DESARROLLO DE UN PROXY RDP. IMPLEMENTACION DEL MAPEADO DE DISCOS LOCALES. GESTIÓN Y SINCRONIZACIÓN DEL PORTAPAPELES.

AUTOR : JAVIER CAVERNI IBÁÑEZ
PONENTE: ÁLVARO ALESANCO

Departamento de Ingeniería Electrónica y Comunicaciones

Índice

RESUMEN

A lo largo de esta memoria se van a exponer las diferentes etapas en las que ha consistido este Proyecto Fin de Carrera. Antes de todo cabe destacar que la realización de éste proyecto ha sido llevada a cabo en el marco de unas prácticas en empresa, más concretamente en una empresa dedicada al ámbito de la seguridad informática y que está basada en París (Francia).

Es pues, en el marco de éstas prácticas que se definieron los objetivos, así como las diferentes etapas a llevar a cabo para obtener una solución satisfactoria al final de dicho periodo de prácticas.

El objetivo de estas prácticas era realizar un aporte y una mejora sustancial al producto Wallix Admin Bastion desarrollado en el departamento de I+D de la empresa Wallix. Más concretamente, el módulo en el cual se debían desarrollar dichas aportaciones, era el módulo encargado de gestionar las conexiones entre dos equipos distantes a través del protocolo Remote Desktop Protocol (RDP) propietario de Microsoft.

Dichas evoluciones, que son los objetivos presentados a continuación, eran condición sine qua non para el lanzamiento comercial de la versión 2.1 del producto Wallix Admin Bastion ya que eran esperadas por una gran parte de los clientes de la empresa en la que las prácticas se desarrollaron.

La memoria que se presenta a continuación, tiene como fin realizar una explicación detallada de las etapas seguidas por el alumno, desde la definición de los objetivos llevada a cabo por el director del departamento de I+D, pasando por las diferentes etapas de análisis y desarrollo de la solución, hasta llegar a la etapa de definición de las etapas futuras.

INTRODUCCION

Este proyecto fin de carrera (PFC) tiene como objetivo principal el desarrollo de nuevas funcionalidades dentro del protocolo de comunicaciones RDP (Remote Desktop Protocol).

Nuestro proyecto se enmarca dentro de un programa de prácticas en empresa. La empresa en la cual hemos desarrollado el proyecto está basada en París y se llama Wallix. Dicha empresa pertenece al grupo iF Research y desarrolla productos destinados a dotar de seguridad al Sistema de Información de los clientes mediante. La estrategia general del desarrollo de dichos productos esta altamente ligada a la comunidad denominada “Open Source”. Ambos hechos, el desarrollo de las prácticas en empresa y el de que la empresa esté ligada al mundo “Open Source” determinan una serie de restricciones, que citamos a continuación:

- Funcionalidades básicas de conexión entre equipo remoto y servidor existentes gracias a un proxy RDP desarrollado por la comunidad “Open Source”.
- Empresa y proyecto altamente ligado a la comunidad “Open Source”, lo que va a definir las herramientas a utilizar para nuestro proyecto así como los objetivos secundarios alcanzados durante el mismo.
- Ausencia de una especificación funcional del protocolo Remote Desktop Protocol hasta Enero de 2009, debido a que dicho protocolo es propietario de la empresa Microsoft.
- Desarrollo del proxy RDP en el marco de un producto “Open Source” destinado a dotar de seguridad el Sistema de Información de los clientes.

CAPITULO I: PROBLEMÁTICA GENERAL DEL PROYECTO Y ORGANIZACIÓN DE LA DOCUMENTACIÓN

INTRODUCCIÓN

En este capítulo se va a realizar una presentación general de la problemática del proyecto y a definir la estructura de organización del documento del PFC. Además de ello se analizarán las contribuciones del proyecto y los objetivos definidos al inicio del mismo.

PROBLEMÁTICA GENERAL DEL PROYECTO

Como se ha comentado en la introducción, éste PFC, que tiene como objetivos ella implementación de nuevas funcionalidades de un proxy RDP, ha sido realizado dentro del marco de unas prácticas desarrolladas en una empresa. Dicha singularidad ha hecho que la definición de los objetivos, así como el desarrollo de las diferentes fases hasta llegar a su conclusión, hayan sido definidos en la misma línea del producto que engloba el proxy RDP que debe ser adaptado.

Dicho producto es conocido públicamente como Wallix Admin Bastion (WAB) y se trata de un proxy multi-protocolo con funcionalidades de autentificación, control de acceso, almacenaje de datos y generación de informes de actividad. Es pues, en este marco, en el que tratamos de hacer evolucionar el proxy RDP para adaptarlo a la especificación proporcionada por Microsoft en Enero de 2009. Cabe destacar, que el proxy RDP existente en el producto Wallix Admin Bastion (WAB) se basa en el código fuente del proyecto XRDP, proyecto desarrollado en el marco de la comunidad “Open Source”.

La funcionalidad principal del producto Wallix Admin Bastion (que pasaremos a denominar a partir de ahora como WAB) es situarse como puerta de acceso del sistema de información de una sociedad, dotando así al acceso al Sistema de Información del cliente (SI), de todas las ventajas de autentificación, control de acceso y trazabilidad de sesiones. Así pues, con esta arquitectura, las empresas prestatarias de servicios ocasionales y los empleados externos al Sistema de Información del cliente, deben acceder inicialmente al producto conocido como WAB para, una vez autentificados y debidamente trazados si fuera necesario, poder acceder a los recursos internos del Sistema de Información.

Wallix Admin Bastion va a realizar la autentificación de los usuarios mediante la consulta de un anuario LDAP (***L**ightweight **D**irectory **A**cces **P**rotocol*), permitiendo o rechazando el acceso al sistema de información. Si, como resultado de la consulta a

dicho anuario, el usuario es rechazado, Wallix Admin Bastion generará una alarma que será enviada a los Administradores del Sistema de Información. La autentificación se realiza a través de la comunicación con un motor de búsqueda ligado a un anuario LDAP en el cual figuran los usuarios que pueden tener acceso a los recursos del Sistema de Información del cliente así como los permisos de cada usuario con respecto a cada recurso.

Una vez el usuario ha sido autentificado gracias a las informaciones del anuario LDAP, el producto Wallix Admin Bastion será el encargado de almacenar las sesiones gráficas de acceso a los recursos del Sistema de Información (vía el protocolo RDP), así como las informaciones correspondientes a las sesiones iniciadas vía el protocolo SSH (Secure Shell), si así ha sido especificado por los administradores del Sistema de Información para el usuario ha utilizado dichos recursos. El conjunto de estos datos quedan entonces disponibles para el posterior análisis de los administradores del sistema.

A continuación, en la figura 1, se presenta el esquema general de utilización del producto Wallix Admin Bastion.

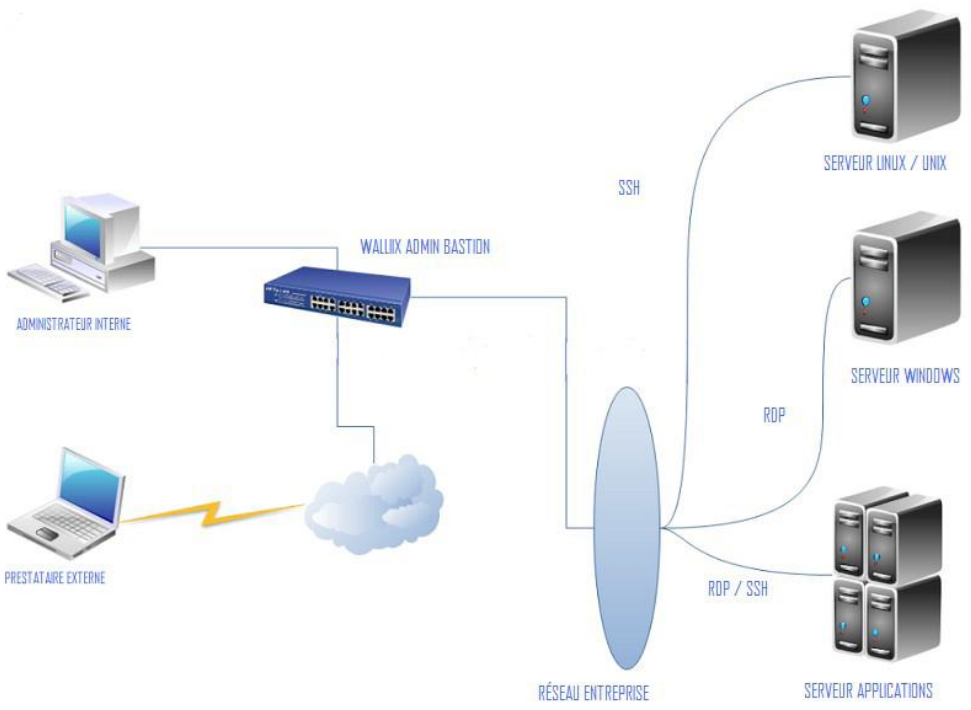

Figura 1. Esquema general de utilización del producto Wallix Admin Bastion

Concretamente, dentro de este marco general del producto y de la problemática asociada, el proyecto fin de carrera que ha sido desarrollado trata de mejorar y

adaptar el proxy RDP que está imbuido en nuestro producto y así hacerlo conforme a la especificación de Microsoft publicada en Enero de 2009.

El objetivo de dicho protocolo es permitir la comunicación entre dos equipos de un modo gráfico a través de los denominados "Microsoft Terminal Server Clients". El equipo local, llamado cliente, va a conectarse a una aplicación situada en el equipo distante, ejecutarla y recuperar los datos de dicha ejecución para mostrarlos en el terminal cliente como si la aplicación estuviera instalada de forma local. Así pues, podemos definir el tipo de arquitectura en la cual se basa dicho protocolo en una arquitectura cliente – servidor ligera, debido a que la mayor parte de la carga se realiza en el servidor, lo cual nos permite, ejecutar aplicaciones con gran carga de trabajo, desde un cliente ligero.

Este protocolo, iniciado y desarrollado por Microsoft, está basado en el protocolo ITU.share (también conocido como T-128) y permite acceder a las siguientes funcionalidades:

- Visualización en modo gráfico de una sesión Windows.
- Posibilidad de redirigir y utilizar de forma local los dispositivos del equipamiento cliente en el equipamiento distante.

Un apunte final, pero no menos importante, es que la especificación de Microsoft del protocolo RDP fue publicada en Enero de 2009 con lo que todo el trabajo anterior hecho por la comunidad "Open Source" ha sido desarrollado a través de métodos de ingeniería inversa.

Dichos trabajos cuentan entre sus proyectos con:

- Rdesktop: aplicación que simula un cliente RDP en entorno Linux.
- XRDP: aplicación que simula un servidor RDP en entorno Linux, dicha aplicación ha sido llevada a cabo gracias al estudio previo realizado en el proyecto Rdesktop.

OBJETIVOS INICIALES DEL PROYECTO

El objetivo global del proyecto es hacer evolucionar el proxy RDP que está presente en el producto Wallix Admin Bastion creado por Wallix, para establecerlo conforme a la especificación del protocolo RDP publicada por Microsoft en Enero de 2009. Dicha especificación presenta una serie de funcionalidades propias al protocolo RDP que no se encontraban disponibles en el proxy RDP y cuyo desarrollo se convirtió en los objetivos principales de nuestro proyecto:

- Desarrollo de la sincronización del portapapeles entre un equipo cliente y un servidor remoto.
- Desarrollo del mapeo de discos locales de un equipo cliente en un servidor remoto.
- Desarrollo de la gestión de impresoras y periféricos a distancia.

El protocolo de comunicaciones sobre el que se basa nuestro proyecto es el protocolo RDP. Dicho protocolo fue definido inicialmente por la empresa Microsoft quien actualmente es todavía propietaria de la patente del mismo.

Para poder llevar a cabo nuestro proyecto, una serie de reglas de diseño y de requisitos asociados al producto dentro del cual vamos a desarrollar el proyecto se imponen de una manera implícita. Dichas limitaciones son, por ejemplo, el uso de herramientas ligadas al mundo "Open Source" o el uso de un lenguaje de programación determinado (lenguaje C en nuestro caso).

Desde un punto de vista técnico, y debido a la arquitectura impuesta por el producto en el cual nosotros tenemos que desarrollar nuestro proyecto; el desarrollo del proxy RDP debe hacerse en el marco del proyecto XRDP que implementa un servidor RDP en el entorno Linux.

Una vez expuestas dichas limitaciones técnicas debemos aprovechar el código fuente "Open Source" del proyecto XRDP propuesto por los ingenieros de la comunidad "Open Source" para adecuarlo a la especificación funcional definida por Microsoft y con ello contribuir a la evolución de dicho proyecto.

ORGANIZACIÓN DE LA DOCUMENTACIÓN

Respecto a la organización de la documentación, este informe está estructurado de la siguiente manera. El primer capítulo ha sido dedicado a la presentación de tema principal del proyecto así como las limitaciones ligadas a nuestro proyecto desde un punto de vista técnico.

En el segundo capítulo se va realizar una presentación de la problemática del proyecto así como de la solución adoptada finalmente para llevar a cabo la tarea propuesta inicialmente.

Finalmente, en el tercer capítulo, vamos a presentar las conclusiones finales de nuestro proyecto así como las posibles mejoras propuestas para una continuación de nuestro proyecto de desarrollo de un proxy para el protocolo RDP.

Una vez concluida la organización de los capítulos que componen el PFC, vamos a pasar a describir la organización de los anexos que componen dicho PFC. En el primer anexo, vamos a presentar la sociedad en la cual se ha desarrollado el periodo de prácticas así como una presentación inicial del proyecto en el que se enmarcan estas prácticas.

En el segundo anexo, se va presentar la continuación de nuestro desarrollo del proxy RDP así como su posterior evolución a un nuevo proyecto que todavía no ha sido publicado en el mundo “Open Source”, pero que ha sido bautizado con el nombre de “Redemption”.

En el tercer anexo se va a realizar un análisis de la especificación abierta por Microsoft en Enero de 2009 y que nos ha servido para la total comprensión del protocolo así como para el desarrollo de las nuevas funcionalidades.

Finalmente, en el cuarto anexo, se realizará una pequeña presentación de los principales módulos del proyecto “Rdesktop” sobre el cual ha sido basado el proyecto “XRDP” que es a su vez, la base de nuestro PFC. Dicho proyecto “Rdesktop” es fruto del desarrollo de un cliente, en el entorno de la comunidad “Open Source”, basado en una distribución Unix, para el protocolo RDP.

CAPÍTULO II: PRESENTACIÓN DE LA PROBLEMÁTICA DEL PROYECTO Y DESARROLLO DE LA SOLUCIÓN ADOPTADA

INTRODUCCIÓN

Durante este capítulo, vamos a realizar la presentación de la problemática del proyecto así como el desarrollo de la solución adoptada. Para ello, es necesario recordar el objetivo principal del proyecto, este no es otro que la realizar la evolución de un proxy RDP para añadirle nuevas funcionalidades que lo adapten a la especificación publicada por Microsoft en Enero de 2009. Concretamente, dichas funcionalidades de "copiar/pegar" y de redirección de dispositivos del ordenador local al ordenador remoto.

Este desarrollo del proxy RDP está incluido en un producto de la sociedad Wallix llamado Wallix Admin Bastion (WAB). Una de las principales características de dicho producto es que está formado única y exclusivamente por componentes "Open Source", lo que impone ciertas restricciones al entorno de trabajo de nuestro PFC. Dicho entorno de desarrollo nos va a permitir conocer el mundo "Open Source" así como numerosas de sus distribuciones comerciales como por ejemplo Debian y Ubuntu.

Así pues, la elección de la tecnología para desarrollar nuestro proyecto, así como la base sobre la cual debíamos trabajar, estaba ya tomada a nuestra llegada al mismo. Una versión del proyecto "XRDP" funcionando en modo proxy RDP dentro del producto Wallix Admin Bastion había sido desarrollada e implementada por uno de los ingenieros que forma parte del departamento R&D de Wallix. Dicho proxy es la base que debemos completar para tener un proxy funcional que siga las especificaciones liberadas por Microsoft a correspondientes a las diversas funcionalidades del protocolo RDP.

Dicho proyecto XRDP está destinado a la construcción de un servidor RDP totalmente funcional y disponible para entornos UNIX / Linux. Nuestro objetivo es, pues, desarrollar un proxy en el marco de este proyecto XRDP para posteriormente, y como valor añadido al propio valor académico del proyecto, poder ofrecerlo como aporte a la comunidad "Open Source".

ESTADO INICIAL DEL PROYECTO XRDP

El proyecto XRDP, al ser un proyecto "Open Source" seguido por una amplia comunidad de gente interesada en dicho proyecto, está en continuo desarrollo. Dicho proyecto tiene como objetivo el desarrollo de un servidor para el protocolo propietario de Microsoft denominado RDP bajo entornos diferentes a Windows.

La última versión estable distribuida y sobre la que nos basamos para el estudio del código, fue la versión 0.4.1. Estando en un estado de desarrollo avanzado la siguiente versión 0.5.0. Las principales características de esta última versión estable son el soporte para la versión 5.0 del protocolo RDP, la interacción con los clientes de Windows XP y Vista así como el correcto funcionamiento del servidor RDP en distintas distribuciones de Linux como Ubuntu, Debian, RedHat, OpenSuse y Solaris.

Por el contrario, las mayores limitaciones a nivel de la versión 0.4.1, las encontramos a nivel de la gestión de los canales virtuales sobre el que se soportan la mayoría de las aplicaciones del protocolo RDP. El establecimiento y correcta gestión de estos canales virtuales, nos permitirá tener acceso a las funcionalidades que por el momento no están desarrolladas como son:

- La gestión de la aplicación denominada como portapapeles, para el buen funcionamiento de la funcionalidad "copiar y pegar" entre una sesión Windows y una sesión Linux.
- La redirección de los dispositivos de almacenamiento como el sistema de ficheros del equipo local en el equipo remoto, que nos permitirán tener acceso a los ficheros del ordenador local para ejecutarlos en las aplicaciones del servidor RDP remoto.
- La redirección del sonido y de las impresoras que nos permitirán disponer de ellas para usarlas en el servidor remoto

Finalmente otra de las características que faltaba por desarrollar en esta versión 0.4.1, es un servidor X que nos permita usar RDP en modo proxy, para finalmente conectarse al servidor final RDP. Dicho desarrollo de un servidor X, es el principal eje de investigación y avance del proyecto XRDP secundado por sus responsables.

ESTUDIO TÉCNICO SOBRE EL PROTOCOLO RDP

INTRODUCCIÓN

Inicialmente se han presentado ciertas generalidades del proyecto XRDP a partir del cual se va a desarrollar nuestro proyecto fin de carrera, pero en este apartado vamos a

analizar más en profundidad el protocolo sobre el cual vamos a construir nuestro proxy, dicho protocolo se denomina RDP (Remote Desktop Protocol).

El funcionamiento de dicho protocolo es bastante sencillo, y está basado en la arquitectura clásica de cliente-servidor. La información generada por la aplicación, que está ejecutándose en el servidor remoto, es convertida al formato RDP y enviada a través de la red al terminal cliente. Dicho terminal cliente se encarga de interpretar la información recibida a través de la red para posteriormente tratarla y presentarla en el terminal como si la aplicación estuviera ejecutándose de manera local.

Por otro lado, la información de los datos introducidos por el usuario a través del terminal cliente, como por ejemplo los caracteres introducidos a través del teclado o los movimientos del ratón, son redirigidos al servidor para su posterior tratamiento e interacción con la aplicación. Dichos datos, son comprimidos por motivos de reducción de ancho de banda, utilizando un algoritmo de compresión definido en función de la versión del protocolo utilizado. Además de ser comprimidos, dichos datos, podrán ser encriptados para dotar al protocolo de una mayor seguridad en función de la elección realizada por el cliente en el momento de la conexión.

Por último, cabe destacar que este protocolo está basado sobre el estándar T.128 del que vamos a realizar una breve descripción a continuación.

PROTOCOLO T.128

El protocolo T.128 también conocido como T.share, es un protocolo de aplicaciones repartidas que autoriza la visualización de una aplicación informática ejecutándose en un punto dado sobre la forma de una sesión ejecutándose en un sitio remoto.

Esta recomendación de la UIT-T utiliza los servicios proporcionados por las recomendaciones T.122 para los servicios de comunicación multi-punto, y también los servicios proporcionados por la recomendación T.124 para el control de conferencias de modo genérico.

Uno de los puntos principales de la recomendación T.128, es que cada uno de los puntos remotos puede tomar el control de una aplicación informática compartida. Este tipo de organización permite la visualización y el control a distancia de una aplicación de la misma manera que si dicha aplicación se estuviera desarrollando de manera local.

A continuación, una vez explicado el funcionamiento de base del protocolo y presentadas las recomendaciones internacionales sobre las que dicho protocolo se apoya, se va a presentar la arquitectura funcional del proyecto XRDP para la construcción de un servidor RDP adaptado a entornos UNIX / Linux.

MODO DE FUNCIONAMIENTO DEL PROTOCOLO RDP

El protocolo RDP y por consiguiente el proyecto XRDP (implementación "open source" del mismo), se basa en una arquitectura corriente denominada cliente-servidor. Es decir, por una parte tenemos un cliente encargado de iniciar la comunicación y de enviar peticiones de información a un servidor que va a encargarse de devolver las informaciones requeridas por dicho cliente.

En nuestro caso, el protocolo que usan el cliente y el servidor para comunicarse entre ellos, no es otro que el protocolo RDP.

En la primera fase de establecimiento de la sesión RDP, el servidor RDP en el cual va a estar ejecutándose la aplicación, llamado a partir de ahora simplemente servidor, va a permanecer inicialmente a la escucha en el puerto 3389, dicho puerto es el definido por defecto para aceptar las conexiones entrantes de los diferentes clientes basadas en el protocolo RDP, aunque puede ser modificado a través de un fichero de configuración.

Por su parte el cliente RDP, llamado a partir de ahora cliente, va a ser el encargado de iniciar la comunicación enviando diversas informaciones al servidor sobre la versión del protocolo deseada, los dispositivos a redirigir así como otras correspondientes al nivel de seguridad a utilizar durante la comunicación.

Una vez el cliente ha iniciado la comunicación con el servidor, dicho servidor va a definir un identificador de sesión que posteriormente intercambiará con el cliente y le identificará durante todos los intercambios de información de la sesión RDP. Una vez que dicho identificador ha sido establecido y comunicado, el servidor va a hacer uso de un gestor de sesiones para solicitar al cliente los datos necesarios para la identificación y acceso al servidor remoto, entre estas informaciones se encuentran la resolución de pantalla, el puerto a través del cual se realizará la comunicación, así como el nombre de usuario y la contraseña necesarias para abrir la conexión en dicho servidor remoto.

Si las informaciones de identificación son intercambiadas correctamente, el cliente dispondrá de acceso al servidor para poder realizar las tareas convenientes durante el transcurso de la sesión RDP.

Tras analizar el comportamiento inicial del servidor, podemos pasar a analizar el punto de vista del cliente. Dicho cliente enviará inicialmente las informaciones ligadas a las entradas estándar de dicho dispositivo, es decir las entradas de teclado y ratón principalmente que nos van a permitir la interacción del cliente con la aplicación que se está desarrollando, de forma remota, en el servidor.

Dependiendo de la versión del protocolo, el cliente podría enviar diferentes informaciones relacionadas con los periféricos a redirigir hacia el servidor como pueden ser informaciones de sistema de ficheros, informaciones de sonido, etc. Dichas informaciones deben estar disponibles si la versión del protocolo negociado entre cliente y servidor es superior a la versión 4, por el contrario, el servidor RDP para entornos UNIX / Linux que constituye el proyecto XRDP, no soporta actualmente ninguna de estas entradas adicionales, ya que está basado sobre la versión 4 del protocolo. Dicho soporte de entradas de periféricos adicionales es uno de nuestros objetivos.

Por otro lado, desde el punto de vista del servidor, al menos en esta etapa de desarrollo del proyecto XRDP, las únicas informaciones destinadas al cliente son las informaciones gráficas que deberán ser reconstruidas en el cliente posteriormente para poder obtener las imágenes generadas por la aplicación distante. Básicamente, el servidor se encarga de devolver las informaciones que vienen dadas como resultado de la interacción del cliente con la aplicación desarrollada en dicho servidor. Ejemplos de estas informaciones pueden ser, nuevos mapas de bits que vienen dados como resultado de la apertura de una nueva ventana en la sesión Windows remota, informaciones de texto resultado de la funcionalidad de “copiar/pegar” entre cliente y servidor, etc.

Un detalle importante en el protocolo RDP es que la comunicación entre el cliente y el servidor en el marco del protocolo RDP se realiza a través de diferentes canales virtuales. Estos canales virtuales que son establecidos al inicio de la conexión entre el cliente y el servidor, nos permitirán gestionar el intercambio de información que está ligado a la gestión de periféricos (exceptuando el ratón y teclado que se gestionan a través del canal principal de comunicación), a la gestión del portapapeles, etc.

Dichos canales pueden ser negociados entre ambas partes al inicio de la conexión o durante el transcurso de la sesión. Estos canales virtuales, son propuestos por el cliente durante la etapa de conexión del protocolo RDP. Cabe destacar la presencia de una especialización en la gestión de los canales virtuales, para ser mas exactos, tenemos un canal dedicado al audio, otro dedicado a la gestión del portapapeles compartido entre cliente y servidor, otro canal dedicado a los dispositivos externos tales como dispositivos USB o Sistemas de Ficheros, con lo que la gestión de dichos canales es vital para el posterior uso de las funcionalidades previstas en los

documentos técnicos que componen la especificación del protocolo RDP redactados por Microsoft.

Una vez anunciados por el cliente, el servidor debe confirmar que ha recibido la lista de canales propuesta por el cliente. El siguiente paso, en lo que respecta a la negociación de los canales, es la confirmación individual de cada uno de los canales presentes en la lista negociada. Dicha confirmación se lleva a cabo por el cliente en una serie de mensajes denominados Client Join Request - Client Join Confirm.

Finalmente cuando los canales han sido establecidos y la etapa de conexión ha sido finalizada, el cliente y el servidor pueden comenzar a intercambiar mensajes con los datos referentes a las entradas / salidas estándar del protocolo o a entradas diferentes a las estándar provenientes de la redirección de dispositivos.

Para finalizar con el análisis del protocolo RDP y de su arquitectura, vamos a presentar algunas características de seguridad y compresión de datos que hacen interesante este protocolo para una arquitectura cliente ligero-servidor como la que describe este protocolo.

SEGURIDAD EN EL PROTOCOLO RDP

Un aspecto a destacar en el protocolo RDP es el de la seguridad de los datos transmitidos entre el cliente y el servidor ya que esta posibilidad es una de las ventajas que tiene el protocolo RDP con respecto a otros protocolos que tratan la misma problemática.

Dependiendo del nivel de seguridad escogido en el inicio de la conexión tenemos diferentes esquemas de cifrado de datos entre el cliente y el servidor, a continuación se van a presentar las diferentes alternativas que existen de acuerdo al nivel de seguridad negociado durante el establecimiento de la conexión:

- **Seguridad RDP por defecto:** dicha opción de seguridad permite utilizar hasta cuatro niveles diferentes de encriptado de datos, dependiendo de los “flags” intercambiados durante el proceso de negociación llevado a cabo entre cliente y servidor. Durante este proceso, el cliente procede a anunciar al servidor los niveles de encriptado que es capaz de interpretar y el servidor le va a responder con la elección del encriptado de los paquetes a llevar a cabo durante todo el resto de la sesión. Dicha elección del nivel de encriptado debe estar contenida en la lista que puede ser gestionada por el cliente, ya que de lo contrario el intercambio de los datos entre el cliente y el servidor seria incoherente y no sería posible la correcta interpretación de los datos enviados ni por el cliente ni por el servidor.

Los cuatro niveles presentes para la seguridad RDP por defecto son los siguientes:

- *Bajo:* en el cual todos los datos que circulan del cliente al servidor son encriptados teniendo en cuenta el tamaño máximo de la clave de cifrado soportada por el cliente.
- *Compatible con el cliente:* sigue el mismo principio que el nivel anterior pero esta vez los datos que circulan entre el servidor y el cliente son también encriptados.
- *Alto:* en el cual los datos que circulan en ambos sentidos son cifrados teniendo en cuenta el tamaño máximo de la clave de cifrado presente en el servidor, si el cliente no puede soportar este tamaño de llave de cifrado es desconectado, la conexión es rechazada.
- *FIPS (Federal Information Processing Standard):* todo el flujo de datos que circula entre cliente y servidor es cifrado siguiendo el estándar "Federal Information Processing Standard 140-1", los clientes que no pueden soportar dicho estándar no pueden conectarse al servidor, ya que de lo contrario las informaciones intercambiadas durante la sesión serían indescifrables tanto por el cliente como por el servidor.

Como resumen, se puede decir que los tres primeros niveles de encriptado de la seguridad RDP por defecto utilizan un encriptado basado en el algoritmo RC4, por el contrario, el último nivel de seguridad está basado en el algoritmo triple DES. A continuación, en la figura 2, se presenta una tabla descriptiva de los niveles de encriptado usados en este protocolo:

Selected Encryption Level	Selected Encryption Method	Data Encryption	MAC Generation
None (0)	None (0x00)	None	None
Low (1)	40-Bit (0x01) 56-Bit (0x08) 128-Bit (0x02)	Client-to-server traffic only using RC4	Client-to-server traffic only using MD5 and SHA-1
Client Compatible (2)	40-Bit (0x01) 56-Bit (0x08) 128-Bit (0x02)	Client-to-server and server-to-client traffic using RC4	Client-to-server and server-to-client traffic using MD5 and SHA-1
High (3)	128-Bit (0x02)	Client-to-server and server-to-client traffic using RC4	Client-to-server and server-to-client traffic using MD5 and SHA-1
FIPS (4)	FIPS (0x10)	Client-to-server and server-to-client traffic using Triple DES	Client-to-server and server-to-client traffic using SHA-1

Figura 2. Tabla resumen de los niveles de encriptado presentes en el protocolo RDP

- El segundo tipo de seguridad soportada por el protocolo RDP es la denominada seguridad mejorada. Dicha elección de seguridad va a permitir que todos los

intercambios de datos sean realizados por un protocolo externo de seguridad. Dichos protocolos pueden ser los siguientes:

- TLS, **T**ransport **L**ayer **S**ecurity.
- CredSSP, que es una mezcla del protocolo TLS junto con Kerberos y NTLM (**NT L**AN **M**anager).

A su vez, este segundo tipo de seguridad también soporta algunos de los métodos descritos en el apartado anterior, como son los niveles: compatible con el cliente, alto y FIPS.

COMPRESIÓN DE DATOS EN EL PROTOCOLO RDP

Otra ventaja muy interesante proporcionada por el protocolo RDP, es la posibilidad de realizar compresión de datos para poder disminuir en todo lo posible el ancho de banda usado durante la sesión RDP entre el cliente y el servidor. Puesto que la arquitectura del protocolo RDP está basada en poseer un cliente ligero y que sea el servidor el que lleve la mayor parte de carga del trabajo, es interesante también tener en cuenta el ancho de banda que puede consumir dicho tipo de arquitectura ya que todas las informaciones generadas en el servidor han de ser reenviadas al cliente vía la red.

No se va a tratar en profundidad el tema de la compresión de datos en el protocolo RDP, pero si podemos realizar una breve descripción de las características básicas de la compresión de datos utilizado en el protocolo RDP. El algoritmo usado para la compresión de dichos datos es una variación del protocolo desarrollado por Microsoft MPPC (**M**icrosoft **P**oint-to-**P**oint-**C**ompression).

Existen dos niveles diferentes de compresión aceptados en la especificación del protocolo RDP, el primero, es el denominado MPPC-8K que proporciona un buffer de almacenamiento de 8Kb. El segundo, es una variación del estándar que proporciona un buffer de 64 Kb e implementa una codificación de Huffman perfeccionada para flujos de bits.

La diferencia del tamaño de los buffers es muy importante ya que el protocolo MPPC se basa en la búsqueda y coincidencia de una secuencia, que se encuentra en los datos existentes en el buffer. Así pues, cuantos más datos existan en el buffer, más probabilidad de coincidencia con la secuencia a comprimir, y por tanto más eficiente será la compresión.

ARQUITECTURA DEL PROYECTO XRDP

A continuación, se va a realizar una presentación de la arquitectura del proyecto XRDP. Lo primero que es necesario decir, es que dicho proyecto XRDP está programado en el lenguaje de programación C.

En lo que respecta a las generalidades del proyecto XRDP, que es el encargado de implementar un servidor RDP en el dominio UNIX / Linux, se va a realizar una pequeña presentación de su arquitectura para una mejor comprensión de cómo se realizan los intercambios de mensajes entre el cliente y el servidor, representado en este caso por XRDP. Cabe destacar que el objetivo principal del proyecto XRDP es la de funcionar como servidor RDP bajo una arquitectura UNIX / Linux. Por lo tanto, uno de nuestros objetivos, es modificarlo para que dicho proyecto pueda ser utilizado también como proxy RDP y no sólo como servidor final.

Inicialmente el proyecto está estructurado de la siguiente manera: un bloque principal denominado "core XRDP" que nos va a permitir realizar las primeras etapas de la conexión RDP, posteriormente nos encontramos un gestor de sesiones que nos va a permitir realizar la autentificación en el servidor (actualmente se realiza mediante un módulo de autentificación PAM), finalmente nos encontramos tres librerías diferentes que serán cargadas dinámicamente en el momento de la elección del tipo de conexión requerida.

A continuación, en la figura número 3, se presenta un pequeño esquema de cómo está organizado el proyecto XRDP.

Figura 3. Esquema general de organización del proyecto XRDP

Como se puede observar, y como ha sido mencionado anteriormente, el proyecto está compuesto de cuatro módulos principales más el gestor de sesiones que no se representa en la figura precedente debido a que depende del tipo de librería cargada.

- **Módulo XRDP :** es el módulo encargado de recibir la conexión entrante procedente del cliente. Para ello, inicialmente, va a abrir un socket en el puerto 3389 en escucha de las conexiones que lleguen a este puerto (recordemos que el puerto 3389 es el utilizado por el protocolo RDP por defecto para realizar las conexiones aunque dicho puerto puede ser modificado directamente en los ficheros de configuración). Una vez la conexión establecida, a nivel de socket Linux, y dicho socket está dispuesto a recibir los datos, el programa va creando una serie de estructuras (conocidas como estructuras en el lenguaje de programación C que es el lenguaje en el cual está programado dicho proyecto) para que le permiten almacenar las diferentes informaciones a nivel de conexión, sesión, canales utilizados, etc…. Todos estos datos transmitidos durante la etapa de conexión y que posteriormente serán utilizados para realizar la compresión o no de los paquetes, enviar los datos por uno u otro canal establecido, realizar el encriptado de datos, etc.

  Una vez las primeras etapas de la conexión a nivel de red han sido realizadas, el servidor va a abrir una ventana de conexión que nos va a solicitar la elección de una de las librerías para continuar el proceso de conexión y poder establecer la sesión necesaria para el intercambio de datos entre el cliente y el servidor. La elección de dicho tipo de conexión nos va a permitir elegir el módulo que va a ser lanzado a continuación y por tanto el tipo de conexión realizada entre el cliente y el servidor (un ejemplo de pantalla de conexión se presenta a continuación en la figura 4). A continuación se encuentra una breve explicación de cada uno de los diferentes módulos de conexión.

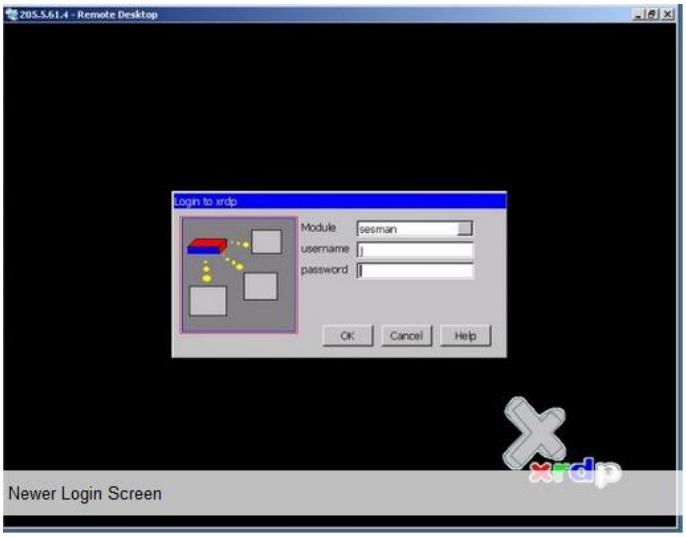

Figura 4. Pantalla inicial de conexión del programa XRDP

- **Módulo Xvnc :** dicho módulo va a ser el único que va a requerir el uso del gestor de sesiones que está también presente en el proyecto XRDP. El funcionamiento es el siguiente, una vez que hemos escogido la opción Xvnc-sesman en el menú desplegable de las opciones de conexión presentadas por el servidor RDP, el módulo XRDP, presentado anteriormente, va a establecer una conexión a través del puerto 3350 con el gestor de sesiones incorporado implícitamente en el proyecto XRDP. Dicho gestor de sesiones va a realizar la autentificación de los datos introducidos en las casillas de usuario y contraseña en el momento de la conexión. Si la autentificación ha resultado positiva, el gestor de sesiones va a lanzar un servidor VNC (**V**irtual **N**etwork **C**omputing). Una vez lanzado dicho servidor, el módulo XRDP, ejecutándose en paralelo al ser un programa que usa diferentes hilos de ejecución para cada conexión, va a encargarse de lanzar la librería libvnc.so de manera dinámica. Una vez lanzada dicha librería, el funcionamiento del programa es el siguiente: todas las informaciones de entradas estándar del protocolo, es decir entradas procedentes del ratón y del teclado, van a ser tratadas directamente por la librería vnc.

  Dichas entradas van a ser transformadas en formato RFB (**R**emote **F**rame**B**uffer) para ser transmitidas posteriormente al servidor VNC lanzado por el gestor de sesiones, así pues, dicho servidor va a ser el encargado de procesar dichas entradas y generar las salidas estándar como son las salidas de gráficos generadas en el servidor, dichas salidas serán enviadas a la librería libvnc que va a interpretarlas y posteriormente transformarlas en formato RDP para enviarlas al cliente.

  La ventaja de usar esta combinación de RDP y VNC es que VNC nos va a proporcionar la interfaz gráfica en el servidor y RDP nos va a permitir transformar esta información para poder comprimirla y encriptarla mejorando así el ancho de banda consumido y la seguridad del protocolo.

- **Módulo RDP :** inicialmente el módulo RDP es un cliente RDP desarrollado en el interior del proyecto XRDP, que recordemos es un proyecto dedicado a ser utilizado como servidor RDP en entornos Unix / Linux no adaptados a este tipo de tecnología desarrollada por Microsoft. Al elegir este módulo en el menú deslizante presentado por el servidor XRDP en el momento de la conexión, debemos introducir también una dirección IP sobre la que se realizará la conexión. Si se hace un análisis funcional de este módulo, desde un punto de vista externo al proyecto XRDP, dicho módulo es una especie de proxy RDP, ya que nos permite realizar una conexión hacia un tercer equipo que será considerado entonces como servidor RDP final. Dicha segunda conexión, permite al proyecto XRDP recibir una conexión de un cliente dicho “Microsoft”,

lanzar nuestro módulo RDP y establecer una nueva conexión con un tercer equipo distante que pasara a ser el servidor final en el que se ejecutaran todas las aplicaciones. En este caso concreto, el proyecto XRDP a través de su módulo RDP funciona como un "proxy RDP". Una vez analizada dicha funcionalidad global del proyecto XRDP a través de este módulo, resulta fácil deducir que es en este módulo en el que deberemos realizar la gran parte de nuestro trabajo para poder adaptarlo a las especificaciones requeridas, ya que no olvidemos, nuestro proyecto consiste desarrollar diversas funcionalidades y en adaptar un proxy RDP a las especificaciones liberadas por Microsoft con el objetivo de hacer este proxy RDP compatible con la mayoría de los clientes RDP presentes en el mercado .

Sin embargo, si bien acabamos de decir que dicho módulo puede comportarse como una especia de "proxy RDP" hace falta remarcar algunas diferencias importantes por las cuales no se puede considerar éste módulo como un verdadero proxy RDP. La primera y principal diferencia es que dicho módulo realiza una nueva conexión hacia un tercer equipo basándose únicamente en la dirección IP, lo cual quiere decir, entre otras cosas, que dicho módulo no tiene en cuenta la versión del protocolo de la conexión entrante, ni la lista de canales presentados por el cliente RDP para ser negociados con el servidor RDP, recordemos que dichos canales son utilizados para la gestión de periféricos, etc.

Cabe destacar entonces, que la ausencia de la negociación de dichos canales imposibilita las diferentes funcionalidades de redirección de periféricos del cliente al servidor, mapeado de discos locales en el servidor y la funcionalidad de cortar y pegar que son los objetivos principales de nuestro proyecto fin de carrera.

El funcionamiento de XRDP en este aspecto es el siguiente, una vez que hemos elegido el módulo RDP en el menú deslizante presentado por el protocolo a través de la ventana de conexión vista anteriormente, se va a proceder a la introducción de ciertos parámetros obligatorios para la realización de la segunda conexión hacia el servidor remoto, dicho dato obligatorio es la conocida como dirección IP de destino.

Una vez recuperado este dato, el módulo principal del programa XRDP va a ser el encargado de lanzar la librería librdp.so. Dicha librería va a realizar una conexión con el equipo distante que pasará a considerarse servidor RDP. La conexión, previamente lanzada por la librería librdp.so interna al proyecto XRDP, es realizada con los parámetros por defecto cargados en el fichero de configuración de dicha librería (dichos parámetros son el tamaño de la ventana

de conexión, la versión del protocolo utilizada, los canales de transmisión de datos usados en la conexión, etc.…).

Uno de los puntos principales a tratar en este proyecto fin de carrera para poder lograr alcanzar nuestros objetivos, es el relativo a la versión del protocolo tratada por el proyecto XRDP en el momento del desarrollo de nuestro proyecto, dicha problemática reside en que este cliente RDP, que resulta ser la librería librdp.so, va a realizar la conexión usando una versión del protocolo RDP que es la versión 4, la cual no nos permite usar la redirección ni tampoco el encriptado ni la compresión de datos.

Así pues, los datos procedentes del cliente RDP son tratados por el módulo principal XRDP en función de la versión del protocolo y del nivel de seguridad negociado entre el cliente RDP y el programa XRDP (ya que recordemos la segunda conexión realizada hacia el servidor RDP se basa en la versión 4 del protocolo), los paquetes son descifrados y descomprimidos, las informaciones de teclado y ratón son analizadas y pasadas como parámetros a cierto módulos de la librería librdp.so que serán los encargados de volver a encapsular y comprimir dichos datos siguiendo la especificación de Microsoft para la versión 4 del protocolo RDP. Una vez las tramas de datos constituidas por ciertos módulos de la librería librdp.so, otros módulos de la misma librería serán los encargados de transmitir estas informaciones, a través de la versión 4 del protocolo RDP, hasta el tercer equipo denominado servidor RDP.

En los párrafos anteriores hemos explicado como el programa XRDP, a través de su módulo RDP realiza el envío de las informaciones entre el cliente y el servidor, por el contrario, a continuación se va a pasar a realizar el análisis del flujo de datos inverso, es decir, ¿cómo son tratados los datos procedentes del servidor hacia el cliente?. Inicialmente, las informaciones procedentes de este nuevo tercer equipo, son recuperadas por la librería librdp.so, analizadas siguiendo la versión 4 del protocolo y enviadas al módulo principal que se encarga de volver a cifrarlas y comprimirlas de acuerdo con las políticas de cifrado y compresión negociadas entre el cliente y el programa XRDP al inicio de la conexión. Dichos datos serán posteriormente encapsulados en paquetes y enviados al cliente a través de la red.

En resumen, y como principal idea que debemos retener al margen del funcionamiento interno del programa XRDP , es que debido al desarrollo realizado sobre el módulo RDP, tenemos dos sesiones presentes en nuestra conexión entre cliente – xrdp – servidor. Dichas sesiones no utilizan por defecto la misma versión del protocolo lo que nos puede llevar a una pérdida de prestaciones en nuestra conexión final entre el cliente y el servidor. Alguna de

estas prestaciones pueden ser la gestión de canales virtuales, o la pérdida de un nivel de seguridad elevado entre el cliente y el servidor final.

- **Módulo XUP:** dicho módulo se encuentra en una fase de desarrollo inicial, el objetivo de dicho módulo es lanzar utilizar un servidor X para mostrar las imágenes generadas por el servidor. Dicho módulo no ha sido estudiado en profundidad en la realización de este proyecto pero, a nivel general, podemos comentar que el comportamiento es similar al del módulo Xvnc pero lanzando un servidor X, lo que nos permite el ahorro del uso del protocolo VNC para la gestión de imágenes en el servidor.

ALTERNATIVAS AL PROTOCOLO RDP

En el inicio de nuestro proyecto, y durante la fase de documentación que nos llevo los tres primeros meses, pudimos observar diferentes alternativas al protocolo RDP para realizar la gestión de equipos distantes mediante una interfaz gráfica. Dichos meses fueron empleados en la lectura de la especificación, la comprensión del protocolo y la comprensión del código del proyecto XRDP, además de la realización de diversas tareas de mejora de dicho código en el entorno del producto Wallix Admin Bastion. A continuación se va a realizar una descripción un poco mas detallada de algunas de las alternativas encontradas:

Virtual Network Computing (VNC)

La primera alternativa que analizamos, y que es en parte utilizada en el proyecto XRDP, es la denominada Virtual Network Computing que está implementada en diversos clientes-servidores del protocolo VNC tanto para entornos Windows como para entornos Linux. Dicha alternativa tiene diversas ventajas como el hecho de tener una mayor presencia en todo tipo de sistemas operativos debido a que es un protocolo ampliamente desarrollado para todo tipo de plataformas. La tecnología VNC se basa en el protocolo RFB (Remote Frame Buffer) cuyo funcionamiento pasamos a detallar a continuación.

Dicho protocolo está basado en el envío de actualizaciones cada vez que tiene lugar un evento en el servidor VNC. Esta implementación es bastante restrictiva cuando nos encontramos en un entorno en el que existen muchos intercambios de datos de manera consecutiva, como por ejemplo cuando estamos visualizando un video en pantalla completa. En este caso, tenemos una enorme cantidad de intercambios lo que nos lleva a tener un gran consumo de ancho de banda y, como consecuencia directa, una elevada carga de tratamiento de datos en el sistema.

Otro punto que cabe destacar en este protocolo, es la falta de encriptado de los datos implícita al protocolo RFB, con lo que, si se tiene la necesidad de transmitir las

informaciones de manera segura, se hace necesario establecer un canal VPN o SSH antes de realizar el intercambio de los datos a través del protocolo VNC. En nuestro caso ambas restricciones, el ancho de banda y el nivel de seguridad requerido, son básicas en nuestro producto, y es por ello que los diseñadores iniciales del producto Wallix Admin Bastion descartaron esta solución de gestión gráfica de equipos remotos.

Otras alternativas

Existen otras alternativas que pudimos conocer y analizar a lo largo de nuestra fase de documentación y que citamos a continuación, sin entrar demasiado en detalles en ninguna de ellas ya que no es el objetivo de este proyecto realizar un análisis exhaustivo de las diferentes alternativas de gestión remota de equipos a través de protocolos denominados gráficos.

- **XDMCP (X Display Manager Control Protocol):** protocolo utilizado en redes para conectar un terminal con un servidor de gráficos, principalmente utilizado en ambientes UNIX / Linux lo que resulta un punto de bloqueo crítico ya que dicho protocolo no serviría para nuestro propósito de poder gestionar equipos Windows a través de nuestro producto Wallix Admin Bastion.
- **Protocolo SSH (Secure SHell):** combinado con el proyecto X11forwarding que está bajo licencia BSD (Berkeley Software Distribution) y está basado sobre el protocolo X11 y que permite la interacción gráfica en red entre un usuario, denominado cliente, y una computadora, denominada servidor. Dicho protocolo X fue desarrollado inicialmente para dotar a los sistemas UNIX de una interfaz gráfica, pero debido a la existencia de un estándar definido para el protocolo X, este protocolo ha sido desarrollado para diversas plataformas y sistemas operativos.

CONCLUSIÓN

En este capítulo se ha realizado una descripción general del protocolo RDP sobre el que vamos a trabajar durante todo este proyecto. Se ha comenzado por dar una visión general de la arquitectura del proyecto así como su modo de funcionamiento. A continuación se ha proporcionado una visión del estado actual del proyecto XRDP y las funcionalidades sobre las que vamos a trabajar durante nuestro proyecto.

Finalmente, se han analizado algunas de las alternativas posibles y realizado una crítica y un posterior análisis de las restricciones intrínsecas a nuestro proyecto para así determinar la elección de la tecnología escogida en el seno del producto Wallix Admin Bastion.

En el capítulo siguiente se va a detallar todo el proceso seguido durante el proyecto, es decir, se va a comenzar por detallar las etapas de documentación, posteriormente se va a realizar un análisis de la solución adoptada y modificaciones y aportaciones de nuestro proyecto al proyecto general XRDP.

CAPITULO III: PLANTEAMIENTO Y DESARROLLO DE LA SOLUCIÓN.

INTRODUCCIÓN

En este capítulo se va a realizar un recorrido por las diferentes tareas llevadas a cabo a lo largo del proyecto. Se analizará cada una de estas tareas junto con su justificación técnica y en la parte final de dicho capítulo, se expondrán las conclusiones del proyecto.

El objetivo inicial de este proyecto es el desarrollo de nuevas funcionalidades para realizar la adaptación de un proxy RDP de acuerdo a las especificaciones proporcionadas por Microsoft.

Cabe destacar que el desarrollo de dicho proxy RDP, y por consecuencia de nuestro proyecto fin de carrera, se va a realizar en unas condiciones un tanto particulares. La primera de esas condiciones es que,, dicho proyecto, se enmarca dentro de unas prácticas en empresa, y es en el seno del desarrollo de uno de los productos de dicha empresa, en el cual se va realizar la adaptación del proxy RDP ya existente para adecuarlo a las especificaciones liberadas por Microsoft en Enero de 2009. Como consecuencia de la adaptación a dichas especificaciones, se va a dotar a dicho proxy de una amplia compatibilidad con una gran parte de los dispositivos presentes en el mercado en el momento del desarrollo de nuestro proyecto fin de carrera lo que aporta un aliciente tanto académico como profesional a nuestro proyecto.
La principal funcionalidad que debe ser desarrollada en el marco de nuestro proyecto fin de carrera, y que fue definida por nuestro responsable del proyecto en la empresa, es la capacidad de sincronizar el sistema de portapapeles entre el cliente y el servidor, y que nos permitirá realizar la funcionalidad de “copiar/pegar” entre ambos equipos. Otros de los objetivos a desarrollar son las funcionalidades de montaje de un sistema de ficheros local en el servidor distante, la adaptación del módulo RDP a posteriores versiones del protocolo, redirección del sonido entre el cliente y el servidor, etc.

ETAPA INICIAL DEL DESARROLLO

Durante la primera etapa del desarrollo del proyecto, se realizó principalmente un trabajo de documentación y adaptación al entorno del mundo UNIX/Linux.

Durante un periodo inicial que cubrió las dos primeras semanas, nos limitamos a realizar la instalación, documentación y personalización de las diferentes herramientas que serían utilizadas durante el desarrollo del proyecto. Entre dichas herramientas se

encuentran, un sistema operativo Ubuntu 8.10 con una posterior migración a Ubuntu 9.04, que no son sino el nombre de dos de las distribuciones Linux comerciales más conocidas por su simplicidad y su estabilidad. Posteriormente, se siguió un proceso de introducción al uso de varias herramientas de seguimiento de proyectos de desarrollo común como Subversion (SVN) para la gestión de proyectos comunes, Bugzilla para la gestión de los diferentes errores encontrados en el producto Wallix Admin Bastion (WAB) y Trac, una herramienta para la gestión y planificación de las funcionalidades previstas para siguientes versiones del producto escrita en Python y distribuida sobre una licencia BSD (Berkeley Software Distribution). Estas dos últimas herramientas fueron sustituidas por una sola a lo largo del proyecto, dicha herramienta es conocida bajo el nombre de "Redmine". La totalidad de las herramientas utilizadas durante el desarrollo del proyecto están fuertemente ligadas al mundo Open Source ya que es una de las limitaciones impuestas por nuestra empresa a la hora de realizar el proyecto como podrá verse en el Anexo I en el que se realiza una breve presentación de la empresa y sus proyectos con el objetivo de dar una idea global del entorno en el cual se desarrolló el proyecto fin de carrera.

Una vez concluida la primera fase de conocimiento básico de las herramientas a utilizar a lo largo del proyecto, se procedió a la instalación del proyecto XRDP. Dicho proyecto, ya existente a nuestra llegada a la empresa en la que realizamos nuestras prácticas, es la base sobre la que deberemos desarrollar nuestras funcionalidades que van a componer el cuerpo principal de nuestro proyecto.

El primer paso fue realizar la descarga de la última versión estable en el momento del comienzo del proyecto. Seguidamente, y tras solucionar algunos problemas de instalación, pasamos a realizar una serie de pruebas de utilización del programa desde el punto de vista de un usuario, para empezar a comprender el funcionamiento del protocolo RDP.

ETAPA DE DOCUMENTACIÓN SOBRE EL PROTOCOLO RDP Y SOBRE LA SOLUCIÓN XRDP.

La segunda etapa de nuestro proyecto es la etapa de documentación. Durante dicha etapa fueron realizadas numerosas consultas a diversas fuentes sobre el protocolo RDP. Principalmente las consultas que se realizaron, estuvieron basadas en los proyectos Rdesktop (recordemos que Rdesktop es un proyecto que implemente un cliente RDP en entornos UNIX / Linux) y XRPD (proyecto destinado a implementar un servidor RDP en entornos UNIX / Linux).

Antes de realizar un análisis del código fuente de los proyectos citados en el párrafo anterior, se realizó una detallada consulta a la especificación proporcionada por Microsoft en Enero de 2009. En esta especificación quedan completamente descritas todas y cada una de las etapas de conexión y de establecimiento de la sesión RDP entre el cliente y el servidor así como el detalle de todos y cada uno de los mensajes intercambiados entre el cliente y el servidor para cada una de las posibles acciones realizadas por el protocolo RDP. El nivel de detalle de dicha especificación es muy elevado, lo que hace que existan numerosas referencias cruzadas a otros documentos, y dificulta la lectura y comprensión de dicha especificación. Sin embargo, también se puede destacar la existencia de ciertos campos pertenecientes a determinadas tramas intercambiadas durante la sesión RDP cuyo uso no está completamente definido, lo que hace que a la hora de implementar algunas de las nuevas funcionalidades, dichos campos se reserven para implementaciones futuras.

La especificación del protocolo RDP revelada por Microsoft en Enero de 2009 consta de 20 documentos con una extensión aproximada de 2000 páginas detallando cada uno de ellos, una funcionalidad o conjunto de funcionalidades incluida en el protocolo RDP. Así pues existen diversos documentos entre los que podemos citar el documento encargado de detallar la etapa de conexión y de intercambio básico de entradas y salidas entre cliente y servidor (denominado MS-RDPBCGR); el documento encargado de detallar el establecimiento de un canal y el intercambio de mensajes dedicado a la redirección de un sistema de ficheros local (denominado MS-RDPEFS), otro documento encargado de definir todos y cada uno de los mensajes y los canales virtuales que son establecidos para la sincronización de la herramienta de portapapeles entre cliente y servidor, etc.

Una vez finalizada la lectura y comprensión de esta vasta y compleja especificación, se procedió a una lectura y análisis del código fuente proporcionado por el proyecto XRDP con el objetivo de obtener una visión global y completa del funcionamiento del mismo.

Para realizar de una forma correcta esta segunda fase de documentación, en un primer momento se comenzó a trabajar sobre el código de la última versión estable aparecida en Junio de 2008, para posteriormente, trabajar sobre la versión más reciente y en vías de desarrollo disponible a través de CVS (**C**oncurrent **V**ersion **S**ystem) en la página web del proyecto XRDP (http://sourceforge.net/projects/xrdp/) y que fue la que posteriormente se utilizó durante todo el proyecto.

La lectura y comprensión del código fuente (aproximadamente unas 30.000 líneas de código C), para un ingeniero de telecomunicaciones con unas nociones no muy elevadas de programación en C, llevo prácticamente un mes y medio de trabajo. Dicha carga de trabajo incluye un aprendizaje de un nivel elevado de programación en C, así como una adaptación de los conocimientos de C adquiridos durante la carrera a la programación orientada a estructuras. La lectura y comprensión es bastante intuitiva, debido principalmente al buen uso de la declaración de variables, estructuras y

funciones si bien es cierto que el nivel de programación usado en el proyecto es bastante elevado.

Una vez terminada la lectura y comprensión del código fuente del proyecto XRDP, sobre el cual se desarrolló nuestra solución, y con la finalidad de tener una visión detallada de todos los intercambios realizados entre el cliente y el servidor durante una conexión RDP, se paso a realizar un análisis del código fuente del proyecto Rdesktop, que recordemos, implementa un cliente RDP para entornos UNIX / Linux.

Dicho análisis fue llevado a cabo de una manera menos exhaustiva y concienzuda debido a que buena parte del código del proyecto XRDP está basado en código del proyecto Rdesktop (http://sourceforge.net/projects/rdesktop/).

El objetivo final de la lectura de ambos códigos, tanto del proyecto XRDP que implementa un servidor RDP, como del proyecto Rdesktop, que implementa un cliente RDP, es la total comprensión del protocolo RDP, así como el conocimiento de todas las informaciones que se intercambian en cada una de las tramas durante la sesión RDP entre cliente y servidor y el modo en que dichas tramas son interpretadas por cada una de ambas entidades.

La adquisición de dichos conocimientos resultó obligatoria a la hora del desarrollo de nuestro PFC, ya que para la implementación de nuestras las nuevas funcionalidades, se debe conocer perfectamente todo lo relativo a la gestión de los canales en una sesión RDP.

Por último, cabe destacar, una vez finalizada la lectura de toda la documentación, que la gran parte del código de ambos proyectos ha sido fruto de un gran y arduo ejercicio de retro-ingeniería debido a que ambos proyectos datan de años antes de la publicación de la especificación del protocolo RDP en Enero de 2009 año por Microsoft, en parte obligada por la legislación internacional.

ETAPA DE MANIPULACIÓN DEL CÓDIGO FUENTE DE XRDP Y DESARROLLO DE LA PRIMERA SOLUCIÓN ADOPTADA.

Tras haberse finalizado la primera fase de documentación del protocolo RDP correspondiente a la lectura y comprensión del código de los proyectos XRDP y Rdesktop, se procedió a desarrollar una serie de test funcionales para una comprensión óptima de dicho código. Una vez finalizada dicha fase, continuamos por la etapa de concepción de la solución al problema planteado. Dicha fase de test, fue concebida junto con mi director de proyecto y durante la misma realizamos una

implementación de una arquitectura cliente – proxy – servidor en nuestro equipo local. Una vez dicha arquitectura fue operacional sobre nuestro puesto de trabajo, pasamos a realizar diversos test de conexión a través de los diferentes módulos que componen el proyecto XRDP, así como con diferentes juegos de datos de configuración. Mediante la interpretación de las tramas intercambiadas entre el cliente – proxy – servidor durante esta etapa, pudimos obtener un nivel de comprensión más elevado del protocolo RDP.

El propósito de este proyecto es el desarrollo de nuevas funcionalidades de un proxy RDP en el ámbito del proyecto XRDP. Una de las funcionalidades principales que se debe desarrollar es la de montaje de un sistema de ficheros local en el servidor remoto.

Dicha funcionalidad puede ser descrita del siguiente modo, en el momento del establecimiento de la conexión RDP entre el cliente y el servidor a través de nuestro proxy RDP, dicho cliente tiene la opción de re-direccionar diversos dispositivos hacia el servidor distante, algunos de estos dispositivos son la memoria USB, la impresora, el sonido y el sistema de ficheros. El montaje remoto y posterior sincronismo de dicho sistema de ficheros entre el cliente y el servidor es la base para poder realizar la copia de archivos entre el cliente y el servidor distante. Dicha transferencia de archivos es posible ya que, si somos capaces de montar un sistema de ficheros que va a ser re-direccionado al servidor, podremos crear archivos ejecutándose en el servidor cuyos cambios se verán reflejados en el mismo sistema de ficheros del cliente.

Una vez definido el objetivo inicial de nuestro proyecto, y tras una reunión con el director del equipo de I+D y mi director de proyecto, se definió un entorno técnico a nivel de plataformas disponibles para el desarrollo del proyecto que contaba con tres partes bien diferenciadas. La primera plataforma es una máquina local destinada a ser el cliente RDP que lanzará las conexiones pertinentes (dicha máquina será una máquina virtual creada en mí puesto de trabajo con el fin de simular un cliente RDP). La elección de lanzar el cliente RDP, ya sea a través de un SO Windows instalado sobre una máquina virtual en mi ordenador local, o sobre un sistema Linux instalado de la misma manera, tiene como objetivo evitar la pérdida de informaciones ante un eventual cierre inesperado de la sesión RDP.

La segunda plataforma, es un servidor dedicado a ejercer las funciones de proxy RDP, en la que se va a instalar el programa XRDP que deberemos desarrollar para la implementación de las nuevas funcionalidades, dicha máquina se encuentra instalada sobre el mismo puesto de trabajo que el cliente RDP. Finalmente una tercera plataforma, en la que tendremos instalado un servidor Windows 2003 para realizar los ensayos de conexiones y poder determinar si los cambios realizados funcionan correctamente o no. Una vez probados los cambios sobre un servidor Windows 2003, intentaremos ampliar la compatibilidad hacia otros sistemas Windows tales como Windows Vista y Windows Seven.

Tras haber sido definida la arquitectura de trabajo junto con el director del equipo I+D y mi director de proyecto, se prosiguió por dar comienzo a la etapa de concepción de la solución para el desarrollo del proxy RDP. El primer paso que fue establecido fue contactar con el ingeniero encargado de desarrollar y administrar el proyecto XRDP y que es actualmente también la persona encargada de dirigir el proyecto,. Tras unos primeros intercambios de nuestros objetivos y un posterior informe detallado de lo que se deseaba realizar hacer, nos aportó algunas pistas sobre el desarrollo presente y futuro del proyecto XRDP, del cual él es el máximo responsable y de cómo podríamos integrar nuestro trabajo en dicho proyecto.

Dicho proyecto XRDP está compuesto de una parte principal (denominada núcleo del programa) y tres librerías dinámicas además de un gestor de sesiones. Como también se comentó anteriormente, nuestro objetivo es hacer evolucionar el proxy RDP dentro del marco del producto Wallix Admin Bastion que es un proxy multi-protocolos con características de trazabilidad, autentificación y gestión y auditoría de conexiones.

Es pues, en el marco de este producto Wallix Admin Bastion, en el cual se ha comenzado el desarrollo de nuestro proyecto. Inicialmente, en dicho producto, el programa XRDP está siendo utilizado de la siguiente manera: el producto Wallix Admin Bastion, que pasará a ser denominado como WAB (su nombre comercial) a partir de este momento, lanza inicialmente el programa XRDP junto con la librería Xvnc. ¿En que se traduce esto?, dicho funcionamiento se traduce en que para cada conexión entrante a nuestro producto WAB, el programa XRDP va a recoger los datos de conexión, principalmente el nombre de usuario y la contraseña, y va a aplicarle las reglas definidas por el motor de ACL (Acces Control List), propio al producto WAB, para definir si el usuario tiene derecho a conectarse al equipo distante que ha solicitado o no. Dicha verificación de reglas y autentificación será realizada por el gestor de sesiones como se comentó en el capítulo anterior.

Una vez que el usuario ha sido autentificado, pasará a abrirse una nueva sesión. En este caso, se lanzará un nuevo servidor VNC que será el encargado de tratar las imágenes provenientes del servidor RDP distante. Por el momento como se ha podido observar, todo se está desarrollando en la máquina que tiene instalado el programa XRDP, por lo cual no existe, de momento, la noción de proxy ya que no hemos realizado la conexión con el equipo distante. Es en este momento cuando entran en acción otros bloques constitutivos del producto WAB que pasaremos a explicar a continuación.

En el mismo momento en el que el producto WAB es instalado y el programa XRDP es lanzado, se lanza también un cliente VNC que nos permitirá visualizar todas las imágenes que son tratadas por el servidor VNC lanzado por el gestor de sesiones, es a través de este cliente VNC que vamos a poder lanzar un cliente RDP que realizará la conexión hacia el equipo remoto dando así a nuestro producto la noción de proxy RDP requerida.

A continuación se presenta un esquema general del posicionamiento del programa XRDP en el marco de nuestro producto WAB (véase figura 5). Con el objetivo de dar una visión mas precisa del esquema presentado a continuación, se va a realizar un análisis del diagrama de flujo siguiente.

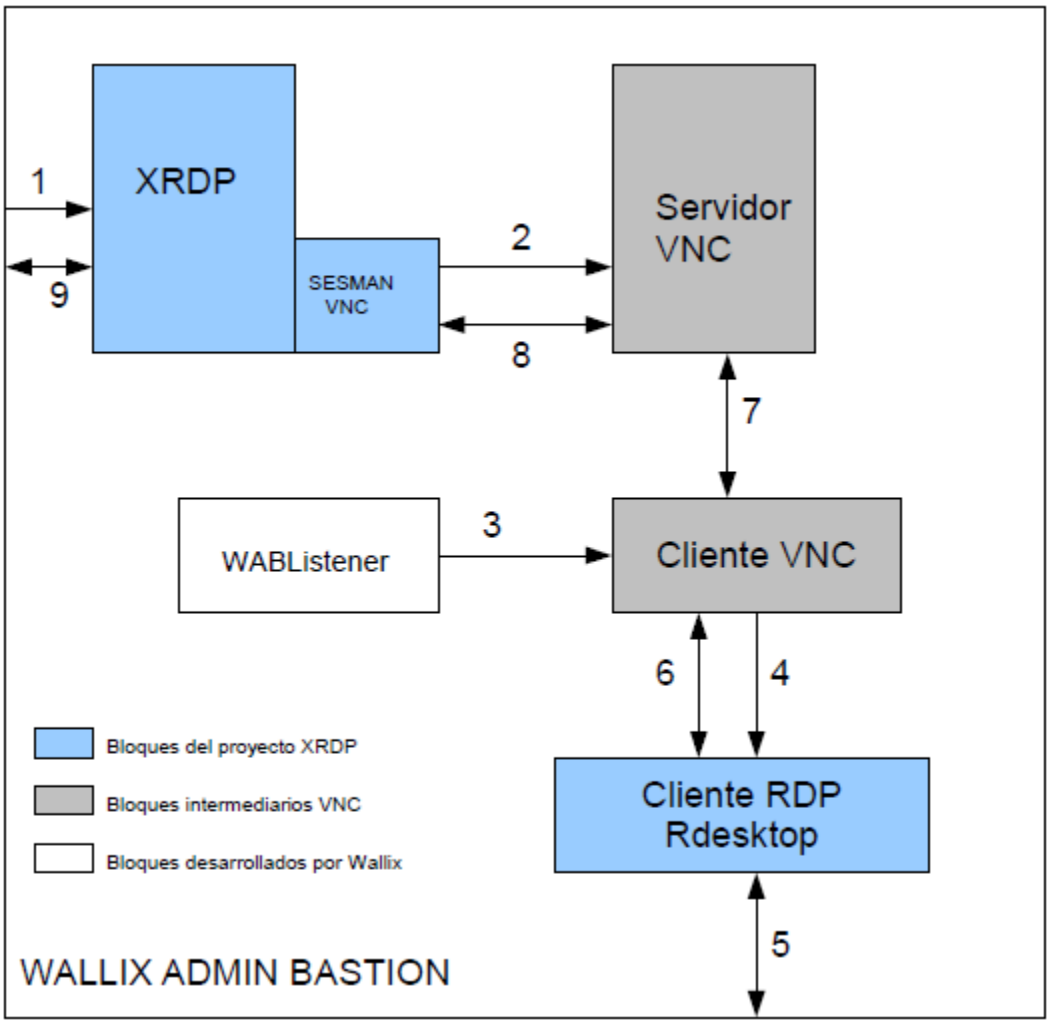

Figura 5. Integración del proyecto XRDP en el producto Wallix Admin Bastion

1. Nuestro producto WAB recibe una tentativa de conexión a través del puerto predeterminado para el protocolo RDP con lo que redirige esta tentativa hacia el bloque denominado como núcleo XRDP instalado en nuestro producto. A continuación, dicho bloque del programa recibe la conexión, crea un hilo de ejecución para tratar esta conexión como explicamos en el capítulo anterior y posteriormente lanza el módulo sesman-Xvnc para tratar dicha conexión. Como se explicó en el capítulo anterior, durante este proceso, el gestor de sesiones denominado sesman va a efectuar las tareas de autentificación y de puesta en ejecución de un servidor VNC.

2. Dicho servidor VNC, que será el encargado de gestionar las sesiones gráficas entre el cliente y el servidor final, es lanzado con unos parámetros por defecto definidos por el gestor de sesiones y se establece en un modo de escucha de conexión en espera de recibir una conexión VNC por parte de algún cliente VNC. Cabe destacar que las informaciones que circulan entre el bloque XRDP y el servidor VNC lanzado por el gestor de sesiones se hacen siguiendo el protocolo VNC. Dichas informaciones, que circulan entre el cliente y el bloque XRDP son interpretadas por el módulo VNC y son transformadas a paquetes RDP que serán enviados vía XRDP hacia el cliente que inicio la comunicación en 1.

3. Paralelamente al lanzamiento del servidor VNC realizado por el bloque XRDP, el bloque denominado WABListener, que es interno al producto Wallix Admin Bastion, va a ser el encargado de lanzar un cliente VNC que posteriormente realizará la conexión con el servidor VNC lanzado por XRDP con el objetivo final de establecer de una cierta manera un proxy RDP entre el cliente y el servidor final, de una manera transparente para el usuario.

4. En el siguiente paso, y gracias a la ayuda del cliente VNC, se va a realizar el lanzamiento del cliente RDP para entornos UNIX / Linux, cabe recordar que el producto denominado WAB va instalado sobre un distribución del SO Linux denominada Debian Lenny.

5. En este punto, el cliente Rdesktop va a realizar la conexión hacia el equipo distante, a través del protocolo RDP mediante los parámetros que han sido transmitidos por el bloque WABListener.

6. Una vez la conexión ha sido realizada de una manera satisfactoria, todas las informaciones recibidas de parte del equipo remoto por el cliente Rdesktop van a ser tratadas y transmitidas como información al cliente VNC, a través de un servidor X que es ejecutado por el mismo cliente Rdesktop que interviene durante toda la etapa de ejecución de la sesión.

7. Dichas informaciones gráficas, siempre en formato VNC, son transmitidas de cliente a servidor VNC en este paso durante toda la sesión RDP.

8. En este punto, todas las informaciones VNC que han sido transmitidas del servidor VNC al módulo VNC presente en XRDP van a ser tratadas, y retransmitidas hacia el cliente que inició la conexión, pero esta vez según el protocolo RDP que es el protocolo usado por el cliente para conectarse al servidor final a través de nuestro “proxy” RDP cuyo funcionamiento hemos descrito a continuación.

9. Finalmente dichos paquetes RDP formateados en el módulo VNC incluido en XRDP, son transmitidos al equipo local y así tenemos como resultado una conexión RDP de principio a fin pasando por nuestra arquitectura de producto WAB.

En resumen, el producto Wallix Admin Bastion, recupera las informaciones procedentes del cliente según el protocolo RDP, y a través del mecanismo que ha sido explicado en los párrafos anteriores, se encarga de dirigir las informaciones hacia el servidor final sobre el cual el cliente ha solicitado la conexión, siempre siguiendo el protocolo RDP. El hecho de no disponer de un verdadero proxy RDP al interior del producto WAB, como consecuencia de no existir como tal en el proyecto XRDP, hace que deba ser utilizada una arquitectura como la que hemos mostrado anteriormente.

Una vez explicada la arquitectura de nuestro producto en lo que respecta a los módulos usados y a su funcionamiento interno, nuestra solución debía centrarse pues, en cómo gestionar los canales virtuales necesarios para realizar el desarrollo de la redirección de un sistema de ficheros entre el cliente y el equipo distante. Cabe recordar en este punto que las entradas y salidas estándar del protocolo (véase entradas de ratón y teclado para el cliente y salidas gráficas para el servidor RDP), son gestionadas a través de un canal estándar que es iniciado y gestionado en el cuerpo principal de XRDP.

Como puede comprobarse gracias a la explicación dada anteriormente, la forma en que el proyecto XRDP está siendo utilizada en el producto WAB no representa exactamente la noción de un proxy RDP pura. Para resumir la explicación anterior, todas las informaciones procedentes del cliente RDP, son transformadas al protocolo VNC, asimismo, todas las informaciones procedentes del servidor final RDP son también transformadas al protocolo VNC dentro de nuestro proyecto WAB, con lo cual, el concepto de proxy puramente dicho no existe ya que para realizar la conexión RDP de principio a final, ha sido necesario pasar a través del protocolo VNC y además realizar el lanzamiento de un cliente Rdesktop en el interior de nuestro producto WAB, para emular dicho comportamiento de proxy. Esto es debido a que el proyecto XRDP, tiene como objetivo desarrollar un servidor RDP sobre entornos UNIX / Linux y no desarrollar un proxy RDP sobre entornos UNIX / Linux, de ahí que nosotros hayamos debido usar esta transformación al protocolo VNC intermedia.

Para realizar la correcta gestión de los demás canales virtuales que podrían permitirnos obtener las funcionalidades de redirección de dispositivos, gestión del portapapeles para la función de copiar y pegar y sonido principalmente, y tras haber realizado un estudio del código como explicamos en capítulos anteriores, se decidió utilizar un módulo adyacente denominado chansrv.c. Después de tomar contacto con el ingeniero al cargo del proyecto XRDP ambos decidimos centrar nuestros esfuerzos en desarrollar este módulo para tratar todos los mensajes provenientes de canales virtuales distintos al canal estándar de RDP. Así pues, todos los mensajes de canales virtuales una vez iniciada la conexión deberán ser tratados por dicho módulo chansrv.c.

A continuación se va a realizar una breve explicación del funcionamiento de este módulo. Este módulo, denominado “chansrv” en el código del proyecto XRDP, es lanzado por el gestor de sesiones al mismo tiempo que es lanzado el servidor VNC.

Nuestra primera solución pues deberá consistir en desarrollar este módulo para aceptar los mensajes entrantes de todos los canales virtuales. Una vez realizada esta tarea, será necesario desarrollar un módulo concreto para tratar los mensajes de intercambio presentes en la especificación Microsoft del protocolo RDP denominada "Remote Desktop Protocol: File System Channel Virtual Extension" (cuyo contenido será tratado en los anexos de este proyecto fin de carrera). El desarrollo de las funciones necesarias para adaptar el módulo chansrv.c a la recepción de mensajes globales de inicio y confirmación de canales junto con el desarrollo de las funciones necesarias para el intercambio de mensajes y datos del canal denominado como "rdpdr" de uso exclusivo para la redirección del sistema de ficheros hizo que se realizara el desarrollo de unas 1500 líneas de código aproximadamente.

Una vez finalizada dicha etapa de codificación de mensajes de intercambio y adaptación del módulo "chansrv", se pasó a realizar la segunda etapa que no es otra que la etapa consistente a la integración de los mensajes recibidos a través del canal "rdpdr" exclusivamente utilizado para los mensajes de redirección de sistema de ficheros.

Para esta segunda etapa y tras una consulta con el ingeniero responsable del desarrollo del proyecto XRDP, ambos decidimos, en colaboración, analizar la posibilidad de integrar el proyecto denominado como FUSE (Filesystem in Userspace) en el proycto XRDP para la gestión y sincronización del sistema de ficheros del cliente y del servidor RDP. Tras un primer análisis de dicho proyecto FUSE (disponible a través de la página web http://fuse.sourceforge.net/) se realizó la presentación de la posibilidad de integrar dicho proyecto en la arquitectura de nuestro producto WAB. Después de una serie de reuniones para el estudio de la viabilidad de esta integración, el equipo I+D decidió rechazarla debido a que el objetivo final de dicho producto era de dar el servicio requerido por el cliente con el menor número de dependencias posibles, hecho que no sería respetado si se hubiese realizado la adaptación del proyecto FUSE a nuestro producto.

Como resultado de esta primera etapa de desarrollo, que finalmente no decidió ser utilizada de forma práctica en el producto WAB, y tras haber enviado el código desarrollado al ingeniero encargado del desarrollo del proyecto XRDP, dicho responsable nos comunicó que el código generado para el intercambio de todos los mensajes especificados en la especificación de Microsoft referente al sistema de ficheros (aproximadamente unas 2000 líneas de código), iba a ser integrado en versiones posteriores del proyecto XRDP, con lo cual dicha primera solución, no aportó los resultados finales requeridos al proyecto fin de carrera, pero contribuyó de forma activa al desarrollo del proyecto "Open Source" denominado XRDP así como a la comprensión de la documentación relativa a la gestión de canales virtuales al interior del protocolo RDP.

SEGUNDA ETAPA DE DESARROLLO, DESARROLLO DEL MODULO RDP COMO PROXY RDP.

Tras un segundo análisis de las alternativas que podíamos desarrollar en ese momento, se decidió aprovechar el código ya desarrollado en la librería denominada "librdp.so" que está presente en el código del proyecto XRDP y cuyo funcionamiento ha sido explicado en capítulos anteriores de esta memoria. En dicha librería, como comentamos en el apartado anterior, se puede tener acceso a un cliente RDP, bajo la versión 4, que permitirá conectarse al equipo distante de una manera transparente para el usuario, es decir, que en cierto modo, dicha librería actúa como una especie de proxy RDP de cliente a servidor, con la única pero importante diferencia, que la sesión establecida entre el producto WAB y el servidor final se efectuará siguiendo la versión 4 del protocolo RDP.

A continuación se describen los diferentes problemas que han sido tratados a la hora de considerar dicho módulo como un verdadero proxy RDP.

El primer problema a tratar fue el siguiente, como explicamos en el capítulo anterior, cuando se realiza el lanzamiento de lo que ha sido denominado el núcleo del programa XRDP junto con la librería librdp.so el proceso seguido es el siguiente, el cliente RDP lanza una conexión que es recibida por el programa XRDP, dicho módulo XRDP se encarga de establecer la conexión con el cliente así como de desarrollar las primeras etapas de la secuencia de conexión publicadas en la especificación.
Una vez que el módulo XRDP ha recibido las informaciones de conexión del cliente así como la dirección IP del equipo destino, pasa a lanzar el módulo RDP en forma de librería dinámica como explicamos en el capítulo anterior.

Una vez que dicha librería ha sido lanzada, va ser la encargada de lanzar una nueva conexión hacia el servidor distante. Dicha nueva conexión no recupera los datos iniciales que el cliente ha transmitido al módulo XRDP con lo que se trata de una nueva conexión con unos parámetros definidos por defecto. Además de ello, dicha conexión es lanzada bajo la forma de versión 4.0 del protocolo RDP que, entre otras cosas, no permite ningún tipo de redirección como dijimos en el capítulo anterior, al no declarar en la etapa inicial de la conexión ningún otro canal diferente del canal principal de comunicación, lo que hace que ningún canal virtual alternativo esté disponible para realizar la redirección de dispositivos.

Una vez que la conexión entre la librería "librdp" es realizada con el equipo distante, todas las informaciones procedentes del cliente son desencapsuladas, desencriptadas y descomprimidas por el módulo XRDP y después son transmitidas a la librería librdp.so para que las realice el proceso inverso pero siempre siguiendo la especificación Microsoft definida para la versión protocolo RDP 4.0.

Por el contrario, las informaciones procedentes del servidor distante RDP son tratadas por la librería en forma de paquetes RDP versión 4.0 y transmitidas informaciones al módulo principal XRDP para ser encapsuladas siguiendo la versión y las especificaciones de seguridad definidas entre el cliente y el módulo XRDP antes de ser retransmitidas.

A continuación se presenta un esquema detallado del intercambio de los diferentes mensajes entre los diferentes bloques constitutivos de la sesión RDP (ver figura 6). La conexión entre el cliente RDP y el módulo principal XRDP es realizada siguiendo el conjunto de reglas de seguridad, versión del protocolo, número de canales a utilizar y tipo de compresión definidas en el momento de la conexión.

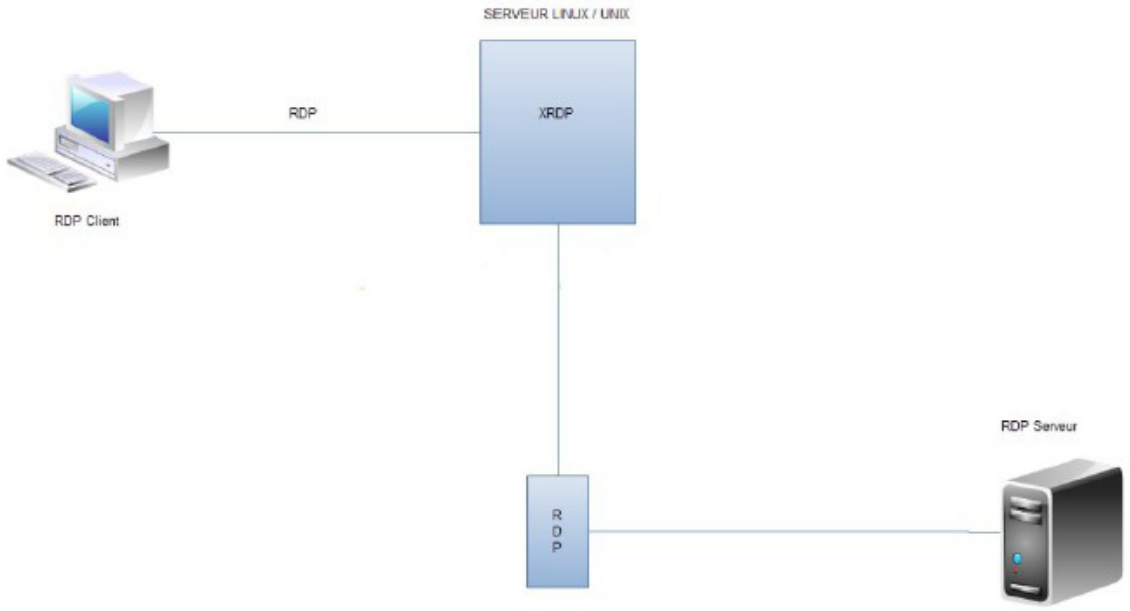

Figura 6. Flujo de mensajes entre los módulos XRDP y la librería RDP

Por otro lado, la conexión entre el módulo RDP y el servidor RDP se realiza siguiendo unos parámetros predeterminados que son definidos por el módulo RDP que funciona como cliente de la conexión módulo RDP – Servidor RDP.

Por último todas las informaciones que circulan entre el módulo XRDP y el módulo RDP son informaciones que carecen de protocolo debido a que son informaciones que han sido tratadas previamente por cada uno de los módulos para extraer únicamente los datos RDP y poder adaptarlos a cada una de las conexiones, según los datos procedan del cliente o del servidor.

Una vez analizado exactamente el modelo de funcionamiento de dicho módulo, se decidió adaptar el código del módulo RDP para crear un verdadero proxy RDP, para

ello, el primer objetivo era recuperar todas las informaciones de la conexión definida entre el cliente y el módulo XRDP.

Dichas informaciones son transmitidas durante la etapa de establecimiento de conexión entre el cliente y el módulo XRDP. El conjunto de informaciones que se deben recuperar son las siguientes: nivel de seguridad y encriptado negociado, conjunto de capacidades de cliente y servidor negociadas, número de canales virtuales negociados en la conexión, nivel de la compresión negociado entre cliente y servidor, informaciones gráficas referentes a la sesión y la versión del protocolo negociada entre el cliente y el servidor.

Una vez recuperadas todas estas informaciones y transmitidas al módulo RDP, el segundo objetivo es el de realizar la adaptación el módulo RDP a versiones posteriores a la versión 4 del protocolo RDP, debido a que inicialmente la única versión soportada era la versión 4 y esta no es suficiente para realizar un proxy totalmente funcional en lo que a redirección de dispositivos se refiere.

Para realizar la adaptación de dicho módulo, inicialmente creado como un cliente RDP versión 4.0, ha sido necesario trabajar en paralelo con la especificación proporcionada por Microsoft para determinar cual eran las diferencias entre los mensajes intercambiados entre el cliente y el servidor en versiones 4.0 y posteriores.

Una vez identificadas dichas diferencias, sobre todo a nivel de niveles de seguridad, gestión de canales virtuales y compresión de paquetes, pasamos a modificar función por función cada una de las funciones presentes tanto en la recepción de los paquetes por parte del servidor RDP como la recepción de las informaciones de datos RDP provenientes del módulo XRDP y que deben de ser transformadas en paquetes de acuerdo a la versión previamente recuperada.

Durante toda esta primera fase de recuperación de la información del cliente por el módulo XRDP y adaptación del módulo RDP a versiones posteriores del protocolo RDP, estuvimos en continuo contacto con el ingeniero encargado del desarrollo del proyecto XRDP como hemos comentado en numerosas ocasiones.

Las principales modificaciones realizadas se corresponden con la integración de certificados X.509, sobre la que todavía se está trabajando y que será integrada en el proyecto XRDP según su responsable principal una vez esté finalizada, como parte de las funciones de seguridad requeridas para ciertos niveles de seguridad de versiones superiores a la 4 del protocolo RDP.

Posteriormente modificamos las funciones de recepción y transmisión de paquetes a todos los niveles de capas componentes del protocolo, es decir a nivel TCP (Transfert Control Protocol), a nivel MCS (Multipoint Communication Service), a nivel SEC (Securisation) y finalmente a nivel RDP (Remote Desktop Protocol) para poder recibir correctamente paquetes de versiones posteriores a la 4 del protocolo RDP.

Una vez finalizado el conjunto de modificaciones realizado precedentemente, se va a pasar a realizar la modificación las funciones encargadas de transmitir las capacidades negociadas entre el cliente y el servidor debido a que hay ciertas capacidades que sólo son negociadas a partir de la versión 5 del protocolo.

Una de las modificaciones más importantes para la adaptación del módulo RDP a la versión 5 y posteriores es la gestión de canales virtuales que inicialmente no estaba prevista durante la versión 4 de dicho protocolo. Para ello ha sido necesario modificar todo el proceso de conexión establecida entre el módulo RDP y el servidor RDP final para ser capaces de recuperar la lista de canales negociados entre el cliente RDP y el módulo XRDP, transmitir dicha lista de canales como si fuera propuesta por el módulo RDP actuando como cliente RDP, recibir la confirmación de dicha lista de canales de parte del servidor RDP y finalmente realizar un intercambio de mensajes durante la etapa de conexión destinados a confirmar uno a uno cada uno de los canales anunciados en la lista de canales virtuales susceptibles de ser usados durante la sesión RDP.

Una vez que el código del módulo RDP ha sido totalmente adaptado a la versión 5 del protocolo RDP, se pasó a implementar las funciones que nos permitirán realizar la redirección de los datos entre el módulo XRDP y el módulo RDP. Se decidió desarrollar una función que recupera los datos procedentes del módulo XRDP y posteriormente los encapsula y los encripta de acuerdo al marco general de la conexión negociado en el inicio y es capaz de enviarlos al servidor RDP. Dicha función ha sido desarrollada y es completamente funcional.

Finalmente, y para terminar de gestionar correctamente los diferentes canales virtuales negociados durante la fase de establecimiento de la sesión, se pasó a desarrollar la función complementaria a la definida anteriormente. Esta función complementaria, denominada en nuestro código “rdp_process_redirect_data”, es la encargada de recuperar los paquetes procedentes del servidor, desencriptarlos y desencapsularlos, enviarlos al módulo XRDP a través del canal virtual previamente establecido y confirmado y que dicho módulo XRDP envíe el paquete al cliente RDP final. Dicha función ha sido completamente desarrollada y es funcional.

Así pues, una vez desarrolladas todas estas funciones, somos capaces de gestionar todas las informaciones procedentes de los canales virtuales definidos en la especificación de Microsoft. A través de dichos canales virtuales, somos capaces de gestionar las nuevas funcionalidades que nos habían sido propuestas como objetivo de nuestro PFC como son la redirección de discos locales y la función de copiar y pegar entre el cliente y el servidor RDP. Como complemento a dichas funcionalidades y debido a que la gestión de las informaciones que circulan a través de los canales virtuales es totalmente operativa, podemos incluir como objetivos secundarios no propuestos inicialmente en nuestro PFC, la gestión remota de impresoras, la gestión remota del sonido así como el montaje de dispositivos USB a distancia.

El desarrollo de dichas funcionalidades secundarias para el PFC fueron recibidas muy positivamente en el entorno de la empresa debido a que aportaban un gran valor añadido al producto WAB.

Para finalizar, otro valor añadido al PFC, es la compatibilidad con nuevas versiones de SO Windows en términos de servidor. Esto es decir, hasta la fecha, la arquitectura que estaba propuesta como proxy RDP a nivel del producto WAB y que ha sido definida en la parte inicial de dicho capítulo, sólo permitía conectarse a servidores RDP tales como Windows 2003 y Windows XP. Al realizar la adaptación y conseguir hacer evolucionar la librería librdp.so hacia la versión 5 del protocolo RDP, hemos conseguido que dicha librería que nos permite actuar como proxy RDP, sea compatible con Windows Vista y Windows Seven así como con Windows 2008 server.

CRONOLOGÍA DEL PROYECTO.

Como último apartado de este capítulo, y con el fin de presentar las diferentes etapas en el tiempo durante el cual realizamos nuestro PFC, se van a presentar la evolución cronológica de las diferentes etapas del proyecto. Cabe destacar que las prácticas en la empresa Wallix en las cuales se enmarca el desarrollo de nuestro PFC comenzaron la segunda semana de febrero del año 2009.

Febrero 2009 : etapa inicial del desarrollo del proyecto, durante la cual se realizó una adaptación al entorno UNIX/Linux así cómo una primera etapa de comprensión de la arquitectura y funcionamiento del producto Wallix Admin Bastion.

Marzo – Abril 2009 : etapa de documentación y comprensión del protocolo RDP, así como comprensión del código del proyecto XRDP en el cual se basa la solución adoptada.

Mayo 2009 : etapa de desarrollo de la primera solución adoptada, análisis de la solución y de sus problemas asociados.

Junio 2009 – Agosto 2009 : etapa de desarrollo de la solución definitiva, integración en el producto Wallix Admin Bastion y etapa de validación funcional.

A continuación se presenta un diagrama de Gantt resumiendo las diferentes fases del proyecto, así como las fechas en las cuales tuvo lugar (ver figura 7).

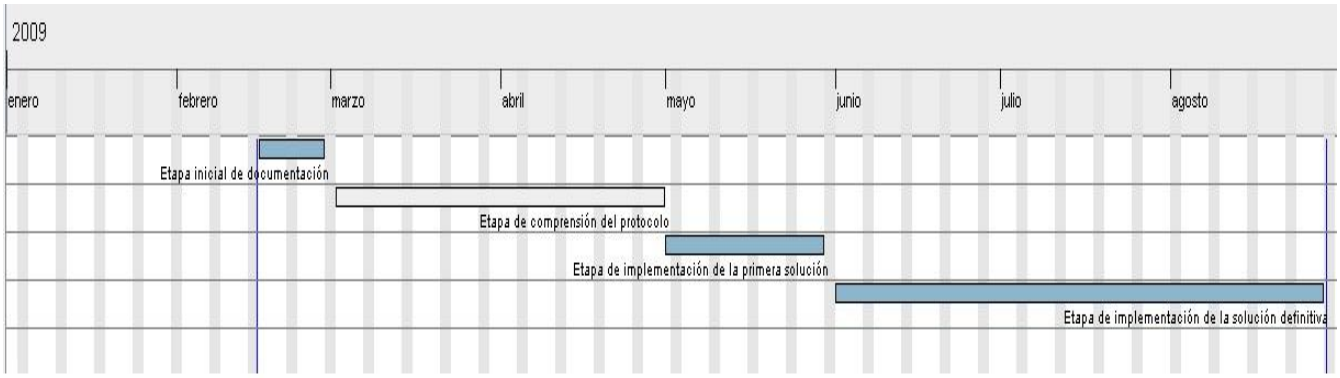

Figura 7. Diagrama de Gantt del proyecto

CAPITULO IV: CONCLUSIÓN TÉCNICA DEL PROYECTO Y ANÁLISIS DE LA EVOLUCIÓN FUTURA DEL MISMO

INTRODUCCIÓN

En este capítulo, se va a realizar una conclusión técnica del proyecto. En dicha conclusión técnica se analizarán el cumplimiento de los objetivos definidos al inicio del PFC además de un análisis de las etapas intermedias realizadas para el cumplimiento de dichos objetivos.

ANÁLISIS DEL CUMPLIMIENTO DE LOS OBJETIVOS INICIALES

Cabe recordar cuales eran los objetivos definidos para este proyecto fin de carrera. El objetivo general era adecuar el proxy RDP que se encontraba en el producto Wallix Admin Bastion con el fin de adecuar dicho proxy a las especificaciones publicadas por Microsoft en Enero de 2009.

Desde un punto de vista más funcional, dichas evoluciones se sintetizan en el desarrollo de las funcionalidades siguientes :

- Desarrollo de la sincronización del portapapeles entre un equipo cliente y un servidor remoto.
- Desarrollo del mapeo de discos locales de un equipo cliente en un servidor remoto.
- Desarrollo de la gestión de impresoras y periféricos a distancia.

Dichos objetivos fueron cumplidos y permitieron al alumno obtener un contrato de duración indefinida tras el aporte importante realizado a la evolución del producto Wallix Admin Bastion y que permitió responder a unas necesidades incipientes en el mercado en el cual dicho producto está implantado.

Finalmente cabe destacar que tanto mi director de proyecto en la universidad de Telecom SudParis, como la presidenta y el tribunal que evalúo mi proyecto , así como mi director de proyecto en la empresa dieron una apreciación muy positiva del trabajo realizado durante el proyecto debido a una serie de dificultades técnicas que serán detalladas en el apartado siguiente. Otro punto a destacar es que la presentación realizada en la universidad Télécom Sud Paris realizada en Septiembre de 2009 se saldó con una nota de 14 / 20.

DETALLE DE LAS ETAPAS INTERMEDIAS NECESARIAS PARA EL CUMPLIMIENTO DE LOS OBJETIVOS INICIALES

En este apartado se va a realizar una explicación detallada de las diferentes fases llevadas a cabo para la consecución de dicho proyecto :

- Amplia comprensión del protocolo RDP así como de diversos protocolos relacionados con la gestión gráfica de servidores remotos, como resultado del estudio de las diferentes posibilidades realizado durante la primera fase del proyecto.

- Mejora del conocimiento de entornos UNIX/Linux, concretamente de la distribución Ubuntu así como conocimientos adquiridos sobre herramientas de gestión de proyectos de software a utilizados en el entrono laboral como SVN, Trac y Bugzilla.

- Desarrollo del módulo XRDP con el objetivo de obtener la dirección IP del cliente RDP y transmisión de dicha información al gestor de sesiones para su posterior utilización. Esta funcionalidad, que fue una de las tareas encargadas por el responsable de nuestro proyecto, fue desarrollada en paralelo en la etapa de documentación inicial sobre el protocolo RD.

- Desarrollo del módulo XRDP, notablemente a nivel de las funciones presentes en el módulo denominado “lang.c” con el objetivo de añadir la funcionalidad de gestión de los teclados belga y suizo en el módulo XRDP que no se encontraban presentes en la versión inicial del proyecto sobre la cual comenzamos a trabajar, dicha versión era la versión 0.4.1 del proyecto XRDP. Esta funcionalidad fue desarrollada en paralelo en la etapa de documentación inicial sobre el protocolo RDP.

- Desarrollo de diversas funciones necesarias para el intercambio de todos los mensajes presentes en la especificación “Remote Desktop Protocol: File System Virtual Channel Extension”. Dichas funciones con total de casi 2000 líneas de código serán incluidas en versiones posteriores del proyecto XRDP habiendo realizado de esta manera un alto nivel de contribución a dicho proyecto. Como se comentó en el apartado anterior, dichas funciones, presentes en el módulo “chansrv.c”, tienen como objetivo comenzar el desarrollo de dicho módulo, el cual, una vez terminado su desarrollo en el seno del proyecto XRDP, tendrá como función gestionar todos y cada uno de los canales virtuales establecidos entre el cliente y el servidor RDP (representado por el proyecto XRDP).

- Desarrollo de funciones necesarias para la recuperación del marco característico de la sesión durante la conexión entre el cliente RDP y el módulo XRDP. Asimismo, desarrollo de otras funciones necesario para realizar el paso

de dichas informaciones entre el módulo XRDP y el módulo RDP. Parte de este marco característico de la sesión entre cliente y servidor RDP, corresponde a la lista de canales virtuales negociados entre cliente y servidor al inicio de la conexión, otra de las informaciones importantes es el nivel de compresión de datos, la dirección IP del cliente, etc.

- Desarrollo de las funciones de encriptado y compresión de datos presentes en todas las capas del protocolo para adaptarlas a versiones posteriores a la versión 4 del protocolo RDP. Como se mencionó anteriormente, el proyecto XRDP realiza un análisis de los paquetes recibidos, a través del "socket" con el cual está conectado al cliente o al servidor, en todos y cada uno de los diferentes niveles del protocolo. Así pues, cuando un paquete es recibido por el proxy RDP va a realizar un análisis a nivel de las capas TCP, MCS, SEC y finalmente RDP. La modificación de las funciones de encriptado y compresión de datos a todos estos niveles del protocolo es necesaria para poder realizar correctamente el análisis de los datos recibidos por el cliente o el servidor.

- Desarrollo de funciones de recepción y transmisión de paquetes en todas las capas del protocolo para adaptar la transmisión de paquetes a versiones posteriores a la versión 4 del protocolo RDP. Como para el caso anteriormente mencionado de recepción de paquetes, se ha debido adaptar el proceso de generación de paquetes RDP a todos los niveles del protocolo para poder asegurar una transmisión adaptada a versiones superiores de la versión 4 del protocolo.

- Desarrollo de funciones para realizar la gestión de los canales virtuales necesarios para la redirección de dispositivos. Dicha gestión de canales ha sido realizada para adaptar también el módulo RDP a versiones posteriores a la 4 del protocolo RDP.

- Desarrollo de las funciones necesarias para realizar la redirección de los paquetes que tienen como origen el servidor RDP y como destino el cliente RDP pasando por los módulos XRDP y RDP. Dichas funciones como ya se ha explicado anteriormente son las requeridas para recuperar los paquetes desencriptarlos y desencapsularlos y pasarlos al módulo XRDP para que sean enviados siguiendo el canal virtual correspondiente hasta el cliente RDP. Son las funciones que van a permitirnos recuperar las informaciones que pasan por cada uno de los canales virtuales negociados entre el cliente y el servidor al inicio de la conexión, y una vez estas informaciones recuperadas, tratarlas del mismo modo que las informaciones que circulan a través del canal principal (que recordemos que son las informaciones gráficas así como las informaciones de teclado y ratón).

- Desarrollo de las funciones necesarias para realizar la redirección de los paquetes que tienen como origen el cliente RDP y como destino el servidor RDP

pasando por los módulos XRDP y RDP. Como es lógico, en el punto anterior hemos hablado de las funciones que permiten recuperar las informaciones procedentes del servidor RDP hacia el cliente RDP, y en este punto se presentan las funciones que nos permiten realizar el análisis inverso, es decir, recuperar y procesar las informaciones procedentes del cliente RDP y con destino el servidor RDP final.

- Adaptación de la librería denominada como "librdp.so" a la versión 5 del protocolo RDP lo que implicó una mayor compatibilidad con nuevas versiones del SO Windows desarrollado por Microsoft.

Como se expuso en el capítulo 2, los objetivos iniciales del proyecto fueron definidos como muy optimistas por el director del equipo de I+D de la empresa Wallix en la cual desarrollé mi proyecto fin de carrera debido a numerosos puntos que se detallan a continuación:

- Ausencia de proyectos similares en la comunidad "Open Source" con lo cual no existen referencias sobre avances en proyectos similares, así como un desarrollo del proyecto XRDP a través de la comunidad "Open Source" que no coincidía exactamente con las necesidades planteadas para este proyecto, lo que hizo que la ayuda que la comunidad pudiera prestar para nuestro proyecto no fuera las más adecuada como es lógico.

- Ausencia de documentación clara y concisa con lo que respecta al protocolo RDP. Si bien es cierto que Microsoft proporcionó a la comunidad la especificación del protocolo RDP en Enero de 2009, dicha especificación no está bien redactada a juicio personal debido a que tiene gran cantidad de referencias a otras muchas especificaciones y eso hace que su lectura y comprensión no sea demasiado evidente. Además de ello, cabe destacar que, si bien la especificación fue publicada en 2009, esta permanece sin ser actualizada y por el contrario el protocolo RDP ha evolucionado desde la versión 4 hasta la versión 7 disponible en Windows 7 sin haber realizado ninguna modificación de dicha especificación.

- Gran cantidad de trabajo de investigación partiendo de una base técnica no demasiado amplia debido a no contar con documentos técnicos que expliquen claramente el funcionamiento del protocolo RDP en sus distintas versiones así como no poder contar con documentos técnicos relativos al proyecto XRDP, únicamente con su código fuente lo que hizo que el proceso de comprensión y lectura fuera enormemente largo y disminuyó el tiempo requerido para la parte de desarrollo de la solución.

EVOLUCIÓN DEL PROYECTO TRAS EL CUMPLIMIENTO DE LOS OBJETIVOS DEFINIDOS INICIALMENTE

En apartado se va a pasar a describir la evolución que siguió el proyecto una vez los objetivos definidos al inicio del mismo fueron alcanzados.

Para entender dicho proceso de evolución se va a comenzar por realizar una breve explicación de la situación e implicación de alumno en la empresa Wallix en la cual se desarrolló este proyecto fin de carrera así como la interacción con el ingeniero encargado del desarrollo y administración del proyecto XRDP.

Una vez finalizado el periodo de 6 meses de contrato de prácticas con el objetivo de desarrollar el proxy RDP en el marco del producto WAB, fue ofrecido al alumno un contrato de trabajo indefinido por la empresa al considerar que el trabajo realizado en dicho campo había progresado rápidamente y que dicho alumno se había convertido en el experto y referente en tecnología RDP de la empresa y hacia el que se dirigían las preguntas cuando de éste campo se trataban.

Así pues, tras dos meses de trabajo suplementarios sobre el proyecto se dotó a nuestro proxy RDP de una serie de funcionalidades complementarias como son la mejora en términos de compatibilidades de dicho proxy RDP, cabe recordar que al inicio de nuestro proyecto, el producto Wallix Admin Bastion era capaz de interactuar con versiones de servidores Windows 2003 y anteriores y tras una serie de evoluciones, dicho producto WAB es capaz de interactuar con equipos cuyo sistema operativo Windows 7 y anteriores.

Tras haber conseguido estos objetivos de mejora de la compatibilidad y mejora de la estabilidad del producto , un segundo ingeniero del equipo de Investigación y Desarrollo de la empresa Wallix empezó a trabajar conmigo para el desarrollo y la estabilidad del proxy RDP en el marco del producto WAB.

Tras una primera fase de reuniones en las que participamos ambos junto con el jefe de estrategia y Marketing y el jefe del equipo de Investigación y Desarrollo se comenzó a fijar los ejes de desarrollo del proxy así como las prioridades a integrar en dicho desarrollo como futuras compatibilidades, aumento de la seguridad de dicho protocolo, etc. Una vez definidos los ejes de desarrollo para los próximos meses pasamos a tener una serie de contactos con el ingeniero encargado del desarrollo del proyecto XRDP, y al ver que las contribuciones de nuestro proyecto no estaban en la misma línea que las contribuciones del proyecto XRDP se decidió, que una vez el

desarrollo fuera completado, se iba a concebir nuestro proxy como un nuevo proyecto que vería la luz en el mundo "Open source" en los próximos meses.

Dicho proyecto que se bautizó con el nombre de "Redemption" (http://www.technoaddict.fr/index.php/2010/06/wallix-sapprete-a-livrer-un-proxyrdp-opensource-a-la-communaute/) nos permitiría establecer nuestros propios ejes de desarrollo así como aportaría a Wallix una importante reputación en el mundo de la Seguridad Informática Open Source, al realizar una contribución de gran calado.

Una de las decisiones más importantes para este nuevo proyecto "Redemption" que se basa sobre el proyecto XRDP, es la del lenguaje de programación escogido para su desarrollo (cabe recordar que el proyecto XRDP está realizado en lenguaje C y estaba compuesto de unas 30000 líneas de código al inicio de nuestro PFC). Así pues y tras analizar las interacciones de nuestro proxy RDP con el resto del producto WAB, ambos decidimos que el lenguaje a escoger que se adaptaba más a nuestras necesidades es el lenguaje C++. Dicha decisión fue tomada tras el exhaustivo estudio del código, que aún programado en C, tenía un estilo orientado a objetos debido al amplio uso de estructuras y punteros.

Tras adoptar esta decisión, se pasó a realizar una larga etapa de evolución de código desde un código inicial en C hacia un código final en C++. Dicha etapa de re-programación del proxy RDP, nos permitió tener un profundo conocimiento de todas y cada una de las funciones del programa, lo que nos resulto muy útil para la resolución de problemas existentes sobre el proxy desarrollado en versiones anteriores del producto WAB.

Una vez finalizada esta larga etapa de re-escritura del código del proyecto, nos centramos en la estabilidad y en el desarrollo de nuevas funcionalidades. Concretamente, yo me concentré sobre la parte de trazabilidad y desarrollé módulos que permitían realizar la visualización en tiempo real de las sesiones que estaban siendo ejecutadas a través de nuestro proxy, así como la posibilidad de creación y almacenaje de sesiones ejecutadas a través de nuestro proxy en formato flv (Flash Video).

Finalmente, tras un año y medio de trabajo en Wallix sobre dicho proxy, el alumno decidió poner fin a su relación con la empresa para seguir trabajando en otros campos relacionados con las telecomunicaciones.

ANEXOS

ANEXO I. PRESENTACIÓN DE LA SOCIEDAD

LA SOCIEDAD WALLIX

La sociedad Wallix es un editor de soluciones con gran presencia en la comunidad "Open Source". Dichas soluciones tienen como objetivo reforzar la seguridad del Sistema de Información (SI) de la empresa así como gestionar dicha infraestructura. A nivel humano, Wallix está considerada como una Pequeña y Mediana Empresa (PME) en constante crecimiento en Europa y EEUU.

Wallix desarrolla productos destinados a dotar de seguridad a los sistemas de información de sus empresas, los flujos de datos y los accesos a dichos sistemas de información. Dichos productos están basados en su experiencia como editor de soluciones para ambientes heterogéneos.

El desarrollo de estos productos tiene como objetivo cubrir las necesidades del mercado en materia de seguridad de sistemas de información de las empresas, desde un punto de vista del mundo "Open Source", lo que le confiere un amplio margen de reducción de costes en relación con otras soluciones propietarias.

La estrecha relación entre Wallix y la comunidad universitaria a través de su colaboración con centros de Investigación y desarrollo (I+D) a nivel universitario, dotan a Wallix y a sus productos de una continua mejora y de un continuo progreso a nivel tecnológico.

Finalmente, Wallix pone a disposición de sus clientes un centro de ayuda para sus clientes en inglés y en francés, que tiene como misión responder a todas las dudas técnicas y de sus clientes en el menor espacio de tiempo posible.

ORGANIZACIÓN DE LA SOCIEDAD

La sociedad forma parte del grupo IF Research y está estructurada internamente del siguiente modo:

- Departamento de Marketing y comercial:

Encargado de la prospección y posicionamiento del producto tanto a nivel nacional como a nivel internacional. Cabe destacar la presencia en países como Inglaterra, Alemania y Bélgica además de Francia, a nivel europeo así como la expansión, durante el último año, a Estados Unidos.

- Departamento de ingeniería Investigación y Desarrollo (I+D) y productos:

  Encargado del desarrollo del producto desde la etapa de concepción, con ayuda del equipo de marketing para integrar las peticiones de los clientes, hasta la etapa de test internos. Es en este departamento donde he trabajado durante el periodo de mi proyecto fin de carrera desarrollando el proxy RDP dentro del producto llamado Wallix Admin Bastion (WAB).

- Departamento de soporte técnico:

  Encargado del soporte técnico de los productos así como de las etapas de test de carga y de producción. Este departamento está encargado también de la instalación del producto en el lugar requerido por el cliente así como de realizar la formación a los clientes a propósito de las funcionalidades del producto.

- Departamento de contabilidad y recursos humanos:

  Encargado de la contabilidad y del proceso de firma de contratos de los nuevos empleados, así como gestión de nóminas y de subvenciones.

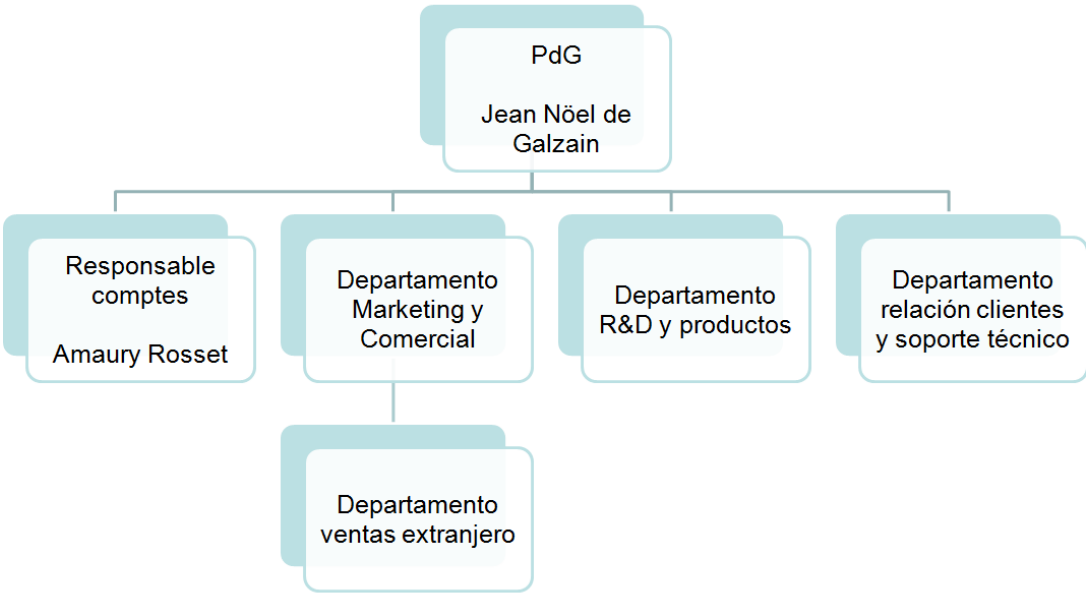

Figura 8. Organigrama de la empresa Wallix

PRODUCTOS DESARROLLADOS POR LA SOCIEDAD WALLIX

Principalmente los productos desarrollados por Wallix para proponer soluciones de seguridad para sus clientes son los siguientes:

- "Total Secure": está basado sobre una arquitectura GNU / Linux, concretamente sobre la arquitectura propuesta por la distribución Debian Lenny. Además de esto se proponen diversos paquetes "Open Source" debidamente desarrollados para dotar de seguridad el SI de la empresa gracias a sus funcionalidades de filtrado de direcciones IP, "firewall" y proxy HTTP.

- "Wallix Log Box": una solución de colecta, centralización y almacenamiento de los eventos producidos en el Sistema de Información del cliente. Dichos eventos pueden tener diferentes orígenes como los dispositivos de seguridad (Firewalls, Proxys) o los servidores de aplicaciones (Mensajería). Además de esta función de colecta de eventos, Wallix Log Box, es capaz de realizar búsquedas de texto a través de todos los eventos almacenados, lo que permite realizar un primer análisis en caso de fallo del Sistema de Información.

- "Walix Admin Bastion (WAB)": es el producto en el cual se encuadra éste proyecto fin de carrera. Wallix Admin Bastion es un proxy multi-protocolos con funcionalidades de autentificación, de control de acceso, de trazabilidad, almacenaje de datos y posibilidad de realizar auditorías. Este producto soporta los protocolos Secure Shell (SSH) y Remote Desktop Protocol (RDP) destinados a la gestión de equipos a distancia. La política de autentificación se hace mediante la consulta de un anuario LDAP (Lightweight Directory Acces Protocol) donde se almacenan los derechos que cada usuario tiene sobre cada equipo, además el control de acceso puede restringirse también mediante la imposición de reglas sobre la IP o sobre las franjas horarias sobre los equipos distantes que componen el sistema de información de la empresa.

  Cabe destacar que el uso del protocolo SSH está sobre todo destinado a controlar equipos desde un modo consola, es decir tecleando los comandos en el "Shell" de entrada. Por otro lado el uso del protocolo RDP está destinado al control gráfico de las sesiones remotas, principalmente sobre servidores con tecnología Microsoft al ser RDP un protocolo propietario de Microsoft. Finalmente, dicho producto está basado sobre una distribución Linux Debian Lenny, sobre la cual son instalados cada uno de los paquetes que componen el producto. Dicha distribución, se caracteriza por poseer un elevado nivel de seguridad y estabilidad lo que confiere un excelente soporte para el funcionamiento del producto.

WALLIX Y LA COMUNIDAD "Open Source"

Como ya hemos mencionado anteriormente, Wallix es una empresa, editor de soluciones para la seguridad de los Sistemas de Información de sus clientes, enmarcada en el ámbito de la comunidad "Open Source".

Las soluciones de Wallix están basadas, por lo tanto, en aplicaciones "Open Source". Sus ingenieros están completamente involucrados en esta comunidad, lo que hace que al mismo tiempo que realizan contribuciones a nivel personal a dicha comunidad, también realizan contribuciones en nombre de la empresa. Dicha filosofía de la empresa permite tener productos en constante evolución, además de ser más competitivos a nivel económico.

Una muestra de ésta filosofía es la aportación de Wallix como co-autora del libro "Les Modèles economiques du logicielle libre" y mantiene una estrecha colaboración con la asociación APRIL (Association pour Promouvoir et Défendre le logiciel libre) pionera del mundo "Open Source" en Francia.

Finalmente cabe destacar también que Wallix es socia en el proyecto "Admin Proxy" que tiene como objetivo el desarrollo de una solución, para los administradores de sistemas de información, capaz de uniformizar el acceso a los sistemas de información de la empresa además de a sus aplicaciones.

Como complemento de dicha colaboración con la comunidad Open Source, Wallix se asocia también con diferentes entidades universitarias para fomentar la investigación y el desarrollo en el ámbito de la seguridad de los Sistemas de Información.

ANEXO II. RESUMEN DE LA ESPECIFICACIÓN TÉCNICA DE MICROSOFT A PROPOSITO DEL PROTOCOLO RDP.

En este segundo anexo se va a presentar los documentos constitutivos de la especificación de Microsoft sobre el protocolo RDP que resultó muy útil en la comprensión del funcionamiento de dicho protocolo. Si bien es cierto que dicha especificación carece de detalles en ciertos momentos y en otros resulta muy complicada de comprender debido a la ingente cantidad de referencias cruzadas entre dichos documentos. A pesar de las dificultades, la lectura de dicha especificación resultó vital para la comprensión de todos los mensajes intercambiados en cada una de las tramas que circulaban entre cliente y servidor.

Remote Desktop Protocol es un protocolo que permite a un usuario conectarse sobre un ordenador a través de Microsoft Terminal Services . Existen clientes para casi todas las versiones de Windows y otros sistemas operativos, como Linux.

Dicho protocolo RDP se basa en el protocolo de T.share UIT (también conocido como T.128), la primera versión de RDP (versión 4.0) se introdujo con Terminal Services en Windows NT 4.0 Server. Por el contrario la versión más reciente (versión 6.1) está disponible sobre Windows Server 2008.

Las características más importantes son el soporte de color de 24 bits, cifrado con 128 bits basado sobre el algoritmo RC4, soporte para sonido, la función de copiar y pegar de archivos el equipo local entre el equipo remoto, el mapeo de puertos serie y paralelo y la gestión de una impresora conectada al equipo remoto.

En la especificación de Microsoft que está disponible en la web de Microsoft http://msdn.microsoft.com/en-us/library/cc240445 (PROT.10). aspx, podemos distinguir diversos documentos relacionados con la especificación de RDP (Remote Desktop Protocol).

El conjunto de todos los documentos que se presentan a continuación definen la especificación del protocolo RDP, pero cada documento se concentra sobre una parte más específica de dicho protocolo, con lo que para lograr una total comprensión del protocolo, vamos a necesitar comprender todos y cada uno de los documentos que vamos a nombrar a continuación:

- Remote Desktop Protocol: Basic Audio Input Redirection Virtual Channel Extensión
- Remote Desktop Protocol: Clipboard Virtual Channel Extension
- Remote Desktop Protocol: Basic Connectivity and Graphics Remoting Specification
- Remote Desktop Protocol: Basic Composited Remoting V2 Specification
- Remote Desktop Protocol: Basic Audio Output Virtual Channel Extension

- Remote Desktop Protocol: Basic Audio Input Redirection Virtual Channel Extension
- Remote Desktop Protocol: Clipboard Virtual Channel Extension
- Remote Desktop Protocol: Desktop Composition Virtual Channel Extension
- Remote Desktop Protocol: DirectX Virtual Channel Extension
- Remote Desktop Protocol: Dynamic Channel Virtual Channel Extension
- Remote Desktop Protocol: File System Virtual Channel Extension
- Remote Desktop Protocol: Graphics Device Interface (GDI) Acceleration Extension
- Remote Desktop Protocol: Licensing Extension
- Remote Desktop Protocol: Multiparty Virtual Channel Extension
- Remote Desktop Protocol: Print Virtual Channel Extension
- Remote Desktop Protocol: Plug and Play Virtual Channel Extension
- Remote Desktop Protocol: Session Selection Extension
- Remote Desktop Protocol: Remote Programs Virtual Channel Extension
- Remote Desktop Protocol: Smart Card Virtual Channel Extension
- Remote Desktop Protocol: Serial Port Virtual Channel Extension
- Remote Desktop Protocol: Video Redirection Virtual Channel Extension
- Remote Desktop Protocol: XML Paper Specification (XPS) Print Virtual Channel Extension

A continuación vamos a pasar a detallar las características principales de cada uno de los documentos que componen la especificación de Microsoft para tener una idea global del protocolo RDP y su funcionamiento.

REMOTE DESKTOP PROTOCOL: BASIC CONNECTIVITY AND GRAPHICS REMOTING SPECIFICATION.

Este documento va a definir el modo de interacción entre la máquina remota (denominada servidor RDP) y el usuario en relación al intercambio de información gráfica, y la redirección de la entrada estándar del usuario hacia sistema remoto.

Se va a comenzar por detallar la secuencia de conexión que se describe en este documento y, a continuación se introducirán diversos conceptos como la gestión de canales virtuales para el tratamiento de dispositivos a distancia, la compresión de datos, y la gestión de entradas estándar como el teclado y el ratón para terminar por la gestión de las salida gráfica proporcionada por el servidor distante.

El objetivo de la secuencia de conexión entre el cliente y el servidor en el marco del protocolo RDP es el intercambio de parámetros específicos de cada uno y el posterior

establecimiento de parámetros comunes entre los dos que no serán modificados a lo largo de la sesión. Para ser más precisos detalle, dicha secuencia tiene las siguientes fases:

- Inicio de la conexión, a través de mensajes basados en el protocolo X.224.
- Intercambio básico del marco de la conexión, con mensajes denominados “MCS Connect Initial / Response” basados en la especificación T.125.
- Conexión del canal, con el objetivo de agregar el canal a la comunicación entre cliente y servidor y además añadir a dicha sesión todos los canales que están presentes en el paquete denominado “Generic Conference Control Protocol” (GCC). Una vez que de todos los canales están confirmados, todos los mensajes que circulen entre el cliente y el servidor deben ser encapsulados en un paquete denominado X224 Data PDU.
- Inicio de la conexión segura RDP, si durante la fase de intercambio de paquetes de GCC (encapsulado en paquetes conocidos como “MCS Connect Initial / Response”) se decide recurrir a los mecanismos seguridad, el cliente y el servidor iniciarán un intercambio de claves asegurar la conexión y tras esto todos los entre cliente y servidor serán transmitidos de manera cifrada.
- Intercambio seguro del marco de la conexión, a partir de este momento, todos los datos serán transmitidos de manera cifrada a través de paquetes denominados “Client Info PDU”.
- « Licensing » es una parte de la conexión opcional que sirve para transferir una licencia del servidor al cliente. El objetivo de este intercambio es el envío de la licencia desde el cliente al servidor para la validación de las conexiones futuras.
- Intercambio de capacidades entre el servidor y el cliente para poner en común los tipos de capacidades que ambos compartirán durante la sesión.
- Finalización de la conexión, una vez recibido el paquete denominado “Font Map PDU” por el cliente, éste puede empezar a enviar datos relacionados con el teclado y el ratón además, después de recibir la el paquete denominado “Font List” el servidor puede comenzar a enviar las informaciones gráficas generadas por las aplicaciones en curso en el servidor hacia el cliente para su posterior tratamiento.

Para completar cabe remarcar que existe una parte relativa a la gestión de errores que se producen durante la conexión como por ejemplo la indisponibilidad de la red, la no gestión de ciertas propiedades gráficas, etc.

A continuación, se pasará a explicar el funcionamiento de los canales virtuales que nos van a permitir enviar ciertas informaciones suplementarias entre cliente y servidor y que van a ser esenciales para lograr los objetivos de nuestro PFC, ya que es a través de estos canales virtuales que se va a realizar la sincronización de los portapapeles del cliente y el servidor para desarrollar la funcionalidad de copiar / pegar, o por ejemplo para ser capaces de realizar la gestión de discos locales a distancia o gestión de impresoras así como la redirección del sonido.

El número máximo de canales virtuales es de 31, y estos son establecidos en la fase de cambio del marco básico de la conexión. La comunicación entre el cliente y el servidor está impedida hasta la finalización de la secuencia. Los datos deben ser enviados por tramas de 1600 bytes y cabe destacar que el tratamiento de los datos de cada canal virtual es totalmente independiente (es por ello que durante nuestro PFC debimos desarrollar ciertas funciones que nos permitieran distinguir el tráfico que circulaba por cada uno de estos canales para poder tratarlo de una manera distinta).

La compresión de datos no se negocia a nivel de la secuencia de conexión, lo cual quiere decir que tanto el cliente como el servidor tienen que ser capaces de comprimir los datos por ellos mismos de manera obligatoria por términos de reducción del uso del ancho de banda lo que nos da un incremento de las prestaciones con respecto a otros protocolos de gestión gráfica.

La compresión de datos se basa en el protocolo MPPC (Microsoft Point-to-Point Compression) y también se usan variaciones del algoritmo de compresión RLE (Run Length Encoding) para la compresión de los mapas de bits enviados desde el servidor al cliente. Cabe recordar que las entradas estándar del protocolo RDP están gestionadas en el cliente a través de los periféricos de teclado y ratón y por el contrario la salida estándar es la salida gráfica proporcionada por las aplicaciones lanzadas en el servidor y que están siendo ejecutadas a distancia desde el cliente.

Las entradas básicas de teclado y ratón se presentan en dos formatos diferentes, el denominado como “slow- path” (similar al formato T.128) y el conocido como “fast-path” (diseñado para reducir el uso del ancho de banda). La información enviada desde el cliente al servidor está incluida en el mensaje intercambiado entre ambos y que es conocido como “Input Event PDU”. Por otra parte, la información enviada desde el servidor al cliente consiste principalmente en informaciones gráficas como mapas de bits.

Por último, se presenta otra cuestión que se aborda en este trabajo que es el control de la salida de gráficos desde el servidor. Este control de informaciones gráficas generadas desde el servidor nos permitirá gestionar el refresco de los mapas de bits por zonas, es decir, no va a ser necesario enviar todo el mapa de bits completo si sólo ha sido modificada una zona del mismo.

REMOTE DESKTOP PROTOCOL: CLIPBOARD VIRTUAL CHANNEL EXTENSION SPECIFICATION.

El objetivo de éste documento es la descripción de las funcionalidades que van a permitir al usuario de comprender el sistema de gestión del portapapeles, así como su sincronización y los tipos de datos que pueden ser gestionados a través del protocolo RDP.

Así, para poder lograr esto, el protocolo especifica la manera en que ambos sistemas de gestión de portapapeles sean sincronizados para poder realizar una operación de copiar / pegar entre ambos sistemas, para ello, ambos portapapeles deben estar sincronizados para tener la misma información en ambos elementos de la conexión.

A continuación se va a realizar una explicación a alto nivel de ésta operación. Las acciones que cada aplicación debe realizar son las siguientes:

- La colocación de datos en el portapapeles
- Extracción de datos del portapapeles
- Lista de los formatos de datos disponibles cada portapapeles
- Registro de las actualizaciones del sistema de portapapeles

A continuación haremos una lista de los diferentes tipos de datos que puede tener un sistema de portapapeles. El primer tipo de datos es el tipo genérico, dicho tipo de datos permitirá el envío de estas informaciones de forma transparente de un punto de la conexión al otro.

El segundo tipo de datos es la paleta que va a contener un conjunto de descriptores de color (conocidos como tripletas RGB (Red-Green-Blue)) y serán codificados por el protocolo para ser transmitidos de una manera específica.

En tercer caso vamos a tener el tipo de datos conocido como metarchivo que se define como un conjunto de estructuras que pueden contener imágenes en un formato independiente, así que tendremos un conjunto de instrucciones de diseño que van a permitirnos obtener estos datos.

Para terminar con la presentación de los tipos de datos, debemos remarcar el tipo “File Stream”, que es un tipo de datos que encapsula un archivo y lo almacena en un formato específico de gran tamaño.

Como acabamos de explicar, es necesario mantener los sistemas de portapapeles sincronizados, porque la meta es tener la misma información en los dos sistemas.

Para lograrlo, es necesario enviar las actualizaciones cada vez que se produce una modificación en el contenido de uno de los dos sistemas y para ello utilizamos el mensaje denominado como “Clipboard Formats”.

Para continuar se van a presentar las dos secuencias principales que encontraremos asociadas al canal dedicado para la gestión del portapapeles:

- Secuencia de inicio, que tiene como objetivo el intercambio de las capacidades entre el cliente y el servidor, el marco general de funcionamiento del protocolo para este canal y los estados iniciales de los portapapeles de cliente y servidor.

- Secuencia de transferencia de datos: con dos tipos de mensajes bien diferenciados, el primero con el objetivo de copiar una secuencia de datos en el portapapeles (a través del paquete denominado “Format List PDU”), una vez enviado dicho paquete, una actualización y sincronización del sistema de portapapeles es llevada a cabo. El segundo tipo de mensajes es el encargado de realizar la recuperación de datos desde el portapapeles y transferirlos hacia el sistema que los requiera (dicha transferencia se realiza a través del paquete denominado “Format Request PDU”).

REMOTE DESKTOP PROTOCOL: FILE SYSTEM VIRTUAL CHANNEL EXTENSION

El propósito de esta especificación es canalizar el acceso del sistema de ficheros del cliente hacia el sistema de ficheros del servidor a través de un canal virtual denominado RDPDR. La forma de sincronización y montaje del sistema de ficheros del cliente hacia el servidor, se basa en el envío de tramas de consulta y respuesta de entrada / salida entre ambos sistemas de ficheros.

Para ser más precisos, vamos a mostrar la secuencia de inicio del protocolo que nos permita hacer este tipo de conexiones entre ambos sistemas de ficheros básicas para realizar el montaje del sistema de ficheros del cliente en el servidor. En el primer intercambio de paquetes, ambas partes, servidor y cliente van a poner en común la versión del protocolo presente en cada uno de ellos.

Tras esto, en una segunda fase, el cliente y el servidor van a intercambiarse las denominadas capacidades, es decir, los tipos de paquetes que van a poder intercambiarse durante toda la sesión sin causar inestabilidades en otros canales virtuales por una mala interpretación de los datos recibidos en estos paquetes.

El último intercambio de mensajes lo iniciará al cliente con la información de cada uno de los dispositivos disponibles para el uso de éste canal, dichos dispositivos deben ser confirmados por el servidor a través de un mensaje denominado “Server Device Announce / Response”. Si alguno de los dispositivos anunciados por el cliente, no es confirmado por el servidor, implicará que no se abrirá un canal asociado para dicho dispositivo y por lo tanto dicho dispositivo no aparecerá declarado en el servidor en el momento de la sesión.

Una vez los canales han sido confirmados, todas las operaciones realizadas sobre alguno de los dispositivos confirmados, se realizan a través del envío de mensajes de información / confirmación, así pues, si un canal RDPDR ha sido confirmado para un dispositivo de memoria USB, dicho dispositivo introducido en el cliente, aparecerá montado en el servidor, con lo que los archivos presentes en dicha memoria USB podrán interactuar con las aplicaciones del servidor y ser modificados, siendo dichas modificaciones realizadas al mismo tiempo en el cliente.

Por último, en relación con las dependencias de éste documento, se puede decir que los canales virtuales RDPDR se encapsulan en un canal estático que debe haber sido totalmente establecido en el inicio de la conexión. Además de esto, varios de los mensajes descritos en esta especificación, se utilizarán en otras especificaciones relacionadas con el uso de las impresoras, los puertos y las tarjetas inteligentes.

BIBLIOGRAFÍA

Página WEB del proyecto XRDP
http://sourceforge.net/projects/xrdp/

Página WEB del proyecto Rdesktop
http://sourceforge.net/projects/rdesktop/

Página WEB de la ITU (de la especificación T-128)
http://www.itu.int/rec/T-REC-T.128/e

Página WEB Wikipedia
http://en.wikipedia.org/wiki/Rdesktop
http://en.wikipedia.org/wiki/Xrdp

Página WEB de Microsoft (obtención de la especificación Microsoft)

http://msdn.microsoft.com/en-us/library/cc216513(v=PROT.10).aspx

Especificación del protocolo RFB y descripción del protocolo VNC

http://www.realvnc.com/docs/rfbproto.pdf

http://virtuallab.tu-freiberg.de/p2p/p2p/vnc/ug/protocol.html

Libro sobre la programación en C

El Lenguaje de Programación C, Kernighan y Ritchie (Pearson)

Documentación sobre el entorno UNIX / LINUX

http://www.linuxfromscratch.org/