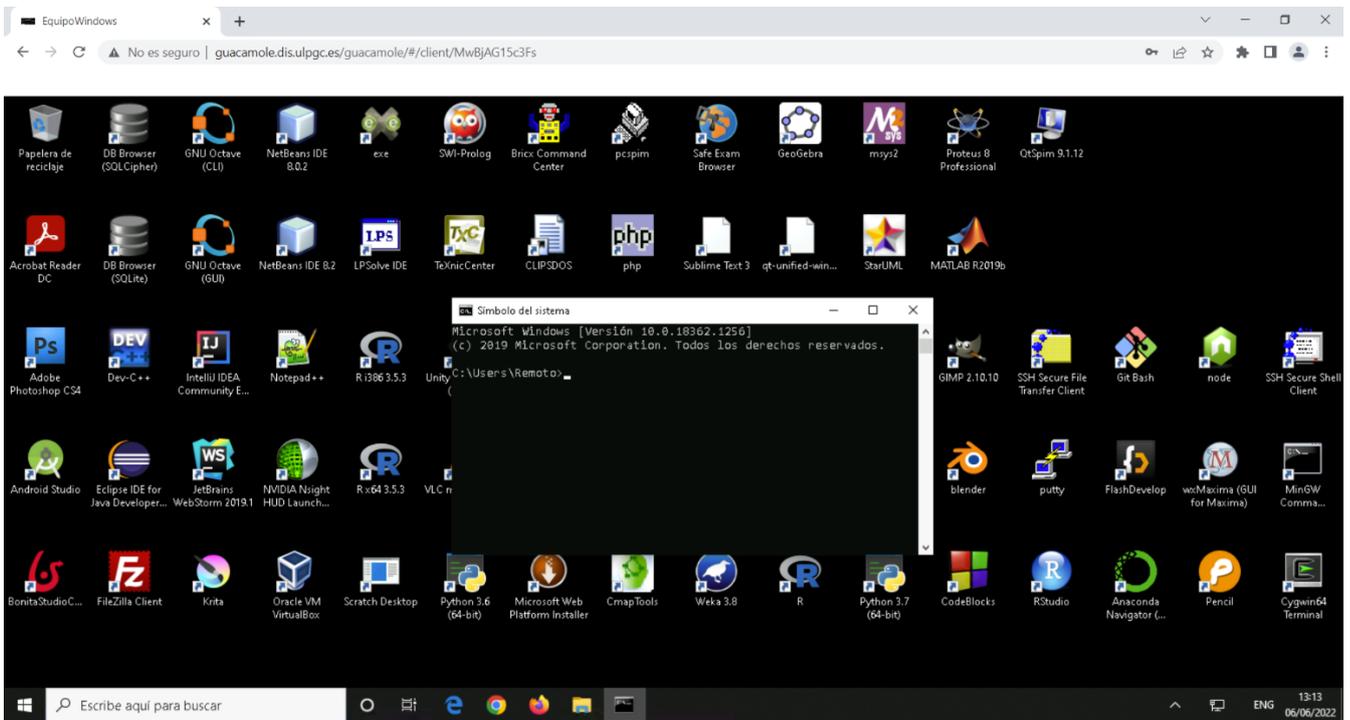
Modo seguridad:

Desactivar autenticación:

Ignorar certificado del servidor:

**Figura 58:** Características de la conexión “Equipo Windows”.

Como se puede observar, se especificó en la conexión la dirección IP del equipo, así como el puerto 3389 que usa por defecto el protocolo RDP. Adicionalmente, tal y como se realizó para las máquinas virtuales, se decidió no establecer un usuario en la propia conexión para forzar la doble autenticación, debido a cuestiones de seguridad. Además, se seleccionó el modo de seguridad NLA configurado por defecto en los equipos y se habilitó el parámetro “Ignorar certificado del servidor” para evitar posibles errores relacionados con que el servidor exigiera conexiones cifradas. Para dicho usuario del sistema a utilizar en esta conexión RDP, se creó uno denominado “Remoto” y se le estableció una contraseña para poder autenticarse en el servidor. El resultado de la conexión RDP se puede observar a continuación.



**Figura 59:** Resultado de la conexión “Equipo Windows”.

## 2.6.- Competencias Adicionales

El desarrollo de este TFT ha acarreado la adquisición de competencias adicionales que se corresponden con tecnologías secundarias estudiadas con el fin de mostrar como integrarlas con Apache Guacamole. El estudio en profundidad de la integración de estas tecnologías se puede encontrar en la sección “*Anexo: Manual de Usuario de Apache Guacamole*”, pero estas se describirán brevemente a continuación.

-**Telnet [21]**: Es un protocolo de texto (terminal) con bastante antigüedad en el mercado que permite las conexiones remotas a través de línea de comandos. Tiene serias desventajas, como por ejemplo la falta de cifrado de sus conexiones, y es por ello por lo que se recomienda no hacer uso de él en sistemas modernos. Se ha estudiado la forma de integrar este protocolo con Guacamole por si es requerido en alguna situación puntual para un uso académico, pues no se recomienda hacer uso de él debido a las desventajas mencionadas.

-**SSH (*Secure Shell*) [23]**: Es, al igual que Telnet, un protocolo de texto (terminal) que permite las conexiones remotas a través de línea de comandos. Surgió a modo de reemplazo del protocolo Telnet debido a sus desventajas. Sus conexiones son cifradas y, es por ello, junto a otras ventajas, por lo que es uno de los protocolos más utilizados en la actualidad. Se ha estudiado la forma de integrar este protocolo con Guacamole por si es requerido para conexiones remotas en las que no sea necesario una interfaz gráfica.

-**LDAP (*Lightweight Directory Access Protocol*) [31]**: Es un protocolo basado en estándares que se ubica sobre los protocolos TCP (*Transfer Control Protocol*) e IP (*Internet Protocol*) y permite a los usuarios realizar diversas operaciones en un servidor denominado “*Directorio*”, como almacenar y obtener datos, buscar datos en base a unos criterios o realizar procesos de autenticación, entre otros. La implementación de código abierto más conocida es *OpenLDAP* [32]. Se ha estudiado esta tecnología como alternativa de sistema de autenticación a CAS, con el objetivo de reflejar la forma de configurarlo por si es requerido en líneas futuras de este TFT.

-**RADIUS (*Remote Authentication Dial in User Service*) [34]**: es un protocolo de autenticación, autorización y contabilidad (AAA por sus siglas en inglés). Posee soporte para diversos protocolos de autenticación como PPP (*Point-to-Point Protocol*), PAP (*Password Authentication Protocol*), CHAP (*Challenge Handshake Authentication Protocol*), entre otros. La implementación de código abierto más conocida es *FreeRADIUS* [35]. Al igual que LDAP, se ha estudiado esta tecnología como alternativa de sistema de autenticación a CAS, con el objetivo de reflejar la forma de configurarlo por si es requerido en líneas futuras de este TFT.

### 3.- Conclusiones y Trabajos futuros

El desarrollo de este TFT ha contribuido en gran medida a adquirir conocimientos en nuevas tecnologías adicionales a los planteados en el plan de estudios.

Gracias al trabajo realizado, se ha conseguido conocer, instalar y configurar una tecnología no muy conocida que posee grandes utilidades para la sociedad en la actualidad. Esto debido a, entre otros motivos, el auge que posee hoy en día las nuevas formas telemáticas de realizar la jornada laboral, surgidas en algunas organizaciones de forma voluntaria y en otras de manera obligatoria por la reciente situación de pandemia mundial.

Esta nueva forma de trabajo es un precedente de lo que será el futuro del ámbito profesional, pues cada vez más son las organizaciones que deciden reducir costes y desarrollar formas de trabajo que no requieran la presencialidad.

Por ello, conocer el funcionamiento de la tecnología estudiada en este proyecto supone una ventaja a la hora de construir un portfolio para el ámbito laboral, pues no muchos profesionales conocen de su existencia y su utilidad para una organización es elevada.

Adicionalmente, el desarrollo de este trabajo fin de título ha supuesto la adquisición de conocimientos adicionales sobre tecnologías secundarias, debido a la capacidad de integración de estas con Apache Guacamole. Estas tecnologías incluyen los sistemas de autenticación CAS (*Central Authentication Service*), LDAP (*Lightweight Directory Access Protocol*) y RADIUS (*Remote Authentication Dial in User Service*) y los protocolos Telnet y SSH (*Secure Shell*) que, a pesar de conocerlos gracias al plan de estudios, el conocimiento sobre ellos ha sido ampliado.

En referencia al grado de consecución de los objetivos inicialmente establecidos, se puede observar que los hitos planteados al inicio del documento traducidos a las diferentes fases del proyecto se han llevado a cabo de forma correcta. Estos hitos eran los siguientes:

- H1. Diseño de la infraestructura básica en la que instalar y ejecutar el servicio Apache Guacamole.
- H2. Instalación del servicio Apache Guacamole.
- H3. Establecimiento de los requerimientos para la instalación del software cliente en los equipos utilizados en las actividades prácticas.
- H4. Instalación del software cliente en los equipos utilizados en las actividades prácticas.
- H5. Pruebas y evaluación del rendimiento del sistema.

El primer hito de diseño de la infraestructura donde instalar y ejecutar el nuevo servicio puede verse reflejado con el trabajo realizado en las tareas de análisis de requisitos del CPD (Centro de Procesamiento de Datos) del Departamento de Informática y Sistemas (DIS) y de diseño de la infraestructura, llevadas a cabo en las fases de Estudio /Análisis y Diseño/Desarrollo/Implementación respectivamente.

El segundo hito de instalación del servicio se corresponde con el trabajo realizado durante la fase de Diseño/Desarrollo/Implementación, donde se instaló y configuró el servicio de forma básica, para posteriormente agregarle extensiones para hacer uso de sistemas de autenticación y almacenamiento presentes en el mercado.

El tercer y cuarto hito se asocia a las tareas de instalación de servidores VNC (*Virtual Network Computing*) y la habilitación de conexiones RDP (*Remote Desktop Protocol*) llevadas a cabo con el objetivo de probar posteriormente el funcionamiento del servicio.

El último hito, correspondiente a dichas pruebas de funcionamiento, se ve representado con las labores realizadas en la fase de Evaluación/Validación/Prueba con el objetivo de evaluar el nuevo servicio, que incluyen las pruebas iniciales sobre máquinas virtuales implementadas con la tecnología KVM y las realizadas sobre equipos físicos que reflejan que el objetivo principal de este proyecto (Proporcionar acceso remoto a los equipos docentes del DIS) se cumple de forma correcta.

Con respecto a líneas futuras de este proyecto, son numerosas las posibilidades de configuración que posee Apache Guacamole. A continuación, se proponen algunas de ellas:

- Implementación de los protocolos SSH y Telnet (este último se recomienda que sea solo para situaciones puntuales) en Apache Guacamole.
- Integración del sistema de autenticación LDAP con Apache Guacamole como medio de identificación de usuarios y, opcionalmente, de almacenamiento de la información de sus conexiones.
- Integración del sistema de autenticación RADIUS con Apache Guacamole como medio de identificación de usuarios.
- Configuración de Apache Guacamole para que haga uso de conexiones cifradas.

Apache Guacamole es una tecnología con grandes posibilidades que le confieren de un gran valor para las organizaciones hoy en día. El desarrollo de este TFT ha ayudado a conocer cómo funciona y cómo integrarla en infraestructuras tecnológicas existentes, lo que supone una gran ventaja para el futuro laboral.

## 4.- Anexo: Manual de Usuario de Apache Guacamole

### 1.- Introducción

En este manual se describirán los diferentes pasos a seguir para instalar y configurar Apache Guacamole [3], así como para su soporte para diversos protocolos y sistemas de autenticación presentes en el mercado. Apache Guacamole [6] es un servicio web basado en HTML5 que permite el acceso remoto a sistemas informáticos haciendo uso de protocolos de escritorio remoto (como VNC (*Virtual Network Computing*) o RDP (*Remote Desktop Protocol*)), entre otros.

Apache Guacamole [7] está compuesto por una aplicación web y varios componentes de bajo nivel. Los usuarios se conectan a un servidor de Apache Guacamole con su navegador. El cliente Guacamole, programado en el lenguaje de programación JavaScript, es servido a los usuarios por parte de un servidor web integrado en el servidor de Guacamole. Una vez cargado, el cliente se conecta de nuevo hacia el servidor a través de HTTP (*Hypertext Transfer Protocol*) usando el protocolo Guacamole.

El protocolo Guacamole implementa un conjunto de protocolos de escritorio remoto existentes. Añadir soporte para un determinado protocolo de escritorio remoto conlleva la creación de una capa intermedia que “traduce” la comunicación entre el protocolo Guacamole y el protocolo de escritorio remoto. Esta capa intermedia es *guacd*.

*guacd* es el núcleo de Apache Guacamole que dinámicamente carga el soporte para protocolos de escritorio remoto conectándolos a escritorios remotos basándose en instrucciones recibidas desde la aplicación web.

*guacd* es un proceso catalogado como demonio del sistema [8] que se instala junto a Apache Guacamole y se ejecuta en segundo plano, escuchando conexiones TCP (*Transmission Control Protocol*) procedentes de la aplicación web. No comprende ningún protocolo de escritorio remoto concreto, pero implementa suficientes funciones pertenecientes al protocolo Guacamole para determinar que soporte de protocolo necesita cargar y con qué argumentos. Una vez el protocolo se carga, este se ejecuta independientemente de *guacd* y posee el control completo de la comunicación entre sí mismo y la aplicación web hasta que el cliente finaliza.

*guacd* y todos los protocolos de escritorio remoto dependen de una librería común, *libguac*, que facilita y hace más abstracta la comunicación a través del protocolo Guacamole.

Tanto el protocolo Guacamole como el demonio *guacd* proporcionan independencia de protocolos evitando que ni el cliente Guacamole ni la aplicación web deban saber qué protocolo de escritorio remoto se está realmente usando.

## 1.1.- Apache Guacamole en este informe

En este informe, se detallarán, en primer lugar, los pasos para instalar Apache Guacamole en un sistema Linux con distribución CentOS7, explicando como instalar el *Servlet* (se hará uso de *Apache Tomcat*) requerido para su parte cliente, así como la forma de construir y compilar dicha parte cliente y su parte servidor.

Posteriormente, se describirá la estructura de directorios y los ficheros de configuración que utiliza Apache Guacamole, así como la forma para crear conexiones usando los diferentes protocolos para los que ofrece soporte (VNC (*Virtual Network Computing*), RDP (*Remote Desktop Protocol*), SSH (*Secure Shell*), Telnet).

Para finalizar, se explicarán algunos de los sistemas de autenticación de usuarios que proporciona Apache Guacamole. Concretamente se hablará de la autenticación por defecto, de la autenticación vía Base de Datos, la autenticación vía LDAP (*Lightweight Directory Access Protocol*) y la autenticación vía RADIUS (*Remote Authentication Dial in User Service*).

## 2.- Apache Guacamole

### 2.1.- Instalación de Apache Guacamole

Para instalar Apache Guacamole, se requiere en primer lugar de un *Servlet* para aplicaciones Java debido a que la aplicación web de la parte cliente de Apache Guacamole está escrita en este lenguaje de programación. Se seleccionará como *Servlet Apache Tomcat* [11] en su versión 9 debido a que su versión 10, de 20 de enero de 2022, requiere de un proceso de migración de aplicaciones debido al cambio de plataforma *Java EE* a *Jakarta EE* que, debido a su reciente publicación, se ha decidido no implementar para evitar posibles inconvenientes de estabilidad.

Tras instalar el *Servlet* mencionado, el siguiente paso es construir y compilar desde el código fuente la parte servidor de Apache Guacamole y, tras eso, instalar la parte cliente, para lo que existen dos formas posibles, la primera construyendo y compilando desde el código fuente al igual que la parte servidor y la segunda añadiendo al *Servlet* un binario con extensión de fichero *.war* que contiene toda la aplicación web de Apache Guacamole.

### 2.1.1.- Instalación del Servlet (Apache Tomcat 9)

En este apartado del manual se describirán los pasos a seguir para instalar el Servlet *Apache Tomcat* en un sistema CentOS7. Para ello, se hará uso de los recursos indicados en [38,39,40].

1. Actualizar el sistema e instalar el JDK de Java con la versión 8 como mínimo, si ya no está instalado:

```
-yum update
-yum install -y java-1.8.0-openjdk-devel
-java -version
```

Si se desea instalar otra versión, como la 11, esta se puede instalar y configurar como versión por defecto (elegir opción correspondiente a Java 11) haciendo uso de las siguientes ordenes:

```
-yum install -y java-11-openjdk-devel
-alternatives --config java
-java -version
```

2. Añadir como variable de entorno la variable **\$JAVA\_HOME**, que indicará la ruta del directorio *home* de Java, indicando la ruta siguiente y aplicando la variable:

```
-gedit /etc/environment
    JAVA_HOME=/etc/alternatives/jre
-source /etc/environment
-echo $JAVA_HOME
```

3. Añadir al usuario *tomcat*, que operará el servicio, al sistema, creando un grupo con el mismo nombre, estableciéndole directorio home en */opt/tomcat*, copiando el esqueleto de su directorio a partir de */dev/null* y con el shell */bin/false*:

```
-useradd -m -U -k /dev/null -d /opt/tomcat -s /bin/false tomcat
```

4. Descargar Apache Tomcat desde el sitio web indicado en [41] seleccionando la opción *tar.gz* del subapartado “*Core*” de la versión 9 correspondiente.

5. Desempaquetar el archivo descargado y ubicarlo en el directorio del usuario *tomcat* (*/opt/tomcat*):

```
-tar xf apache-tomcat-9.X.X.tar.gz -C /opt/tomcat/
```

6. Crear un enlace simbólico del directorio extraído para tener un nombre uniforme del directorio independientemente de la versión del software:

```
-ln -s /opt/tomcat/apache-tomcat-9.X.X/ /opt/tomcat/apache-tomcat
```

7. Establecer como propietario del directorio extraído y de todos los ficheros contenidos en el al usuario *tomcat*:

```
-chown -R tomcat: /opt/tomcat/apache-tomcat/
```

8. Añadir el servicio *tomcat9* a *Systemd* creando un fichero (*tomcat9.service*), con la siguiente orden, que contenga lo que se muestra a continuación de dicha orden:

```
-gedit /etc/systemd/system/tomcat9.service
```

```
[Unit]
```

```
Description=Tomcat 9.0 servlet container para CentOS 7
```

```
After=network.target
```

```
[Service]
```

```
Type=forking
```

```
User=tomcat
```

```
Group=tomcat
```

```
Environment="JAVA_OPTS=-Djava.security.egd=file:///dev/urandom"
```

```
Environment="CATALINA_BASE=/opt/tomcat/apache-tomcat"
```

```
Environment="CATALINA_HOME=/opt/tomcat/apache-tomcat"
```

```
Environment="CATALINA_PID=/opt/tomcat/apache-tomcat/temp/tomcat.pid"
```

```
Environment="CATALINA_OPTS=-Xms512M -Xmx1024M -server -XX:+UseParallelGC"
```

```
ExecStart=/opt/tomcat/apache-tomcat/bin/startup.sh
```

```
ExecStop=/opt/tomcat/apache-tomcat/bin/shutdown.sh
```

```
[Install]
```

```
WantedBy=multi-user.target
```

9. Iniciar y habilitar en el arranque el servicio *tomcat9* y añadir una regla al cortafuegos que permita la comunicación por el puerto 8080, debido a que es el que usa por defecto el *Servlet*:

```
-systemctl start tomcat9
```

```
-systemctl enable tomcat9
```

```
-firewall-cmd --permanent --add-port=8080/tcp
```

```
-firewall-cmd --reload
```

10. Comentar el siguiente bloque en los archivos XML (*eXtensible Markup Language*) indicados en las órdenes, con el objetivo de permitir el acceso desde la red a la administración del *Servlet*:

```
<!--
  <Valve className="org.apache.catalina.valves.RemoteAddrValve"
    allow="127\.\d+\.\d+\.\d+|:1|0:0:0:0:0:0:1" />
-->
-gedit /opt/tomcat/apache-tomcat/webapps/manager/META-INF/context.xml
-gedit /opt/tomcat/apache-tomcat/webapps/host-manager/META-INF/context.xml
```

11. Añadir la siguiente línea al fichero XML de usuarios de Apache Tomcat (*tomcat-users.xml*) estableciendo nombre de usuario y contraseña para crear un usuario con roles suficientes para gestionar las aplicaciones web administrativas que se incluyen en la instalación del *Servlet*:

```
-gedit /opt/tomcat/apache-tomcat/conf/tomcat-users.xml
<user username="nombre" password="XXXXXXXX" roles="admin-gui,manager-gui"/>
```

12. Opcionalmente, se puede establecer un FQDN (*Full Qualified Domain Name*) al equipo donde se está realizando la instalación:

```
-hostnamectl set-hostname <<nombre>>
-reboot
```

13. Con el FQDN indicado o con la dirección IP de la máquina, se puede acceder por vía del navegador a *http://FQDN:8080/* o bien *http://IP:8080/* donde se podrá observar la página principal de Apache Tomcat en donde se encuentran, entre otros aspectos, las aplicaciones administrativas cuya autenticación se estableció en el paso 11.

14. Opcionalmente, si se desea cambiar el tamaño de las aplicaciones web en Tomcat, se debe aumentar el tamaño que se muestra en el bloque siguiente en el archivo indicado en la orden:

```
-gedit /opt/tomcat/apache-tomcat/webapps/manager/WEB-INF/web.xml
<multipart-config>
  <!-- 50MB max -->
  <max-file-size>52428800</max-file-size>
  <max-request-size>52428800</max-request-size>
  <file-size-threshold>0</file-size-threshold>
</multipart-config>
```

## 2.1.2.- Instalación del servidor de Apache Guacamole (*guacamole-server*)

En este apartado del informe se describirán los pasos a seguir para construir y compilar desde el código fuente la parte servidor de Apache Guacamole (*guacamole-server*) siguiendo la guía indicada en [42].

### 1. Actualizar el sistema:

-yum update

2. Se deben instalar una serie de dependencias, algunas necesarias y otras opcionales, para permitir el correcto funcionamiento de Apache Guacamole. Las dependencias, divididas en requeridas y opcionales, son las siguientes:

#### Requeridas

**-Cairo (paquete cairo-devel):** Lo utiliza la API (*Application Programming Interface*) en C de Apache Guacamole, *libguac*, para el renderizado de gráficos.

**-libjpeg-turbo (paquete libjpeg-turbo-devel):** Lo utiliza *libguac* para proporcionar soporte para JPEG (*Joint Photographic Experts Group*).

**-libpng (paquete libpng-devel):** Lo utiliza *libguac* para crear imágenes PNG (*Portable Network Graphics*), el tipo de imagen principal usado por el protocolo Guacamole.

**-libtool (paquete libtool):** Se utiliza durante el proceso de construcción. Crea librerías compiladas que son necesarias para Apache Guacamole.

**-libuuid (paquete libuuid-devel):** Lo utiliza *libguac* para asignar identificadores internos y únicos a cada usuario y conexión. Estos identificadores son la base para el soporte de conexiones compartidas.

#### Opcionales

**-FFmpeg (paquete ffmpeg-devel):** Lo emplea la utilidad *guacenc* para codificar retransmisiones de video cuando traduce las grabaciones de las sesiones de Apache Guacamole. La mencionada utilidad no será construida sin esta dependencia.

**-FreeRDP (paquete freerdp-devel):** Se requiere la versión 2.0.0 o más reciente de esta dependencia para proporcionar soporte para RDP (*Remote Desktop Protocol*).

**-Pango (paquete pango-devel):** Apache Guacamole hace uso de esta librería de disposición de textos para renderizar texto para los protocolos que requieren de terminal.

**-libssh2 (paquete libssh2-devel):** Se requiere para proporcionar soporte a los protocolos SSH (*Secure Shell*) y SFTP (*Secure File Transfer Protocol*).

**-libtelnet (paquete libtelnet-devel):** Se requiere para proporcionar soporte al protocolo Telnet.

**-libVNCServer (paquete libvncserver-devel):** Proporciona la librería *libvncclient*, requerido para el soporte de VNC (*Virtual Network Computing*).

**-libwebsockets (paquete libwebsockets-devel):** Se requiere para proporcionar soporte para Kubernetes.

**-PulseAudio (paquete pulseaudio-libs-server):** proporciona la librería *libpulse*, utilizada por el soporte para VNC de Guacamole para proporcionar audio experimental.

**-OpenSSL (paquete openssl-devel):** Se requiere para proporcionar soporte para SSL (*Secure Sockets Layer*) y TLS (*Transport Layer Security*), esquemas de cifrado usados normalmente para el tráfico web. También es necesaria para proporcionar soporte SSH.

**-libvorbis (paquete libvorbis-devel):** Se requiere para proporcionar soporte para *Ogg Vorbis*, un estándar gratuito para la compresión de sonido.

**-libwebp (paquete libwebp-devel):** Se requiere para proporcionar soporte para crear imágenes *WebP*.

Para instalar las dependencias indicadas se debe ejecutar la siguiente orden:

```
-yum install cairo-devel libjpeg-turbo-devel libpng-devel libtool libuuid-devel
freerdp-devel pango-devel libssh2-devel libvncserver-devel pulseaudio-libs-devel
openssl-devel libvorbis-devel libwebp-devel
```

Como se puede observar, en la orden no se encuentran las dependencias *ffmpeg-devel*, *libwebsockets-devel* y *libtelnet-devel*. Esto es debido a que los repositorios base de CentOS7 no poseen paquetes para ellas o requieren de un fichero con extensión *.rpm* determinado para su instalación. Por ello se instalan a continuación los repositorios EPEL (*Extra Packages for Enterprise Linux*) y se instala de manera local un fichero con extensión *.rpm* con el objetivo de permitir la correcta instalación de estas dependencias. Para todo lo comentado, ejecutar las siguientes órdenes:

```
-yum install epel-release
-yum install libtelnet-devel
-yum install libwebsockets-devel
-yum localinstall --nogpgcheck https://download1.rpmfusion.org/free/el/rpmfusion-free-release-7.noarch.rpm
-yum install ffmpeg-devel
```

Además, se debe instalar el siguiente paquete para que posteriormente un script incluido en el paquete *guacamole-server* detecte correctamente que puede proporcionar soporte para VNC (*Virtual Network Computing*), ya que, en caso contrario, no lo proporciona:

```
-yum install libcrypt-devel
```

3. Descargar desde [43] el fichero *guacamole-server-1.4.0.tar.gz*. Desempaquetarlo con la siguiente orden y acceder al directorio extraído:

```
-tar -xzf guacamole-server-1.4.0.tar.gz
-cd guacamole-server-1.4.0/
```

4. Ejecutar el script denominado *configure* que se encuentra en dicho directorio añadiendo opcionalmente el directorio de inicio, tal y como se muestra en la siguiente orden. Este script comprobará las dependencias instaladas en el sistema e indicará al final de la ejecución qué funcionalidades puede implementar la instalación de Apache Guacamole. Se debe comprobar que, como requisito mínimo, las dependencias obligatorias aparezcan con un “yes” a su lado:

```
./configure --with-init-dir=/etc/init.d
```

5. Por último, se deben ejecutar las tres órdenes siguientes para construir y compilar *guacamole-server* y para actualizar la memoria caché del sistema con las nuevas librerías:

```
-make
-make install
-ldconfig
```

### 2.1.3.- Instalación del cliente de Apache Guacamole (*guacamole-client*)

En este apartado del informe se describirán los pasos a seguir para la instalación de la parte cliente de Apache Guacamole (*guacamole-client*) según la misma guía usada en el apartado anterior. La parte cliente se puede instalar de dos formas, la primera construyendo y compilando al igual que se hizo para la parte servidor, para lo que se necesita Apache Maven, o, por el contrario, descargar un archivo con extensión *.war* y añadirlo al directorio del *Servlet*. Se mostrarán a continuación las dos formas debido a que, aunque la segunda es relativamente más sencilla que la primera, sistemas de autenticación como RADIUS (*Remote Authentication Dial in User Service*), cuyos detalles se exponen en la sección “2.3.4-Autenticación por RADIUS”, no poseen de extensión disponible en el sitio web de Guacamole y, por ello, para su implementación se necesita construir la parte cliente incluyéndola en el proceso.

Los pasos a seguir para la **primera forma** de instalación (construyendo y compilando el código fuente) son los siguientes [42,45]:

1. Descargar Apache Maven desde el sitio web indicado en [46]. El archivo se denomina *apache-maven-X.X.X-bin.tar.gz* donde las “X” indican la versión más reciente. Si se realiza desde línea de comandos, la versión que se instala es antigua (procede de los repositorios de CentOS7) y puede ocasionar conflictos con algunos *plugins* que se instalarán posteriormente en la construcción.

2. Desempaquetar el archivo en el directorio */opt* y crear un enlace simbólico para omitir la versión del software en el propio nombre del fichero:

```
-tar xf apache-maven-X.X.X-bin.tar.gz -C /opt
-ln -s /opt/apache-maven-X.X.X/ /opt/maven
```

3. Establecer las variables de entorno necesarias para Apache Maven creando un fichero denominado *maven.sh* en la ruta indicada en la siguiente orden y añadiendo en él lo que se muestra a continuación de dicha orden:

```
-gedit /etc/profile.d/maven.sh
export JAVA_HOME=/etc/alternatives/jre
export M2_HOME=/opt/maven
export MAVEN_HOME=/opt/maven
export PATH=${M2_HOME}/bin:${PATH}
```

4. Proporcionar permisos al fichero para que sea ejecutable, cargar las variables de entorno con la orden *source* y comprobar la versión de Apache Maven:

```
-chmod +x /etc/profile.d/maven.sh
-source /etc/profile.d/maven.sh
-mvn -version
```

5. Descargar desde [43] el fichero *guacamole-client-1.4.0.tar.gz*. Desempaquetarlo con la siguiente orden y acceder al directorio extraído:

```
-tar -xzf guacamole-client-1.4.0.tar.gz
-cd guacamole-client-1.4.0/
```

6. Ejecutar la siguiente orden para que Apache Maven construya y empaquete todos los componentes, produciendo un fichero con extensión *.war* que contiene la aplicación web completa. Si se requiere implementar en un futuro el sistema de autenticación RADIUS, ejecutar la segunda orden en su lugar:

```
-mvn package
-mvn clean package -Plgpl-extensions
```

7. El fichero con extensión *.war* que contiene la aplicación web completa obtenido se encuentra ubicado en el subdirectorio *guacamole/target* del directorio donde se ejecutó la orden anterior. Se debe copiar el archivo con extensión *.war* renombrándolo si es posible, para omitir el nombre de la versión, al directorio del *Servlet* destinado al almacenamiento de las aplicaciones web. En el Servlet Apache Tomcat dicho directorio es el indicado en la siguiente orden:

```
-cp guacamole-1.4.0.war /opt/tomcat/apache-tomcat/webapps/guacamole.war
```

8. Reiniciar el servicio *tomcat9* e iniciar el servicio *guacd* con las órdenes siguientes:

```
-systemctl restart tomcat9
-systemctl start guacd ó /etc/init.d/guacd start
```

9. Acceder al *Servlet* vía el navegador y a la ruta indicada en este para la aplicación web de Apache Guacamole.

Los pasos a seguir para la **segunda forma** de instalación (a partir de un fichero que contiene toda la aplicación web) son los siguientes [42]:

1. Descargar desde [43] el archivo con extensión *.war*.

2. Copiar el archivo con extensión *.war* renombrándolo si es posible, para omitir el nombre de la versión, al directorio del *Servlet* destinado al almacenamiento de las aplicaciones web. En el Servlet Apache Tomcat dicho directorio es el indicado en la siguiente orden:

```
-cp guacamole-1.4.0.war /opt/tomcat/apache-tomcat/webapps/guacamole.war
```

3. Reiniciar el servicio *tomcat9* e iniciar el servicio *guacd* con las órdenes siguientes:

```
-systemctl restart tomcat9
```

```
-systemctl start guacd ó /etc/init.d/guacd start
```

4. Acceder al *Servlet* vía el navegador y a la ruta indicada en este para la aplicación web de Apache Guacamole.

## 2.2.- Configuración de Apache Guacamole

Apache Guacamole requiere de una serie de tareas de configuración de usuarios y conexiones para permitir cumplir con el objetivo de la tecnología: Proporcionar acceso remoto a sistemas informáticos para los usuarios.

En esta sección del informe se describirán, haciendo uso del sitio web de Apache Guacamole y su página dedicada a configurar el servicio [47], todos aquellos elementos cuya existencia y razón de ser es necesario conocer para configurar correctamente Apache Guacamole.

Concretamente, se hablará de su directorio *GUACAMOLE\_HOME*, de sus ficheros de configuración *guacamole.properties* y *guacd.conf* y de cómo configurar conexiones y usuarios para permitir hacer uso del soporte para los protocolos VNC (*Virtual Network Computing*), RDP (*Remote Desktop Protocol*), SSH (*Secure Shell*) y Telnet.

### 2.2.1.- Directorio GUACAMOLE\_HOME (/etc/guacamole)

*GUACAMOLE\_HOME* es el nombre del directorio de configuración de Apache Guacamole y se encuentra ubicado en el directorio */etc/guacamole* por defecto. Todos los ficheros de configuración, extensiones y otros componentes residen en este directorio. La estructura de este directorio consiste en los siguientes ficheros opcionales:

**-guacamole.properties:** Es el fichero principal de configuración de Apache Guacamole. Las propiedades establecidas en este fichero indican como Apache Guacamole se conecta a *guacd* y configuran el comportamiento de las extensiones de autenticación instaladas.

**-logback.xml:** Apache Guacamole usa un sistema de registro de actividades del sistema (logs) denominado *Logback*. Por defecto, Apache Guacamole muestra los mensajes de log en la consola, pero se puede cambiar este comportamiento proporcionando este fichero.

**-extensions/:** Directorio donde se ubican todas las extensiones instaladas para Apache Guacamole. Se cargarán desde aquí todos los ficheros *.jar* presentes en el arranque del servicio.

**-lib/:** Directorio de búsqueda de aquellas librerías requeridas por las extensiones de Apache Guacamole. Se marcarán como disponibles para todas las extensiones todos los ficheros con extensión *.jar*.

La localización del directorio *GUACAMOLE\_HOME* se puede sobrescribir siguiendo tres métodos:

1. Creando un directorio denominado *.guacamole* en el directorio *home* del usuario que ejecuta el *Servlet*.
2. Especificando en la variable de entorno *GUACAMOLE\_HOME* la ruta completa de un directorio alternativo.
3. Especificando en la propiedad del sistema *guacamole.home* la ruta completa de un directorio alternativo.

### 2.2.2.- Fichero *guacamole.properties*

*guacamole.properties* es un fichero de configuración opcional usado por la aplicación web de Apache Guacamole. Se usa para configurar Apache Guacamole en situaciones cuando las opciones por defecto son insuficientes o para proporcionar información de configuración adicional para las extensiones. Posee una serie de propiedades:

**-api-session-timeout:** Indica la cantidad de tiempo en minutos máxima permitida para que las sesiones de Apache Guacamole (en concreto sus *tokens* de autenticación) sean válidas a pesar de la inactividad. Si se omite, las sesiones expirarán tras 60 minutos de inactividad.

**-api-max-request-size:** Indica el número máximo de bytes a aceptar en el cuerpo de cualquier solicitud HTTP (*Hypertext Transfer Protocol*). Si se indica 0, significa que no se debe aplicar ningún límite a los bytes recibidos. Si se omite, las solicitudes se limitarán a 2MB por defecto. Este límite no se aplica a las subidas de ficheros.

**-allowed-languages:** Indica, mediante una lista de abreviaturas de idioma separadas por coma, los idiomas a elegir en la interfaz de Apache Guacamole.

**-enable-environment-properties:** Si se establece esta propiedad a “*true*”, Apache Guacamole evaluará primero su entorno para obtener el valor de cualquier propiedad de configuración dada, anteponiendo esta operación a usar valores especificados en el fichero *guacamole.properties* o a usar los valores por defecto.

**-extension-priority:** Indica, mediante una lista de nombres de extensiones separada por comas, qué extensiones deben ser cargadas en un orden específico.

**-guacd-hostname:** La interfaz del anfitrión (*host*) por la que el demonio proxy (*guacd*) escuchará. Si se omite, esta interfaz será *localhost*.

**-guacd-port:** El puerto por el que el demonio proxy (*guacd*) escuchará. Si se omite, este puerto será el 4822.

**-guacd-ssl:** Si se establece esta propiedad a “*true*”, Apache Guacamole requerirá de cifrado SSL(*Secure Sockets Layer*)/TLS(*Transport Layer Security*). Se necesita de un certificado SSL y una clave privada. Si se habilita esta opción, *guacd* debe ser configurado mediante la línea de comandos para usar SSL. Por defecto, la comunicación entre la aplicación web y *guacd* será no cifrada.

**-skip-if-unavailable:** Indica, mediante una lista de identificadores de proveedores de sistemas de autenticación separada por comas, los sistemas de autenticación permitidos para fallar internamente sin abortar el proceso de autenticación. Por defecto, Apache Guacamole aborta el proceso de autenticación si alguno de los sistemas destinados a este proceso falla.

Una configuración típica de este fichero que indica explícitamente los valores para *guacd-hostname* y *guacd-port* sería la siguiente:

```
# Hostname and port of guacamole proxy
guacd-hostname: localhost
guacd-port:      4822
```

### 2.2.3.- Fichero *guacd.conf*

*guacd.conf* es el fichero de configuración para el demonio proxy *guacd* de Apache Guacamole y se ubica en el directorio */etc/guacamole*. El formato de este fichero es el siguiente:

```
#
# guacd configuration file
#
[daemon]
pid_file = /var/run/guacd.pid
log_level = info
[server]
bind_host = localhost
bind_port = 4822
#
# The following parameters are valid only if
# guacd was built with SSL support.
#
[ssl]
server_certificate = /etc/ssl/certs/guacd.crt
server_key = /etc/ssl/private/guacd.key
```

Cada sección y propiedad se define de la siguiente manera:

#### **[daemon] section**

**-pid\_file:** Es el nombre del fichero donde el PID (*Process Identifier*) del proceso principal de *guacd* debe ser escrito. Se suele usar para los scripts de inicio, que requieren monitorizar el estado de *guacd*. Si se especifica este parámetro, el usuario que ejecuta *guacd* debe poseer suficientes permisos para crear o modificar el archivo especificado, o el arranque del servicio fallará.

**-log\_level:** El máximo nivel al que *guacd* registrará mensajes de log. Si se omite, el nivel por defecto será *info*. Los valores válidos son: *trace*, *debug*, *info*, *warning* y *error*.

#### **[server] section**

**-bind\_host:** La interfaz del anfitrión (*host*) a la que *guacd* debe ligarse cuando escucha para conexiones entrantes. Si se omite, *guacd* se ligará a *localhost* y solo las conexiones hacia el propio servidor que sustenta *guacd* tendrán éxito.

**-bind\_port:** El puerto al que *guacd* debe ligarse cuando escucha para conexiones entrantes. Si se omite, el puerto será el 4822.

#### [ssl] section

**-server\_certificate:** El nombre del fichero que contiene el certificado SSL para el cifrado del protocolo Guacamole. Si se especifica, el cifrado SSL se habilitará y la aplicación web de Apache Guacamole requerirá ser configurada para usar SSL en el archivo *guacamole.properties*.

**-server\_key:** El nombre del fichero que contiene la clave privada para el cifrado SSL del protocolo Guacamole. Si se especifica, el cifrado SSL se habilitará y la aplicación web de Apache Guacamole requerirá ser configurada para usar SSL en el archivo *guacamole.properties*.

La configuración de *guacd* puede ser alterada adicionalmente a través de la línea de comandos, teniendo preferencia las configuraciones realizadas por esta vía sobre las establecidas en el fichero de configuración. Para obtener más detalles de las opciones de la orden destinada a tal fin, denominada igualmente *guacd*, ejecutar la orden *man guacd*.

### 2.2.4.- VNC (*Virtual Network Computing*)

VNC es el primer protocolo para el que Guacamole tuvo soporte [47]. No es tan rápido como RDP, pero varios servidores VNC son adecuados y, además, VNC es más rápido sobre Guacamole que operando sobre sí mismo debido al bajo uso de ancho de banda. El soporte para VNC se proporciona en la librería *libguac-client-vnc* si las dependencias opcionales para este protocolo son instaladas.

Como parámetros de red, VNC posee los siguientes:

**-hostname:** El nombre o dirección IP del sistema servidor al que Apache Guacamole debe conectarse.

**-port:** El puerto en el que el sistema servidor escucha para conexiones de este protocolo, usualmente 5900 o 5900 + un número que indica la conexión (*display number*). Por ejemplo, si el *display number* es 1 (representado como :1), el puerto será 5901.

**-autoretry:** El número de veces que se debe reintentar la conexión antes de abandonarla y retornar un error. Si es una conexión inversa, posibles en VNC si se configura de la forma correspondiente, es el número de intentos permitidos del proceso de conexión.

Con respecto a la autenticación, el estándar VNC define solo la autenticación basada en contraseña. Existen otros mecanismos, pero estos no son estándar. Guacamole proporciona soporte para la autenticación basada en contraseña (**password**) y la autenticación basada en nombre de usuario y contraseña (**username** y **password**).

VNC posee también una serie de parámetros destinados a proporcionar y configurar distintas funcionalidades como son las *display settings* (**color-depth**, **swap-red-blue**, **cursor**, **encodings**, **read-only**, **force-lossless**), las opciones de posibles repetidores VNC que puedan existir (**dest-host**, **dest-port**), las opciones de las conexiones inversas (**reverse-connect**, **listen-timeout**), el soporte de audio con *PulseAudio* (**enable-audio**, **audio-servername**) y la codificación del portapapeles (**clipboard-encoding**).

Una configuración simple, usando el sistema de autenticación por defecto que ofrece Apache Guacamole (*user-mapping.xml*) que se explica en la sección “2.3.-Autenticación de usuarios” de este informe, es:

```
<connection name="Nombre único">
  <protocol>vnc</protocol>
  <param name="hostname"> <<IP | Nombre Host>> </param>
  <param name="port">5901</param>
</connection>
```

Esta conexión, de nombre “*Nombre único*” usará el protocolo VNC para conectarse al anfitrión proporcionado por su dirección IP o nombre de *host* por su puerto 5901.

## Instalación de servidor VNC

Los equipos informáticos a los que se desee realizar conexiones remotas deben disponer de un servidor VNC instalado que permita responder a las solicitudes del cliente VNC (incluido en el soporte de Apache Guacamole para este protocolo). Uno de estos servidores que, según [47], poseen buen rendimiento es *TigerVNC* [16].

Este servidor se puede instalar de forma sencilla en un sistema CentOS 7 y los pasos a seguir [55] para ello son los siguientes:

1. Actualizar el sistema e instalar el servidor *TigerVNC*:

```
-yum update
-yum install tigervnc-server
```

2. Copiar el archivo del servicio para *Systemd* desde la ubicación donde se guarda al instalar el servidor en el paso anterior hacia el directorio de *Systemd*:

```
-cp /usr/lib/systemd/system/vncserver@.service /etc/systemd/system/vncserver@.service
```

3. Editar el fichero copiado y modificar donde indica “*USER*” añadiendo el nombre de usuario que se desea utilizar en las conexiones remotas:

```
[Unit]
Description=Remote desktop service (VNC)
After=syslog.target network.target

[Service]
Type=simple
# Clean any existing files in /tmp/.X11-unix environment
ExecStartPre=/bin/sh -c '/usr/bin/vncserver -kill %i > /dev/null 2>&1 || :'
ExecStart=/usr/bin/vncserver_wrapper USER %i
ExecStop=/bin/sh -c '/usr/bin/vncserver -kill %i > /dev/null 2>&1 || :'

[Install]
WantedBy=multi-user.target
```

4. Establecer la contraseña con la que el usuario indicado en el paso anterior iniciará sesión en el servidor. Para ello, se debe cambiar la sesión actual en el sistema a ese usuario (primera orden a continuación) y establecer la contraseña con la segunda orden mostrada:

```
-su usuario
-vncpasswd
```

5. Recargar los servicios “*demonio*” del sistema:

```
-systemctl daemon-reload
```

6. Añadir al cortafuegos el puerto que se usará en la conexión remota, teniendo en cuenta el factor del *display number*. En este caso, el *display number* será :1 y, por tanto, el puerto será 5901. Recargar el cortafuegos tras este paso:

```
-firewall-cmd --permanent --zone=public --add-port=5901/tcp
-firewall-cmd --reload
```

7. Por último, iniciar el servicio indicando tras el “@” el *display number* a utilizar:

```
-systemctl start vncserver@:1
```

### 2.2.5.- RDP (*Remote Desktop Protocol*)

RDP es un protocolo más complejo y tiende a ser más rápido que VNC debido al uso de las operaciones de *caching*. El soporte para RDP [47] se proporciona en la librería *libguac-client-rdp* si las dependencias opcionales para este protocolo son instaladas.

Como parámetros de red, RDP posee los siguientes:

**-hostname:** El nombre o dirección IP del sistema servidor al que Apache Guacamole debe conectarse.

**-port:** El puerto en el que el sistema servidor escucha para conexiones de este protocolo. Si no se indica, se establecerá por defecto el puerto 3389 o el puerto por defecto de *VMConnect*, 2179, dependiendo del modo de seguridad seleccionado.

Con respecto a la autenticación, RDP la proporciona a través del uso de un nombre de usuario, una contraseña y un dominio opcional. Además, todas sus conexiones están cifradas.

La mayoría de los servidores RDP proporcionan una opción de inicio de sesión de manera gráfica si los parámetros “usuario”, “contraseña” y “dominio” son omitidos. Una excepción a esto es el uso de NLA (*Network Level Authentication*) [48], que realiza todo el proceso de autenticación externamente a la sesión de escritorio, con la ausencia de una interfaz gráfica. Los servidores de esta índole son gestionados por Guacamole de dos formas.

La primera forma es proporcionar el nombre de usuario y la contraseña internamente en la configuración de la conexión, ya sea haciendo uso de valores estáticos o transfiriendo credenciales de Guacamole con *tokens* y autenticación LDAP (*Lightweight Directory Access Protocol*). Alternativamente, si las credenciales no se establecen en la configuración de la conexión, Guacamole intentará preguntar al usuario por las credenciales de forma interactiva si las versiones de *guacd* y el cliente Guacamole lo permiten. Si ninguno de los componentes lo permite y las credenciales no están configuradas, la autenticación NLA fallará.

Los parámetros de autenticación son los siguientes:

**-username:** Nombre de usuario para la autenticación, si existe. Es un parámetro opcional.

**-password:** Contraseña para la autenticación, si existe. Es un parámetro opcional.

**-domain:** Dominio para la autenticación, si existe. Es un parámetro opcional.

**-security:** El modo de seguridad a usar en la conexión RDP. Este modo indica como se cifrarán los datos y qué tipo de autenticación, si existe, se llevará a cabo.

Por defecto, el modo de seguridad se selecciona en base a un proceso de negociación que determina cual soportan tanto cliente como servidor. Los valores para este parámetro son:

*any:* Selecciona el modo de seguridad automáticamente en base a los protocolos de seguridad soportados por el cliente y el servidor. Esta es la opción por defecto.

*nla:* Este modo usa cifrado TLS (*Transport Layer Security*) y requiere que se envíen con antelación el nombre de usuario y la contraseña. A diferencia del modo RDP, la fase de autenticación se realiza antes de que comience la sesión de escritorio remoto, evitando la necesidad de que el servidor Windows asigne un número significativo de recursos para usuarios que puede que no sean autorizados.

*nla-ext:* Este modo hace uso de NLA extendido. Es idéntico a NLA con la excepción de que se requiere que el servidor envíe hacia el cliente un *Early User Authorization Result* inmediatamente después de la operación *NLA handshake*.

*tls:* La autenticación y el cifrado se implementa mediante TLS (*Transport Layer Security*), también referido como RDSTLS. Este modo se utiliza normalmente en una configuración con balanceo de carga en la que el servidor RDP inicial puede redireccionar la conexión a otro servidor RDP.

*vmconnect:* El proceso de negociación que selecciona el modo de seguridad en base a los protocolos de seguridad soportados por el cliente y el servidor se limita a protocolos soportados por *Hyper-V/VMConnect*.

*rdp:* Es una opción para el cifrado *RDP Legacy*, usado para servidores Windows antiguos o cuando se desea una pantalla de inicio de sesión de Windows. Las nuevas versiones de Windows poseen este modo deshabilitado por defecto y solo aceptarán NLA a excepción de que se configure explícitamente de otra forma.

**-ignore-cert:** Si se establece a “*true*” el certificado devuelto por el servidor será ignorado, incluso si este no puede ser validado. Esta opción es útil si se confía completamente en el servidor y en la conexión, así como que se conoce que el certificado no puede ser validado.

**-disable-auth:** Si se establece a “*true*”, la autenticación llevada a cabo mientras se produce la conexión se deshabilitará. Cualquier otra autenticación forzada por el servidor remoto seguirá habilitada.

RDP posee también una serie de parámetros para funcionalidades adicionales como son la normalización del portapapeles (**normalize-clipboard**), las opciones de sesión (**client-name, console, initial-program, server-layout, timezone**), las opciones de *Display* (**color-depth, width, height, dpi, resize-method, force-lossless**), la redirección de dispositivos (**disable-audio, enable-audio-input, enable-touch, enable-printing, printer-name, enable-drive, disable-download, disable-upload, drive-name, drive-path, create-drive-path, console-audio, static-channels**), entre otros. Todos estos parámetros se pueden observar en la sección “*RDP*” de [47].

Una configuración simple, usando el sistema de autenticación por defecto que ofrece Apache Guacamole (*user-mapping.xml*) que se explica en la sección “2.3.-Autenticación de usuarios” de este informe, es:

```
<connection name="Nombre único">
  <protocol>rdp</protocol>
  <param name="hostname"> <<IP | Nombre Host>> </param>
  <param name="port">3389</param>
</connection>
```

Esta conexión, de nombre “*Nombre único*” usará el protocolo RDP para conectarse al anfitrión, dado por su dirección IP o nombre de *host*, por su puerto 3389.

## Habilitar RDP

Los equipos informáticos a los que se desee realizar conexiones remotas deben disponer de un servidor RDP instalado y configurado. Si los equipos poseen como sistema operativo *Windows* y, en el caso de *Windows 10*, las versiones *Windows 10 Pro* o *Windows 10 Enterprise*, se dispone de un servidor integrado cuya habilitación puede realizarse de forma nativa. Para la habilitación de estas, se debe acceder a *Configuración-Sistema-Escritorio Remoto* y habilitar la opción deslizando el botón presente para ello. Se recomienda no cambiar opciones de configuración como la autenticación NLA que aparece habilitada por defecto. Por otro lado, el cortafuegos debe permitir las conexiones remotas por el puerto correspondiente.

Si no se dispone de servidor integrado, se debe instalar alguna de las soluciones de terceros disponibles en el mercado y habilitar las conexiones de la forma adecuada.

### 2.2.6.- SSH (*Secure Shell*)

A diferencia de los protocolos anteriores, SSH es un protocolo de texto. Su implementación en Apache Guacamole [47] consiste en una combinación de una emulación de terminal realizada en el servidor y renderizada remotamente en el cliente (conexión remota) y un cliente SSH. El soporte para SSH se proporciona en la librería *libguac-client-ssh* si las dependencias opcionales para este protocolo son instaladas.

Por defecto, Apache Guacamole no realiza ninguna verificación o identificación de host antes de establecer conexiones SSH. Para hacer uso de estas verificaciones, Guacamole incluye dos métodos para verificar la identidad SSH (y SFTP (*Secure File Transfer Protocol*)) del sistema servidor.

El primer método consiste en leer un fichero ubicado en **GUACAMOLE\_HOME** denominado *ssh\_known\_hosts*. Este fichero debe poseer el formato del fichero *known\_hosts* de un estándar *OpenSSH*. Si el fichero está presente, las identidades de los host remotos son verificadas haciendo uso de las claves incluidas en el fichero en el momento de la conexión.

El segundo método consiste en pasar un parámetro de conexión que contiene una entrada *known\_hosts* de *OpenSSH* para el host al que se quiere conectar.

Para las conexiones SSH se usa el parámetro *host-key* y para las conexiones SFTP asociadas con RDP y VNC se usa el parámetro *sftp-host-key*.

Como parámetros de red, SSH posee los siguientes:

**-hostname:** El nombre o dirección IP del sistema servidor al que Apache Guacamole debe conectarse.

**-port:** El puerto en el que el sistema servidor escucha para conexiones de este protocolo. Si no se indica, se establecerá por defecto el puerto 22.

**-host-key:** La entrada *known hosts* para el sistema servidor. Es un parámetro opcional que, proporcionado, provocará que se realicen procesos de verificación de la identidad del sistema servidor.

**-server-alive-interval:** Por defecto el cliente SSH no envía peticiones *keepalive* hacia el servidor. Con este parámetro se permite la configuración del intervalo, en segundos, a partir del cual la conexión del cliente envía paquetes *keepalive* hacia el servidor. Por defecto su valor es 0, que desactiva el envío de paquetes y el valor mínimo es 2.

Con respecto a la autenticación de usuarios, SSH proporciona autenticación a través de contraseñas y a través de claves públicas. Además, proporciona el método “*NONE*” que permite la autenticación proporcionando únicamente el nombre de usuario.

Para que Apache Guacamole proporcione autenticación a través de claves públicas, debe tener acceso a la clave privada y, si se establece, a su *passphrase*. Si la clave privada requiere de esta *passphrase* y no se ha establecido, se preguntará al usuario para establecerla. Si no se proporciona clave privada, Guacamole intentará autenticar a través de contraseña, leyendo a partir de los parámetros de la conexión, si se proporcionan, o preguntando al usuario. Los parámetros válidos son el nombre de usuario, su contraseña, una clave privada y una *passphrase*.

SSH también permite, a través de ciertos parámetros, ejecutar una orden en lugar de un *shell* (parámetro **command**), así como opciones de internacionalización y localización (parámetros **locale** y **timezone**).

SSH proporciona, además, soporte para la transferencia de archivos usando SFTP (*Secure File Transfer Protocol*), para lo que dispone de una serie de parámetros:

**-enable-sftp**: Si se establece a “*true*”, se permitirá al usuario la carga y descarga de ficheros desde el servidor SSH usando SFTP. Guacamole incluye una utilidad, **guacctl**, que controla estas transferencias cuando se ejecutan en el servidor SSH a través de la conexión del usuario.

**-sftp-root-directory**: Es el directorio que se muestra a los usuarios conectados a través del explorador de archivos de Guacamole. Si se omite, se usará el directorio raíz.

**-sftp-disable-download**: Si se establece a “*true*”, las descargas de archivos desde el servidor SSH estarán deshabilitadas. Requiere que SFTP esté habilitado para tener efecto.

**-sftp-disable-upload**: Si se establece a “*true*”, las cargas de archivos desde el servidor SSH estarán deshabilitadas. Requiere que SFTP esté habilitado para tener efecto.

Una configuración simple, usando el sistema de autenticación por defecto que ofrece Apache Guacamole (*user-mapping.xml*) que se explica en la sección “2.3.-Autenticación de usuarios” de este informe, es:

```
<connection name="Nombre único">
  <protocol>ssh</protocol>
  <param name="hostname"> <<IP | Nombre Host>> </param>
  <param name="port">22</param>
</connection>
```

Esta conexión, de nombre “*Nombre único*” usará el protocolo SSH para conectarse al anfitrión dado por su dirección IP o nombre de *host* por su puerto 22.

## Instalación de Servidor SSH

Los equipos informáticos a los que se desee realizar conexiones remotas deben disponer de un servidor SSH instalado que permita responder a las solicitudes del cliente SSH, incluido en el soporte de Apache Guacamole para este protocolo. Para el sistema CentOS 7, este servicio viene normalmente preinstalado y solo se debe habilitar. Sin embargo, si no es así, los pasos a seguir para su instalación [57] son los siguientes:

1. Actualizar el sistema e instalar el servidor SSH:

```
-yum update  
-yum install openssh-server
```

2. Opcionalmente, si se desea disponer de un cliente SSH en el propio servidor para realizar pruebas de conexión:

```
-yum install openssh-clients
```

3. Añadir el servicio al cortafuegos y recargarlo:

```
-firewall-cmd --permanent --zone=public --add-service=ssh  
-firewall-cmd --reload
```

4. Por último, iniciar el servicio y, si se desea, habilitarlo en el arranque del sistema:

```
-systemctl start sshd  
-systemctl enable sshd
```

### 2.2.7.- Telnet

Telnet es un protocolo de texto que proporciona funcionalidades similares a SSH. Por naturaleza, es un protocolo no cifrado y no proporciona soporte para la transferencia de ficheros. Apache Guacamole [47] realiza, al igual que para SSH, una emulación de una terminal en el lado servidor que se renderiza en la parte cliente. El soporte para Telnet se proporciona en la librería *libguac-client-telnet* si las dependencias opcionales para este protocolo son instaladas.

Como parámetros de red, Telnet posee los siguientes:

**-hostname:** El nombre o dirección IP (*Internet Protocol*) del sistema servidor al que Apache Guacamole debe conectarse.

**-port:** El puerto en el que el sistema servidor escucha para conexiones de este protocolo. Si no se indica, se establecerá por defecto el puerto 23.

Con respecto a la autenticación de usuarios, Telnet no proporciona vías estándar de autenticación. La autenticación depende completamente del proceso de inicio de sesión que se ejecuta en el sistema servidor. Apache Guacamole proporciona mecanismos de transferencia de nombre de usuario y contraseña no estándares cuyo funcionamiento depende también del proceso de inicio de sesión que se ejecuta en el sistema servidor. Los parámetros de este mecanismo, en caso de usarse, se pueden observar en la sección “*Telnet*”, concretamente en “*Authentication*” de [47].

Una configuración simple, usando el sistema de autenticación por defecto que ofrece Apache Guacamole (*user-mapping.xml*) que se explica en la siguiente sección de este informe, es:

```
<connection name="Nombre único">
  <protocol>telnet</protocol>
  <param name="hostname"> <<IP | Nombre Host>> </param>
  <param name="port">23</param>
</connection>
```

Esta conexión, de nombre “*Nombre único*” usará el protocolo Telnet para conectarse al anfitrión dado por su dirección IP o nombre de *host* por su puerto 23.

## **Instalación de servidor Telnet**

Los sistemas informáticos a los que se desee realizar conexiones remotas deben disponer de un servidor Telnet instalado que permita responder a las solicitudes del cliente Telnet, incluido en el soporte de Apache Guacamole para este protocolo.

Este servidor se puede instalar de forma sencilla en un sistema CentOS 7 y los pasos a seguir para ello [58] son los siguientes:

1. Actualizar el sistema e instalar el servidor Telnet:

```
-yum update
-yum install telnet-server
```

2. Opcionalmente, si se desea disponer de un cliente Telnet en el propio servidor para realizar pruebas de conexión:

```
-yum install telnet
```

3. Añadir el servicio al cortafuegos y recargarlo:

```
-firewall-cmd --permanent --zone=public --add-service=telnet  
-firewall-cmd -reload
```

4. Por último, iniciar el servicio y, si se desea, habilitarlo en el arranque del sistema:

```
-systemctl start telnet.socket  
-systemctl enable telnet.socket
```

## 2.3.- Autenticación de usuarios

En esta sección del informe se describirán y detallarán algunos de los sistemas de autenticación de usuarios y gestión de conexiones de los que dispone Apache Guacamole.

En primer lugar, se explicará el método por defecto que ofrece Apache Guacamole para la autenticación de usuarios y la gestión de sus conexiones, parcialmente descrito en los apartados anteriores.

Tras realizar esta descripción, se procederá a detallar el sistema de autenticación por Base de Datos para el que se utilizará la base de datos *MariaDB* [24].

A continuación, se continuará describiendo el sistema de autenticación por LDAP (*Lightweight Directory Access Protocol*) para finalmente explicar el sistema de autenticación por RADIUS (*Remote Authentication Dial in User Service*).

### 2.3.1.- Autenticación por defecto (*user-mapping.xml*)

Por defecto, el módulo de autenticación de Apache Guacamole consiste en una asociación entre nombres de usuario y configuraciones establecidas. Este módulo lee los nombres de usuario y contraseñas a partir de un fichero XML (*Extensible Markup Language*) denominado *user-mapping.xml* y ubicado en *GUACAMOLE\_HOME*.

El módulo siempre está habilitado, pero solo lee a partir del fichero XML si este existe. Además, posee la prioridad más baja en actuación con respecto a otros sistemas de autenticación que puedan estar implementados.

Cada usuario se especifica con una etiqueta **<authorize>**. Esta etiqueta contiene todas las conexiones autorizadas para cada usuario concreto, estando cada una representada con la etiqueta **<connection>**. Cada conexión posee un protocolo para el que Apache Guacamole ofrece soporte y una serie de parámetros relacionados con este protocolo. El protocolo se denota con la etiqueta **<protocol>** y sus parámetros con la etiqueta **<param>**.

En la etiqueta **<authorize>** se especifica el nombre de usuario y su contraseña. Dicha contraseña puede ser representada en texto plano o mediante su representación *hash* obtenida tras aplicar el algoritmo MD5 (*Message-Digest 5*). Si se desea añadir un nuevo usuario, bastará con añadir un nuevo bloque con esta etiqueta. Los cambios, una vez guardados, se aplicarán inmediatamente debido a que Apache Guacamole vuelve a leer el fichero sin necesidad de reiniciar el *Servlet*.

Para añadir conexiones a un usuario, los pasos a seguir dependen de dos situaciones. En primer lugar, si el usuario sólo poseerá una conexión, se necesita únicamente especificarla con las etiquetas **<protocol>** y **<param>**. Si el usuario poseerá más de una conexión, se requiere hacer uso de la etiqueta **<connection>** para cada una. Un ejemplo del formato de este fichero es el siguiente:

```
<user-mapping>
  <!-- Per-user authentication and config information -->
  <authorize username="USERNAME" password="PASSWORD">
    <protocol>vnc</protocol>
    <param name="hostname">localhost</param>
    <param name="port">5900</param>
    <param name="password">VNCPASS</param>
  </authorize>

  <!-- Another user, but using md5 to hash the password
  (example below uses the md5 hash of "PASSWORD") -->
  <authorize
    username="USERNAME2"
    password="319f4d26e3c536b5dd871bb2c52e3178"
    encoding="md5">

    <!-- First authorized connection -->
    <connection name="localhost">
      <protocol>vnc</protocol>
      <param name="hostname">localhost</param>
      <param name="port">5901</param>
      <param name="password">VNCPASS</param>
    </connection>

    <!-- Second authorized connection -->
    <connection name="otherhost">
      <protocol>vnc</protocol>
      <param name="hostname">otherhost</param>
      <param name="port">5900</param>
      <param name="password">VNCPASS</param>
    </connection>

  </authorize>
</user-mapping>
```

**Figura 60:** Formato del fichero *user-mapping.xml*. [47] (Manual de Usuario)

### 2.3.2.- Autenticación por Base de Datos (*MariaDB*)

Apache Guacamole permite usar Bases de Datos como sistema de autenticación [49], existiendo soporte para *MariaDB*, *MySQL*, *PostgreSQL* y *SQL Server*. Este soporte se proporciona a través de las extensiones disponibles en su sitio web. Usar una base de datos como sistema de autenticación proporciona características adicionales, como la posibilidad de usar grupos de conexiones y la existencia de una interfaz administrativa basada en la aplicación web de Guacamole. Los cambios realizados a usuarios y conexiones se aplican inmediatamente, sin la necesidad de que los usuarios reinicien su sesión para ver las nuevas conexiones.

Mientras que la mayoría de las extensiones de autenticación funcionan independientemente, la autenticación por base de datos puede actuar en un rol subordinado, permitiendo que los usuarios y grupos de usuarios pertenecientes a otras extensiones de autenticación sean asociados con conexiones internas a la base de datos. Los usuarios y grupos de usuarios son considerados idénticos a aquellos que residan en la base de datos si poseen los mismos nombres y el resultado de la autenticación de otra extensión será de confianza si es exitoso. Por tanto, un usuario con una cuenta registrada en varios sistemas de autenticación podrá observar datos de cada sistema cuando inicie sesión.

Para hacer uso de la extensión de autenticación por base de datos, se necesita:

- Alguna de las bases de datos mencionadas anteriormente. Se hará uso de *MariaDB* en este informe.
- Suficientes permisos para crear nuevas bases de datos, para crear nuevos usuarios y para concederles permisos de usuario.
- Acceso red a la base de datos desde el servidor Guacamole.

Para implementar este sistema de autenticación por base de datos usando *MariaDB* se deben seguir los siguientes pasos:

1. Descargar desde el mismo sitio web donde se descarga la parte cliente y servidor de Apache Guacamole (indicado en [43]) el fichero *guacamole-auth-jdbc-1.4.0.tar.gz* que se corresponde con la extensión correspondiente de Guacamole. Este fichero contiene, de forma empaquetada, diferentes directorios correspondientes a cada Base de Datos para la que Guacamole proporciona soporte. Cada directorio posee un subdirectorio denominado **schema/** y un fichero con extensión *.jar* (es la propia extensión de Guacamole). Se debe desempaquetar el fichero *guacamole-auth-jdbc-1.4.0.tar.gz* de la siguiente forma:

```
-tar -xzf guacamole-auth-jdbc-1.4.0.tar.gz
```

En el caso de *MariaDB*, el directorio **schema/** es *mysql/schema/*, y contiene scripts SQL para inicializar la base de datos, así como un directorio “*upgrade*” destinado a la actualización para distintas versiones de Guacamole. Además, el fichero *.jar* correspondiente es *mysql/guacamole-auth-jdbc-mysql-1.4.0.jar* y debe ser ubicado en **GUACAMOLE\_HOME/extensions**.

2. Adicionalmente, se requiere de un *driver* JDBC (*Java Database Connectivity*) para que la aplicación web de Guacamole, programada en lenguaje Java, pueda conectarse a través de este mecanismo a la base de datos. Este driver se debe obtener del sitio web indicado en [50], seleccionando la opción “**Plataforma independiente**”, debido a que CentOS7 no se halla en las opciones del desplegable, y descargando el fichero con extensión *.tar.gz* presente. Posteriormente, se debe desempaquetar este fichero y ubicar el archivo con extensión *.jar* en **GUACAMOLE\_HOME/lib**. En el sitio web indicado, se puede observar que el JDBC descargado es *mysql-connector-java-8.0.28.jar* y no el correspondiente a *MariaDB*. Esto es debido a que, tras realizar varias pruebas, no se detectaba el JDBC de *MariaDB* por parte de Guacamole, a diferencia del correspondiente a *MySQL*, que funciona debido a que *MariaDB* es una bifurcación de *MySQL* y, por ello, poseen compatibilidades entre sí.

3. A continuación, se debe crear la base de datos en la que el módulo de autenticación por base de datos de Apache Guacamole almacenará los datos de autenticación, así como un usuario requerido solo para acceder a los datos y manipularlos. Para ello, instalar en primer lugar *MariaDB* en CentOS7 [51] de la siguiente forma.

3.1. Actualizar el sistema.

```
-yum update
```

3.2. Añadir un nuevo repositorio para *yum* con el objetivo de instalar la versión 10.6, recomendada en [24], de *MariaDB*.

```
-gedit /etc/yum.repos.d/mariadb-10.6.repo
[mariadb]
name = MariaDB 10.6 para CentOS 7
baseurl = http://yum.mariadb.org/10.6.7/centos7-amd64
gpgkey=https://yum.mariadb.org/RPM-GPG-KEY-MariaDB
gpgcheck=1
```

### 3.3. Guardar los cambios del fichero y actualizar las listas de paquetes.

```
-yum update
```

### 3.4. Instalar el servidor de *MariaDB*.

```
-yum install -y MariaDB-server
```

### 3.5. Iniciar y habilitar en el arranque del sistema el nuevo servicio.

```
-systemctl start mariadb
-systemctl enable mariadb
```

3.6. Ejecutar el script de seguridad *mariadb-secure-installation*. Responder a las preguntas de la forma siguiente: modificar la contraseña, si se desea, del usuario *root*, denegar la autenticación por *socket*, eliminar usuarios anónimos, configurar el acceso del usuario *root* como exclusivamente local, eliminar las bases de datos de test y aplicar todos los cambios de forma inmediata.

```
-mariadb-secure-installation
```

3.7. Establecer el juego de caracteres a la codificación para el idioma español, *utf8mb4*, accediendo al fichero indicado en la siguiente orden y añadiendo la línea mostrada a continuación en la sección *[mysqld]*.

```
-gedit /etc/my.cnf.d/server.cnf
-character_set_server=utf8mb4
```

3.8. Por último, reiniciar el servicio.

```
-systemctl restart mariadb
```

4. Una vez instalado y configurado el nuevo servicio, el siguiente paso es iniciar sesión en el monitor MySQL como usuario *root* y crear la base de datos de Guacamole. Posteriormente, se deben ejecutar los scripts SQL (*Structured Query Language*) *001-create-schema.sql* y *002-create-admin-user.sql* incluidos en el directorio **schema/**. La forma de realizar estos pasos es la de la siguiente figura donde se crea una base de datos denominada *guacamole\_db* y se ejecutan los scripts mencionados.

```
$ mysql -u root -p
Enter password:
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 233
Server version: 5.5.29-0ubuntu0.12.10.1 (Ubuntu)

Copyright (c) 2000, 2012, Oracle and/or its affiliates. All rights reserved.

Oracle is a registered trademark of Oracle Corporation and/or its
affiliates. Other names may be trademarks of their respective
owners.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

mysql> CREATE DATABASE guacamole_db;
Query OK, 1 row affected (0.00 sec)

mysql> quit
Bye
$ ls schema/
001-create-schema.sql 002-create-admin-user.sql upgrade
$ cat schema/*.sql | mysql -u root -p guacamole_db
Enter password:
$
```

**Figura 61:** Creación de la base de datos y ejecución de consultas. [49] (Manual de Usuario)

5. Para que Guacamole sea capaz de ejecutar consultas en la base de datos, se debe crear un nuevo usuario y concederle suficientes privilegios para gestionar los contenidos de todas las tablas de la base de datos. Este usuario necesita los permisos *SELECT*, *UPDATE*, *INSERT* y *DELETE* en todas las tablas de la base de datos. En la siguiente figura, se puede observar las diferentes consultas SQL a ejecutar donde “*guacamole\_user*” y “*some\_password*” se indican a modo de ejemplo, pero deben cambiarse.

```
$ mysql -u root -p
Enter password:
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 233
Server version: 5.5.29-0ubuntu0.12.10.1 (Ubuntu)

Copyright (c) 2000, 2012, Oracle and/or its affiliates. All rights reserved.

Oracle is a registered trademark of Oracle Corporation and/or its
affiliates. Other names may be trademarks of their respective
owners.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

mysql> CREATE USER 'guacamole_user'@'localhost' IDENTIFIED BY 'some_password';
Query OK, 0 rows affected (0.00 sec)

mysql> GRANT SELECT,INSERT,UPDATE,DELETE ON guacamole_db.* TO 'guacamole_user'@'localhost';
Query OK, 0 rows affected (0.00 sec)

mysql> FLUSH PRIVILEGES;
Query OK, 0 rows affected (0.02 sec)

mysql> quit
Bye
$
```

**Figura 62:** Creación del usuario para Apache Guacamole y concesión de permisos. [49] (Manual de Usuario)

6. Para configurar Apache Guacamole para usar la autenticación por Base de Datos, se deben añadir una serie de propiedades al fichero *guacamole.properties*. Estas propiedades son específicas de cada base de datos y las que son absolutamente requeridas son las siguientes:

**-mysql-hostname:** Nombre de host o dirección IP del servidor que sustenta la base de datos.

**-mysql-database:** Nombre de la base de datos creada (*guacamole\_db* en este caso).

**-mysql-username:** Nombre del usuario creado (*guacamole\_user* en este caso).

**-mysql-password:** Contraseña del usuario creado (*some\_password* en este caso).

Una configuración mínima de *guacamole.properties* para realizar conexiones a una base de datos ubicada de manera local es la siguiente:

```
# MySQL properties
mysql-hostname: localhost
mysql-database: guacamole_db
mysql-username: guacamole_user
mysql-password: some_password
```

7. Existen parámetros opcionales que permiten controlar como Guacamole se conecta al servidor de base de datos (**mysql-port**, **mysql-driver**, **mysql-server-timezone**, **mysql-ssl-mode**, **mysql-ssl-trust-store**, **mysql-ssl-trust-password**, **mysql-ssl-client-store**, **mysql-ssl-client-password** ) que se pueden observar en detalle en la guía usada en este apartado [49] en su sección “*Configuring Guacamole for database authentication*”.
8. Asimismo, existe una serie de parámetros para distintas funcionalidades, como la gestión de contraseñas, la gestión de conexiones concurrentes, la restricción de la autenticación a únicamente usuarios de la base de datos, entre otros. Todos estos parámetros pueden observarse en [49] en los subapartados correspondientes.
9. Para finalizar la instalación, se debe reiniciar el *Servlet* para que Guacamole cargue las nuevas extensiones. Una vez reiniciado, si Guacamole funciona correctamente significa que no existen errores de configuración en la extensión de autenticación por base de datos.
10. Para iniciar sesión, los script SQL ejecutados anteriormente crean un usuario por defecto denominado “*guacadmin*” con una contraseña por defecto “*guacadmin*”. Este usuario se corresponde con el administrador principal y posee todos los permisos para gestionar usuarios y conexiones, entre otras opciones. Una vez se verifique el correcto inicio de sesión, se debe modificar su contraseña.
11. Si se desea modificar los datos que soportan el módulo de autenticación ejecutando consultas SQL manualmente, se dispone de información de las tablas creadas por los scripts SQL en la sección “*Modifying data manually*” de [49].
12. Para obtener instrucciones detalladas sobre la gestión de usuarios y conexiones a través de la interfaz web, se debe acudir a la sección “*Administration*” [52] del sitio web de Apache Guacamole.

### 2.3.3.- Autenticación por LDAP

Apache Guacamole ofrece soporte para la autenticación LDAP (*Lightweight Directory Access Protocol*) [59] por vía de una extensión disponible en su sitio web. Esta extensión permite que los usuarios y conexiones sean almacenados directamente en un directorio LDAP. Si se posee un sistema de autenticación central que usa LDAP, el soporte de Guacamole puede ser una buena forma de permitir a los usuarios usar sus nombres de usuario y contraseñas existentes en este sistema para iniciar sesión en Guacamole. Para usar la autenticación LDAP se necesita:

-Un directorio LDAP como almacenamiento para toda la información de autenticación, como *OpenLDAP*.

-La posibilidad de modificar el esquema del directorio LDAP.

## Instalación de OpenLDAP

Para instalar y configurar un directorio LDAP, concretamente *OpenLDAP*, se deben seguir los pasos siguientes [60]:

1. Actualizar el sistema e instalar los componentes requeridos:

```
-yum update
-yum install openldap-clients openldap-servers
```

2. Iniciar y habilitar en el arranque del sistema el nuevo servicio:

```
-systemctl start slapd
-systemctl enable slapd
```

3. Crear la contraseña del usuario administrador que se establecerá posteriormente. Importante guardar el *hash* que devuelve la ejecución de la orden:

```
-slappasswd
```

4. Crear un fichero con extensión *.ldif* con el siguiente contenido agregándole el *hash* de la contraseña guardado para establecer la contraseña raíz en el directorio LDAP. Añadir posteriormente el fichero creado al directorio con la orden que se muestra a continuación tras el contenido del fichero:

```
dn: olcDatabase={0}config,cn=config
changetype: modify
add: olcRootPW
olcRootPW: {SSHA}xxxxxxxxxxxxxxxxxxxxxxxxxxxx
```

```
-ldapadd -Y EXTERNAL -H ldapi:/// -f fichero.ldif
```

5. Añadir los siguientes ficheros de esquema estándar de LDAP necesarios para la autenticación de usuarios.

```
-ldapadd -Y EXTERNAL -H ldapi:/// -f /etc/openldap/schema/cosine.ldif
-ldapadd -Y EXTERNAL -H ldapi:/// -f /etc/openldap/schema/nis.ldif
-ldapadd -Y EXTERNAL -H ldapi:/// -f /etc/openldap/schema/inetorgperson.ldif
```

6. A continuación, se muestra un contenido ejemplo para agregar a un fichero con extensión *.ldif* con el objetivo de configurar el dominio raíz y un administrador de las distintas Bases de Datos del directorio LDAP estableciendo aquellas características necesarias, como contraseñas, listas de acceso para el administrador, sufijos del dominio, entre otros. El contenido mostrado debe personalizarse según las características del directorio LDAP a instalar:

```
dn: olcDatabase={1}monitor,cn=config
changetype: modify
replace: olcAccess
olcAccess: {0}to *
    by dn.base="gidNumber=0+uidNumber=0,cn=peercred,cn=external,cn=auth" read
    by dn.base="cn=admin,dc=local,dc=lan" read
    by * none
```

```
dn: olcDatabase={2}hdb,cn=config
changetype: modify
replace: olcSuffix
olcSuffix: dc=local,dc=lan
```

```
dn: olcDatabase={2}hdb,cn=config
changetype: modify
replace: olcRootDN
olcRootDN: cn=admin,dc=local,dc=lan
```

```
dn: olcDatabase={2}hdb,cn=config
changetype: modify
add: olcRootPW
olcRootPW: {SHA}xxxxxxxxxxxxxxxxxxxxxxxxxxxx
```

```
dn: olcDatabase={2}hdb,cn=config
changetype: modify
add: olcAccess
olcAccess: {0}to attrs=userPassword,shadowLastChange
  by dn="cn=admin,dc=local,dc=lan" write
  by anonymous auth
  by self write
  by * none
olcAccess: {1}to dn.base=""
  by * read
olcAccess: {2}to *
  by dn="cn=admin,dc=local,dc=lan" write
  by * read
```

#### 7. Añadir el archivo a la configuración de *OpenLDAP*:

```
-ldapmodify -Y EXTERNAL -H ldapi:/// -f fichero.ldif
```

8. Tal y como se mencionó en el paso 6, se muestra a continuación el contenido de un fichero a modo de ejemplo para crear nodos de usuario, nodos de grupo y superadministradores de bases de datos:

```
dn: dc=local,dc=lan
objectClass: top
objectClass: dcObject
objectclass: organization
o: Example Inc.
dc: local

dn: ou=user,dc=local,dc=lan
objectClass: organizationalUnit
ou: user

dn: ou=group,dc=local,dc=lan
objectClass: organizationalUnit
ou: group
```

```
dn: cn=admin,dc=local,dc=lan
objectClass: organizationalRole
cn: admin
description: Directory Administrator
```

9. Añadir el archivo a la configuración de *OpenLDAP* bajo el rol de administrador:

```
-ldapadd -x -D cn=admin,dc=local,dc=lan -W -f fichero.ldif
```

10. Por último, permitir en el cortafuegos el nuevo servicio:

```
-firewall-cmd --permanent --add-service=ldap
-firewall-cmd --reload
```

Para la gestión del directorio LDAP, se puede hacer uso de gestores como *Apache Directory Studio* [61], compatible con *OpenLDAP*, que facilitan las diferentes acciones que se deseen realizar.

## Autenticación de Usuarios

Apache Guacamole autentica a los usuarios en el servidor LDAP realizando un intento de enlace entre el nombre de usuario y contraseña proporcionados en el inicio de sesión y aquellos propios del servidor LDAP. Si el enlace resulta exitoso, el conjunto de conexiones disponibles de Guacamole es consultado hacia el directorio LDAP ejecutando una consulta LDAP como el usuario enlazado. Cada conexión Guacamole se representa en el directorio LDAP como un tipo de grupo especial: *guacConfigGroup*. Los atributos asociados con el grupo definen el protocolo y los parámetros de conexión, y se permite a los usuarios el acceso a la conexión solo si están asociados con dicho grupo. Esta arquitectura posee una serie de beneficios:

1. Los usuarios pueden usar sus nombres de usuario y contraseñas ya existentes en LDAP.
2. Las conexiones de Guacamole pueden ser gestionadas con la misma herramienta usada para gestionar el directorio LDAP.
3. Las restricciones de seguridad existentes pueden limitar la visibilidad/accesibilidad de las conexiones de Guacamole.
4. El acceso a las conexiones puede ser fácilmente concedido o revocado, debido a que cada conexión está representada por un grupo.

La autenticación LDAP puede configurarse con Apache Guacamole de dos formas dependiendo si se dispone, instalado y configurado, de soporte para Base de Datos y se desea almacenar los datos de configuración de conexiones en dicha Base de Datos o, por el contrario, no se dispone de soporte para Base de Datos o se desea guardar estos datos en el propio directorio LDAP.

Si se ha instalado el soporte de Apache Guacamole para autenticación LDAP y para autenticación por Base de Datos, Guacamole intentará, automáticamente, autenticar a los usuarios que intenten iniciar sesión contra los dos sistemas de autenticación.

Las cuentas de usuario y grupos se considerarán equivalentes a aquellas incluidas en una Base de Datos si sus nombres únicos de usuarios y grupos, determinados por los parámetros *ldap-username-attribute* y *ldap-group-name-attribute* que se explicarán posteriormente, son idénticos. Si ocurre esto, una cuenta de usuario puede ver aquellos objetos pertenecientes al directorio LDAP y a la Base de Datos que estén asociados a su cuenta.

Además, si una cuenta con permisos de administrador incluida en la Base de Datos, como el usuario *guacadmin* proporcionado en el soporte para este sistema de autenticación, posee una cuenta de usuario correspondiente en el directorio LDAP con suficientes permisos, la interfaz web de Guacamole incluirá a los usuarios y grupos de este directorio y permitirá que el administrador asocie conexiones desde la Base de Datos con esas cuentas de usuario del directorio LDAP.

Si no se ha instalado el soporte de Apache Guacamole para autenticación por Base de Datos, o si se ha instalado, pero no se desea almacenar datos de conexiones en ella y si en el directorio LDAP, se deben realizar algunas modificaciones requeridas en el esquema LDAP aplicando alguno de los ficheros de esquema proporcionados desde el sitio web de Apache Guacamole. Dichos ficheros definen una clase de objeto adicional denominada *guacConfigGroup* que contiene toda la información de configuración para una conexión determinada y que puede ser asociada con usuarios y grupos. Cada conexión de este grupo será accesible solo por aquellos usuarios que sean miembros o aquellos usuarios que sean miembros de grupos asociados.

Los ficheros de esquema proporcionados se describen a continuación. De ellos, solo uno es necesario en base a si se utiliza el directorio *OpenLDAP* u otro directorio de los disponibles en el mercado.

**-guacConfigGroup.schema:** Un fichero que describe el esquema y cumple con los estándares. Este fichero se puede usar en cualquier directorio que cumpla con el estándar RFC-2252. [62]

**-guacConfigGroup.ldif:** Es un fichero LDIF (*LDAP Data Interchange Format*) compatible con *OpenLDAP*. Fue creado a partir del fichero con extensión *.schema* comentado anteriormente.

A continuación, se muestran las ordenes necesarias para añadir el fichero con extensión *.ldif* al directorio *OpenLDAP*. Además, se presenta una orden, a modo de comprobación, que realiza una búsqueda y debería obtener la nueva clase de objeto.

```
-ldapadd -Q -Y EXTERNAL -H ldapi:/// -f schema/guacConfigGroup.ldif
-ldapsearch -Q -LLL -Y EXTERNAL -H ldapi:/// -b cn=schema,cn=config dn
```

## Instalación y Configuración del sistema de autenticación LDAP

La extensión LDAP de Apache Guacamole se puede obtener desde el sitio web donde se descarga la parte cliente y servidor de Apache Guacamole (indicado en [43]) y se corresponde con el fichero denominado *guacamole-auth-ldap-1.4.0.tar.gz*. Este fichero contiene, de forma empaquetada, el fichero *guacamole-auth-ldap-1.4.0.jar* que es la extensión en sí y un directorio denominado **schema/** que contiene los ficheros de esquema LDAP. Entre estos, se encuentra un fichero con extensión *.ldif*, compatible con *OpenLDAP*, así como un fichero con extensión *.schema* compatible con el estándar RFC-2252 [62]. El fichero con extensión *.schema* puede ser transformado en el fichero con extensión *.ldif* automáticamente.

Para instalar el sistema de autenticación LDAP, se debe ubicar el fichero *guacamole-auth-ldap-1.4.0.jar*, mencionado anteriormente, en el directorio **GUACAMOLE\_HOME/extensions** y se debe configurar Apache Guacamole para usar la autenticación LDAP de la forma comentada a continuación.

Para configurar Apache Guacamole para usar la autenticación LDAP, se presenta a continuación algunas de las propiedades importantes a añadir en el fichero *guacamole.properties*, de las cuales solo *ldap-user-base-dn* es obligatoria.

**-ldap-hostname:** El nombre de host del servidor LDAP. Si se omite, se usará “*localhost*” por defecto.

**-ldap-port:** El puerto donde escucha el servidor LDAP. Si se omite, se usará el puerto estándar de LDAP o su versión cifrada LDAPS, en base al método de cifrado especificado en el siguiente parámetro. LDAP sin cifrado hace uso del puerto 389 y LDAPS hace uso del puerto 636.

**-ldap-encryption-method:** El mecanismo de cifrado que Guacamole debe usar cuando se comunica con el servidor LDAP. Los valores pueden ser: “*none*” para LDAP sin cifrar, “*ssl*” para cifrado SSL/TLS o “*starttls*” para STARTTLS. Si se omite, no se hará uso de cifrado.

**-ldap-search-bind-dn:** El nombre distinguido (DN) del usuario al que Guacamole debe enlazar cuando autentica usuarios que tratan de iniciar sesión. Si se especifica, Guacamole consultará al directorio LDAP para determinar el DN de cada usuario que inicia sesión. Si se omite, el DN de cada usuario se derivará directamente usando el DN especificado en el parámetro *ldap-user-base-dn*.

**-ldap-search-bind-password:** La contraseña a proporcionar al servidor LDAP cuando se enlaza como el valor indicado en *ldap-search-bind-dn* para autenticar otros usuarios. Esta propiedad solo se usa si establece la propiedad *ldap-search-bind-dn*. Si se omite, pero se especifica *ldap-search-bind-dn*, Guacamole intentará enlazarse con el servidor LDAP sin hacer uso de contraseña.

**-ldap-user-base-dn:** Es el nombre distinguido (DN) base para todos los usuarios de Guacamole. Esta propiedad es la única que es requerida establecer para el funcionamiento del soporte LDAP. Si se establece el parámetro *ldap-search-bind-dn*, los usuarios de Guacamole solo tienen que estar ubicados en el subárbol del DN base de usuarios especificado. Si no se establece dicho parámetro, todos los usuarios de Guacamole deben ser descendientes directos de este DN base, pues este DN se concatenará al nombre de usuario para derivar el DN de usuario.

**-ldap-username-attribute:** El atributo o los atributos que contienen el nombre de usuario entre todos los objetos de usuario de Guacamole en el directorio LDAP. Usualmente y por defecto, el valor de esta propiedad es “*uid*”. Si se desean múltiples atributos, se debe proporcionar el parámetro *ldap-search-bind-dn*. Si no se proporciona, al igual que en el parámetro anterior, se realizará una derivación a partir de *ldap-user-base-dn* y este parámetro para obtener el DN de usuario.

**-ldap-member-attribute:** El atributo que contiene los miembros en todos los objetos de grupo del directorio LDAP. Usualmente y por defecto, el valor de esta propiedad es “*member*”.

**-ldap-member-attribute-type:** Especifica si el atributo establecido en el anterior parámetro identifica al miembro de un grupo en base a su DN o código de usuario. Los valores posibles son “*dn*” (valor por defecto) o “*uid*”.

En la subsección “*Configuring Guacamole for LDAP*” de [59] se pueden observar todos estos parámetros, así como otros que indican en detalle cómo se debe comportar Apache Guacamole con el directorio LDAP. Por ejemplo, existen atributos para limitar el número de resultados en una búsqueda (**ldap-max-search-results**), para establecer el filtro de las búsquedas de usuarios (**ldap-user-search-filter**) o para establecer el límite de tiempo, en segundos, de cualquier operación LDAP (**ldap-operation-timeout**), entre otros. Una configuración mínima de *guacamole.properties* puede ser la siguiente:

```
# LDAP properties
```

```
ldap-user-base-dn: ou=people,dc=example,dc=net
```

Para completar la instalación, se debe reiniciar el Servlet para que Guacamole vuelva a leer el fichero *guacamole.properties* y, si no existen problemas de configuración de la extensión, habilite la autenticación LDAP.

## El esquema LDAP

Apache Guacamole permite la gestión de usuarios y conexiones en un directorio LDAP realizando modificaciones mínimas al esquema estándar de LDAP.

Con respecto a los usuarios, Guacamole no realiza ninguna modificación, pues enlaza a cada usuario que intenta iniciar sesión con un usuario que posea sus credenciales en el directorio LDAP. Si el inicio de sesión es exitoso, Guacamole consulta todas las conexiones asociadas con dicho usuario.

Atendiendo a las conexiones de Guacamole, si se requiere de una modificación en el esquema LDAP para la adición de un nuevo tipo de objeto, *guacConfigGroups*, que representa a las conexiones. Esta clase de objeto, que es una extensión de la clase de objetos *groupOfNames* estándar de LDAP, proporciona dos nuevos atributos adicionales a los proporcionados por dicha clase que son los siguientes:

**-guacConfigProtocol:** Representa al protocolo asociado con la conexión, representado como *vnc* o *rdp*, entre otros. Este atributo es requerido para cada objeto y solo se puede proporcionar una vez.

**-guacConfigParameter:** El nombre y el valor de un parámetro para un protocolo determinado. Se puede proporcionar múltiples veces para la misma conexión.

Por ejemplo, para crear una nueva conexión VNC hacia “*localhost*” por el puerto 5900 y asociarla a los usuarios “*user1*” y “*user2*”, se puede crear un fichero con extensión *.ldif* como el siguiente:

```

dn: cn=Example Connection,ou=groups,dc=example,dc=net
objectClass: guacConfigGroup
objectClass: groupOfNames
cn: Example Connection
guacConfigProtocol: vnc
guacConfigParameter: hostname=localhost
guacConfigParameter: port=5900
guacConfigParameter: password=secret
member: cn=user1,ou=people,dc=example,dc=net
member: cn=user2,ou=people,dc=example,dc=net
    
```

Este fichero puede agregarse al directorio LDAP con la siguiente orden, en la que se añade el fichero autenticándose con un usuario administrador del directorio:

```
-ldapadd -x -D cn=admin,dc=example,dc=net -W -f example-connection.ldif
```

O, puede hacer uso de un entorno gráfico para manipular el directorio LDAP, como el mencionado en este informe, *Apache Directory Studio*, y añadir el nuevo fichero desde sus utilidades.

### 2.3.4.- Autenticación por RADIUS

Apache Guacamole ofrece soporte para delegar la autenticación a un servicio RADIUS (*Remote Authentication Dial in User Service*) [63], como *FreeRADIUS*, con el objetivo de validar combinaciones de nombre de usuario y contraseña y para soportar la autenticación multifactor. Este sistema de autenticación debe ser ubicado en una capa superior a otros sistemas de autenticación disponibles en el sitio web de Apache Guacamole, para proporcionar acceso a conexiones reales.

La extensión para RADIUS depende de software que está cubierto por una licencia LGPL (*Lesser General Public License*), que es incompatible con la licencia Apache 2.0, bajo la cual Apache Guacamole se encuentra. Por tanto, el sitio web de Apache Guacamole no puede proporcionar versiones binarias de la extensión RADIUS.

Para solucionar esto, se debe instalar la extensión junto a la construcción de la parte cliente de Apache Guacamole, indicado en este informe en el apartado “2.1-Instalación de Apache Guacamole” y, concretamente en su subapartado “2.1.3-Instalación del cliente de Apache Guacamole (*guacamole-client*)”.

La extensión RADIUS debe habilitarse explícitamente en el tiempo de construcción para generar los binarios y el fichero con extensión *.jar* correspondiente. Por tanto, en el momento de ejecutar la orden *mvn package* tal y como se indica en dicha sección se debe ejecutar de la siguiente forma:

```
-mvn clean package -Plgpl-extensions
```

Una vez la construcción se complete, la extensión estará ubicada en el directorio *extensions/guacamole-auth-radius/target/* y el fichero se denominará *guacamole-auth-radius-1.4.0.jar*. Este fichero se debe ubicar en el directorio **GUACAMOLE\_HOME/extensions**.

Las extensiones se cargan en orden alfabético y la autenticación se realiza en el orden el que las extensiones fueron cargadas. Si se usa la extensión RADIUS con otra extensión, como la correspondiente para JDBC, se necesitará renombrar a la extensión RADIUS para que sea evaluada previo a la extensión JDBC. También se puede hacer uso del parámetro **extension-priority** en el fichero *guacamole.properties* para especificar que la extensión RADIUS se cargue en primer lugar. Si no se realiza esto, un fallo de autenticación en alguno de los módulos previos puede bloquear al módulo RADIUS para que sea evaluado.

## Configuración de Guacamole para RADIUS

Con respecto a la configuración de Guacamole para que implemente el sistema de autenticación RADIUS, se deben conocer y establecer una serie de propiedades de configuración que proporciona la extensión correspondiente. Estas propiedades definen como se debe comunicar Guacamole al servidor RADIUS para el proceso de autenticación. Como mínimo, se debe conocer nombre del servidor o dirección IP, su puerto de autenticación, el protocolo de autenticación que usa el servidor y el secreto compartido para el cliente RADIUS. Se deben configurar, a su vez, en el servidor RADIUS, estos parámetros para que Guacamole realice el proceso de autenticación de forma correcta. Estos “*items*” deben ser configurados en el fichero *guacamole.properties*:

**-radius-hostname:** El servidor RADIUS contra el que hacer intentos de autenticación. Si no se especifica, estará ubicado por defecto en “*localhost*”.

**-radius-auth-port:** El puerto de autenticación por el que el servicio RADIUS está escuchando. Si no se especifica, será el puerto 1812 por defecto.

**-radius-shared-secret:** El secreto compartido a utilizar cuando se realice la comunicación con el servidor RADIUS. Es un parámetro requerido y la extensión no se cargará si no se especifica.

**-radius-auth-protocol:** El protocolo de autenticación a utilizar cuando se realice la comunicación con el servidor RADIUS. Es un parámetro requerido para que la extensión opere de forma correcta.

Los valores permitidos son: “pap”, “chap”, “mschapv1”, “mschapv2”, “eap-md5”, “eap-tls” y “eap-ttls”. También existe soporte para PEAP (*Protected Extensible Authentication Protocol*) pero debido a una regresión en la implementación *JRadius*, actualmente no se encuentra disponible. Si se especifica “eap-ttls”, se debe establecer el parámetro **radius-eap-ttls-inner-protocol** para configurar apropiadamente el protocolo usado en el túnel EAP TTLS (*Extensible Authentication Protocol in Tunnel Transport Layer Security*).

Además de estos, existen otros parámetros aplicables según cómo se desee personalizar el comportamiento de Guacamole con el servidor RADIUS. Por ejemplo, si se quiere hacer uso de cifrado de conexiones, existen una serie de parámetros para ello (**radius-key-file**, **radius-key-type**, **radius-key-password**, **radius-ca-file**, **radius-ca-type**, **radius-ca-password**, **radius-trust-all**) al igual que para otras funcionalidades como identificar dispositivos NAS (*Network Access Server*) (**radius-nas-ip**) o para configurar el comportamiento en caso de intentos fallidos de conexión (**radius-retries**, **radius-timeout**), entre otros.

Una configuración mínima del fichero *guacamole.properties* para permitir la autenticación RADIUS es la siguiente:

```
extension-priority: radius, mysql
# RADIUS properties
radius-hostname: localhost
radius-auth-port: 1812
radius-shared-secret: XXX
radius-auth-protocol: pap
```

## Instalación y configuración de un servidor RADIUS

Para probar la extensión se necesita de un servidor RADIUS instalado y configurado correctamente.

Una solución gratuita es *FreeRadius* [35], cuyos pasos de instalación [64, 65] son los siguientes:

1. Descargar desde [66] el fichero *freeradius-server-3.X.X.tar.gz*.
2. Descomprimir el fichero:

```
-tar -xzf freeradius-server-3.X.X.tar.gz
```

3. Se debe instalar una dependencia, *Talloc*, que consiste en un alojador de memoria jerárquico necesario para las versiones 3.0.X o superiores. Además, se debe instalar, si no lo está, otra dependencia correspondiente a la Base de Datos instalada. En este informe se ha decidido optar por *MariaDB* por lo que esta dependencia es *mysql-devel*. Para ello:

```
-yum install libtalloc-devel  
-yum install mysql-devel
```

4. Tras esto, se debe ejecutar el script *configure* que viene con el fichero descargado y posteriormente realizar la instalación:

```
-cd freeradius-server-3.X.X/  
-./configure  
-make  
-make install
```

Tras seguir estos pasos, el servidor RADIUS se debe encontrar instalado en el sistema en su versión básica. A continuación, se muestra por pasos una serie de configuraciones básicas [67] para realizar un test con el servidor:

1. Por comodidad, crear un enlace simbólico para facilitar el acceso a los ficheros de configuración del servidor:

```
-ln -s /usr/local/etc/raddb/ /etc/raddb
```

2. Existen dos ficheros en RADIUS para crear clientes (son aquellos dispositivos que se conectan al servidor y cuyo fichero es */etc/raddb/clients.conf*) y usuarios (entidades que pueden autenticarse en el servidor y cuyo fichero es */etc/raddb/users*). El fichero de clientes viene por defecto con una configuración para el cliente *localhost*. Sin embargo, el fichero de usuarios viene con configuraciones para diferentes usuarios, pero están comentadas. Por ello, se presenta a continuación una configuración básica en este fichero para un usuario:

```
-testing Cleartext-Password := "password"
```

Donde *testing* es el nombre del usuario y lo que aparece a continuación es su contraseña en texto plano. La configuración que aparece en el fichero *clients.conf* para el cliente *localhost* es:

```
client localhost {
    ipaddr = 127.0.0.1
    proto = *
    secret = testing123
    require_message_authenticator = no
    nas_type = other
    limit {
        max_connections = 16
        lifetime = 0
        idle_timeout = 30
    }
}
```

Donde:

*-ipaddr*: Es la dirección IP dominio de localhost.

*-proto*: Si no se especifica, se establece UDP (*User Datagram Protocol*) por defecto.

*-secret*: Un secreto compartido para “cifrar” y “firmar” los paquetes entre el NAS y *FreeRADIUS*.

*-require\_message\_authenticator*: Determina si se requiere un mensaje autenticador de los clientes.

*-nas\_type*: El tipo de NAS que se conectará en este cliente. Para *localhost* el valor es *other*.

*-limit*: Para limitar las conexiones TCP(*Transfer Control Protocol*). Para las conexiones UDP este bloque es ignorado.

*-max\_connections*: Límite de conexiones TCP simultáneas.

*-lifetime*: El tiempo de vida, en segundos, de las conexiones TCP.

*-idle\_timeout*: El tiempo ocioso, en segundos, de una conexión TCP.

3. Una vez configurados ambos ficheros, se debe ejecutar el servidor RADIUS en modo *debug* desde otra instancia del *shell*:

```
~/usr/local/sbin/radiusd -X
```

La ubicación del ejecutable puede variar dependiendo del sistema.

4. Para hacer un test básico de funcionamiento, se debe ejecutar la siguiente orden:

```
-radtest testing password localhost 0 testing123
```

Donde:

- testing*: Nombre del usuario establecido.
- password*: Contraseña del usuario establecido.
- localhost*: Cliente establecido.
- 0: valor del atributo Puerto NAS.
- testing123*: Secreto compartido establecido.

Si todo ha ido correctamente debe aparecer como salida un mensaje que contenga las palabras ***Access-Accept***.

Una vez realizada la configuración básica, se procede a mostrar como asociar una Base de Datos donde crear los usuarios que serán autenticados a través de Guacamole. Para ello, se hará uso de la guía indicada en [68].

1. Autenticarse en el monitor de *MariaDB* y crear la Base de Datos a utilizar por el servidor:

```
-mysql -u root -p
-CREATE DATABASE radius;
```

2. Crear el esquema SQL y crear un usuario a partir de los script SQL proporcionados en la descarga (modificarlos como se considere):

```
-mysql -uroot -p radius < mods-config/sql/main/mysql/schema.sql
-mysql -uroot -p radius < mods-config/sql/main/mysql/setup.sql
```

Adicionalmente, debido a que en el script *setup.sql* no se encuentra, se debe proporcionar permiso al usuario creado para insertar registros en una tabla concreta con el objetivo de posteriormente crear registros de usuarios:

```
-GRANT INSERT ON radius.radcheck TO 'radius'@'localhost';
```

3. Editar el fichero */etc/mods-available/sql* (la ubicación del subdirectorio *mods-available* puede variar según la instalación realizada) y realizar las siguientes modificaciones:

```
-dialect= "mysql"
-driver = "rlm_sql_${dialect}"
-En el apartado "mysql" comentar las líneas del subapartado "tls" si no se desea hacer uso de cifrado.
-Descomentar las líneas de la sección "Connection info":
server = "localhost"
port = 3306
```

login = "radius"

password = "<<contraseña>>"

Si se han usado las opciones por defecto, no se debe realizar ninguna otra modificación.

#### 4. Habilitar el módulo sql:

-cd mods-enabled

-ln -s ../mods-available/sql sql

5. Editar el fichero */etc/sites-available/default* (la ubicación del subdirectorio *sites-available* puede variar según la instalación realizada) realizando las siguientes modificaciones:

-Descomentar la línea "sql" en la sección "authorize{}".

-Descomentar la línea "sql" en la sección "accounting{}".

-Descomentar la línea "sql" en la sección "post-auth{}" si se desea guardar registros de intentos de inicio de sesión.

#### 6. Tras esto, insertar un registro de usuario en la Base de Datos:

-mysql -u radius -p

-use radius;

-insert into radcheck (username,attribute,op,value) values("<<nombreusuario>>", "Cleartext-Password", ":", "<<contraseña>>");

#### 7. Realizar un test básico de funcionamiento con el nuevo usuario:

-radtest <<nombreusuario>> <<contraseña>> localhost 0 <<secreto>>

Donde <<secreto>>, si no se ha modificado el fichero *clients.conf*, debería ser *testing123*. Si todo ha ido correctamente debe aparecer como salida un mensaje que contenga las palabras **Access-Accept**.

8. Finalmente, reiniciando el *Servlet* Apache Tomcat para que Apache Guacamole cargue la nueva extensión que permita la autenticación por RADIUS y configurando *guacamole.properties* de manera adecuada, la autenticación vía RADIUS debería estar habilitada para el usuario creado, a pesar de que este no exista en la Base de Datos de Guacamole. Si se quieren añadir conexiones a este usuario, si se debe crear registros de estas en la Base de Datos de Guacamole que se asociarán automáticamente al usuario autenticado por RADIUS.

## 5.- Referencias

- [1] “*Departamento de Informática y Sistemas*”, ULPGC, 2022. [En línea]. Disponible en: <https://www.dis.ulpgc.es/> [Accedido: 27-abr-2022]
- [2] “*Información General sobre el DIS*”, Departamento de Informática y Sistemas, ULPGC, 2022. [En línea]. Disponible en: <https://www.dis.ulpgc.es/contenido/composicion/infogral.asp> [Accedido: 27-abr-2022]
- [3] “*Apache Guacamole*”, 2022. [En línea]. Disponible en: <https://guacamole.apache.org/> [Accedido: 04-mar-2022]
- [4] “*Objetivos y Competencias del GIF*”, Escuela de Ingeniería Informática, ULPGC, 2022. [En línea]. Disponible en: [https://www.eii.ulpgc.es/tb\\_university\\_ex/?q=objetivos-y-competencias-del-gii](https://www.eii.ulpgc.es/tb_university_ex/?q=objetivos-y-competencias-del-gii) [Accedido: 27-abr-2022]
- [5] “*Enfermedad por nuevo coronavirus, COVID-19*”, Ministerio de Sanidad, Gobierno de España, 2022. [En línea]. Disponible en: <https://www.sanidad.gob.es/profesionales/saludPublica/ccayes/alertasActual/nCov/home.htm> [Accedido: 27-abr-2022]
- [6] “*Introduction*”, Apache Guacamole, 2022. [En línea]. Disponible en: <https://guacamole.apache.org/doc/gug/introduction.html> [Accedido: 04-mar-2022]
- [7] “*Implementation and architecture*”, Apache Guacamole, 2022. [En línea]. Disponible en: <https://guacamole.apache.org/doc/gug/guacamole-architecture.html> [Accedido: 04-mar-2022]
- [8] “*daemon(7) - Linux man page*”, 2022. [En línea]. Disponible en: <https://man7.org/linux/man-pages/man7/daemon.7.html> [Accedido: 04-mar-2022]
- [9] “*The CentOS Project*”, 2022. [En línea]. Disponible en: <https://www.centos.org/> [Accedido: 28-abr-2022]
- [10] “*About CentOS7*”, *The CentOS Project*, 2022. [En línea]. Disponible en: <https://www.centos.org/about/> [Accedido: 28-abr-2022]
- [11] “*Apache Tomcat*”, 2022. [En línea]. Disponible en: <https://tomcat.apache.org/> [Accedido: 28-abr-2022]
- [12] “*Jakarta EE*”, 2022. [En línea]. Disponible en: <https://jakarta.ee/> [Accedido: 28-abr-2022]
- [13] “*Java EE at a Glance*”, 2022. [En línea]. Disponible en: <https://www.oracle.com/es/java/technologies/java-ee-glance.html> [Accedido: 28-abr-2022]
- [14] “*VNC Viewer*”, Oficina de Seguridad del Internauta, 2022. [En línea]. Disponible en: <https://www.osi.es/es/herramientas-gratuitas/vnc-viewer> [Accedido: 28-abr-2022]
- [15] “*VNC*”, Wikipedia, 2022. [En línea]. Disponible en: <https://es.wikipedia.org/wiki/VNC> [Accedido: 28-abr-2022]
- [16] “*TigerVNC*”, 2022. [En línea]. Disponible en: <https://tigervnc.org/> [Accedido: 13-mar-2022]

- [17] “*Remote Desktop Protocol (RDP)*”, *Network Encyclopedia*, 2022. [En línea]. Disponible en: <https://networkencyclopedia.com/remote-desktop-protocol-rdp/> [Accedido: 28-abr-2022]
- [18] “*¿Qué es el RDP, y cómo usarlo con seguridad?*”, *NordVPN*, 2022. [En línea]. Disponible en: <https://nordvpn.com/es/blog/acceso-remoto-rdp/> [Accedido: 28-abr-2022]
- [19] “*Remote Desktop Protocol*”, *Wikipedia*, 2022. [En línea]. Disponible en: [https://es.wikipedia.org/wiki/Remote\\_Desktop\\_Protocol](https://es.wikipedia.org/wiki/Remote_Desktop_Protocol) [Accedido: 28-abr-2022]
- [20] “*Logo Telnet*”, *LogoMoose*, 2022. [En línea]. Disponible en: <https://www.logomoose.com/pending/telnet/> [Accedido: 29-abr-2022]
- [21] “*Telnet*”, *Wikipedia*, 2022. [En línea]. Disponible en: <https://es.wikipedia.org/wiki/Telnet> [Accedido: 29-abr-2022]
- [22] “*Logo SSH*”, *flaticon*, 2022. [En línea]. Disponible en: [https://www.flaticon.es/icono-gratis/ssh\\_5225347](https://www.flaticon.es/icono-gratis/ssh_5225347) [Accedido: 29-abr-2022]
- [23] “*¿Cómo funciona el SSH?*”, *Hostinger*, 2022. [En línea]. Disponible en: <https://www.hostinger.es/tutoriales/que-es-ssh> [Accedido: 29-abr-2022]
- [24] “*MariaDB Server: The open source relational database*”, 2022. [En línea]. Disponible en: <https://mariadb.org/> [Accedido: 26-mar-2022]
- [25] “*About MariaDB Server*”, 2022. [En línea]. Disponible en: <https://mariadb.org/about/> [Accedido: 29-abr-2022]
- [26] “*Apereo CAS*”, 2022. [En línea]. Disponible en: <https://www.apereo.org/projects/cas> [Accedido: 29-abr-2022]
- [27] “*Apereo CAS - Identity & Single Sign-On*”, 2022. [En línea]. Disponible en: <https://apereo.github.io/cas/6.5.x/index.html> [Accedido: 29-abr-2022]
- [28] “*¿Qué es un sistema Single Sign-On?*”, Servicio de Informática y Comunicaciones, Universidad de Sevilla, 2022. [En línea]. Disponible en: <https://sic.us.es/servicios/cuentas-y-accesos-los-servicios/integracion-con-ss0/que-es-ss0> [Accedido: 29-abr-2022]
- [29] “*Servicio de autenticación centralizada*”, Universidad de Las Palmas de Gran Canaria (ULPGC), 2022. [En línea]. Disponible en: <https://identificate.ulpgc.es/cas> [Accedido: 29-abr-2022]
- [30] “*Lightweight Directory Access Protocol*”, 2022. [En línea]. Disponible en: <https://ldap.com/> [Accedido: 30-abr-2022]
- [31] “*Learn About LDAP*”, 2022. [En línea]. Disponible en: <https://ldap.com/learn-about-ldap/> [Accedido: 30-abr-2022]
- [32] “*OpenLDAP*”, 2022. [En línea]. Disponible en: <https://www.openldap.org/> [Accedido: 30-abr-2022]
- [33] “*Descubre para qué sirve un servidor RADIUS y su funcionamiento*”, 2022. [En línea]. Disponible en: <https://www.redeszone.net/tutoriales/servidores/que-es-servidor-radius-funcionamiento/> [Accedido: 30-abr-2022]

- [34] “*How Does RADIUS Work?*”, CISCO, 2022. [En línea]. Disponible en: <https://www.cisco.com/c/en/us/support/docs/security-vpn/remote-authentication-dial-user-service-radius/12433-32.html> [Accedido: 30-abr-2022]
- [35] “*FreeRADIUS*”, 2022. [En línea]. Disponible en: <https://freeradius.org/>  
[Accedido: 16-abr-2022]
- [36] “*Kernel Virtual Machine*”, 2022. [En línea]. Disponible en: [https://www.linux-kvm.org/page/Main\\_Page](https://www.linux-kvm.org/page/Main_Page) [Accedido: 30-abr-2022]
- [37] “*NoMachine*”, 2022. [En línea]. Disponible en: <https://www.nomachine.com/es>  
[Accedido: 05-may-2022]
- [38] “*Como instalar Tomcat 9 en CentOS 7*”, 2021. [En línea]. Disponible en: <https://comoinstalar.me/como-instalar-tomcat-en-centos-7/> [Accedido: 04-mar-2022]
- [39] “*Como instalar Java OpenJDK en CentOS 7*”, 2020. [En línea]. Disponible en: <https://comoinstalar.me/como-instalar-java-openjdk-en-centos-7/> [Accedido: 04-mar-2022]
- [40] “*How to install Tomcat 9 on CentOS7*”, 2019. [En línea]. Disponible en: <https://linuxize.com/post/how-to-install-tomcat-9-on-centos-7/> [Accedido: 04-mar-2022]
- [41] “*Tomcat 9 Software Development*”, Apache Tomcat, 2022. [En línea]. Disponible en: <https://tomcat.apache.org/download-90.cgi> [Accedido: 04-mar-2022]
- [42] “*Installing Guacamole natively*”, Apache Guacamole, 2022. [En línea]. Disponible en: <https://guacamole.apache.org/doc/gug/installing-guacamole.html> [Accedido: 05-mar-2022]
- [43] “*Apache Guacamole 1.4.0*”, 2022. [En línea]. Disponible en: <https://guacamole.apache.org/releases/1.4.0/> [Accedido: 05-mar-2022]
- [44] “*Welcome to Apache Maven*”, 2022. [En línea]. Disponible en: <https://maven.apache.org/>  
[Accedido: 10-mar-2022]
- [45] “*How to Install Apache Maven on CentOS 7*”, 2019. [En línea]. Disponible en: <https://linuxize.com/post/how-to-install-apache-maven-on-centos-7/> [Accedido: 10-mar-2022]
- [46] “*Downloading Apache Maven 3.8.5*”, 2022. [En línea]. Disponible en: <https://maven.apache.org/download.cgi> [Accedido: 10-mar-2022]
- [47] “*Configuring Guacamole*”, Apache Guacamole, 2022. [En línea]. Disponible en: <https://guacamole.apache.org/doc/gug/configuring-guacamole.html> [Accedido: 12-mar-2022]
- [48] “*Network Level Authentication*”, 2021. [En línea]. Disponible en: [https://en.wikipedia.org/wiki/Network\\_Level\\_Authentication](https://en.wikipedia.org/wiki/Network_Level_Authentication) [Accedido: 17-mar-2022]
- [49] “*Database authentication*”, Apache Guacamole, 2022. [En línea]. Disponible en: <https://guacamole.apache.org/doc/gug/jdbc-auth.html> [Accedido: 26-mar-2022]

- [50] "MySQL Community Downloads", 2022. [En línea]. Disponible en: <https://dev.mysql.com/downloads/connector/j/> [Accedido: 26-mar-2022]
- [51] "Cómo instalar MariaDB en CentOS 7", 2021. [En línea]. Disponible en: <https://comoinstalar.me/como-instalar-mariadb-en-centos-7/> [Accedido: 26-mar-2022]
- [52] "Administration", Apache Guacamole, 2022. [En línea]. Disponible en: <https://guacamole.apache.org/doc/gug/administration.html> [Accedido: 26-mar-2022]
- [53] "CAS authentication", Apache Guacamole, 2022. [En línea]. Disponible en: <https://guacamole.apache.org/doc/gug/cas-auth.html> [Accedido: 17-may-2022]
- [54] "ClearPass: Credential Caching and Replay", Apache Guacamole, 2022. [En línea]. Disponible en: <https://unicon.github.io/cas/4.2.x/integration/ClearPass.html> [Accedido: 17-may-2022]
- [55] Red Hat Enterprise Linux 7, "System Administrator's Guide", en *Chapter 13: TigerVNC*, 2022, pp.200-207 [En línea]. Disponible en: [https://access.redhat.com/documentation/en-us/red\\_hat\\_enterprise\\_linux/7/html/system\\_administrators\\_guide/index](https://access.redhat.com/documentation/en-us/red_hat_enterprise_linux/7/html/system_administrators_guide/index) [Accedido: 13-mar-2022]
- [56] Red Hat Enterprise Linux 7, "Virtualization Deployment and Administration Guide", en *Chapter 1: System Requirements, Chapter 2: Installing the Virtualization packages, Chapter 3: Creating a Virtual Machine*, 2022, pp.9-36 [En línea]. Disponible en: [https://access.redhat.com/documentation/en-us/red\\_hat\\_enterprise\\_linux/7/pdf/virtualization\\_deployment\\_and\\_administration\\_guide/red\\_hat\\_enterprise\\_linux-7-virtualization\\_deployment\\_and\\_administration\\_guide-en-us.pdf](https://access.redhat.com/documentation/en-us/red_hat_enterprise_linux/7/pdf/virtualization_deployment_and_administration_guide/red_hat_enterprise_linux-7-virtualization_deployment_and_administration_guide-en-us.pdf) [Accedido: 07-may-2022]
- [57] "How to Install / Enable OpenSSH on CentOS 7", 2019. [En línea]. Disponible en: <https://phoenixnap.com/kb/how-to-enable-ssh-centos-7> [Accedido: 19-mar-2022]
- [58] "CentOS / RHEL 7 : How to install and configure telnet", 2022. [En línea]. Disponible en: <https://www.thegeekdiary.com/centos-rhel-7-how-to-install-and-configure-telnet/> [Accedido: 24-mar-2022 ]
- [59] "LDAP authentication", Apache Guacamole, 2022. [En línea]. Disponible en: <https://guacamole.apache.org/doc/gug/ldap-auth.html> [Accedido: 31-mar-2022]
- [60] "Tutorial de configuración rápida de OpenLDAP en CentOS 7", 2022. [En línea]. Disponible en: <https://programmerclick.com/article/70121841523/> [Accedido: 31-mar-2022]
- [61] "Apache Directory Studio", 2022. [En línea]. Disponible en: <https://directory.apache.org/studio/> [Accedido: 02-abr-2022 ]
- [62] "RFC 2252", 2022. [En línea]. Disponible en: <https://www.rfc-editor.org/info/rfc2252> [Accedido: 02-abr-2022]
- [63] "RADIUS Authentication", 2022. [En línea]. Disponible en: <https://guacamole.apache.org/doc/gug/radius-auth.html> [Accedido: 03-abr-2022]
- [64] "Building FreeRADIUS", FreeRADIUS, 2022. [En línea]. Disponible en: <https://wiki.freeradius.org/building/Home> [Accedido: 17-abr-2022]

[65] “*Building on RHEL7 or Centos7*”, FreeRADIUS, 2022. [En línea]. Disponible en: <https://wiki.freeradius.org/building/RHEL%20and%20Centos> [Accedido: 17-abr-2022]

[66] “*Releases*”, FreeRADIUS, 2022. [En línea]. Disponible en: <https://freeradius.org/releases/> [Accedido: 17-abr-2022]

[67] “*Basic Configuration Howto*”, FreeRADIUS, 2022. [En línea]. Disponible en: <https://wiki.freeradius.org/guide/Basic-configuration-HOWTO> [Accedido: 24-abr-2022]

[68] “*guide/SQL HOWTO for freeradius 3.x on Debian Ubuntu*”, FreeRADIUS, 2022. [En línea]. Disponible en: <https://wiki.freeradius.org/guide/SQL-HOWTO-for-freeradius-3.x-on-Debian-Ubuntu> [Accedido: 24-abr-2022]