AUTOMATIZACION DE PROCESOS DE MIGRACION DE NODOS HFC A REMOTE PHY EN CLARO COLOMBIA

KEVIN SEBASTIAN CARDENAS ARAQUE

INGENIERÍA DE TELECOMUNICACIONES

UNIVERSIDAD SANTO TOMÁS

BOGOTÁ

2022

AUTOMATIZACION DE PROCESOS DE MIGRACION DE NODOS HFC A REMOTE PHY EN CLARO COLOMBIA

KEVIN SEBASTIAN CARDENAS ARAQUE

MONOGRAFÍA PARA OPTAR POR EL TÍTULO DE INGENIERA DE TELECOMUNICACIONES

DIRIGIDO POR:

ING. VICTOR MANUEL CASTRO RAMIREZ

INGENIERÍA DE TELECOMUNICACIONES

UNIVERSIDAD SANTO TOMÁS

BOGOTÁ

2022

Nota de aceptación

_____

_____

_____

_____

_____

_____

Firma Docente

_____

Firma Jurado

_____

CIUDAD Y FECHA

TABLA DE CONTENIDO

LISTA DE FIGURAS

LISTA DE ABREVIATURAS

DOCSIS	Data Over Cable Service Interface Specification
HFC	Hybrid Fiber-Coaxial
R-PHY	Remote PHY
CMTS	Cable Modem Termination System
CCAP	Converged Cable Access Platform
CPE	Customer Premises Equipment
DAA	Distributed Access Architecture
OOB	Out of Band
RPD	Remote PHY Device
SSH	Secure Shell
ISP	Internet Service Provider
SFP	Small Form-Factor Pluggable
CM	Cable Modem
MTA	Multimedia Terminal Adapters
MPLS	Multiprotocol Label Switching
HA	High Availability
MAC	Media Access Control
NOC	Network Operations Center
VBA	Visual Basic for Aplications

1. INTRODUCCIÓN

Hoy en día existe una demanda creciente de mayor ancho de banda y menor latencia por parte de los clientes hacia los ISP, esto se debe principalmente a la gran cantidad de servicios que requieren un ancho de banda alto, como por ejemplo, servicios de streaming o de videojuegos en línea. Teniendo presente esta situación, Claro Colombia tiene varios planes para mejorar la calidad de servicio de sus clientes, entre los cuales están la migración de clientes hacia redes FTTH, o en otros casos se realiza una migración de clientes de la red HFC hacia la arquitectura Remote PHY, en donde la capa física (modulación y demodulación de señales) se realiza directamente sobre los nodos ubicados en la calle en vez de realizarse en el CMTS tradicional.

Para apoyar las actividades relacionadas al proyecto Remote PHY, Claro Colombia hace uso de los servicios de ingeniería de ZTE Colombia, principalmente en lo que respecta a servicios de configuración y de administración del proyecto, por lo cual durante el proceso de practica se trabajó sobre dicho proyecto (en el área de implementación de Claro), principalmente en temas relacionados con la validación y asignación de direccionamiento IP necesario para el despliegue de la solución, generación y seguimiento de permisos de Firewall necesarios para asegurar la conectividad, diseños necesarios para la migración de los nodos ubicados en la calle, una vez que los equipos de la cabecera están en estado productivo, entre otras actividades.

La transición hacia Remote PHY es un aspecto crítico para la compañía, ya que el deterioro de la calidad de servicio que se presta en muchas partes del país está provocando que los clientes cambien de proveedor y en algunos casos no es viable realizar la transición de la red HFC actual hacia una red FTTH, por lo que la arquitectura Remote PHY aparece como la opción más conveniente en este tipo de situaciones, especialmente teniendo en cuenta las ventajas que posee, entre ellas la alta disponibilidad (al estar diseñado para poseer redundancia física y lógica), las mayores velocidades que puede ofrecer y el menor consumo energético, de espacio y de refrigeración que ofrece la arquitectura Remote PHY vs los equipos de la red HFC.

2. PLANTEAMIENTO DEL PROBLEMA

El proyecto Remote PHY que se está desarrollando en Claro Colombia no está exento de inconvenientes que se presentan durante todas las fases del mismo, algunos de ellos son imprevistos y otros son por el contrario, temas que se podrían evitar y son generados por reprocesos, adicionalmente debido a la naturaleza del proyecto, en la que se busca actualizar una tecnología previamente existente, pueden existir inconvenientes relacionados con la migración de la red HFC tradicional hacia la red Remote PHY, esto en parte a los procesos que se tienen preestablecidos para la red HFC en la compañía.

Estos imprevistos se generan principalmente debido a particularidades en la infraestructura de parte de la red de Claro y también a demoras en la integración de los equipos nuevos de Remote PHY en los sistemas de gestión que se usan actualmente en la compañía, no obstante, en este documento únicamente se abordara aquellas actividades que se asignaron durante el proceso de prácticas profesionales, específicamente las relacionadas al proceso de segmentación/migración de nodos HFC a RPHY (validar estado y ocupación de los puertos en los equipos, generar los diseños para las nuevas segmentaciones, entre otros).

Estos procesos mencionados anteriormente son, a menudo, un poco repetitivos, por lo que su ejecución tiende a ser algo tedioso y difícil de manejar, principalmente cuando se requiere la elaboración de los diseños es requerida de manera y con poco tiempo para ser desarrollados (como ha sucedido en varias oportunidades) debido a los requerimientos de la compañía, en particular, las metas de migración de nodos que requiere realizar cada mes el área de planta externa de la compañía y el proceso de notificación a los clientes que se debe hacer con una semana calendario de anticipación, lo que se traduce en demoras o errores en la ejecución de estas actividades.

Teniendo en cuenta el contexto anterior, vale la pena poner sobre la mesa la pregunta problema, la cual es la siguiente:

¿Cómo se puede reducir el tiempo necesario para desarrollar las actividades necesarias de una segmentación/migración en la red RPHY de Claro Colombia?

3. OBJETIVOS

3.1. OBJETIVO GENERAL

Mejorar las actividades de segmentación/migración de clientes de la red HFC hacia Remote PHY, automatizando el proceso a través de scripting en Python y Visual Basic con el fin de reducir el tiempo necesario para desarrollar dichas actividades, además de evitar posibles errores y reduciendo los reprocesos que se puedan generar por dichos errores.

3.2. OBJETIVOS ESPECÍFICOS

- Describir de manera general la arquitectura de red HFC, y compararla respecto a la arquitectura Remote PHY
- Mostrar parte del proceso que se lleva a cabo para la implementación de esta arquitectura en Claro Colombia.
- Reducir el tiempo de ejecución de ciertas actividades a través del desarrollo de scripts que permiten automatizar las actividades de segmentación/migración de clientes.

4. JUSTIFICACIÓN

Una de las prioridades más importantes de Claro en estos momentos, es el desarrollo del proyecto Remote PHY, ya que esto es parte de la estrategia que la empresa va a aplicar para mitigar la pérdida de clientes que ha venido sufriendo es el aumento de la velocidad a los clientes por el mismo precio que pagan actualmente, y para poder conseguir este objetivo es vital el uso de Remote PHY, principalmente debido a las mejoras que esta arquitectura posee en términos de ancho de banda hacia los clientes y gasto energético y de espacio en los equipos ubicados en la cabecera.

También es necesario tener presente que una de las fases más importantes, y a su vez más complicadas del proceso de implementación de Remote PHY es la parte de segmentaciones/migraciones desde la red HFC hacia la nueva red de RPHY, ya que hay muchas variables en juego (direccionamiento IP, aprovisionamiento de servicios, disponibilidad de puertos con SFP, programación de ventanas para la migración, entre otros) por lo cual es mucho más probable que se presenten problemas, algunos de los cuales no pueden ser mitigados totalmente, pero si se les puede dar un mejor manejo.

Con este fin, se busca mejorar en términos de tiempo algunos procesos en los que se prestó apoyo durante el proceso de prácticas (generación archivos de diseño y validación de disponibilidad de puertos para agregar nuevos RPDs a los switches de distribución) y así lograr que estas actividades no tengan reprocesos pro errores en los archivos de diseño y script o por temas de marcación errónea o ausencia de marcación en los equipos de RPHY, ya que esto es un tema que sucede con cierta frecuencia y es el culpable de que algunas actividades se migración de clientes se detengan hasta corregir el problema.

Es importante corregir este inconveniente, ya que anteriormente se venía trabajando de manera manual estos procesos, pero debido al crecimiento que tendrá este proyecto, a medida que se implementen más equipos Core en producción, va a ser cada vez más difícil realizar estas validaciones sin llegar a cometer errores o requiriendo validar varias veces la información, adicionalmente, el área de planta externa generalmente solicita una gran cantidad de diseños en un lapso corto de tiempo, para cumplir con las metas que proponen durante cada mes, y por ende esto combinado al hecho de que estas migraciones se deben programar con cierta anterioridad para informar a los clientes, hace que se cuente con poco tiempo para elaborar dichos archivos.

5. MARCO TEORICO

5.1. DOCSIS (Data Over Cable Service Interface Specifications):

DOCSIS es un estándar de telecomunicaciones desarrollado por el consorcio CableLabs, organización la cual es encargada de establecer estándares para el uso de los operadores de telecomunicaciones que usen redes de cable para desplegar servicios. Este estándar ha experimentado varias revisiones, en las cuales se han mejorado algunos de sus aspectos, con el fin de mejorar la calidad del servicio que ofrecen, actualmente se encuentra desplegada la versión 3.1 del estándar DOCSIS, pero ya se está desarrollando el sucesor del mismo, la versión 4.0. A continuación se muestran las velocidades máximas teóricas de los estándares DOCSIS más recientes:

- DOCSIS 3.0: US: 200 Mbps - DS: 1 Gbps - Modulación: 256-QAM (42.88 Mbps por cada canal de 6 MHz)
- DOCSIS 3.1: US: 1.5 Gbps – DS: 10 Gbps – Modulación: 4096-QAM (1.89 Gbps por cada canal de 192 MHz)
- DOCSIS 4.0: US: 10 Gbps – DS: 10 Gbps – Modulación: 4096-QAM (1.89 Gbps por cada canal de 192 MHz)

Los operadores de cable de todo el mundo han adoptado DOCSIS gracias a su capacidad para permitir la transferencia de datos de gran ancho de banda a través de un sistema de cable coaxial existente, inicialmente diseñado para la transmisión de televisión por cable (video). (Software Para Monitoreo, Gestión Y Pruebas, 2022)

En la década de 1950 inicia el despliegue de redes de cable coaxial para servicios CATV (TV por cable) unidireccionales. A mediados de los noventa, se crea CableLabs para la creación de estándares de redes de cable avanzadas, que permitieran también los servicios de telefonía e internet. La estandarización debía permitir la interoperación y reutilización de equipos y procesos, bajando costos, tiempos y dando velocidad al avance de los servicios sobre cable y video digitales. (La Historia de DOCSIS - Intraway Blog | Think Incredible, 2020)

- DOCSIS 1.0: Desarrollado en 1997, es la primera versión del estándar DOCSIS, en el cual se consolida que debe existir una interoperabilidad entre fabricantes, además de implantar el esquema de Cablemodems actuando como clientes y CMTS actuando como servidores o Core. Es una tecnología asimétrica en la cual se define por primera vez el mecanismo que se debe usar para transmitir datos digitales a través de cables coaxiales, teniendo en cuenta que anteriormente únicamente se podrá transmitir CATV analógica mediante estos cables, esto fue un gran avance en cuanto a las telecomunicaciones de esa época.

- DOCSIS 1.1: Es básicamente el mismo estándar DOCSIS 1.0 pero incorpora mejoras en cuanto a calidad de servicio (QoS), abriendo la posibilidad de agregar servicios que requieren de ciertos parámetros de calidad, como la televisión digital, el streaming y las llamadas a través de internet (VoIP), una mejora respecto a DOCSIS 1.0 en donde la metodología para trabajar con este tráfico es Best Effort. Fue desarrollado en el 2001.
- DOCSIS 2.0: Esta versión se desarrolló en el 2003, y consiste en algunas mejoras tales como el aumento de la capacidad de transmisión de datos, teniendo en cuenta el aumento de la demanda de servicios de Internet, especialmente en términos de velocidad de subida, además de soportar IPv6 a nivel de los cablemódems. Otra mejora sustancial es el uso de QAM64 para modulación de señales, una mejora respecto a versiones anteriores que usaban QAM16, entre otras mejoras.
- DOCSIS 3.0: Esta versión del estándar DOCSIS, liberada en el 2006, posee varias ventajas, entre ellas, el uso de IPv6 (dual stack), para así poder disponer de una gran cantidad de IPs respecto a IPv4, ya que esta integración ahora estaba a nivel de señalización DOCSIS. Otra ventaja que poseía, es el hecho de aplicar modulación QAM 256, además de permitir Channel Bonding (una técnica que permite que los cablemódems se conecten a varios canales de Upstream y Downstream como si fueran uno solo para mejorar la velocidad de los suscriptores).
- DOCSIS 3.1: Es la versión actualmente desplegada del estándar DOCSIS, y posee muchas mejoras respecto a versiones anteriores, principalmente con respecto a la eficiencia con la que se trabaja el espectro. Este estándar fue liberado en el 2013, con el fin de mejorar la tecnología para poder competir con otras alternativas que fueron ganando terreno. Algunas de las ventajas que destacan de esta versión son el uso de 1024QAM para modulación además del uso de OFDM (Orthogonal Frequency Division Multiplexing) para aprovechar mejor el espectro disponible en los cables coaxiales, y finalmente se mejoró el proceso de comprobación de errores a través de un nuevo algoritmo. La idea de estas mejoras es que los proveedores de servicio de telecomunicaciones puedan obtener los beneficios de estas mejoras sin necesidad de cambiar la infraestructura actual.

- DOCSIS 4.0: Es la versión recientemente lanzada de DOCSIS (lanzada en el 2017), pero que aún no se ha desplegado por parte de la gran mayoría de cableoperadores, aunque las ventajas que esta versión brinda, hacen que este tipo de redes sean capaces de competir contra otras alternativas, como la fibra

óptica o las comunicaciones móviles. La principal ventaja que tiene esta versión de DOCSIS es el uso de ancho de banda simétrico, es decir, la velocidad de subida y de bajada es de 10 Gbps, una velocidad que es bastante competitiva, teniendo en cuenta las necesidades actuales de velocidad, estabilidad y baja latencia para aplicaciones como streaming de videojuegos o uso en plataformas de video bajo demanda de alta definición, entre otras.

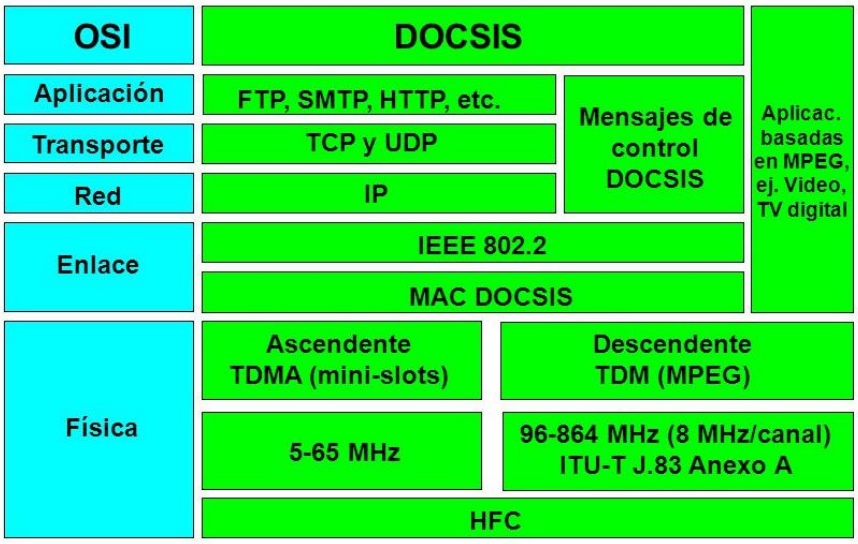


Figura 1. Comparación DOSCIS vs. OSI.

Fuente: https://slideplayer.es/slide/4087070/

5.2. HFC (Hybrid Fiber-Coaxial):

Una red HFC es aquella red de telecomunicaciones que usa cable coaxial, además de fibra óptica análoga o digital para llevar servicios a los clientes, a través de estos dos medios de transmisión lleva las señales de voz, de internet y de televisión desde el headend (cabecera) hasta los clientes finales. Algunas de las características que ofrece esta red son la posibilidad de usar el mismo medio de transmisión para obtener diferentes servicios (a diferencia de los servicios que se ofrecían antes de que se implementará esta tecnología), además de poder ofrecer tanto servicios análogos y digitales, entre otros. La arquitectura de esta red está compuesta de varios elementos pasivos y activos, tales como fibras multimodo, cables coaxiales, amplificadores de RF, combinadores, splitters, nodos ópticos, y multitaps, entre otros. (Redes HFC – Entre Redes Y Servidores, 2009)

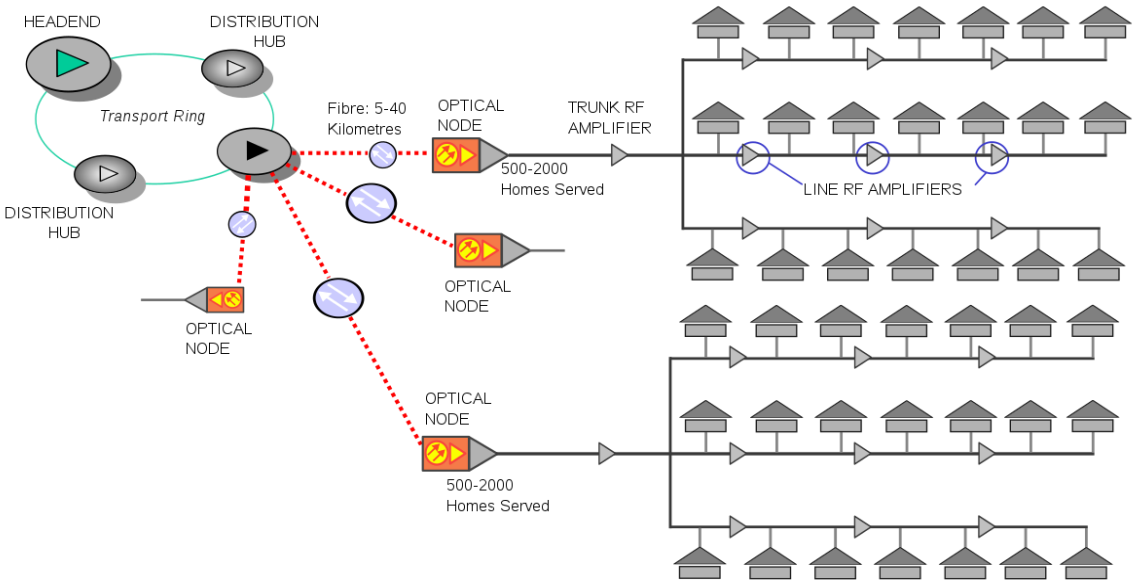


Figura 2. Diagrama de una red HFC

Fuente: https://commons.wikimedia.org/wiki/File:HFC_Network_Diagram.svg

Una red HFC se compone de los siguientes componentes principales:

- CABECERA:

  Es la parte central o el core de este tipo de arquitectura de red, en donde se generan las señales que se distribuirán luego a través de las otras partes de la red, en este punto se encuentran los equipos más poderosos, que conectan la nube privada del proveedor con el resto de la red y otras redes, para poder brindar acceso a los consumidores de dichos servicios. Algunos de los equipos que se

pueden encontrar en esta parte son moduladores, codificadores, entre otros. Otra de las funciones que se cumplen en la cabecera, es la de monitorear el estado de la red, para asegurar que todo funcione de la manera correcta.

- RED TRONCAL:

  Esta es la parte de la red que se conecta desde la cabecera o headend de la red, y termina en la interconexión entre los nodos ópticos, la mayor parte de este tramo de red se compone de enlaces ópticos análogos o digitales, los cuales se conectan hacia los nodos a través de enlaces directos o anillos ópticos, principalmente comentando anillos primarios a secundarios a través de enlaces directos, y luego en dichos nodos convierten la señal óptica en señal eléctrica que posteriormente viajará a través del cable coaxial hasta llegar a los clientes.

- RED DE DISTRIBUCIÓN:

  Esta fase consiste en mayor parte por los equipos eléctricos activos y pasivos que permiten el paso de señales eléctricas, luego de la conversión del dominio óptico al dominio eléctrico, en donde se transporta y amplifica la señal a través de diversos dispositivos eléctricos, como splitters, amplificadores eléctricos, fuentes de poder, acopladores y multitaps.

- RED DE ACOMETIDA DE ABONADO:

  Este es el tramo final de la red, en donde se interconectan los equipos de los usuarios finales con el resto de la red, un buen ejemplo de esta fase de la red es la instalación de cable coaxial interna de los edificios o el tramo de red que se despliega desde el domicilio de un cliente hasta el punto de agregación que se puede encontrar en un poste o una instalación subterránea en la calle.

Los elementos principales que componen la red HFC son:

- CMTS: (Cable Modem Termination System): Es el dispositivo encargado de proporcionar conectividad entre los abonados de la red HFC, y la red de transporte del operador de servicios de telecomunicaciones. Este equipo se ubica generalmente en la cabecera de la red, y permite convertir las señales de protocolos como Ethernet (que se conectan hacia la red de transporte de alta capacidad) con los abonados a través de interfaces RF (con varios equipos intermedios, cabe aclarar).
- Cablemódem: Es el dispositivo que se ubica en la residencia del abonado, y permite obtener servicios como Internet o TV, a través de los enlaces de RF que van hacia los nodos.

- Nodo: Dispositivo ubicado en planta externa, en donde llegan las señales ópticas que después se convertirán en señales eléctricas de RF, para desplegar servicios a los suscriptores de servicios de telecomunicaciones.
- Combinadores/Splitters: Dispositivos encargados combinar o dividir las señales de RF, para posteriormente enviar las señales a través de enlaces ópticos análogos hacia los clientes

5.3. DAA (Distributed Access Architecture):

La Arquitectura de Acceso Distribuida (DAA, por sus siglas en inglés) es una evolución de la arquitectura que se maneja en las redes HFC, en la cual, algunos de los componentes que se encuentran en la Cabecera de la red se migran a partes de la red más cercanas al usuario.

Esta arquitectura permite reducir la cantidad de componentes que se encuentran en la cabecera de la red, con las ventajas que esto supone, en cuanto a espacio necesario para instalar los equipos y los cableados necesarios, además de reducir el consumo de energía que se requiere para hacer funcionar dichos equipos, y, en consecuencia, los clientes notarán una mejora del servicio ya que las funciones que no se ejecutan en la cabecera, se implementan en partes de la red que se encuentran más cerca de los suscriptores del operador, por lo cual pueden obtener ventajas como mejoras en velocidad, confiabilidad y latencia.

Se emplea un enlace de 10Gbps desde la cabecera hasta el nodo óptico. Algunas de las arquitecturas de acceso distribuido que se han desarrollado incluyen Remote PHY y Remote MAC-PHY, en la primera se hace mueve la parte de modulación y demodulación hacia los nodos, permitiendo mejores velocidades y mayores capacidades de modulación, y para el caso de Remote MAC-PHY acerca la capa de procesamiento de señales DOCSIS hacia los clientes, ya que esta parte también se realiza directamente en el nodo, a diferencia de un CMTS tradicional, en donde ambas funcionalidades se ejecutan en la cabecera (CMTS). Una ventaja de estas arquitecturas, es que se pueden desplegar mediante fases, dependiendo de los requerimientos de los operadores de servicios de telecomunicaciones.

Al reubicar la capa física más cerca del usuario final, se mejora la calidad y el rendimiento, así como la velocidad, el nivel de ruido y la modulación, todo ello como resultado directo del cambio. (*Arquitecturas de Acceso Distribuido*, 2021)

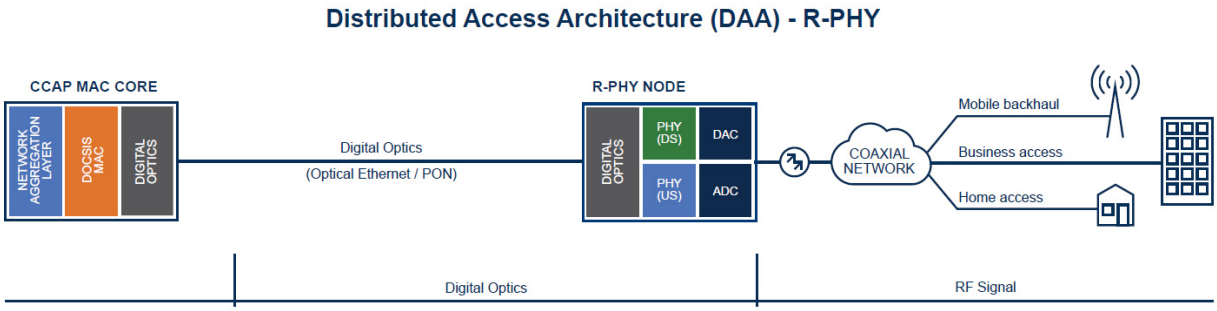


Figura 3. Arquitectura DAA

Fuente: [VECTOR TECHNOLOGIES | Remote PHY Interop testing reaches new level with CCAP Core Simulator]

Otra de las ventajas del uso de arquitecturas de acceso distribuidas (DAA) es el uso de sistema digitales en vez de los tradicionales sistemas análogos usados en HFC (en donde se usan componentes que pueden sufrir atenuaciones que empeoren la calidad de servicio de los clientes como splitters, combinadores, etc.), representando beneficios como el uso de modulaciones digitales para mayor eficiencia o la capacidad de migrar parte de la arquitectura hacia IP para mejorar la visibilidad de la red y la convergencia con otras plataformas como FTTH o 5G. Adicionalmente, las fibras ópticas usadas en las arquitecturas DAA permiten el uso de más longitudes de onda (multimodo) que las fibras análogas para permitir el aumento de la capacidad de transmisión de tráfico, y junto con otros dispositivos permite que se pueda habilitar servicios de 10 Gbps de tráfico.

5.4. REMOTE PHY

Es una arquitectura de acceso distribuida (DAA) en la cual se migra parte de las funciones que se ejecutan en la cabecera (CCAP Core o vCMTS), principalmente las referentes a la modulación y demodulación de señales (capa Física o PHY) más cerca de los clientes finales, específicamente, en los nodos ópticos ubicados en la calle, a los cuales se les agregan estas funciones a través de dispositivos llamados RPDs (Remote PHY Device.

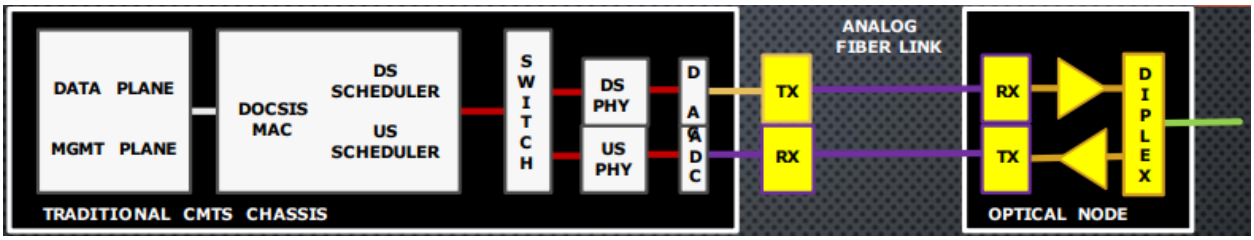


Figura 4. Modelo Tradicional CMTS

Fuente [EncRegTel2019-EvoluciondeHFC.pdf (encuentrosregionales.com)]

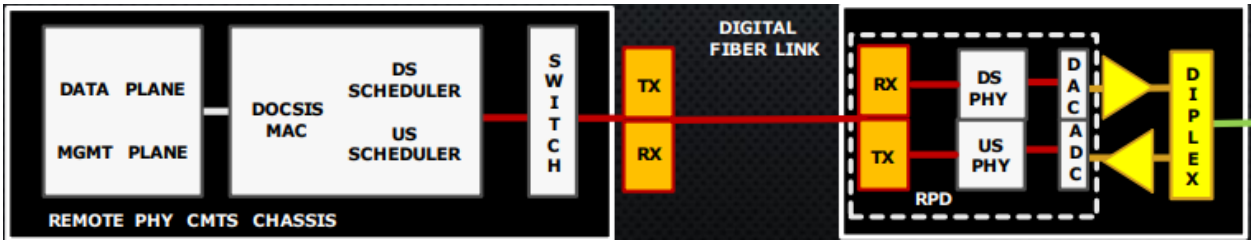


Figura 5. Modelo Remote PHY

Fuente [EncRegTel2019-EvoluciondeHFC.pdf (encuentrosregionales.com)]

Esto supone algunas ventajas, entre las que se encuentra el poder modular las señales con más cantidad de símbolos y también el hecho de que se pueda ajustar los niveles de RF de manera remota, evitando que el personal de planta externa tenga que desplazarse hasta al nodo para realizar dicha tarea. Esta arquitectura está diseñada para ser desplegada sobre la arquitectura de un CMTS tradicional, lo cual facilita la transición para los operadores, que además representa una ventaja, al ya no tener que adicional splitters y combinadores de RF, sino que, en su lugar, se desplegará una red IP Ethernet para envío de tráfico de Upstream y Downstream hacia los vCMTS.

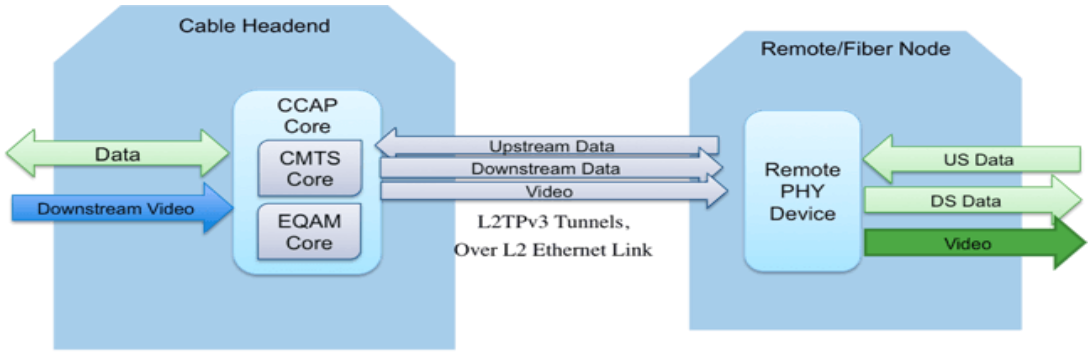


Figura 6. Arquitectura de Alto Nivel Remote PHY

Fuente: [Remote PHY is a Reality (cablelabs.com)]

A menudo, cuando se despliega este tipo de arquitectura, esta se hace de manera paulatina mediante el reemplazo de componentes de la red HFC existente (como, por ejemplo, los nodos ópticos), abaratando costes y mejorando la calidad de servicio de los suscriptores. Adicionalmente, esta arquitectura, en lo que respecta al CCAP Core permite ser virtualizado, a través de microservicios, con las ventajas que esto supone, como mayor rendimiento, menor consumo energético y redundancia lógica y física (a través del uso de orquestadores como Kubernetes y enlaces físicos redundantes), que permiten que en caso de que algún microservicio falle, este sea reemplazado de manera casi imperceptible, un ejemplo podría ser que en caso de que el microservicio encargado de controlar todo lo relacionado con DHCP falle, éste será reemplazado por otro microservicio ubicado en otro servidor físico, o en caso de que algún enlace del CCAP Core falle, este cuente con un enlace de respaldo, que permita que no exista tiempo de inactividad que afecte a los clientes.

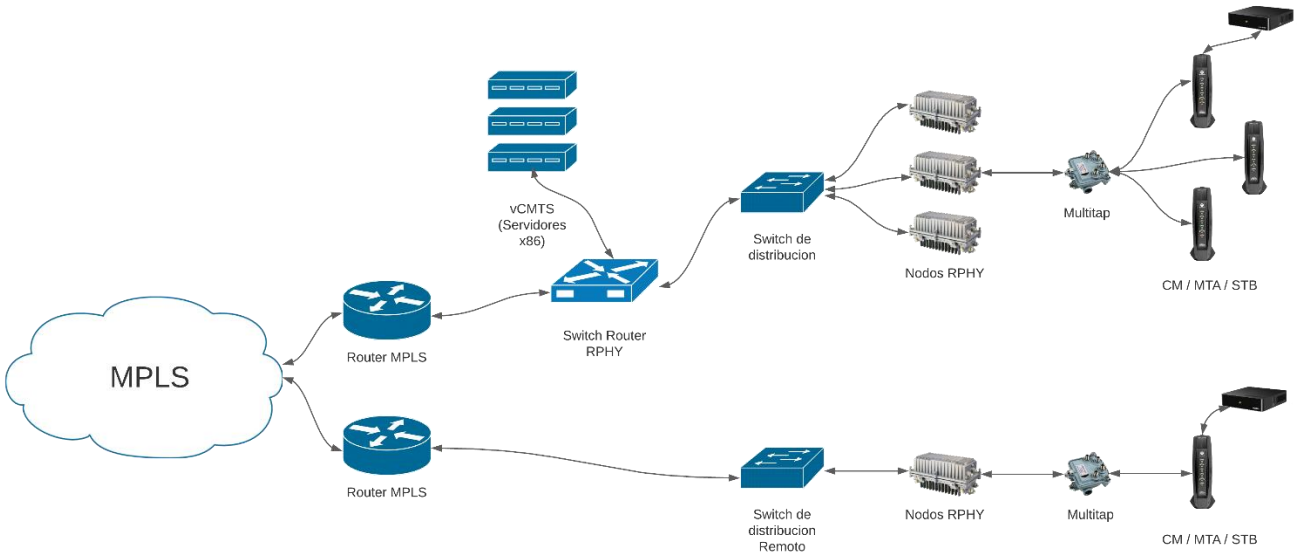


Figura 7. Arquitectura Remote PHY

Fuente: Elaboración propia

La arquitectura Remote PHY que se está implementando en la compañía consiste principalmente en Switch Routers que se conectan a la red MPLS de Claro a través de interfaces ópticas de 100Gbps, y que a su vez se conectan en los equipos Core (vCMTS), los cuales son servidores x86 que ejecutan microservicios mediante Kubernetes y también se conectan a los switches de distribución en caso de que estos se conecten localmente, todos estos enlaces son a través de fibra óptica y usan el protocolo IP, algo que antes no sucedía, ya que anteriormente en la arquitectura HFC se usaban enlaces ópticos, pero estos eran analógicos.

Los switches de distribución interconectan varios nodos ópticos, por lo que cada uno de los puertos requiere tener un SFP óptico de 10Gbps, en este dispositivo se realiza la marcación de cada uno de los nodos que estén ocupados, y en caso contrario, se marca como PUERTO LIBRE. El nodo RPHY contiene los dispositivos necesarios para realizar la conversión de señales ópticas a señales DOCSIS que se distribuirán posteriormente por los cables coaxiales ubicados en la calle hasta llegar a los TAP y finalmente hacia los clientes finales a través de los Cablemodems y los Set-Top box.

La arquitectura en un sitio remoto es similar a la de un sitio core, pero aquí únicamente se conecta el switch de distribución a los nodos y de manera similar al caso anterior se distribuye la señal DOCSIS a los clientes. Es necesario aclarar que los switches remotos dependen de un vCMTS, el cual alcanza a través de la red MPLS.

6. DISEÑO DE LA SOLUCIÓN

El desarrollo correcto del proyecto Remote PHY es un tema crítico a nivel de compañía, a tal punto que a nivel gerencial se le hace un seguimiento detallado, y se busca que este proyecto no tenga algún tipo de contratiempo (de energía, de configuración, de procesos internos, etc.), por lo que es necesario verificar cuales son los aspectos que se podrían mejorar, para evitar contratiempos en los procesos. Adicionalmente a esto, en este proyecto se tiene contempladas varias fases en las cuales se añadirán nuevos equipos core y de sitios secundarios, para poder migrar clientes de la red HFC tradicional, hacia Remote PHY.

Actualmente se tiene contempladas dos fases, durante la primera fase se implementarán sitios Core (en los cuales se instalan los servidores Core (vCMTS), además del switch router y el switch de distribución), durante la segunda fase se implementará únicamente sitios secundarios, los cuales estarán asociados a los sitios Core de la primera fase.

En el largo plazo, se tiene contemplado que esta red crecerá aún más con el tiempo, debido a las ventajas que posee implementar esta arquitectura con respecto a la arquitectura HFC tradicional, con respecto a el espacio que ocupan los CMTS Legacy, y la cantidad de equipos adicionales que se requieren (como por ejemplo, combinadores de RF Splitters, Amplificadores eléctricos RF), algo que permitirá el desmonte de parte de la infraestructura Legacy de la red HFC, lo cual significaba el fin progresivo de la red HFC, al menos a largo plazo, ya que existen lugares donde el único acceso a servicios de telecomunicaciones es a través de la red HFC, y no es viable en términos de Capex y Opex el hecho de reemplazar la tecnología HFC hacia Remote PHY.

Otro dato destacado de la arquitectura Remote PHY que se está desplegando en la red de Claro, es el hecho de que se está utilizando IPv6, en parte debido al agotamiento de direccionamiento IPv4 que se ha venido presentado a nivel mundial, por lo que hoy en día para tratar de mitigar esto, se realizar varias técnicas, entre ellas, el uso de IPv6 y realizar traducciones de IPs a través de NAT.

Es necesario aclarar que obviara la parte relacionada con la compra de los equipos y la fase de planeación del despliegue de la red, ya que no hacen parte del proceso de implementación, sino de etapas más tempranas que ya fueron ejecutadas (al menos en las dos primeras fases).

6.1. PROCESO DE IMPLEMENTACIÓN

El proceso de implementación de Remote PHY en la red de Claro Colombia se refiere a todos los procesos que se deben llevar a cabo para lograr que los equipos de esta arquitectura entren en producción y los clientes finales puedan obtener servicios a través de dichos equipos.

A continuación, se describirán brevemente el proceso de implementación que se lleva a cabo para poner en producción los equipos del proyecto Remote PHY en cada sitio definido:

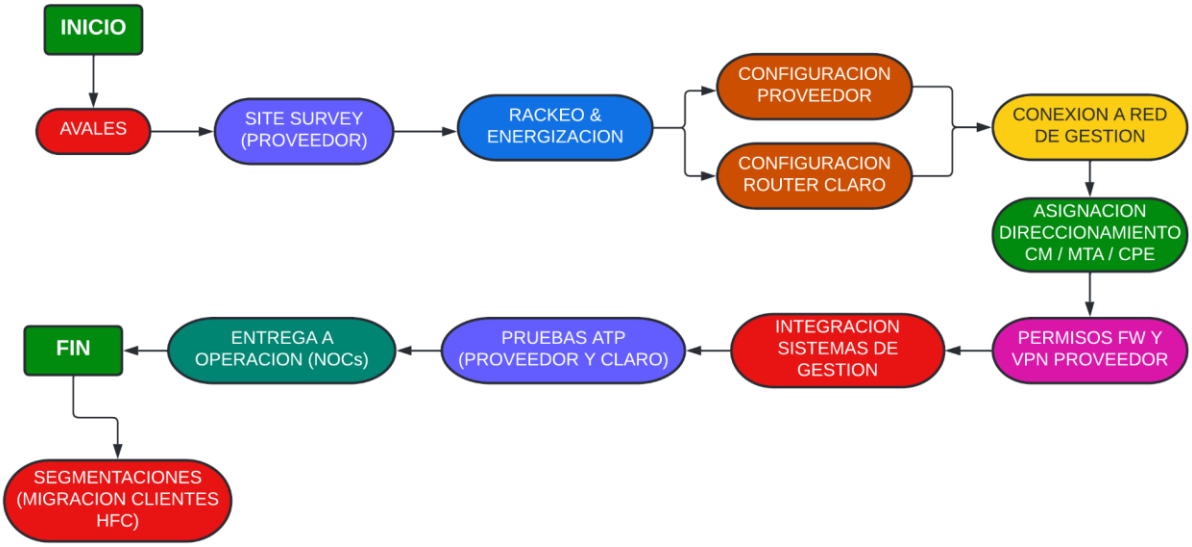


Figura 8. Proceso de implementación Remote PHY

Fuente: Elaboración Propia

6.1.1. SOLICITUD DE AVALES

Durante esta fase, se hacen varias solicitudes a las regionales donde se encuentran los sitios donde se van a instalar los equipos, para que ellos validen si en dicho sitio se cuenta con el espacio, la refrigeración y la potencia eléctrica suficiente para instalar los equipos nuevos en el sitio. Se debe contar con los 3 avales para poder continuar a la siguiente fase.

6.1.2. SITE SURVEY (TSS)

Esta es parte de la fase inicial del proceso de implementación, en la cual, el proveedor (o en su defecto un aliado esté) visita el sitio donde se van a instalar los equipos, con el fin de verificar el lugar donde se instalarán los mismos, además de las distancias que se requieren para interconectar los equipos y algunas otras particularidades que se puedan presentar en el sitio. En este punto, el proveedor debe escoger la solución más óptima para instalar sus equipos, y en caso de que exista algún problema, este es el punto donde se pueden informar, para poder darles solución y continuar con las siguientes fases.

6.1.3. ENERGIZACIÓN Y RACKEO DE EQUIPOS

Una vez que ya se tiene la información de la ubicación del sitio, y del lugar concreto donde se pueden instalar los equipos, el proveedor o el aliado visitan nuevamente el sitio para realizar todas las interconexiones necesarias para realizar la configuración inicial de los equipos. Entre las tareas que se realizan durante esta fase, está la

6.1.4. CONFIGURACIÓN PROVEEDOR

Ya teniendo los equipos listos para realizar la configuración, el proveedor comienza a realizar las configuraciones iniciales que requiere para poner la plataforma en estado operativo o de producción, como, por ejemplo, actualizar el firmware de los equipos, asegurar que encienden correctamente, e incluso aplicar una configuración inicial, aunque no se limita simplemente a la configuración inicial, simplemente desde esta fase se empiezan a realizar las primeras configuraciones en los equipos.

6.1.5. INTERCONEXIÓN TV - INTERNET - VOZ (CONFIGURACION CLARO)

Esta fase puede ir de la mano de la configuración del proveedor, ya que parte de esta configuración consiste en establecer los enlaces necesarios entre el router Edge de Claro y la solución de cada proveedor, para este caso, ya se deben contar con los puertos seleccionados del lado del router de Claro (se maneja en tráfico de datos separado del tráfico de tv y voz, por lo que se requieren varios puertos para los diferentes servicios).

Además, del lado del proveedor y del lado de Claro se requiere que la configuración a nivel de VLANs y VRFs esté correcta desde ambos lados, para así lograr la conectividad necesaria y se puedan obtener los servicios que se le prestarán a los clientes. Esta interconexión se hace hacia los equipos Edge de Claro que permitirán obtener servicios como VoIP, Internet, IPTV, AAA, TACACS, DHCP, PTP, DNS, entre otros, los cuales se obtienen a través de la nube privada de Claro, que funciona a través del protocolo MPLS.

6.1.6. INTERCONEXIÓN RED GESTIÓN OOB

Además de los entre los equipos del proveedor y el router de Claro desde donde se obtendrán los servicios de Internet, IPTV y el servicio de VoIP, se requiere que todos los equipos de la red estén conectados a una red interna, llamada Red de Gestión, y través de la cual se podrá acceder al CLI de los equipos (además de la IP de loopback que debe tener cada equipo), esto es necesario en caso de que existan problemas con la IP de loopback, poder acceder a estos equipos y también para temas de envió de paquetes SNMP a los sistemas de gestión, para el tema de las alarmas. Es necesario tener en cuenta que en algunos casos el switch de gestión no se encuentra en el mismo sitio donde se instalan los equipos Remote PHY, por lo que así mismo se usa la infraestructura de la red móvil de Claro, para lograr llegar a dicho switch.

6.1.7. ASIGNACION DIRECCIONAMIENTO DHCP (CM, MTA, CPE)

Otro tema que se debe tener en cuenta, es el direccionamiento que se va a asignar a los Cablemodems (CM) y MTA de los clientes, el cual también debe estar correctamente configurado y asignado, para poder prestar los servicios a los clientes. Durante esta fase, se asignan varios pool o segmentos correspondientes a clientes residenciales y corporativos, tanto en IPv4 como en IPv6, y también se asignan los segmentos IPv6 que tomarán los switch router de Remote PHY, y que se le entregarán a los RPDs.

Una vez se tenga el consolidado de todos los rangos, se deben agregar dichos rangos a los servidores DHCP de la compañía, y asegurar que exista alcanzabilidad desde los equipos a estos servidores. Adicional a esto, se debe tener en cuenta que durante esta fase también se asignan las direcciones de los PTP, primario y secundario, los cuales también deben tener disponibles y que son vitales en la arquitectura Remote PHY.

6.1.8. PERMISOS FIREWALL & VPN PROVEEDORES

Este proceso consiste en asegurar que se pueda acceder a los equipos, tanto a través de la red de gestión (IP de gestión), como por la red MPLS (IP loopback), con el fin de que se puedan acceder y no exista interrupción de tráfico en algún punto, y en caso de que exista algún problema, validar la causa y solucionarlo. Además de esto, también se solicitan permisos en las VPNs del proveedor para que puedan acceder de forma remota a estos equipos, y puedan seguir realizando sus configuraciones.

6.1.9. PRUEBAS DE SERVICIO PROVEEDOR Y CLARO (ATP)

Ya teniendo en cuenta que se ha avanzado en gran parte, en la implementación del proyecto, se puede iniciar con las pruebas internas por parte del proveedor, en las cuales se garantiza que, del lado de ellos, todo esté funcionando de manera adecuada. Algunas de las pruebas que ellos realizan son, por ejemplo, pruebas de redundancia (ya que los equipos cuentan con esta capacidad tanto a nivel físico como lógico, se debe garantizar que la redundancia esté funcionando de manera adecuada), pruebas de conectividad (en donde se debe asegurar que los CM alcancen los servidores DHCP y obtengan conectividad hacia internet, desde todos los pools que se asignaron anteriormente), entre otras.

Además de estas pruebas se deben realizar otras pruebas entre el proveedor y el NOC, en las cuales ya deben estar listos algunos de los sistemas de gestión que usarán para temas de troubleshooting y mantenimiento de los nuevos equipos, y que el NOC usará para validar las alarmas y las métricas que reciban, mientras el proveedor realiza unas pruebas similares a las que hace el proveedor internamente.

Finalmente se hacen unas pruebas rápidas con la regional donde se instaló dichos equipos, con el fin de asegurar que al momento de migrar clientes actuales (o en su defecto, añadir clientes nuevos) a la red Remote PHY, no exista ningún problema.

6.1.10. INTEGRACIÓN A SISTEMAS DE GESTIÓN

Como se mencionó anteriormente, los sistemas de gestión son una herramienta importante para la compañía en general, ya que a través de ellos se puede obtener información del estado de los equipos (temperatura, carga de procesamiento, conectividad existente, etc.), además de integrarlos a la red de Claro, para que en caso de que suceda algún imprevisto, las áreas correspondientes puedan solucionarlo de

manera correcta. Algunas de estas herramientas permiten a las áreas diagnosticar los problemas que se presenten a nivel de cliente, u en otros casos, permiten que se puedan aplicar cambios a estos equipos, ya que dichos equipos se encontrarán inventariados dentro de los sistemas de la empresa.

6.1.11. ENTREGA DE INFRAESTRUCTURA AL NOC

Esta es la fase final del proceso de implementación, en donde se hace la entrega oficial de la infraestructura, totalmente funcional y lista para que los clientes puedan acceder a los servicios que ofrece Claro, a través de dichos equipos en las áreas del país en donde estén presentes. En este punto, ya se debieron completar todos los anteriores procesos, por lo cual el NOC hace una última revisión y luego se corrigen algunas posibles fallas o issues que se presenten, para finalmente poner los equipos en producción.

6.1.12. SEGMENTACION/MIGRACIÓN REMOTE PHY (MIGRACIÓN CLIENTES HFC)

Esta fase se desencadena inmediatamente después de la entrega al NOC, cuando los equipos ya están en producción y ya se pueden migrar clientes de la red HFC hacia estos. Esta fase no es realizada por el área de implementación, sino que es realizada por personal de planta externa y del NOC directamente, los cuales hacen diferentes tareas como programar la segmentación (migración) de nodos HFC a RPHY, y la parte de planta externa se encarga de realizar el cambio a nivel físico, conectando los equipos correspondientes, los cuales son principalmente, el enlace que va desde el switch de distribución hasta el nodo RPHY, y asegurar que los enlaces funcionen, algo de lo que se encargan todos los involucrados.

6.2. DESARROLLO DE ACTIVIDADES EN EL PROCESO DE IMPLEMENTACION

Es importante aclarar que la etapa del proceso de implementación en la que se trabajara es la de “SEGMENTACION/MIGRACIÓN DE CLIENTES HFC”, en la cual se generan los diseños y los scripts que se van a usar para las segmentaciones, además de la programación de dichas segmentaciones, para lo cual se requiere previamente dichos diseños, además de que para poder elaborar dichos diseños se debe obtener

información de etapas anteriores, como la de “ASIGNACION DIRECCIONAMIENTO DHCP” o la etapa de “INTERCONEXIÓN RED GESTIÓN OOB”, en las cuales se toma todo el direccionamiento de los equipos de Remote PHY, por lo que es necesario.

Toma aproximadamente media hora el desarrollo cada pareja de diseño/script para cada nodo que sea solicitado por parte del área de planta externa, ya que para cada uno de estos, se debe obtener la información del CMTS donde está actualmente conectado el nodo (puerto actual, clientes, nombre, direccionamientos IP, etc.), y la información del equipo Core de Remote PHY en donde se conectara dicho nodo después (puerto libre con SFP, marcación nueva, direccionamientos IP, etc.).

Mas específicamente, el área de planta externa envía un listado por correo electrónico con la programación de los nodos que se segmentaran o migraran el mes actual, este listado se envía el primer día de cada mes y se requiere el desarrollo de la actividad lo más pronto posible. Estos diseños se solicitan por cada regional del país (costa, centro, oriente, noroccidente, etc.) y son aproximadamente 15 diseños por cada regional, pero este valor puede variar dependiendo de las necesidades y la ocupación de los equipos de Remote PHY, adicional a esto, se debe tener en cuenta que en cada regional puede existir más de un sitio en donde se implemente Remote PHY, como por ejemplo la regional Centro, en la cual existen equipos de Remote PHY en 3 sitios de Bogotá y uno ubicado en Cundinamarca.

NODO	REGIONAL	CMTS	MES
ABC	Costa	CMTS1	Febrero
DEF	Oriente	CMTS2	Febrero
GHI	Centro	CMTS3	Febrero
JKL	Noroccident	CMTS2	Febrero
MNO	Noroccident	CMTS3	Febrero
PQR	Noroccident	CMTS1	Febrero
STU	Noroccident	CMTS1	Febrero
ASD	Centro	CMTS1	Febrero
WED1	Costa	CMTS2	Febrero
ASW	Centro	CMTS3	Febrero

Figura 9. Ejemplo de solicitud de generación de diseño/script

Fuente. Elaboración propia

Esto es un problema, especialmente teniendo en cuenta que esta actividad anteriormente se realizaba manualmente por que existían pocos equipos Core, por lo que era mucho más fácil realizarlos. Hoy en día se ha multiplicado casi x4 la cantidad de equipos en producción, por lo que realizar el proceso de manera manual es algo mucho más complejo y la cantidad de diseños que solicitan para cada equipo Core de Remote

PHY crece cada vez más (incluso se tiene contemplada una 2nda fase para el proyecto en la cual se instalaran aún más equipos).

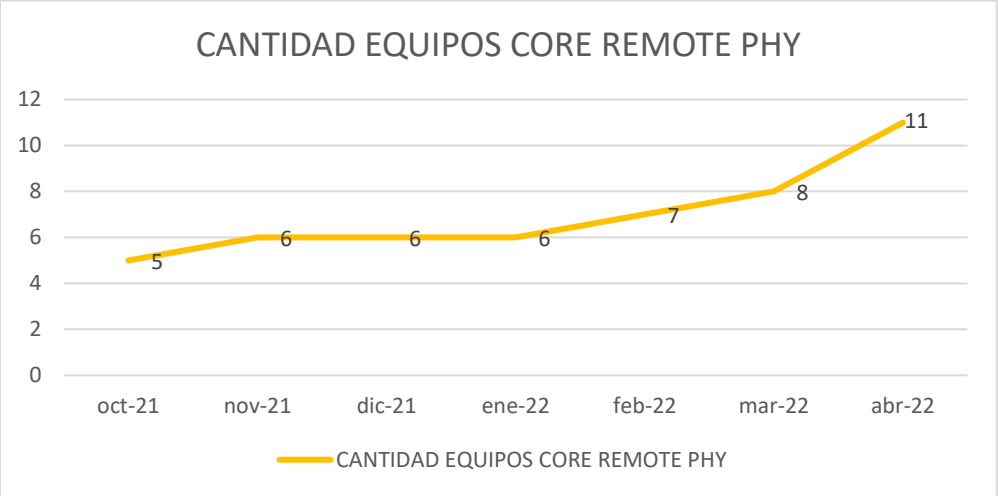


Figura 10. Cantidad equipos Remote PHY

Fuente. Elaboración propia

En primer lugar, se verifica sobre cual equipo de HFC está el nodo a segmentar, a veces este dato se envía junto con la información del nombre del nodo o en otros casos, esta debe ser consultada manualmente en los CMTS que se encuentren en producción. Esta tarea se realizaba manualmente, pero posteriormente se generó un aplicativo en PowerBI (el cual no puede ser mostrado debido a que es información interna de la compañía) en el cual se obtiene parte de la información un poco más fácilmente, específicamente este aplicativo muestra el ID del nodo, el nombre de dicho nodo, la cantidad de clientes que están asociados y los puertos de upstream que está utilizando, esta tarea se demora aproximadamente 8 minutos, si el diseño se encuentra en el desarrollo de PowerBI y 15 minutos si el diseño se busca manualmente en los CMTS.

Una vez validado el CMTS en donde se encuentra el nodo actualmente, se agrega el direccionamiento IP del equipo de Remote PHY (rangos IP de Cablemodems, de MTA, de internet NAT, internet DHCP, e Internet con IP Fija), esta tarea se demora aproximadamente 5 minutos si el diseño se hace sobre un equipo existente y 15 minutos en aquellos sitios que son totalmente nuevos, en los cuales es necesario generar nuevo direccionamiento. Luego de esto, se valida el equipo destino al que se debe migrar el nodo (en la mayoría de los casos, el nombre del equipo origen y destino es similar, por lo que esta información se infiere, al menos en la fase 1 del proyecto) y se revisa en dicho equipo los puertos disponibles.

Finalmente, se copian los archivos de diseño y script desde los archivos de plantilla y se renombran de acuerdo al ID del nodo, y se suben a la carpeta correspondiente de SharePoint

Diariamente el equipo de planta externa solicita el estado y la disponibilidad de los puertos en un equipo especifico, para lo cual es necesario entrar al equipo y ejecutar un comando, y a continuación se toma una captura de pantalla para enviarla mediante correo electrónico o mediante capturas de pantalla en el chat interno de la compañía, algo que es poco práctico. Al existir una gran cantidad de equipos, esta tarea consume una buena cantidad de tiempo, debido a que se solicita información de todos los equipos que están en producción, y para hacer esto es necesario entrar manualmente mediante PuTTY a los equipos Core y Switch de distribución para así tomar captura de pantalla de los puertos disponibles en cada equipo, esta tarea toma alrededor de 10 minutos por cada equipo.

Algunas veces, personal del área de implementación solicita información específica de los equipos, por lo que se debe acceder a revisar la configuración de los equipos, como por ejemplo, la configuración que está corriendo actualmente un equipo especifico, o en otros casos, validar que se nombre correctamente cada uno de los puertos de los nodos de Remote PHY (marcación que se hace al momento de migrar clientes al nodo) para agregar dicha información a los sistemas de gestión de la compañía. En ambos casos, es necesario en primer lugar buscar la IP del equipo necesario e ingresar a ella mediante PuTTY (previamente ingresando al servidor Linux desde donde se accede a los demás equipos)

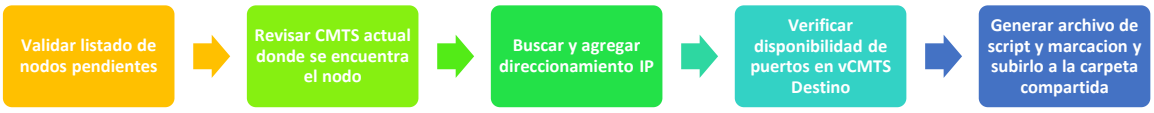


Figura 11. Proceso de generación script/marcación

Fuente. Elaboración propia

A continuación se muestra el tiempo aproximado que se demora cada una de las tareas:

ACTIVIDAD	TIEMPO ESTIMADO
Validar Listado de Nodos Pendientes	5 min
Revisar CMTS actual	8 min
Agregar direccionamiento IP	5 a 10 min
Validar disponibilidad de puertos en CMTS Destino	5 min
Generar script y marcacion y subir archivos a SP	15 min
Validar configuraciones adicionales (marcaciones rf-port)	5 min
Revision de estado puertos para planta externa	5 min
Ingreso a equipos mediante PuTTY	8 min
Busqueda de diseño previamente generado	5 min

Figura 12. Tiempo estimado actividades

Fuente. Elaboración propia

6.2.1. ACCESO A EQUIPOS DE REMOTE PHY

Para acceder a los equipos de Remote PHY, y en general de otros servicios, se debe acceder primero a un servidor Linux intermediario a través de clientes como PuTTY, por lo cual cada vez que se quiera acceder a algún equipo, primero se debe acceder al servidor, y una vez ingresado el nombre de usuario y la contraseña, se podrá acceder a los equipos que sean requeridos, ya sean de FTTH, Remote PHY, IP RAN, o CMTS, a través del comando ssh <user>@<hostname>, y luego ingresar la contraseña correspondiente, la cual en algunos casos no es la misma contraseña que se usó al principio.

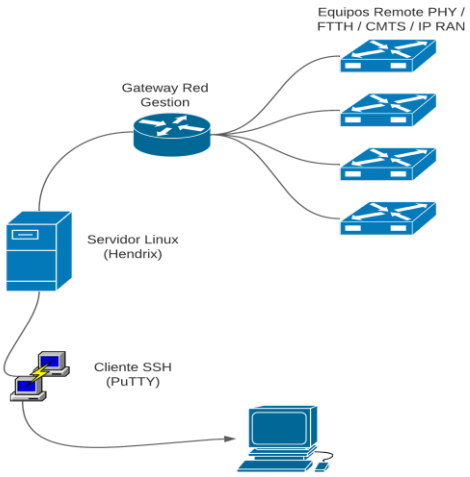


Figura 13. Topología acceso a equipos

Fuente: Elaboración Propia

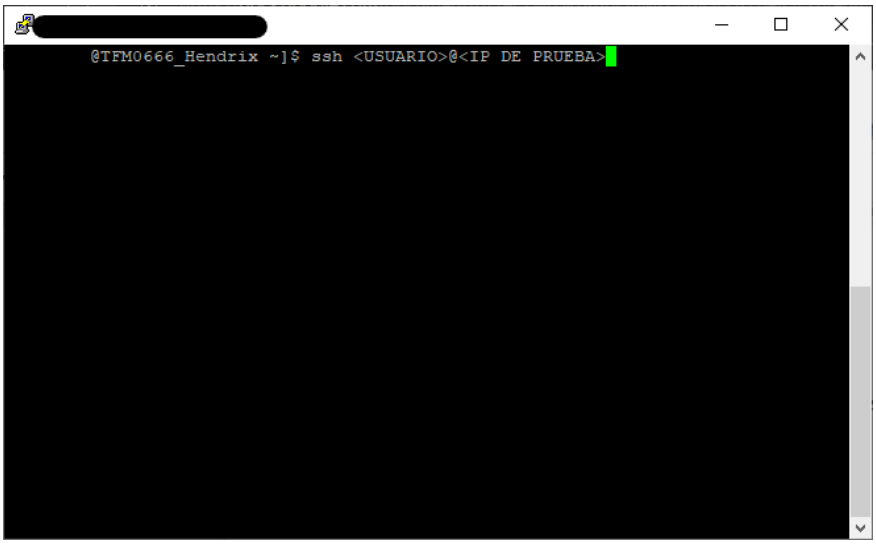


Figura 14. Ventana activa programa PuTTY

Fuente: Elaboración Propia

Una vez dentro del servidor Linux, es necesario obtener la IP del equipo a conectar, y hacer login desde dicho servidor, para lo cual es necesario acceder a un archivo Excel en línea, en donde se encuentran dichas IPs, una tarea que es un poco engorrosa, teniendo en cuenta que la cantidad de equipos a los que se debe acceder es cada vez mayor, debido al crecimiento del proyecto. Por esta razón, es necesario mejorar el proceso con el que se accede a los equipos, y para lograr esto se tienen dos propuestas, la primera es grabar las sesiones de cada equipo de Remote PHY en SecureCRT y, la

segunda opción es generar un archivo txt que contenga la información de los equipos, para que sean más fácilmente accesibles directamente desde el bash del servidor de Linux.

6.3. AUTOMATIZACIONES A DESARROLLAR

Las opciones que se van a desarrollar son, en primer lugar, cambiar la manera en la que se acceden a los equipos por medio de SSH, para poder realizar consultas, además de automatizar parte del proceso de diseño y marcado de las RPDs nuevas que se vayan a segmentar, y obtener archivos de Excel (que posteriormente se compartirán al personal de planta externa en terreno o cualquiera que requiera conocer el estado de ocupación de puertos de un Switch en particular).

Todos estos scripts automatizan tareas que se ejecutan actualmente en el proceso de prácticas, y que ocupan una buena parte del tiempo, por lo que puede hacer mediciones de tiempo y sacar conclusiones, pero no obstante, no es posible mostrar la información confidencial de la compañía, por lo que para el desarrollo de este documento se realizara con datos de muestra.

6.3.1. SELECCIÓN DE NUEVO CLIENTE SSH

Una parte importante de cualquier proceso de implementación y operación de redes de telecomunicaciones es acceder a los equipos que administra, esto es un aspecto crítico, ya que muchas personas deben acceder a la gran cantidad de equipos que tiene la empresa hoy en día. Con el fin de poder acceder a dichos equipos es necesario usar un cliente SSH, ya que a través de este se logra ingresar a los equipos y ver su estado actual y configurarlos en caso de que se requiera.

Tradicionalmente para estos casos se usa el cliente de SSH PuTTY, el cual es una herramienta gratuita que permite acceder a estos equipos de manera fácil y rápida, es decir, es un programa básico para los profesionales de áreas relacionadas con la tecnología. Este programa es bastante popular en el mundo TI, aunque no cuenta con algunas funcionalidades como el guardado automático de contraseñas y sesiones y tampoco permite la automatización a través de scripts. Es por esto que, como parte del diseño de esta monografía, se propone cambiar el uso del cliente SSH PuTTY, por una herramienta que tenga más funcionalidades que permitan un uso más productivo de este tipo de herramienta, que es tan frecuentemente usado por el proyecto, y la compañía en general.

A la hora de seleccionar el cliente SSH, existen varias opciones para reemplazar el uso de PuTTY, como por ejemplo los clientes SecureCRT y Mobaxterm, una vez indagado entre los posibles clientes SSH a utilizar, se considera que la opción más apta para desarrollar la solución es SecureCRT, ya que tiene muchas ventajas respecto a otras alternativas, ya que tiene posibilidad de generar scripts que permitan la automatización de tareas, además de permitir el guardado de sesiones SSH y el envío de comandos personalizados. Este cliente SSH permite crear scripts en el lenguaje de programación VBA (Visual Basic for Applications) o Python 2, aunque existe la posibilidad de integrarlo a una instalación de Python 3.8 que se encuentre instalada en Windows o Linux.

6.3.2. SCRIPTS PARA OBTENER INFORMACIÓN DE RPDS EN EXCEL

Teniendo en cuenta que SecureCRT es una buena opción para el desarrollo de esta propuesta, se selecciona dicho programa, y más específicamente se utilizará el lenguaje de programación VBA, debido a la facilidad de uso y a que la documentación de este programa usa principalmente dicho lenguaje de programación. Además, es compatible con los macros de Microsoft Excel, facilitando la automatización de dichos procesos.

Este desarrollo se llevará a cabo para la versión de SecureCRT de Windows, ya que, aunque existe una versión de este programa para Linux y macOS, el uso de Windows es prácticamente generalizado en la compañía, y más específicamente, en el proyecto, además, para el desarrollo de los scripts se usará el programa Visual Studio Code, debido a su facilidad de uso y extensiones que permiten la compatibilidad con varios lenguajes de programación.

Es importante aclarar que es necesario generar un script para cada tarea que se desee automatizar, en este caso específicamente, se utilizará para obtener información de las RPDs conectadas, tanto en el CCAP Core, como en el Switch de distribución donde se conectaran las mismas. Así mismo, se tiene en cuenta que estos dispositivos deben estar correctamente marcados en ambos dispositivos, y particularmente, se realiza un archivo de Excel cada cierto tiempo para ir comprobando que la relación que hay entre las marcaciones sea la correcta.

6.3.3. SCRIPT PARA AUTOMATIZAR GENERACIÓN DE SCRIPT / DISEÑO DE NODOS

Otra de las tareas que se desea mejorar, y que además es algo repetitiva, es la generación de los diseños y scripts que se usarán al momento de desplegar un nodo

hacia Remote PHY, debido a que se requieren por parte del personal de planta externa y por parte del NOC que interviene en el despliegue de dicho nodo.

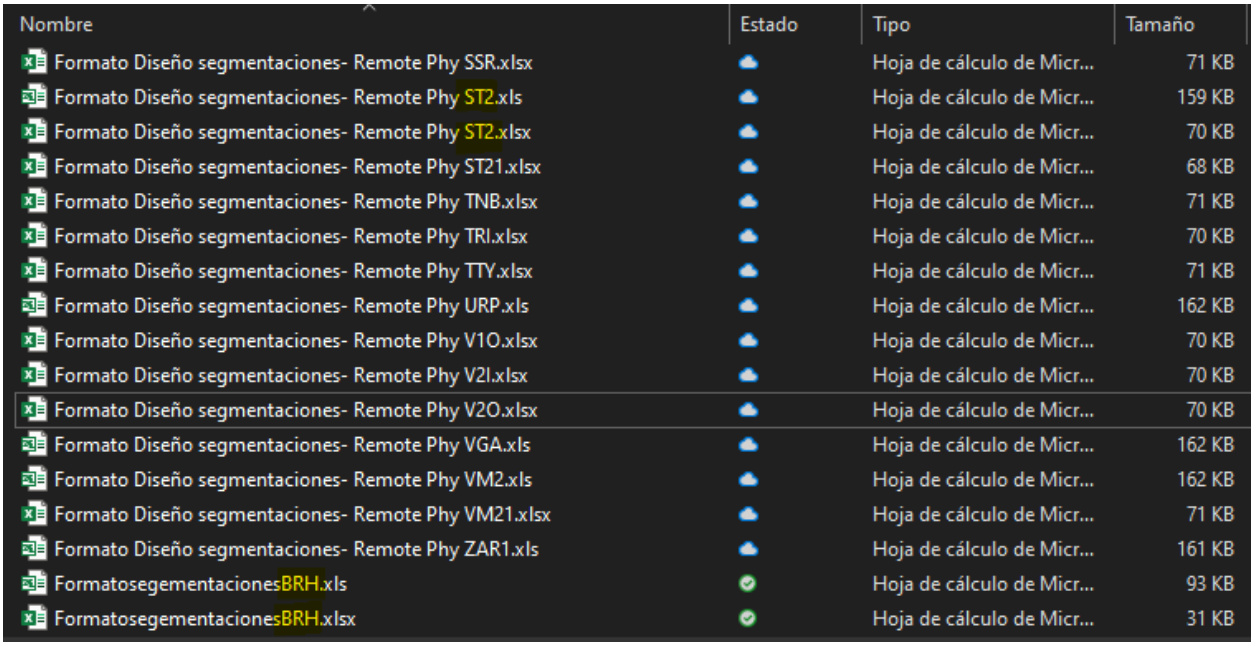
El desarrollo consistirá básicamente en la elaboración de un script que permitirá generar los archivos de script y diseño de cada nodo, para automatizar esta tarea y permitir que se realice de manera más rápida y sencilla, únicamente ingresando los parámetros que cambian, como el ID y el nombre del nodo, así como los puertos correspondientes en los equipos en donde se configura dicho nodo, que son el CCAP Core (vCMTS) y el Switch de Distribución.

7. DESARROLLO

7.1. GENERACIÓN AUTOMÁTICA DE SCRIPT/DISEÑO DE NODOS CON PYTHON

Para el correcto desarrollo de esta propuesta, antes que nada fue necesario realizar una depuración de los diseños de segmentaciones Remote PHY que estaban previamente realizados, ya que en su momento se asignaron algunos puertos que no estaban contemplados o que cambiaron, y por lo tanto se podría presentar incoherencias en los diseños, como por ejemplo que existan varios diseños con puertos duplicados o que tengan nombres incorrectos (algo que en realidad sí sucedió al momento de realizar la implementación), para evitar esto se busca mantener documentados estos diseños para que así estén lo más ordenados posible, facilitando su acceso para todas las áreas involucradas en la operación y mantenimiento del proyecto, principalmente los NOC encargados de dar soporte a estos dispositivos (NOC Cable, NOC Televisión, etc.).

Estos diseños se encuentran en una carpeta compartida online dentro de la compañía, para que todas las áreas de la compañía puedan acceder a ellas, por lo tanto, cualquier cambio que se haga, debe hacerse directamente sobre ese directorio. A continuación, se muestra cómo se encuentra organizada, para este caso, se trata de una subcarpeta donde se encuentran los diseños que se han hecho para Bogotá:



Nombre	Estado	Tipo	Tamaño
Formato Diseño segmentaciones- Remote Phy SSR.xlsx		Hoja de cálculo de Micr...	71 KB
Formato Diseño segmentaciones- Remote Phy ST2.xls		Hoja de cálculo de Micr...	159 KB
Formato Diseño segmentaciones- Remote Phy ST2.xlsx		Hoja de cálculo de Micr...	70 KB
Formato Diseño segmentaciones- Remote Phy ST21.xlsx		Hoja de cálculo de Micr...	68 KB
Formato Diseño segmentaciones- Remote Phy TNB.xlsx		Hoja de cálculo de Micr...	71 KB
Formato Diseño segmentaciones- Remote Phy TRI.xlsx		Hoja de cálculo de Micr...	70 KB
Formato Diseño segmentaciones- Remote Phy TTY.xlsx		Hoja de cálculo de Micr...	71 KB
Formato Diseño segmentaciones- Remote Phy URP.xls		Hoja de cálculo de Micr...	162 KB
Formato Diseño segmentaciones- Remote Phy V1O.xlsx		Hoja de cálculo de Micr...	70 KB
Formato Diseño segmentaciones- Remote Phy V2I.xlsx		Hoja de cálculo de Micr...	70 KB
Formato Diseño segmentaciones- Remote Phy V2O.xlsx		Hoja de cálculo de Micr...	70 KB
Formato Diseño segmentaciones- Remote Phy VGA.xls		Hoja de cálculo de Micr...	162 KB
Formato Diseño segmentaciones- Remote Phy VM2.xls		Hoja de cálculo de Micr...	162 KB
Formato Diseño segmentaciones- Remote Phy VM21.xlsx		Hoja de cálculo de Micr...	71 KB
Formato Diseño segmentaciones- Remote Phy ZAR1.xls		Hoja de cálculo de Micr...	161 KB
FormatosegementacionesBRH.xls		Hoja de cálculo de Micr...	93 KB
FormatosegementacionesBRH.xlsx		Hoja de cálculo de Micr...	31 KB

Figura 15. Carpeta con archivos de diseño segmentaciones

Fuente: Elaboración Propia

Como se puede ver en la imagen, existen algunos diseños que están repetidos, lo cual puede llegar a generar algunos reprocesos entre las áreas, mientras se valida cuál de los dos diseños es el correcto, para luego poder realizar la segmentación correspondiente, ya que para poder aplicar la segmentación del diseño es necesario solicitar un cambio e informar a los clientes que actualmente se encuentren conectados a dicho nodo que se realizará el cambio de tecnología con suficiente tiempo para que lo tengan presente (esto se hace con el fin de que los clientes tengan presente que se desconectara del servicio durante el cambio de tecnología, además de poder realizar rollback en caso de que algo salga mal).

El siguiente paso que se hizo fue ordenar la información de cada nodo en carpetas separadas con el nombre del nodo (ya que cada nodo tiene nombre único), para agrupar y seleccionar los diseños y scripts de configuración que el NOC debe correr cuando se haga el cambio de tecnología, ya que algunos de estos diseños ya se ejecutaron, pero aun así no se corrigieron en su momento.

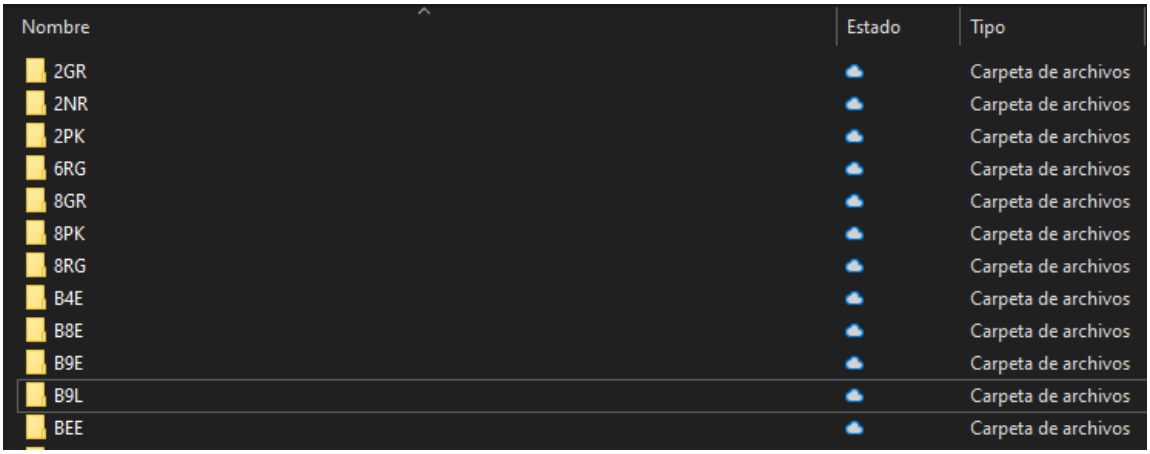


Figura 16. Carpetas ordenadas con ID de nodo

Fuente: Elaboración Propia

En este caso, estos nodos ya se encuentran depurados y ordenados, una tarea que ayuda bastante a conocer cuáles puertos quedan disponibles en las futuras segmentaciones que se asignen a estos Servidores Core, además se debe tener en cuenta que en un futuro cercano se van a habilitar switches secundarios, los cuales se conectan a Servidor Core respectivo a través de la red MPLS de Claro, lo cual debe estar documentando adecuadamente, o de lo contrario puede haber confusiones en el futuro respecto al sitio de la instalación, ya que un solo Servidor Core puede controlar varios Switches, ya sea que estos se encuentren en el mismo sitio, o en un sitio remoto.

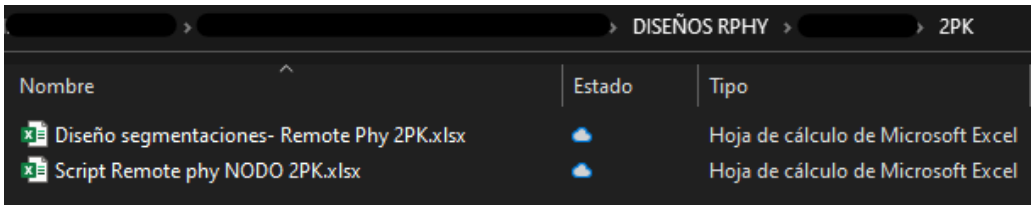


Figura 17. Subcarpeta de ejemplo

Fuente: Elaboración Propia

Cada una de las carpetas debe contener dos archivos, los cuales son el diseño que contiene la información del antes y después de la segmentación, y también el script que se debe aplicar a la hora de realizar la segmentación por parte del NOC. Ya teniendo la información de las carpetas más ordenada, se procede a verificar qué puertos se encuentran asignados según los diseños y según la marcación correspondiente en los equipos.

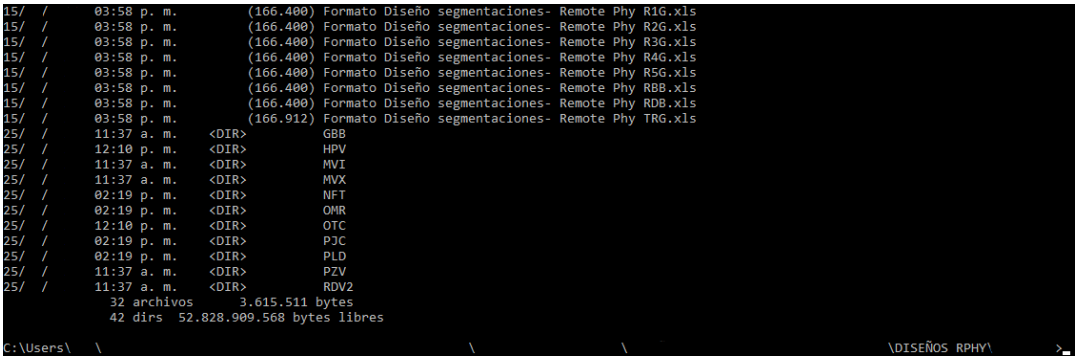


Figura 18. Validación carpetas y archivos generados por cmd

Fuente: Elaboración Propia

Para facilitar el proceso de creación de las carpetas faltantes, se aplica el siguiente comando el cual creará varias carpetas con los nombres de los nodos faltantes:

```
C:\Users\                                                    \DISEÑOS RPHY\
md R1G R2G R3G R4G R5G RBB RDB TRG
```

Figura 19. Comando creación carpetas por lotes

Fuente: Elaboración Propia

Estos cambios se ven reflejados inmediatamente en la carpeta compartida, por lo cual el siguiente paso que se debe realizar es mover los archivos de diseño a su carpeta correspondiente:

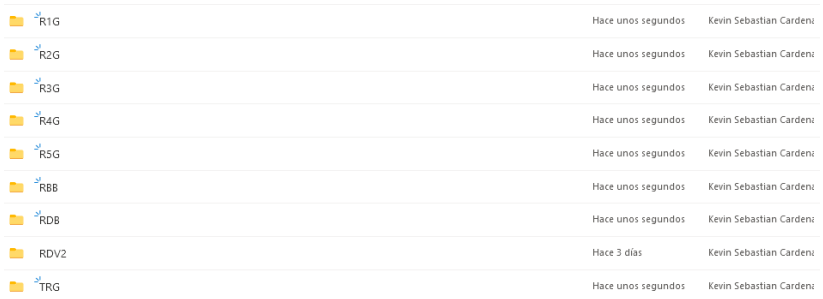


Figura 20. Carpetas actualizadas en SharePoint

Fuente: Elaboración Propia

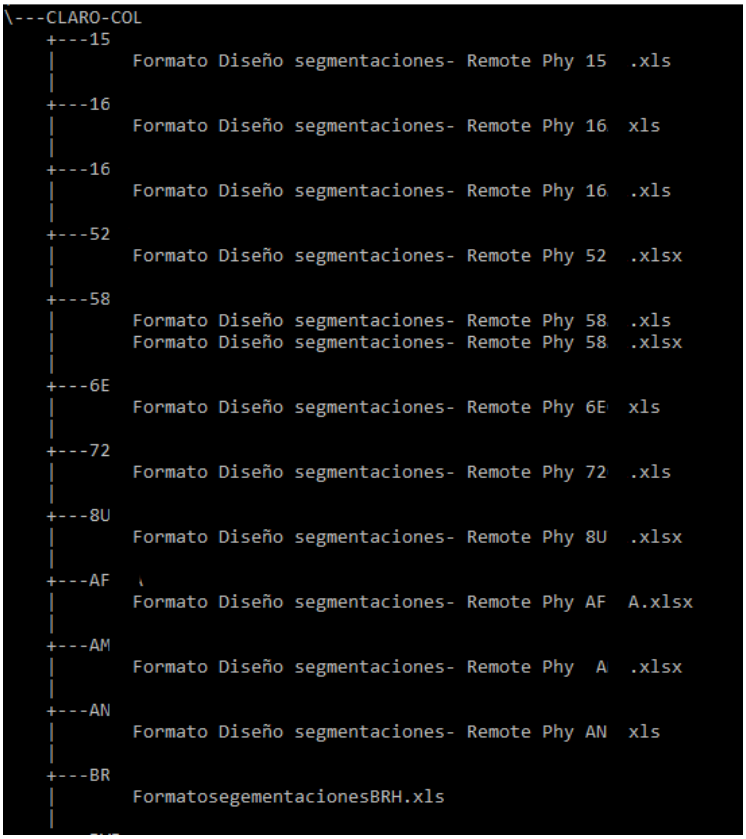


Figura 21. Estructura de carpetas actualizada

Fuente: Elaboración Propia

Una vez que se tenga la información ordenada, ya se puede decidir y recolectar la información de que puerto y vCMTS Core está asignado a cada nodo, y teniendo esta información se hace más fácil el proceso del diseño de nuevas segmentaciones de nodos, ya que se tendrá la certeza de cuáles puertos están verdaderamente libres, y en caso de que existan diseños que aún no estén marcados en el Switch de Distribución, se puede marcar los puertos correspondientes. A continuación, se muestra un ejemplo

de la correlación de marcación que debe existir en el Switch de distribución (DAAS) y el vCMTS (COS):

```
root@              1-DAAS> show interfaces descriptions
Interface       Admin Link Description
xe-0/0/0        up    up   ATP-ATP2F-ATP3F-ATP4F
xe-0/0/1        up    down PUERTO LIBRE
```

Figura 22. Interfaces en Switch Remoto

Fuente: Elaboración Propia

```
EC     @            4-COS>show cable rpd description
VC:VS   MAC-ADDRESS    DESCRIPTION                                SW-VERSION                      STATUS
------- -------------- ------------------------------------------ ------------------------------- --------
2:0     00  .a3  .94   ATP-ATP2F                                  1.24.1.0-1+auto8                online
2:1       20.  48. .6b ATP3F-ATP4F                                1.24.4.0-1+auto2                online
```

Figura 23. Interfaces en vCMTS

Fuente: Elaboración Propia

Cuando el diseño ya está ejecutado, la interfaz correspondiente está activada tanto en modo administrativo como en modo operativo “up up” (en este caso, el puerto marcado como ATP), en cambio, cuando la interfaz está únicamente activada administrativamente significa que únicamente se ha insertado un módulo SFP de 10Gb de capacidad, como en el caso del puerto xe-0/0/1 que se encuentra disponible.

A partir de estos antecedentes, se vio la necesidad de mejorar el proceso de creación de diseños y scripts, ya que esto es una tarea repetitiva y desgastante, algo que se puede automatizar de varias maneras, una de ellas es a través de lenguajes de programación, en este caso se usará Python, debido a las ventajas que posee, como la facilidad de uso y portabilidad.

Para realizar este script de Python es necesario tener algunos archivos previamente creados, los cuales consisten principalmente en:

- Plantillas de Diseño de Nodo:
  - Diseño_Plantilla.xlsx
- Plantilla de Diseño de Script:
  - Script_Plantilla.xlsx
- Archivo Excel con diseños pendientes de creación:
  - EXCEL DISEÑOS PENDIENTES.xlsx
- Script en Python paras realizar la automatización:
  - Script_Generador_Diseños.py

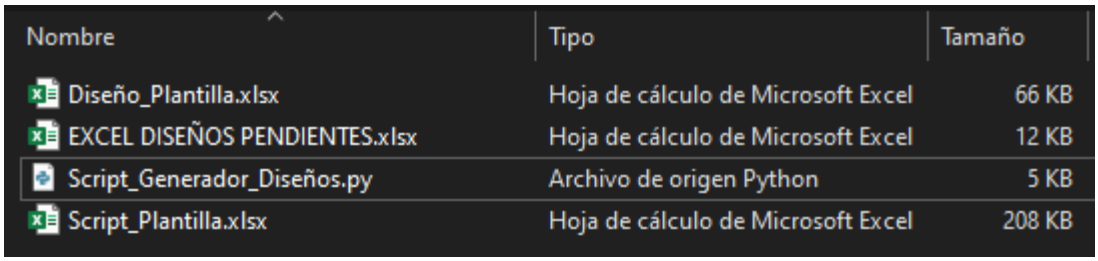


Figura 24. Archivos necesarios para generador de diseños Python

Fuente: Elaboración Propia

En primer lugar, vale la pena mostrar los campos que contiene el archivo *EXCEL DISEÑOS PENDIENTES.xlsx*. Dichos campos corresponden al ID del nodo, el nombre del mismo, el equipo Core destino, el Switch remoto destino, y los puertos correspondientes en los equipos Core y Switch, los cuales se ubicarán en el archivo de la siguiente manera:

	A	B	D	E	F	G
1	**ID NODO**	**NOMBRE**	**CORE DESTINO**	**SW_REMOTO DESTINO**	**PUERTO CORE**	**PUERTO SW_REMOTO**
2	KEV	nombre prueba 1	CORE VCMTS PRUEBA	SW DISTR 1	11	12
3	ALZ	ALFEREZ REAL 6	CORE VCCAP Prueba	SW distr 2	2	4
4	OLA	TOBERIN RESIDENCIAL	BOG-TOBERI-1	TOBERIN-SW-2	1	1
5	USTA	KEVIN CARDENAS	BOG-TOBERI-2	TOBERIN-SW-4	20	13

Figura 25. Datos de entrada archivo Excel

Fuente: Elaboración Propia

Cable aclarar que los valores de la imagen anterior son a modo de ejemplo, ya que por temas de confidencialidad no es posible mostrar la información de los nodos que se van a lanzar a producción

A continuación, se mostrará brevemente el código del archivo *Script_Generador_Diseños.py*, para lograr la automatización deseada

```python
import os, shutil, openpyxl

def obtenerValoresExcel(archivoExcel, col):
    wb = openpyxl.load_workbook(archivoExcel)
    sheet = wb.active
    res = []
    i = 2
    while True:
        column = sheet[col.upper() + str(i)].value
        if column == None:
            break
        else:
            res.append(str(column).upper())
        i = i+1
    return res
```

Figura 26. Código Script_Generador_Diseños.py #1

Fuente: Elaboración Propia

En primer lugar, se importan las librerías necesarias para el desarrollo del script, las cuales son:

- os: Se usa para manipular algunos aspectos del sistema operativo sobre el que se ejecuta el intérprete de Python
- shutil: Esta librería se usa para manipular archivos (copiar, mover, renombrar, etc.)
- openpyxl: Esta librería es usada principalmente para manipular archivos de Microsoft Excel (creación, modificar campos, obtener datos de una tabla, etc.)

A continuación, se genera una función que tomará una columna a la vez, y traerá todos los valores que encuentre del archivo EXCEL DISEÑOS PENDIENTES.xlsx, para convertirlos en listas de Python. Como parámetros de entrada pide el nombre del archivo de Excel a manipular y la letra de la columna sobre la cual se desea obtener los valores. Internamente se crea un for, en donde se va a tomar cada casilla de la columna, y se valida si el campo está diligenciado (es necesario que todos los campos estén correctamente diligenciados antes de aplicar el script)

```python
def crearNombresNodo(idNodo, nombreNodo, SegFisica):
    if SegFisica==True:
        c = [idNodo, idNodo+"2F", idNodo+"3B", idNodo+"4C"]
        d = [nombreNodo, nombreNodo+" 2F", nombreNodo+" 3B", nombreNodo+" 4C"]
    else:
        c = [idNodo, idNodo+"2F", idNodo+"3F", idNodo+"4F"]
        d = [nombreNodo, nombreNodo+" 2F", nombreNodo+" 3F", nombreNodo+" 4F"]
    listNombres = []
    i = 0
    for id_nodo in c:
        listNombres.append("NODO " + id_nodo + " (" + d[i]+")")
        i = i+1
    return listNombres
```

Figura 27. Código Script_Generador_Diseños.py #2

Fuente: Elaboración Propia

La función *crearNombresNodo()* se encarga de recibir el ID del nodo, el nombre completo del mismo y un parámetro True o False, que permitirá validar si los nombres corresponden a un tipo de segmentación u otro, por dentro de la función se hace la validación del tipo de segmentación y según el caso, concatena las marcaciones de una manera u otra, según el estándar que se maneja en la compañía, y almacena el resultado en una lista que será devuelta por la función para usarse más adelante.

```python
def crearCarpetas(idNodo):
    nombreArchivoDiseño = 'Formato Diseño Segmentaciones RPHY - Nodo ' + idNodo + '.xlsx'
    nombreArchivoScript = 'Script RPHY - Nodo ' + idNodo + '.xlsx'

    os.makedirs('DISEÑOS GENERADOS/' + idNodo)

    rutaCompletaDiseño = "DISEÑOS GENERADOS/" + idNodo + "/" + nombreArchivoDiseño
    rutaCompletaScript = "DISEÑOS GENERADOS/" + idNodo + "/" + nombreArchivoScript
    shutil.copy("Diseño_Plantilla.xlsx", rutaCompletaDiseño, follow_symlinks=True)
    shutil.copy("Script_Plantilla.xlsx", rutaCompletaScript, follow_symlinks=True)
    return rutaCompletaDiseño, rutaCompletaScript
```

Figura 28. Código Script_Generador_Diseños.py #3

Fuente: Elaboración Propia

En esta función, se pasa por parámetro el ID del nodo, y se encarga automáticamente de crear una subcarpeta con el nombre del nodo y generar los dos archivos (script y diseño) con el nombre de cada nodo estandarizado. Además de eso, esta función devuelve la ruta generada para el script y el diseño del nodo, los cuales se usarán más adelante.

```python
def escribirExcelArchivoScript(libroExcel, listMarcaciones, portCOS, portDAAS):
    wb = openpyxl.load_workbook(libroExcel)
    sheet = wb.active

    sheet['I3'] = listMarcaciones[0]
    sheet['I4'] = listMarcaciones[1]
    sheet['I6'] = listMarcaciones[2]
    sheet['I7'] = listMarcaciones[3]

    sheet['H4'] = portCOS
    sheet['H5'] = portDAAS

    wb.save(libroExcel)
    wb.close()
```

Figura 29. Código Script_Generador_Diseños.py #4

Fuente: Elaboración Propia

En la función *escribirExcelArchivoScript()* se procede a abrir el archivo de script previamente creado, y se le ingresan las marcaciones correspondientes en cada celda de la tabla de Excel, además de los puertos de equipo Core (vCMTS, o COS, según la nomenclatura del vendor) y el Switch de distribución (en este código se llama DAAS debido a la nomenclatura del vendor).

Todos estos datos entran como parámetros de la función, específicamente se ingresa el archivo de Excel (el script) a modificar, una lista con las 4 marcaciones (de los 4 puertos físicos que tiene el nodo), y el puerto correspondiente en vCMTS y Switch de distribución (ya que, en muchos casos, no son los mismos), y una vez marcados, se guarda y cierra el archivo de Excel.

```python
def escribirExcelArchivoDiseño(libroExcel, listMarcaciones, portCOS, portDAAS, nomCOS, nomDAAS):
    wb = openpyxl.load_workbook(libroExcel)
    sheet = wb.active
```

Figura 30. Código Script_Generador_Diseños.py #5

Fuente: Elaboración Propia

```
62      sheet['J5'] = nomDAAS
63      sheet['J6'] = nomDAAS
64      sheet['J8'] = nomDAAS
65      sheet['J9'] = nomDAAS
66
67      sheet['L5'] = nomCOS
68      sheet['L6'] = nomCOS
69      sheet['L8'] = nomCOS
70      sheet['L9'] = nomCOS
71
72      sheet['K5'] = portDAAS + "/0"
73      sheet['K6'] = portDAAS + "/0"
74      sheet['K8'] = portDAAS + "/0"
75      sheet['K9'] = portDAAS + "/0"
76
77      sheet['M5'] = portCOS + ":0"
78      sheet['M6'] = portCOS + ":0"
79      sheet['M8'] = portCOS + ":0"
80      sheet['M9'] = portCOS + ":0"
81      sheet['N5'] = portCOS + ":0/0"
82      sheet['N6'] = portCOS + ":0/0"
83      sheet['N8'] = portCOS + ":0/0"
84      sheet['N9'] = portCOS + ":0/0"
85      sheet['O5'] = portCOS + ":0/0.0"
86      sheet['O6'] = portCOS + ":0/0.0"
87      sheet['O8'] = portCOS + ":0/0.0"
88      sheet['O9'] = portCOS + ":0/0.0"
89
90      sheet['Q5'] = listMarcaciones[0]
91      sheet['Q6'] = listMarcaciones[1]
92      sheet['Q8'] = listMarcaciones[2]
93      sheet['Q9'] = listMarcaciones[3]
94
95      wb.save(libroExcel)
96      wb.close()
```

Figura 31. Código Script_Generador_Diseños.py #6

Fuente: Elaboración Propia

Finalmente, en esta función se hace algo similar que, en la anterior, se pide por parámetro el libro de Excel a utilizar, la lista de marcaciones generadas con la función *crearNombresNodo(),* además de los nombres y puertos de los equipos vCMTS y Switch de distribución.

Una vez se obtiene esa información se abre el archivo Excel a través de un objeto Workbook de la librería openpyxl, además se selecciona la hoja activa, luego se modifican las celdas necesarias, según la información que entro como parámetro, específicamente en la parte de los puertos, se concatena el número de puerto recibido con un String que se necesita para cumplir con el estándar interno de diseño

```
file = 'EXCEL DISEÑOS PENDIENTES.xlsx'
idNodos, nombres, cos, daas, puerto_cos, puerto_daas = "A", "B", "D", "E", "F", "G"
idNodos_list = obtenerValoresExcel(file, idNodos)
nombres_list = obtenerValoresExcel(file, nombres)
cos_list = obtenerValoresExcel(file, cos)
daas_list = obtenerValoresExcel(file, daas)
puerto_cos_list = obtenerValoresExcel(file, puerto_cos)
puerto_daas_list = obtenerValoresExcel(file, puerto_daas)
```

Figura 32. Código Script_Generador_Diseños.py #7

Fuente: Elaboración Propia

En esta parte del código se declaran algunas variables, como el nombre del archivo de entrada (donde se obtienen los valores de los diseños a generar), los cuales se van obteniendo columna por columna, a cada función se le ingresa el nombre del archivo y la letra de la columna correspondiente.

```
i2 = 0
for i in idNodos_list:
    marcacionPuerto = crearNombresNodo(i, nombres_list[i2], True)
    x = crearCarpetas(i)
    escribirExcelArchivoScript(x[1], marcacionPuerto, puerto_cos_list[i2], puerto_daas_list[i2])
    escribirExcelArchivoDiseño(x[0], marcacionPuerto, puerto_cos_list[i2], puerto_daas_list[i2], cos_list[i2], daas_list[i2])
    print("Generado - ID: "+i)
    i2 = i2 +1
```

Figura 33. Código Script_Generador_Diseños.py #8

Fuente: Elaboración Propia

Ahora se crea una variable i2 que está inicializada en 0, y también se crea un for que va desde 0 hasta la cantidad de nodos pendientes, luego se crea una variable que va a generar las marcaciones correspondientes a cada diseño y las va a almacenar, posteriormente va a generar la carpeta nombrada con el ID del nodo y los archivos de diseño y script, y finalmente va a tomar las marcaciones y las escribirá sobre los archivos Excel, además cada vez que complete este proceso lo va a indicar mediante consola como se muestra a continuación:

```
PS C:\Users\Kevin\Desktop\PYTHON CREAR DISEÑOS> & "C:/Program Files/Python310/python.exe"
"c:/Users/Kevin/Desktop/PYTHON CREAR DISEÑOS/Script_Generador_Diseños.py"
Generado - ID: KEV
Generado - ID: ALZ
Generado - ID: OLA
Generado - ID: USTA
```

Figura 34. Ejecución Script Script_Generador_Diseños.py

Fuente: Elaboración Propia

En la siguiente imagen se puede ver como quedo la estructura de carpetas luego de ejecutar el script, como se puede ver, se generó correctamente la carpeta y los archivos que debe tener internamente:

```
PS C:\Users\Kevin\Desktop\PYTHON CREAR DISEÑOS> tree /f
Listado de rutas de carpetas para el volumen Disco Estado Solido
El número de serie del volumen es 9A92-2816
C:.
    Diseño_Plantilla.xlsx
    EXCEL DISEÑOS PENDIENTES.xlsx
    Script_Generador_Diseños.py
    Script_Plantilla.xlsx

└───DISEÑOS GENERADOS
    ├───ALZ
            Formato Diseño Segmentaciones RPHY - Nodo ALZ.xlsx
            Script RPHY - Nodo ALZ.xlsx

    ├───KEV
            Formato Diseño Segmentaciones RPHY - Nodo KEV.xlsx
            Script RPHY - Nodo KEV.xlsx

    ├───OLA
            Formato Diseño Segmentaciones RPHY - Nodo OLA.xlsx
            Script RPHY - Nodo OLA.xlsx

    └───USTA
            Formato Diseño Segmentaciones RPHY - Nodo USTA.xlsx
            Script RPHY - Nodo USTA.xlsx
```

Figura 35. Estructura generada después de ejecución de Script Script_Generador_Diseños.py

Fuente: Elaboración Propia

Al abrir los archivos generados (en este caso usamos el ID “USTA” como ejemplo) se puede observar que los campos fueron correctamente asignados, tanto en los puertos, como en los nombres.

G	H	I
Tipo de Segmentación		NOMBRES NODOS
remote-phy		NODO USTA (KEVIN CARDENAS)
Puerto COS	20	NODO USTA2F (KEVIN CARDENAS 2F)
Puerto DAAS	13	
		NODO USTA3B (KEVIN CARDENAS 3B)
		NODO USTA4F (KEVIN CARDENAS 4F)

Figura 36. Archivo Script Ejemplo

Fuente: Elaboración Propia

BOG-TOBERI-2

DESPUES

Tipo de Segmentacion	SW REMOTO	PUERTO	VCMTS DESTINO					NOMBRE
remote-phy	TOBERIN-SW-4	13/0	BOG-TOBERI-2	20:0	20:0/0	20:0/0.0		NODO USTA (KEVIN CARDENAS)
	TOBERIN-SW-4	13/0	BOG-TOBERI-2	20:0	20:0/0	20:0/0.0		NODO USTA2F (KEVIN CARDENAS 2F)
	TOBERIN-SW-4	13/0	BOG-TOBERI-2	20:0	20:0/0	20:0/0.0		NODO USTA3B (KEVIN CARDENAS 3B)
	TOBERIN-SW-4	13/0	BOG-TOBERI-2	20:0	20:0/0	20:0/0.0		NODO USTA4F (KEVIN CARDENAS 4F)

Figura 37. Archivo Diseño Ejemplo

Fuente: Elaboración Propia

7.2. ACCESO A LOS EQUIPOS DE RPHY:

Como se comenta anteriormente, SecureCRT permite guardar las sesiones, incluyendo datos como usuario y contraseña, para que, a la hora de acceder a los equipos, ya q no sea necesario ingresar los valores, optimizando el tiempo, y permitiendo el acceso automático a los equipos, mejorando la productividad del personal del proyecto, con este fin, a continuación, se mostrará como realizar las configuraciones necesarias para automatizar el login a los equipos:

- Para crear una nueva sesión se abre el panel de sesiones y a continuación se da clic en el botón +:

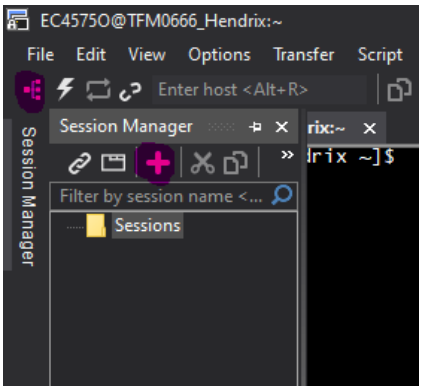


Figura 38. Pasos para crear nueva sesión #1

Fuente: Elaboración Propia

- A continuación, se abre el asistente de creación de sesiones, se selecciona la conexión SSH2 y luego se da clic en siguiente:

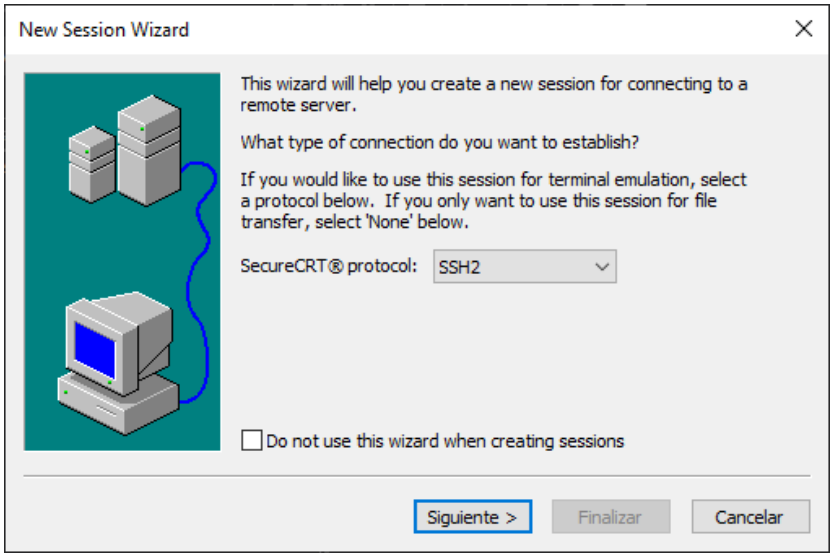


Figura 39. Pasos para crear nueva sesión #2

Fuente: Elaboración Propia

- Ahora se diligencian los campos de Hostname, Puerto y Usuario, luego clic en Siguiente:

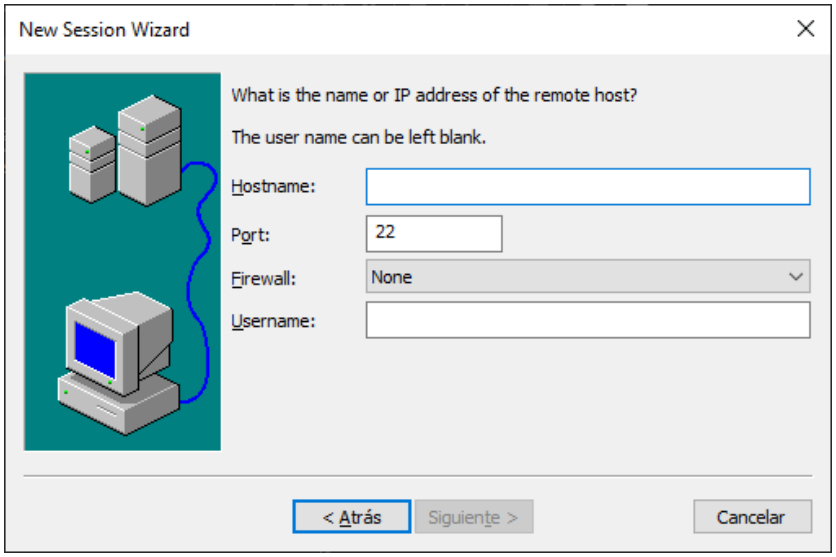


Figura 40. Pasos para crear nueva sesión #3

Fuente: Elaboración Propia

- Ahora se deja la opción por defecto y se da clic de nuevo en siguiente:

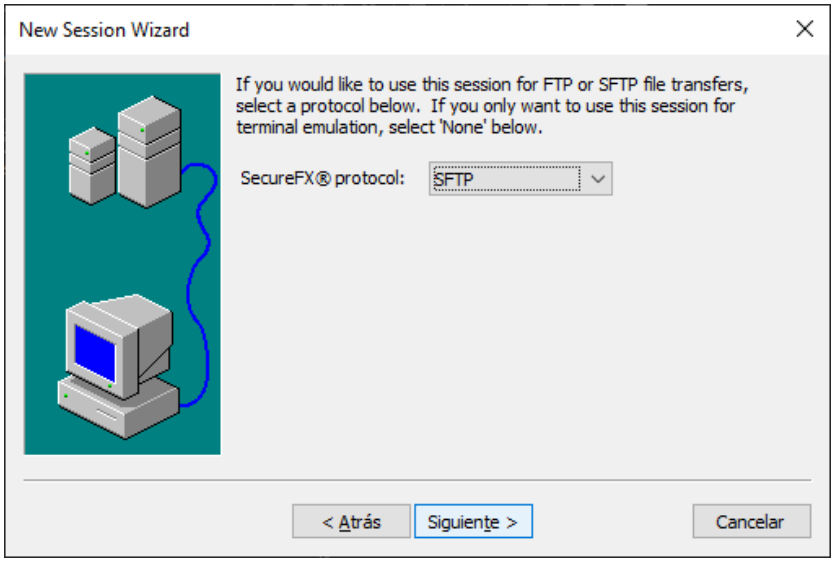


Figura 41. Pasos para crear nueva sesión #4

Fuente: Elaboración Propia

- En este punto se debe diligenciar el nombre de la sesión (para este caso de esta propuesta se sugiere poner el nombre del equipo al que se accederá) y si se requiere una descripción:

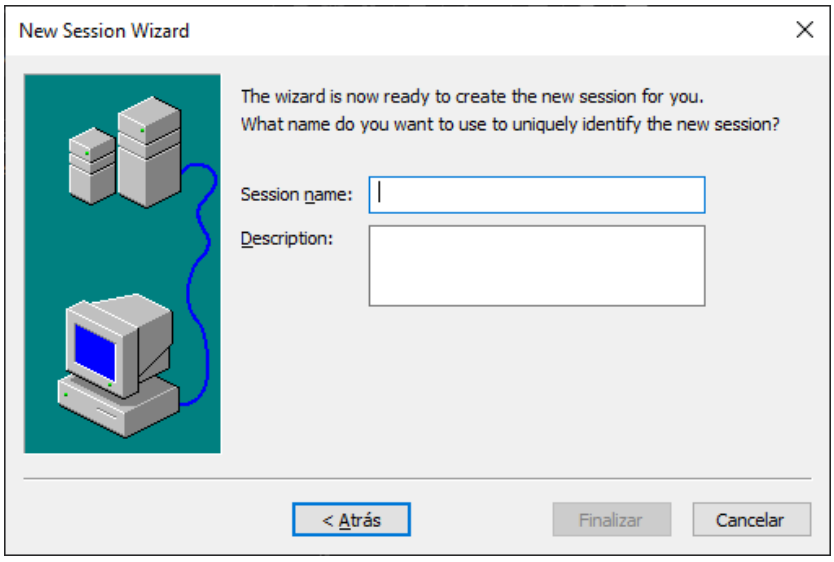


Figura 42. Pasos para crear nueva sesión #5

Fuente: Elaboración Propia

- Ahora la sesión quedara creada, y se puede acceder al equipo simplemente dando doble clic sobre el nombre:

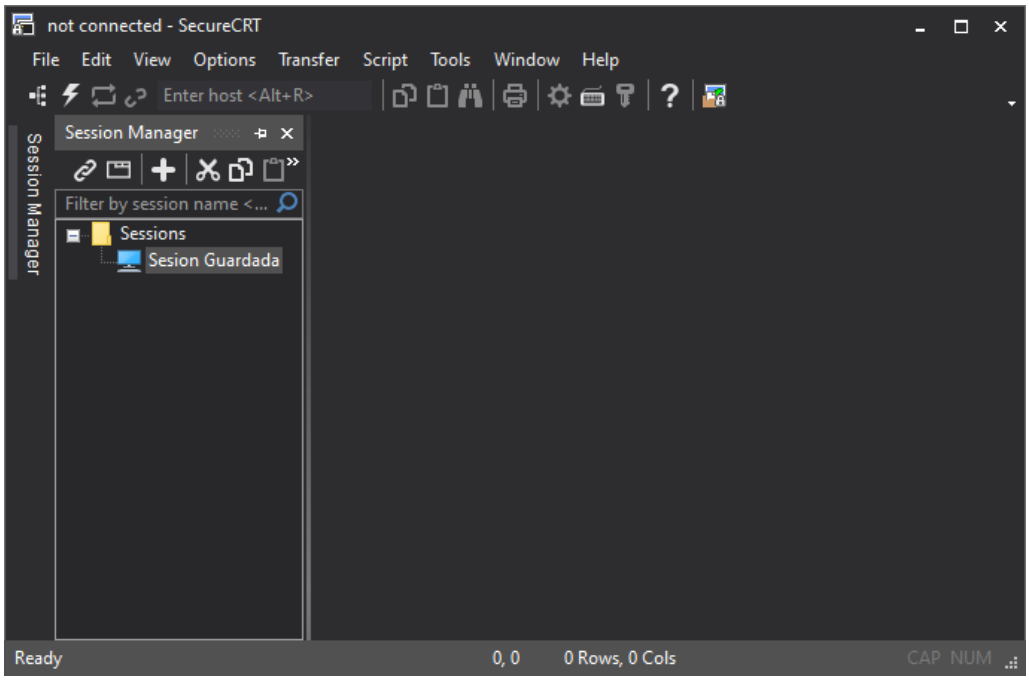


Figura 43. Pasos para crear nueva sesión #6

Fuente: Elaboración Propia

- La primera vez que se abra la sesión, se solicitara añadir las claves correspondientes, para evitar que vuelva a salir la advertencia se debe dar clic en “Accept & Save”:

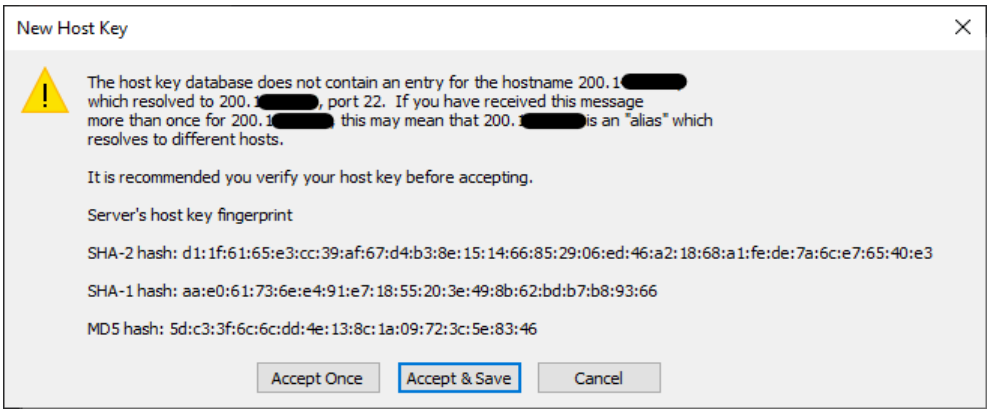


Figura 44. Pasos para crear nueva sesión #7

Fuente: Elaboración Propia

- A continuación, se solicitará el usuario y la contraseña, para evitar que se pida de nuevo, se debe dar clic en “Save password”:

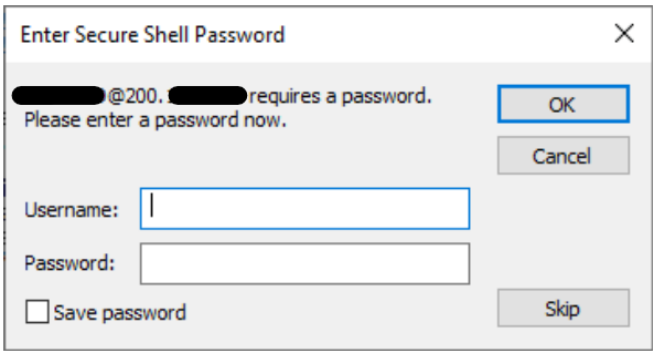


Figura 45. Pasos para crear nueva sesión #8

Fuente: Elaboración Propia

- Ya con esta configuración se puede acceder al servidor intermediario simplemente dando doble clic sobre el nombre de la sesión, pero para automatizar el acceso a los equipos e Remote PHY es necesario realizar un paso adicional:

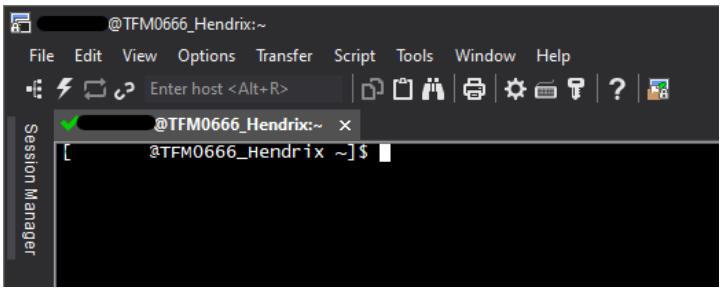


Figura 46. Pasos para crear nueva sesión #9

Fuente: Elaboración Propia

- Este paso consiste en dar clic derecho sobre la sesión, y hacer clic en la opción de propiedades:

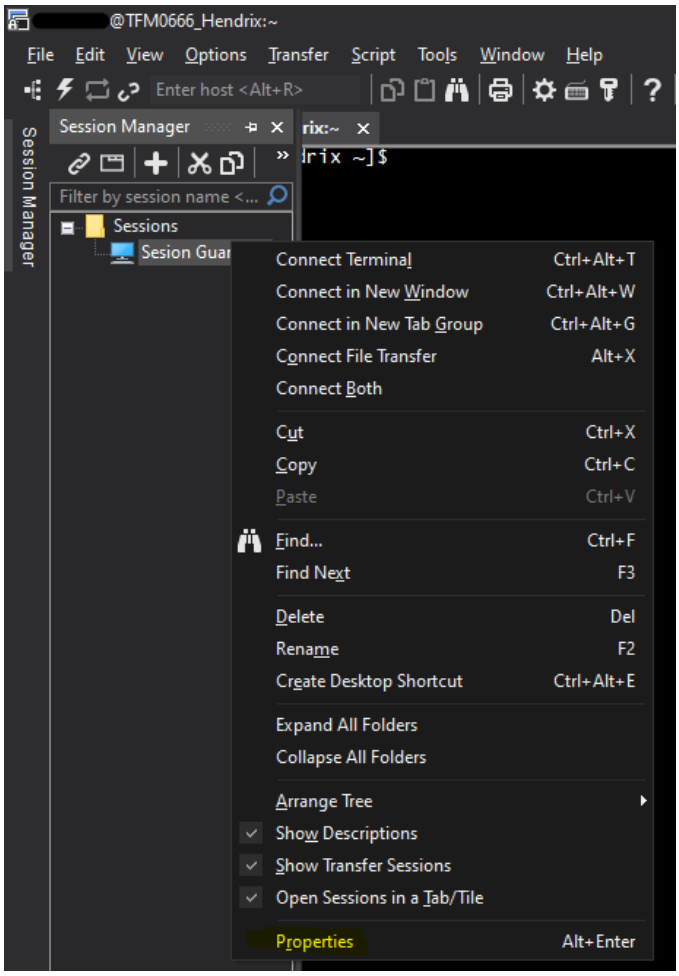


Figura 47. Pasos para crear nueva sesión #10

Fuente: Elaboración Propia

- Una vez, se despliegue esta ventana, en el menú de la parte izquierda dar clic en la segunda opción, llamada “Logon Actions”, la cual mostrará los siguientes campos:

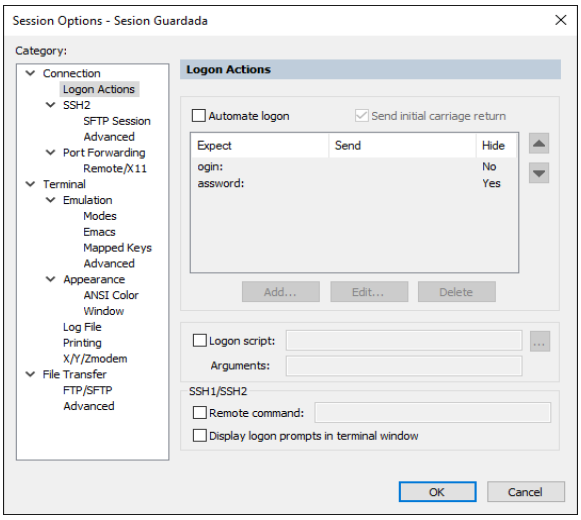


Figura 48. Pasos para crear nueva sesión #11

Fuente: Elaboración Propia

- Los campos deben quedar de manera similar a la siguiente imagen, para que se acceda automáticamente a los equipos:

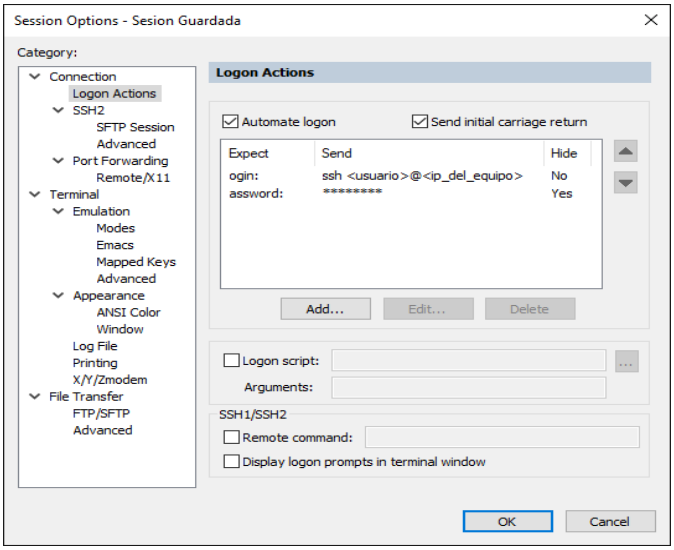


Figura 49. Pasos para crear nueva sesión #12

Fuente: Elaboración Propia

Así se podrá automatizar el ingreso a los equipos de la solución Remote PHY, mejorando el tiempo en el que se logra acceder a dichos equipos.

Otra de las mejoras que se proponen para mejorar el tiempo que toma acceder a los equipos es el uso de un archivo de texto generado directamente en la consola Linux desde donde se accede a los equipos, para facilitar el acceso a la información de los equipos de la plataforma, ya que normalmente las IPs se obtienen de un archivo Excel compartido en SharePoint, algo poco intuitivo y que a veces toma más tiempo del que se esperaría, para una tarea tan simple y haciendo perder productividad.

Para solucionar ese inconveniente se planteó mantener todos los equipos en un archivo .txt directamente en el servidor Linux, para facilitar la tarea de obtener las IPs de los equipos que se desea acceder en un momento dado. Con este fin se crea una tabla en Excel con los datos básicos de los equipos

Los datos de la siguiente tabla son a modo de ejemplo, pero sirven de guía sobre el desarrollo de la solución:

HOSTNAME	IP GESTION	IP LOOPBACK	COMUNIDAD SNMP	CIUDAD	PROVEEDOR
SOA-SOA-H-01-COS	172.16.1.1	n/a	SOAH1COS	SOACHA	PROVEEDOR1
SOA-SOA-H-01-CRE	172.16.1.2	172.20.1.1	SOAH1CRE	SOACHA	PROVEEDOR1
SOA-SOA-H-01-DAAS	172.16.1.3	172.20.1.2	SOAH1DAAS	SOACHA	PROVEEDOR1
BOG-FON-H-VCCAP1	172.16.1.4	172.20.1.3	BOGFONVCCAP1	BOGOTA	PROVEEDOR2
BOG-FON-H-TOR1	172.16.1.5	172.20.1.4	BOGFONVCCAP2	BOGOTA	PROVEEDOR2

Figura 50. Ejemplo direcciones IP de Equipos RPHY

Fuente: Elaboración Propia

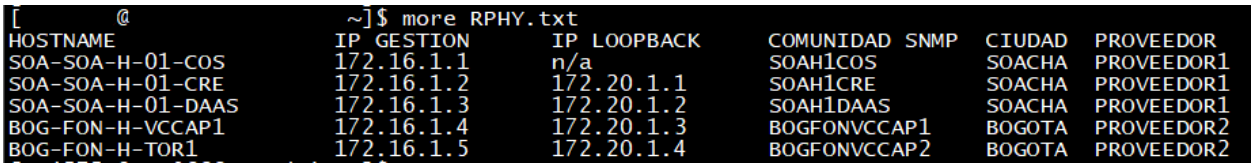
Con los datos de la tabla anterior se puede generar unas líneas de comando que permitirán crear un archivo de texto directamente en el servidor, que contenga la información que se usa más a menudo para poder acceder a los equipos por medio de dicho servidor intermedio, principalmente las IPs de loopback y de gestión

```
touch RPHY.txt
echo "HOSTNAME      IP GESTION      IP LOOPBACK     COMUNIDAD SNMP      CIUDAD      PROVEEDOR" >> RPHY.txt
echo "SOA-SOA-H-01-COS  172.16.1.1  n/a  SOAH1COS  SOACHA  PROVEEDOR1" >> RPHY.txt
echo "SOA-SOA-H-01-CRE  172.16.1.2  172.20.1.1  SOAH1CRE  SOACHA  PROVEEDOR1" >> RPHY.txt
echo "SOA-SOA-H-01-DAAS   172.16.1.3  172.20.1.2  SOAH1DAAS  SOACHA  PROVEEDOR1" >> RPHY.txt
echo "BOG-FON-H-VCCAP1  172.16.1.4  172.20.1.3  BOGFONVCCAP1  BOGOTA  PROVEEDOR2" >> RPHY.txt
echo "BOG-FON-H-TOR1  172.16.1.5  172.20.1.4  BOGFONVCCAP2  BOGOTA  PROVEEDOR2" >> RPHY.txt
```

Figura 51. Comandos a ejecutar en el bash

Fuente: Elaboración Propia

Una vez que se apliquen estos comandos en el shell, se puede revisar rápidamente como quedó organizado el archivo, a través del comando “more RPHY.txt”, el cual mostrará todo el contenido del archivo:

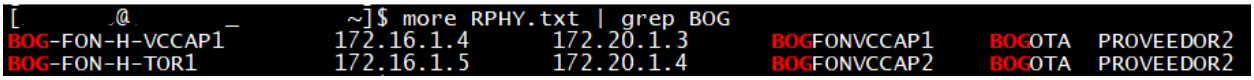


```
[       @                ~]$ more RPHY.txt
HOSTNAME                IP GESTION      IP LOOPBACK     COMUNIDAD SNMP  CIUDAD  PROVEEDOR
SOA-SOA-H-01-COS        172.16.1.1      n/a             SOAH1COS        SOACHA  PROVEEDOR1
SOA-SOA-H-01-CRE        172.16.1.2      172.20.1.1      SOAH1CRE        SOACHA  PROVEEDOR1
SOA-SOA-H-01-DAAS       172.16.1.3      172.20.1.2      SOAH1DAAS       SOACHA  PROVEEDOR1
BOG-FON-H-VCCAP1        172.16.1.4      172.20.1.3      BOGFONVCCAP1    BOGOTA  PROVEEDOR2
BOG-FON-H-TOR1          172.16.1.5      172.20.1.4      BOGFONVCCAP2    BOGOTA  PROVEEDOR2
```

Figura 52. Ejemplo ejecución comando en bash

Fuente: Elaboración Propia

En dado caso que se requiera únicamente información específica, se puede aplicar un comando de filtrado como el que se ve en la imagen a continuación:



```
[       @                ~]$ more RPHY.txt | grep BOG
BOG-FON-H-VCCAP1        172.16.1.4      172.20.1.3      BOGFONVCCAP1    BOGOTA  PROVEEDOR2
BOG-FON-H-TOR1          172.16.1.5      172.20.1.4      BOGFONVCCAP2    BOGOTA  PROVEEDOR2
```

Figura 53. Ejemplo con comando de filtrado

Fuente: Elaboración Propia

El manejo de este archivo es especialmente útil para aquellos que, por una razón u otra, solo pueden acceder a los equipos de manera ocasional, para temas de troubleshooting, a través de clientes SSH como PuTTY, que son bastante básicos.

Este archivo puede ser fácilmente editable desde la consola de Linux con comandos como “vim” u otros ya que a medida que aumente la cantidad de equipos de la plataforma Remote PHY, es necesario actualizar el archivo.

7.3. SCRIPTS AUTOMATIZACIÓN DE TAREAS EN SECURECRT

Como se expuso anteriormente, el programa SecureCRT es una poderosa herramienta a la hora de manipular sesiones SSH, ya que tiene múltiples métodos y funciones con los que se puede manipular variables, obtener información y automatizar tareas, incluso pudiendo extraer datos hacia otros programas, como Microsoft Excel. Para el desarrollo de esta propuesta, se harán 4 casos de uso, aunque a futuro se pueden diseñar más scripts para automatizar tareas específicas:

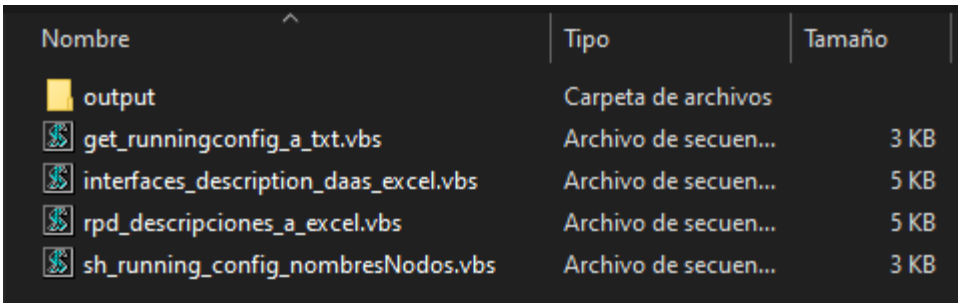


Figura 54. Archivos de scripts para SecureCRT

Fuente: Elaboración Propia

Los scripts que se generaron para automatizar dichas tareas son los siguientes:

- **get_runningconfig_a_txt.vbs:** Este script permite obtener la salida del comando "show running-config" y la almacena en un archivo de texto automáticamente
- **interfaces_description_daas_excel.vbs:** Obtiene la descripción y estado de las interfaces de un Switch de distribución y las guarda en un archivo de Excel, separando cada parámetro en una columna diferente.
- **rpd_descripciones_a_excel.vbs:** Ejecuta el comando que obtiene las RPDs conectadas a un equipo en una sesión previamente creada junto con otras variables como la MAC del dispositivo o el nombre de cada RPD, también está separado cada dato en una columna separada.
- **sh_running_config_nombresNodos.vbs:** Este script permite tomar los nombres completos de los nodos que ya se encuentren activos en cada vCCAP CMTS Core y los agrega a un archivo Excel.

A continuación, se describe el funcionamiento de cada uno de los scripts mencionados anteriormente:

7.3.1. SCRIPT GET_RUNNINGCONFIG_A_TXT.VBS

Para todos los scripts, las dos primeras líneas permiten a SecureCRT saber el motor de scripting con el que deben manejar el archivo, esto es principalmente útil cuando estos archivos no tienen la extensión vba, aunque en este caso es opcional, ya que dichos archivos si están nombrado con la extensión.

La primera función que se codifica es *obtenerNombreEquipo()*, en esta función se encarga de obtener el nombre del equipo, a través de la captura de la última línea que se muestra en el CLI (en donde se muestra el nombre y el puntero donde se escriben los comandos) y posteriormente se separa la parte que representa el usuario de la parte que

nombra al equipo (generalmente separados por arroba), en este código se tiene en cuenta para varios equipos, ya que algunos poseen el carácter # como delimitador y otros poseen el carácter @ y >, por lo que se trabaja estas cadenas de texto de manera diferente.

```
#$Language="VBScript"
#$Interface="1.0"

Function obtenerNombreEquipo()
    Dim nombreEquipo
    'la funcion get() pide 4 parametros, en este caso la posicion actual del cursor - nombre_archivo equipo
    nombreEquipo = crt.Screen.Get(crt.Screen.CurrentRow, 1, crt.Screen.CurrentRow, crt.Screen.CurrentColumn)
    Dim posArroba, posCRE, posNumeral, stringExtraido
    posArroba = InStr(nombreEquipo, "@")
    posCRE = InStr(nombreEquipo, "CRE")
    If posArroba<>0 Then
        Dim posicionInicial, lenghtCaracteres
        posicionInicial = InStr(nombreEquipo,"@")+1
        lenghtCaracteres = InStr(nombreEquipo,">")-InStr(nombreEquipo,"@")-1
        nomSeparado = Mid(nombreEquipo, posicionInicial, lenghtCaracteres)
        obtenerNombreEquipo = nomSeparado
    elseif posCRE<>0 Then
        posNumeral = InStr(nombreEquipo, "#")
        stringExtraido = Mid(nombreEquipo, 1, posNumeral - 1)
        obtenerNombreEquipo = stringExtraido
    End If
End Function
```

Figura 55. Código get_runningconfig_a_txt.vbs #1

Fuente: Elaboración Propia

La función *obtenerFecha()* se encarga de obtener la fecha y la hora actual, para posteriormente usarla en otras partes del script (para este caso se usará en el nombre de los archivos generados)

```
24  Function obtenerFecha()
25      Dim nom, m, d, h, n, s
26      nom = Year(Date)
27      m = Month(Date) : If Len(m)=1 Then m = "0" & m : End If
28      d = Day(Date) : If Len(d)=1 Then d = "0" & d : End If
29      h = Hour(Time) : If Len(h)=1 Then h = "0" & h : End If
30      n = Minute(Time) : If Len(n)=1 Then : n = "0" & n : End If
31      s = Second(Time) : If Len(s)=1 Then : s = "0" & s : End If
32      obtenerFecha=nom&"."&m&"."&d&"-"&h&"."&n&"."&s
33  End Function
```

Figura 56. Código get_runningconfig_a_txt.vbs #2

Fuente: Elaboración Propia

La función *running_config_a_txt()* ejecuta el comando “show running config” en una sesión previamente abierta y espera a que salgan algunos String específicos (los cuales son la parte final de hostname de cada equipo). Adicional a esto, esta parte del código también se encarga de crear un objeto que represente al sistema de archivos y otro objeto que represente al archivo txt, abre el archivo que se ingresa por parámetro en modo “append”, es decir, no lo sobrescribe totalmente, sino que la información que se ingrese a dicho archivo se escribirá debajo de los datos existentes.

```
35  Function running_config_a_txt(rutaTXT)
36      crt.Screen.Synchronous = True
37      crt.Screen.Send "show running-config " & Chr(10)
38      crt.Screen.WaitForString Chr(10)
39
40      Dim stringsEspera
41      stringsEspera = Array( "COS>", "CRE#", "DAAS>", "--More--")
42
43      Dim linea_actual, screenrow, readline
44
45      Dim fs, f
46      Set fs = CreateObject("Scripting.FileSystemObject")
47      Set f = fs.OpenTextFile(rutaTXT, 8, True)
```

Figura 57. Código get_runningconfig_a_txt.vbs #3

Fuente: Elaboración Propia

En esta parte del código se crea un bucle while que va a ir leyendo todas las líneas que se vayan mostrando en la sesión SSH, y toma esos caracteres y los almacena en la variable readline, adicionalmente usa el valor de la variable result (función que permite verificar si la línea actual contiene alguno de los valores de espera previamente creados, es decir, los hostname) el varias instrucciones for para validar si ya se terminó de ejecutar el comando y puede finalizar el ciclo, a medida que va avanzando por el ciclo se va a añadir cada una de las líneas hasta que se encuentre con alguno de los valores de la variable "*stringsEspera*" y salga del ciclo. En este punto se cierra el archivo txt y se finaliza la función.

```
49      linea_actual = 0
50      Do
51        While True
52          result = crt.Screen.WaitForStrings( stringsEspera )
53          screenrow = crt.screen.CurrentRow - 1
54          readline = crt.Screen.Get(screenrow, 1, screenrow, 150)
55          If result = 1 Then
56            crt.Screen.Send Chr(10)
57            Exit Do
58          End If
59          If result = 2 or result = 3 Then
60            crt.Screen.Send Chr(10)
61            Exit Do
62          End If
63          If result = 4 Then
64            crt.Screen.Send Chr(32)
65          End If
66          f.Write readline & chr(10)
67          linea_actual = linea_actual + 1
68        Wend
69      Loop
70      f.Close
71      crt.Screen.Synchronous = False
72  End Function
```

Figura 58. Código get_runningconfig_a_txt.vbs #4

Fuente: Elaboración Propia

En la parte principal del código se crean algunas variables, y se llaman a las funciones anteriormente creadas, la función que devuelve en nombre del equipo y la función que devuelve la fecha y hora actual, las cuales se usarán para nombrar el archivo que se pasará como parámetro de la función *running_config_a_txt()*, y finalmente genera un aviso que indique que ya se terminó el proceso.

```
74  Sub Main
75      Dim nombre_equipo, fecha_actual, ruta_txt
76      nombre_equipo = obtenerNombreEquipo()
77      fecha_actual = obtenerFecha()
78      ruta_txt = "C:\Users\Kevin C\Desktop\script_securecrt\SCRIPTS_SECURECRT\output" & nombre_equipo & "__RUNNING-CONFIG__" & obtenerFecha()
79      running_config_a_txt(ruta_txt)
80      MsgBox "HECHO!!"
81  End Sub
```

Figura 59. Código get_runningconfig_a_txt.vbs #5

Fuente: Elaboración Propia

Para ejecutar este (u otro script) se debe buscar el menú “Script” de la aplicación, y a continuación se debe dar clic en la opción “Run”:

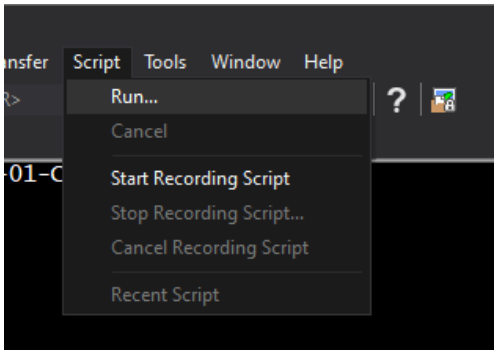


Figura 60. Ejecución Script en SecureCRT

Fuente: Elaboración Propia

Luego, se abrirá una ventana en donde se debe seleccionar el script a ejecutar:

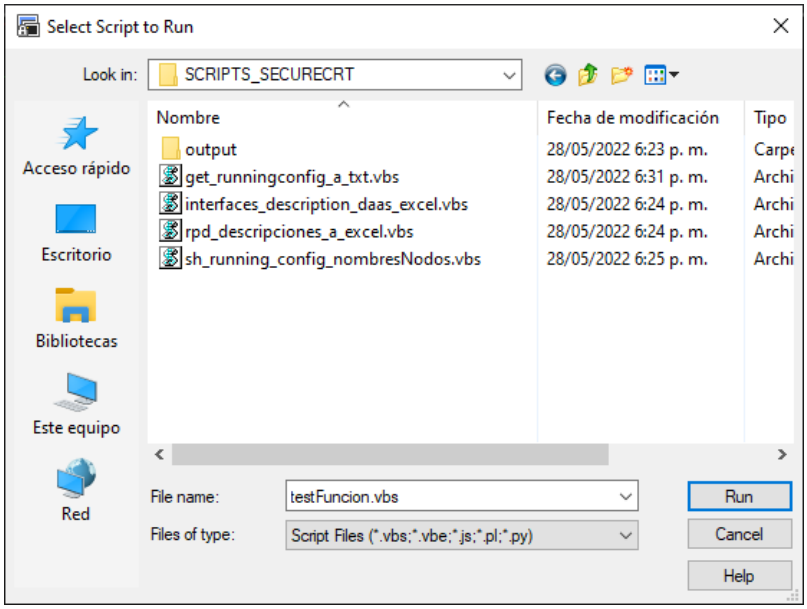


Figura 61. Ventana seleccionar script a ejecutar

Fuente: Elaboración Propia

Como se mencionó anteriormente, se debe tener la sesión en el equipo abierta previamente, una vez hecho esto, se ejecuta el script y como se puede ver en la siguiente imagen, se muestra el mensaje que indica que se ejecutó de manera correcta:

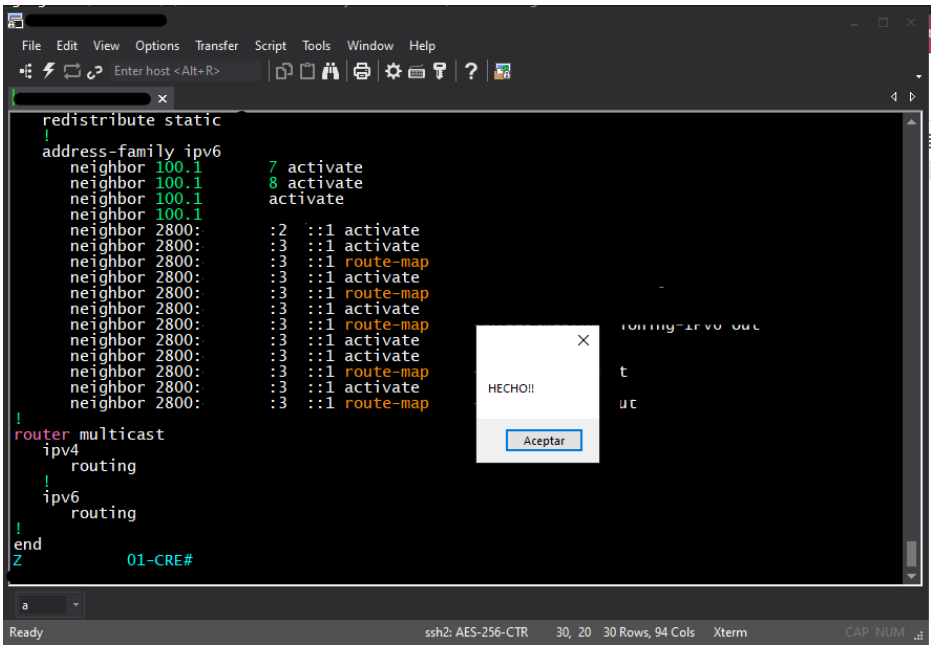


Figura 62. Ejecución de comando en sesión de SecureCRT

Fuente: Elaboración Propia

Para facilitar la ejecución de este, y los demás scripts, se puede crear un botón en la parte inferior del programa, en donde se asigne el script que se requiera, para esto se debe dar clic derecho en la barra inferior y dar clic en “New Button”:

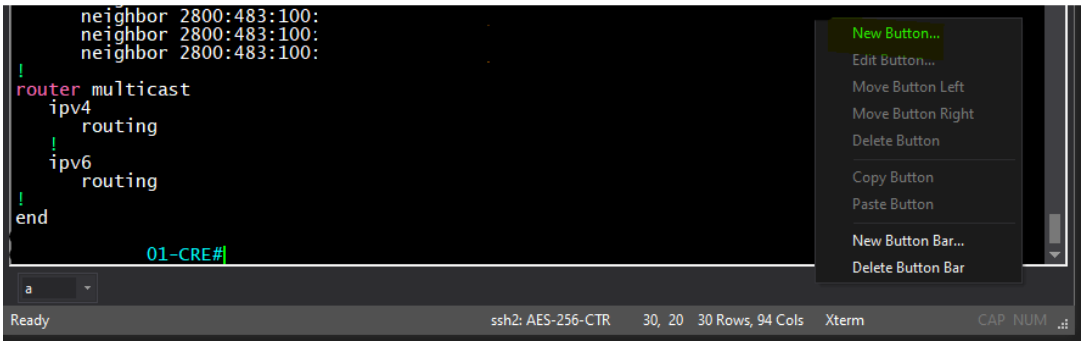


Figura 63. Creación botón para ejecución Script #1

Fuente: Elaboración Propia

Se abrirá la siguiente ventana, en donde se debe seleccionar en la parte de “Function” la opción “Run Script”:

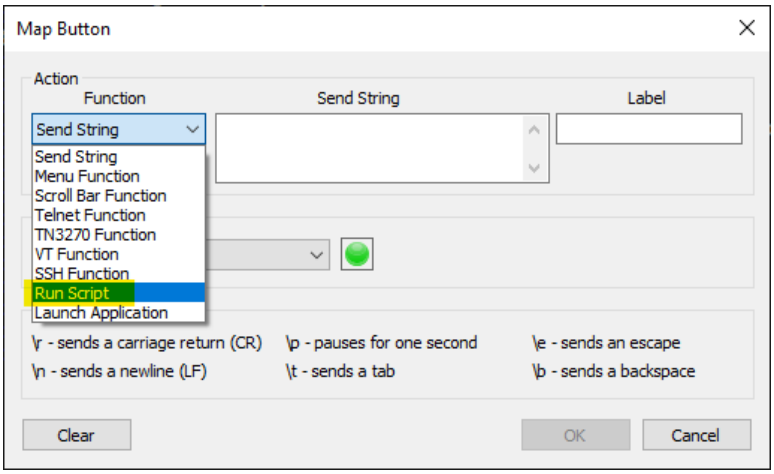


Figura 64. Creación botón para ejecución Script #2

Fuente: Elaboración Propia

Posteriormente, se debe dar clic en el botón con los tres puntos para buscar la ruta del script:

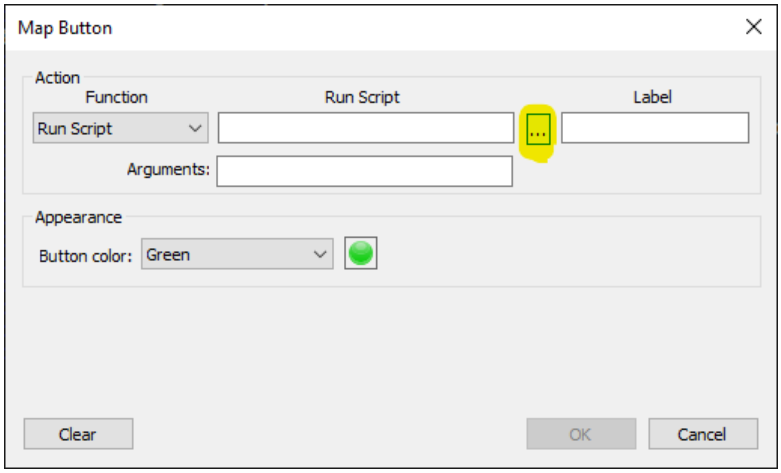


Figura 65. Creación botón para ejecución Script #3

Fuente: Elaboración Propia

El siguiente paso es desplazarse a través de las carpetas del equipo hasta encontrar el script que va a asignar al botón y seleccionarlo:

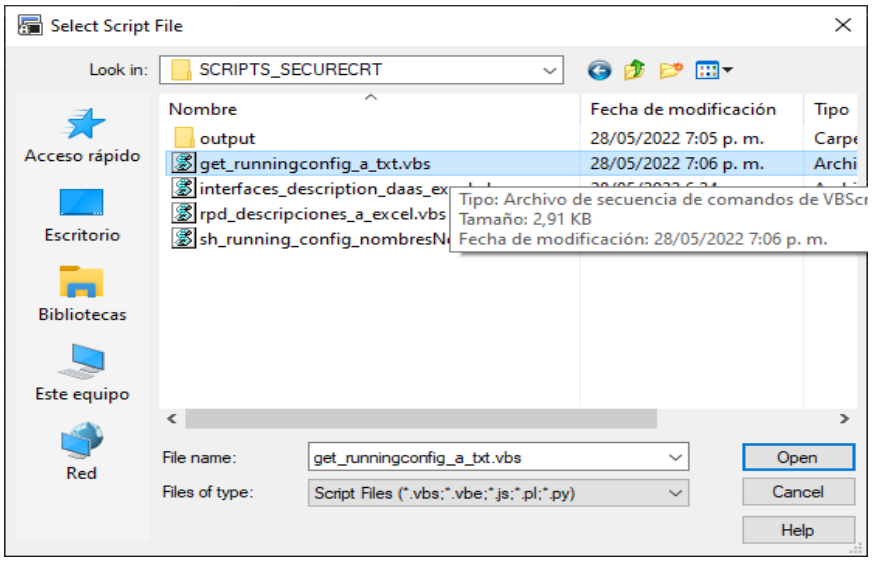


Figura 66. Creación botón para ejecución Script #4

Fuente: Elaboración Propia

Adicionalmente, es recomendable asignar un Label a cada botón que describa la función que realizará dicho script, luego dar click en OK:

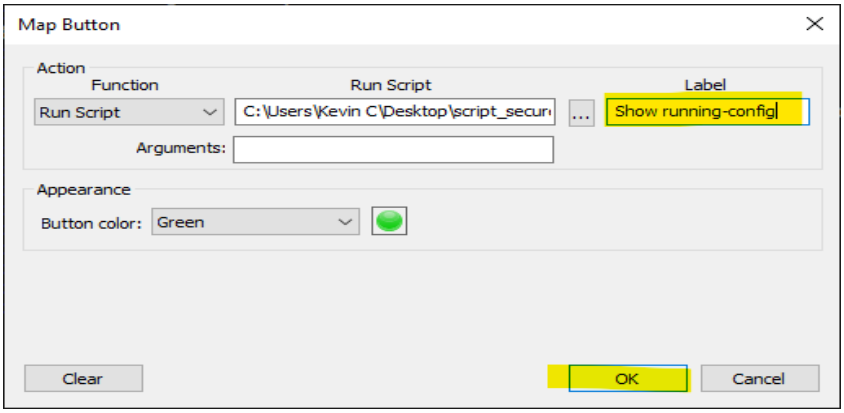


Figura 67. Creación botón para ejecución Script #5

Fuente: Elaboración Propia

El botón ya se encuentra en la parte inferior del programa, por lo que ya se puede ejecutar de manera más fácil:

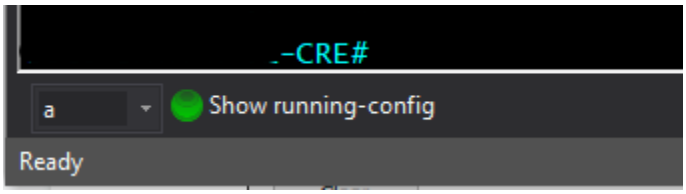


Figura 68. Creación botón para ejecución Script #6

Fuente: Elaboración Propia

7.3.2. SCRIPT INTERFACES_DESCRIPTION_DAAS_EXCEL.VBS

Este segundo script permite obtener la descripción de las interfaces del Switch de distribución en un archivo Excel, y cada parámetro se separa en una columna de Excel. Para realizar esta funcionalidad en primer lugar se asignan las constantes que definen el tipo de motor de script que SecureCRT debe utilizar, posteriormente se usan las dos funciones que se mostraron en el anterior script, para posteriormente usarlas en el nombre del archivo.

```
1   #$Language="VBScript"
2   #$Interface="1.0"
3
4 > Function obtenerNombreEquipo() ···
22  End Function
23
24 > Function obtenerFecha() ···
33  End Function
```

Figura 69. Código interfaces_description_daas_excel.vbs #1

Fuente: Elaboración Propia

A continuación, se crea la función *get_rpd_description_excel()*, que recibe como parámetro el nombre del archivo Excel que se generara, una vez dentro de la función, ésta genera el comando “show interfaces description” que mostrará el estado y la descripción de las interfaces del switch de distribución donde se ejecute dicho script.

A continuación, se crea una variable que contenga todos los valores esperados que indiquen que el comando ya terminó de ejecutarse, a continuación, se generan algunas otras variables, como las relacionadas a las líneas que se van generando a medida que se ejecuta el comando, y varios Array que contienen las posiciones y cantidad de caracteres que deben tomar desde dicho carácter.

También se genera una variable con la cantidad de posiciones del primer Array, para saber cuántas veces se tiene que acceder al archivo de Excel, es decir, que por cada parámetro que muestre el comando (en este caso son 4) se debe acceder a cada columna del archivo de Excel correspondiente. Finalmente, crea un objeto de archivo Excel y crea una hoja dentro del mismo.

```
35  Function get_rpd_description_excel(rutaArchivo)
36      crt.Screen.Synchronous = True
37      crt.Screen.Send "show interfaces description | no-more" & Chr(10)
38      crt.Screen.WaitForString Chr(10)
39
40      Dim stringsEspera
41      stringsEspera = Array( Chr(10), "DAAS>" )
42
43      Dim linea_actual, screenrow, readline
44      Dim posInicialDAAS, len_interfaces_descrip, cantCaracteres
45
46      posInicialDAAS = Array(1, 17, 23, 28)
47      cantCaracteres = Array(16, 5, 5, 55)
48      len_interfaces_descrip = UBound(posInicialDAAS)
49
50      Dim puerto ()
51      Dim admin_state ()
52      Dim link_state ()
53      Dim descripcion ()
54
55      Dim app, wb, ws
56      Set app = CreateObject("Excel.Application")
57      Set wb = app.Workbooks.Add
58      Set ws = wb.Worksheets(1)
```

Figura 70. Código interfaces_description_daas_excel.vbs #2

Fuente: Elaboración Propia

En esta parte del código se genera un While que recorrerá cada una de las líneas del archivo, y posteriormente validará si detecta el String “DAAS>”, ya que esto es el indicativo de que se terminó de ejecutar el comando y ya está el CLI de nuevo disponible, por lo que saldrá del While.

En caso de que no se encuentre dicho string en la línea actual, seguirá con la otra parte del código, en la cual se redimensionan varios Arrays, según la cantidad de líneas que se vayan generando, y luego se crea una función for que extraerá de la línea actual, la cantidad de caracteres necesaria para cada columna que se muestre en el comando y luego lo asignará al Array correspondiente.

```
60    linea_actual = 0
61    Do
62      While True
63        result = crt.Screen.WaitForStrings( stringsEspera )
64        screenrow = crt.screen.CurrentRow - 1
65        readline = crt.Screen.Get(screenrow, 1, screenrow, 150)
66        If result = 2 Then
67          Exit Do
68        End If
69
70        Redim Preserve puerto(linea_actual)
71        Redim Preserve admin_state(linea_actual)
72        Redim Preserve link_state(linea_actual)
73        Redim Preserve descripcion(linea_actual)
74
75        For i=0 To len_interfaces_descrip
76          extr = Mid(readline, posInicialDAAS(i), cantCaracteres(i))
77          extr = Trim(extr)
78          if i=0 Then
79            puerto(linea_actual) = extr
80          ElseIf i=1 Then
81            admin_state(linea_actual) = extr
82          ElseIf i=2 Then
83            link_state(linea_actual) = extr
84          ElseIf i=3 Then
85            descripcion(linea_actual) = extr
86          end if
87        Next
88        linea_actual = linea_actual + 1
89      Wend
90    Loop
```

Figura 71. Código interfaces_description_daas_excel.vbs #3

Fuente: Elaboración Propia

Finalmente, una vez se completa el proceso de llenado de los Arrays, se genera un nuevo for que irá recorriendo las casillas del archivo Excel e irá añadiendo la información correspondiente a cada columna, además de hacer que se autoajuste las columnas y hacer que el formato de las celdas sea de texto, para evitar que los puertos se muestran como fechas u horas.

```
92      for i_fila = 0 to linea_actual-1
93        for j_col=0 to len_interfaces_descrip
94          if j_col=0 Then
95            ws.Cells(i_fila+1, 1).Numberformat = "@"
96            ws.Cells(i_fila+1, 1).Value = puerto(i_fila)
97            ws.Cells(i_fila+1, 1).EntireColumn.AutoFit
98          ElseIf j_col=1 Then
99            ws.Cells(i_fila+1, 2).Numberformat = "@"
100           ws.Cells(i_fila+1, 2).Value = admin_state(i_fila)
101           ws.Cells(i_fila+1, 2).EntireColumn.AutoFit
102         ElseIf j_col=2 Then
103           ws.Cells(i_fila+1, 3).Numberformat = "@"
104           ws.Cells(i_fila+1, 3).Value = link_state(i_fila)
105           ws.Cells(i_fila+1, 3).EntireColumn.AutoFit
106         ElseIf j_col=3 Then
107           ws.Cells(i_fila+1, 4).Numberformat = "@"
108           ws.Cells(i_fila+1, 4).Value = descripcion(i_fila)
109           ws.Cells(i_fila+1, 4).EntireColumn.AutoFit
110         End if
111       Next
112     next
```

Figura 72. Código interfaces_description_daas_excel.vbs #4

Fuente: Elaboración Propia

Finalmente, se guardará el archivo de Excel con las celdas ya modificadas y se cerrará dicho archivo y se mostrará un mensaje que indique que ya se completó el proceso.

```
114     wb.SaveAs(rutaArchivo)
115     wb.Close
116     app.Quit
117     crt.screen.synchronous = false
118     MsgBox "Hecho!!"
```

Figura 73. Código interfaces_description_daas_excel.vbs #5

Fuente: Elaboración Propia

En la función *Main()* de este script se generarán algunas variables, se llamará a la función *obtenerNombreEquipo()* y *obtenerFecha()* y se generará la ruta del archivo, que posteriormente se pasará como parámetro en la función *get_rpd_description_excel()*

```
122  Sub Main()
123    Dim nom1, rut
124    nom1 = obtenerNombreEquipo()
125    rut = "C:\Users\Kevin C\Desktop\script_securecrt\SCRIPTS_SECURECRT\output\INT_DESCR_" & nom1 & "_" & obtenerFecha() & ".xlsx"
126    get_rpd_description_excel(rut)
127  End Sub
```

Figura 74. Código interfaces_description_daas_excel.vbs #6

Fuente: Elaboración Propia

A continuación, se aplica el script a una sesión previamente abierta de un switch de distribución, se ejecutará el comando correspondiente y se mostrará el resultado en el programa:

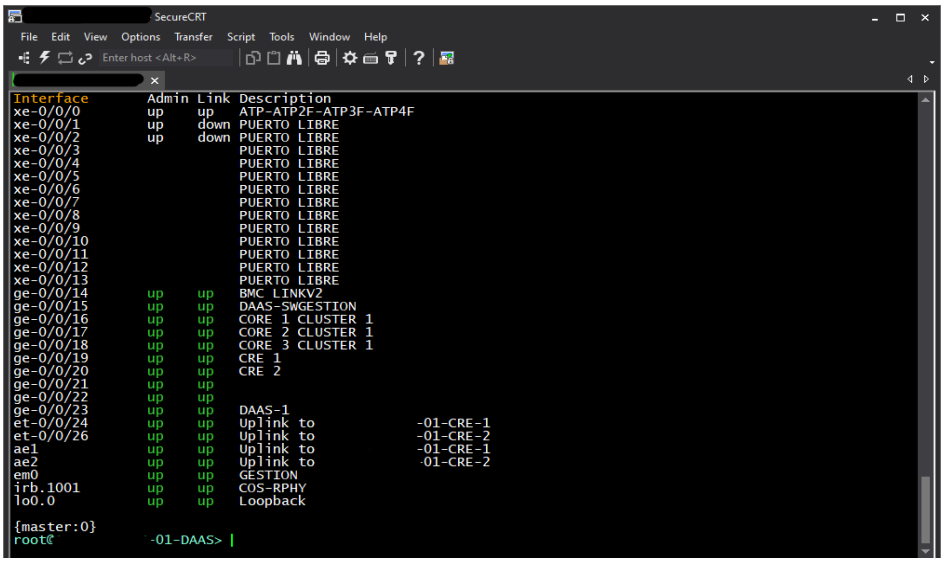


Figura 75. Ejecución script interfaces_description_daas_excel.vbs

Fuente: Elaboración Propia

En la carpeta “output” se generará un archivo Excel con el nombre del equipo y la fecha en la que se generó

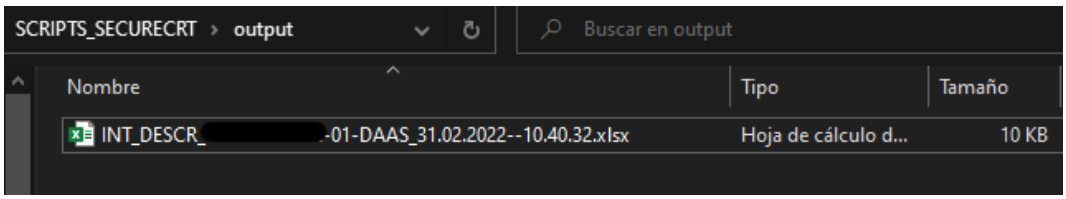
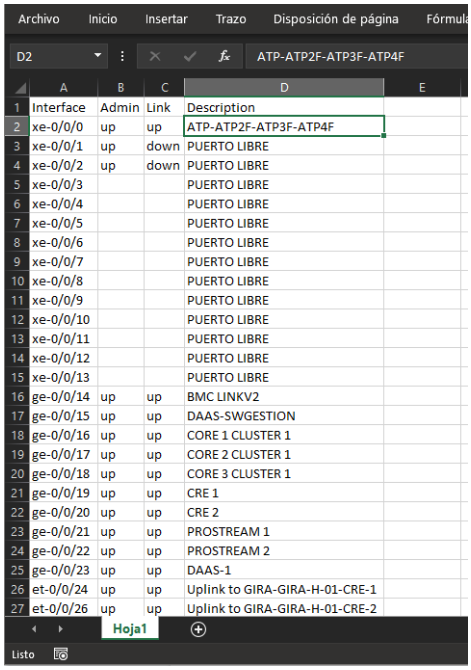


Figura 76. Capeta output con archivo generado - interfaces_description_daas_excel.vbs

Fuente: Elaboración Propia

Al abrir el archivo se puede ver la información del comando, con las cuatro columnas debidamente separadas y automáticamente alineadas



	A	B	C	D	E
1	Interface	Admin	Link	Description	
2	xe-0/0/0	up	up	ATP-ATP2F-ATP3F-ATP4F	
3	xe-0/0/1	up	down	PUERTO LIBRE	
4	xe-0/0/2	up	down	PUERTO LIBRE	
5	xe-0/0/3			PUERTO LIBRE	
6	xe-0/0/4			PUERTO LIBRE	
7	xe-0/0/5			PUERTO LIBRE	
8	xe-0/0/6			PUERTO LIBRE	
9	xe-0/0/7			PUERTO LIBRE	
10	xe-0/0/8			PUERTO LIBRE	
11	xe-0/0/9			PUERTO LIBRE	
12	xe-0/0/10			PUERTO LIBRE	
13	xe-0/0/11			PUERTO LIBRE	
14	xe-0/0/12			PUERTO LIBRE	
15	xe-0/0/13			PUERTO LIBRE	
16	ge-0/0/14	up	up	BMC LINKV2	
17	ge-0/0/15	up	up	DAAS-SWGESTION	
18	ge-0/0/16	up	up	CORE 1 CLUSTER 1	
19	ge-0/0/17	up	up	CORE 2 CLUSTER 1	
20	ge-0/0/18	up	up	CORE 3 CLUSTER 1	
21	ge-0/0/19	up	up	CRE 1	
22	ge-0/0/20	up	up	CRE 2	
23	ge-0/0/21	up	up	PROSTREAM 1	
24	ge-0/0/22	up	up	PROSTREAM 2	
25	ge-0/0/23	up	up	DAAS-1	
26	et-0/0/24	up	up	Uplink to GIRA-GIRA-H-01-CRE-1	
27	et-0/0/26	up	up	Uplink to GIRA-GIRA-H-01-CRE-2	

Figura 77. Archivo Excel generado - interfaces_description_daas_excel.vbs

Fuente: Elaboración Propia

7.3.3. SCRIPT RPD_DESCRIPCIONES_A_EXCEL.VBS

Para el tercer script, el primer paso es similar al del script anterior, definir el lenguaje de scripting que se va a usar y crear las funciones que obtienen la hora actual y el nombre del equipo

```
#$Language="VBScript"
#$Interface="1.0"

Function obtenerNombreEquipo() ...
End Function

Function obtenerFecha() ...
End Function
```

Figura 78. Código rpd_descripciones_a_excel.vbs #1

Fuente: Elaboración Propia

La función get_rpd_description_excel() aplica el comando “show cable rpd description” en la sesión activa, y similar al script anterior, se declaran unos Arrays de valores que va a separar de la línea que se muestra actualmente y así mismo genera una variable con la cantidad de posiciones a separar.

```
35  Function get_rpd_description_excel(rutaArchivo)
36      crt.Screen.Synchronous = True
37      crt.Screen.Send "show cable rpd description | nomore" & Chr(10)
38      crt.Screen.WaitForString Chr(10)
39
40      Dim stringsEspera
41      stringsEspera = Array( Chr(10), "COS>" )
42
43      Dim linea_actual, screenrow, readline
44      Dim posInicialCOS, len_var_rpd_descr, cantTamanoTexto
45
46      posInicialCOS = Array(1, 9, 24, 65, 96, 112)
47      cantTamanoTexto = Array(8, 14, 40, 30, 15, 10)
48      len_var_rpd_descr = UBound(posInicialCOS)
```

Figura 79. Código rpd_descripciones_a_excel.vbs #2

Fuente: Elaboración Propia

En esta parte, se generan Arrays adicionales con las variables que se muestran en el comando, con el fin de almacenarlas a medida que se muestran en la línea de comandos

```
50      Dim puerto ()
51      Dim macAddress ()
52      Dim descripcion ()
53      Dim swVersion ()
54      Dim rpd_status ()
55      Dim clockStatus ()
56
57      Dim app, wb, ws
58      Set app = CreateObject("Excel.Application")
59      Set wb = app.Workbooks.Add
60      Set ws = wb.Worksheets(1)
```

Figura 80. Código rpd_descripciones_a_excel.vbs #3

Fuente: Elaboración Propia

En esta porción de código se genera un ciclo While que irá recorriendo las líneas que se vayan mostrando y, valida que en cada línea no exista el string “COS>”, que indicaría que el comando ya se terminó de ejecutar, en caso de que esto suceda se sale inmediatamente del ciclo. En caso contrario, redimensiona los Arrays para que tengan el tamaño de la cantidad de líneas que se estén generando (mediante un contador).

```
62      linea_actual = 0
63      Do
64        While True
65          result = crt.Screen.WaitForStrings( stringsEspera )
66          screenrow = crt.screen.CurrentRow - 1
67          readline = crt.Screen.Get(screenrow, 1, screenrow, 150)
68          If result = 2 Then
69            Exit Do
70          End If
71
72          Redim Preserve puerto(linea_actual)
73          Redim Preserve macAddress(linea_actual)
74          Redim Preserve descripcion(linea_actual)
75          Redim Preserve swVersion(linea_actual)
76          Redim Preserve rpd_status(linea_actual)
77          Redim Preserve clockStatus(linea_actual)
```

Figura 81. Código rpd_descripciones_a_excel.vbs #4

Fuente: Elaboración Propia

A continuación, se crea un ciclo for que extraerá cada uno de los parámetros del comando y los almacenará en el Array correspondiente

```
79          For i=0 To len_var_rpd_descr
80            extr = Mid(readline, posInicialCOS(i), cantTamanoTexto(i))
81            extr = Trim(extr)
82            if i=0 Then
83              puerto(linea_actual) = extr
84            ElseIf i=1 Then
85              macAddress(linea_actual) = extr
86            ElseIf i=2 Then
87              descripcion(linea_actual) = extr
88            ElseIf i=3 Then
89              swVersion(linea_actual) = extr
90            ElseIf i=4 Then
91              rpd_status(linea_actual) = extr
92            ElseIf i=5 Then
93              clockStatus(linea_actual) = extr
94            end if
95          Next
96          linea_actual = linea_actual + 1
97        Wend
98      Loop
```

Figura 82. Código rpd_descripciones_a_excel.vbs #5

Fuente: Elaboración Propia

Finalmente, una vez que ya están almacenados los parámetros en los Array, se crea un for anidado que irá recorriendo las celdas del archivo Excel e irá formateando cada celda como Texto, agregando el valor correspondiente y autoajustando el tamaño de la columna.

```
100     for i_fila = 0 to linea_actual-1
101       for j_col=0 to len_var_rpd_descr
102         if j_col=0 Then
103           ws.Cells(i_fila+1, 1).Numberformat = "@"
104           ws.Cells(i_fila+1, 1).Value = puerto(i_fila)
105           ws.Cells(i_fila+1, 1).EntireColumn.AutoFit
106         ElseIf j_col=1 Then
107           ws.Cells(i_fila+1, 2).Numberformat = "@"
108           ws.Cells(i_fila+1, 2).Value = macAddress(i_fila)
109           ws.Cells(i_fila+1, 2).EntireColumn.AutoFit
110         ElseIf j_col=2 Then
111           ws.Cells(i_fila+1, 3).Numberformat = "@"
112           ws.Cells(i_fila+1, 3).Value = descripcion(i_fila)
113           ws.Cells(i_fila+1, 3).EntireColumn.AutoFit
114         ElseIf j_col=3 Then
115           ws.Cells(i_fila+1, 4).Numberformat = "@"
116           ws.Cells(i_fila+1, 4).Value = swVersion(i_fila)
117           ws.Cells(i_fila+1, 4).EntireColumn.AutoFit
118         ElseIf j_col=4 Then
119           ws.Cells(i_fila+1, 5).Numberformat = "@"
120           ws.Cells(i_fila+1, 5).Value = rpd_status(i_fila)
121           ws.Cells(i_fila+1, 5).EntireColumn.AutoFit
122         ElseIf j_col=5 Then
123           ws.Cells(i_fila+1, 6).Numberformat = "@"
124           ws.Cells(i_fila+1, 6).Value = clockStatus(i_fila)
125           ws.Cells(i_fila+1, 6).EntireColumn.AutoFit
126         end if
127       Next
128     next
```

Figura 83. Código rpd_descripciones_a_excel.vbs #6

Fuente: Elaboración Propia

Una vez que se finaliza el ciclo for, ya se puede cerrar el libro y se puede mostrar el mensaje que indique que ya se completó el proceso

```
130     wb.SaveAs(rutaArchivo)
131     wb.Close
132     app.Quit
133     crt.screen.synchronous = false
134     MsgBox "Hecho!!"
```

Figura 84. Código rpd_descripciones_a_excel.vbs #7

Fuente: Elaboración Propia

Finalmente, en la función Main() se llaman las funciones de fecha y nombre de equipo y posteriormente se ingresan los valores que devuelven en el nombre del archivo que se generará, y se pasa por parámetro la ruta del archivo a la función principal

```
138  Sub Main()
139    Dim nom1, rut
140    nom1 = obtenerNombreEquipo()
141    rut = "C:\Users\Kevin C\Desktop\script_securecrt\SCRIPTS_SECURECRT\output\LISTA_RPDS_" & nom1 & "_" & obtenerFecha() & ".xlsx"
142    get_rpd_description_excel(rut)
143  End Sub
```

Figura 85. Código rpd_descripciones_a_excel.vbs #8

Fuente: Elaboración Propia

Una vez se tiene la sesión de SecureCRT en uno de los vCMTS, se ejecuta el script, el cual mostrará el resultado del comando por consola y a su vez mostrará el mensaje que indique que ya se terminó el proceso

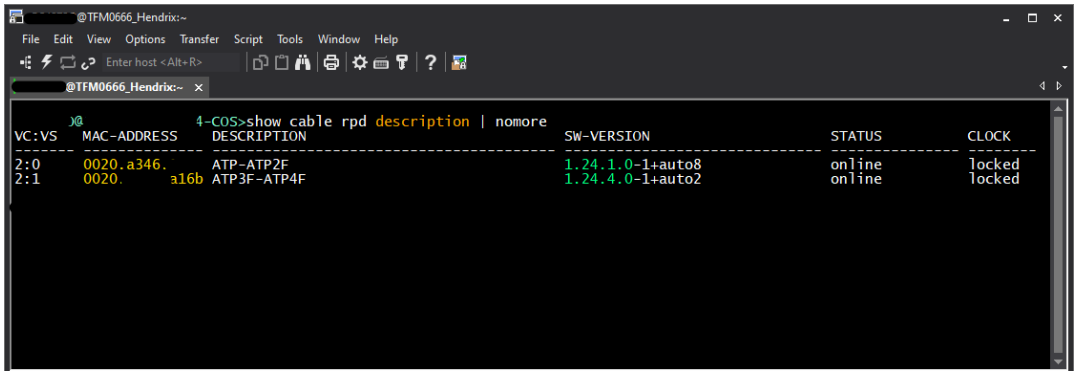


Figura 86. Ejecución script rpd_descripciones_a_excel.vbs

Fuente: Elaboración Propia

Una vez que se termina de ejecutar el script, se valida que en la carpeta output se genera un nuevo archivo, el cual corresponde con el nombre que se pasó por parámetro en la función *get_rpd_description_excel()*

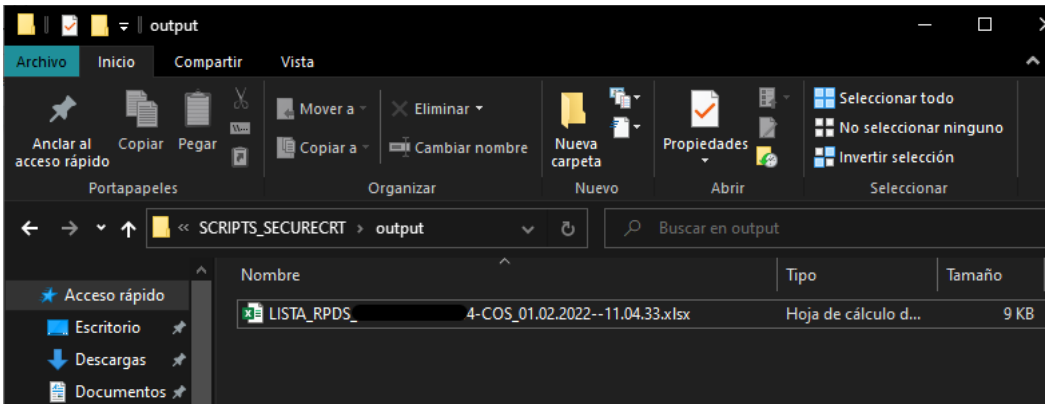


Figura 87. Archivo generado - rpd_descripciones_a_excel.vbs

Fuente: Elaboración Propia

Si se abre el archivo, se puede observar que genera correctamente la información del comando que se mostraba en el programa SecureCRT, y la información de las columnas está debidamente separada y auto alineada.

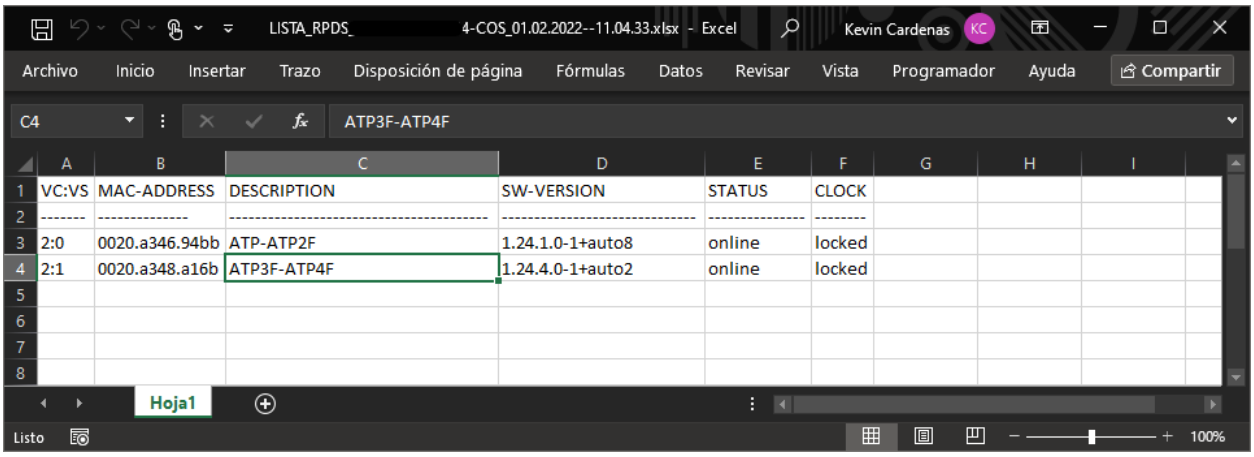


Figura 88. Archivo Excel generado - rpd_descripciones_a_excel.vbs

Fuente: Elaboración Propia

7.3.4. SCRIPT SH_RUNNING_CONFIG_NOMBRESNODOS.VBS

Este último script que se generó, también se inicia de la misma manera que los demás, es decir, con las funciones de *obtenerNombreEquipo()* y *obtenerFecha()*, además del identificador de lenguaje de programación y la interfaz que SecureCRT debe utilizar, ya que también se pueden definir otras, como JavaScript o Python 2.

```
1   #$Language="VBScript"
2   #$Interface="1.0"
3
4 > Function obtenerNombreEquipo() ···
22  End Function
23
24 > Function obtenerFecha() ···
33  End Function
```

Figura 89. Código sh_running_config_nombresNodos.vbs #1

Fuente: Elaboración Propia

En esta parte, al igual que los demás scripts, se ejecuta un comando que permite obtener parte de la configuración de un vCMTS, específicamente la parte del nombre completo de los puertos de cada RPD y así mismo, genera un Array con valores de espera, los cuales son el carácter Enter y el string “COS>”, además de crear el objeto de libro de Excel y añadirle una hoja.

```
35  Function get_cable_rf_port_description(ruta)
36    crt.Screen.Synchronous = True
37    crt.Screen.Send "show running-config cable us-rf-port description | exclude "& Chr(34)& Chr(33) & Chr(34) & Chr(10)
38    ' 33 ! - 34 "
39    crt.Screen.WaitForString Chr(10)
40
41    Dim stringsEspera
42    stringsEspera = Array( Chr(10), "COS>" )
43
44    Dim linea_actual, screenrow, readline
45
46    Dim app, wb, ws
47    Set app = CreateObject("Excel.Application")
48    Set wb = app.Workbooks.Add
49    Set ws = wb.Worksheets(1)
```

Figura 90. Código sh_running_config_nombresNodos.vbs #2

Fuente: Elaboración Propia

A continuación, se crea un While que recorre la salida del comando y verifica si contiene la cadena “COS>” en cuyo caso se saldrá del While, si no es así, posteriormente a eso va almacenando cada una de las líneas del comando directamente en el archivo Excel, y además va a formatear las celdas como texto y las va a alinear automáticamente,

```
51    linea_actual = 0
52    Do
53      While True
54        result = crt.Screen.WaitForStrings( stringsEspera )
55        screenrow = crt.screen.CurrentRow - 1
56        readline = crt.Screen.Get(screenrow, 1, screenrow, 150)
57        If result = 2 Then
58          Exit Do
59        End If
60        ws.Cells(linea_actual+1, 1).Numberformat = "@"
61        ws.Cells(linea_actual+1, 1).Value = readline
62        ws.Cells(linea_actual+1, 1).EntireColumn.AutoFit
63        linea_actual = linea_actual + 1
64      Wend
65    Loop
```

Figura 91. Código sh_running_config_nombresNodos.vbs #3

Fuente: Elaboración Propia

Luego de finalizar el ciclo, se guarda el archivo Excel y se cierra, además se mostrará un mensaje que indique que el proceso terminó

```
67      wb.SaveAs(ruta)
68      wb.Close
69      app.Quit
70      MsgBox "Hecho!!"
71
72      crt.Screen.Synchronous = False
73    End Function
```

Figura 92. Código sh_running_config_nombresNodos.vbs #4

Fuente: Elaboración Propia

En la función *Main()* de este script, se generan algunas variables que recibirán el parámetro generado por la función de fecha y la función de nombre de equipo, además de concatenar la ruta de archivo con el nombre del mismo, para luego pasar esta variable como parámetro en la función *get_cable_rf_port_description()*

```
75  Sub Main
76      Dim nombre_equipo, fecha_actual
77      Dim ruta_carpeta, nombre_archivo, rutacomp
78      Dim arr_cmd, len
79      Dim func_aggLinea
80
81      nombre_equipo = obtenerNombreEquipo()
82      fecha_actual = obtenerFecha()
83      ruta_carpeta = "C:\Users\Kevin C\Desktop\script_securecrt\SCRIPTS_SECURECRT\output\"
84      nombre_archivo = "cable us-rf-port description - " & nombre_equipo & " - " & fecha_actual & ".xlsx"
85      rutacomp = ruta_carpeta & nombre_archivo
86      get_cable_rf_port_description(rutacomp)
87  End Sub
```

Figura 93. Código sh_running_config_nombresNodos.vbs #5

Fuente: Elaboración Propia

Una vez se ejecuta este script en el CLI del vCMTS, se obtiene los nombres de los puertos de cada nodo, y esta información se guarda en el Excel, cuando se termina dicho proceso se ejecuta la parte final de la función que indica con un MsgBox que el proceso concluyó

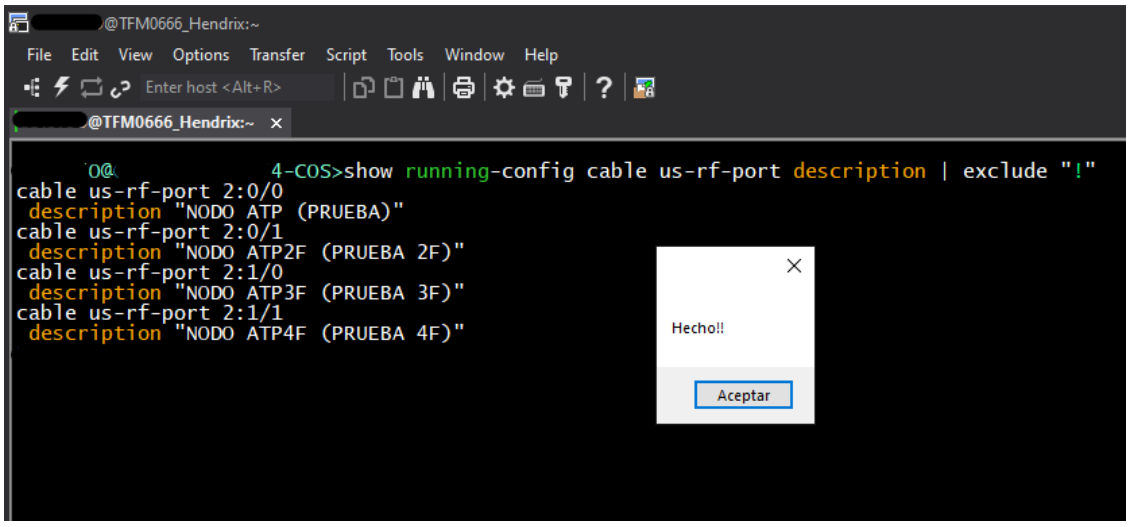


Figura 94. Ejecución script sh_running_config_nombresNodos.vbs

Fuente: Elaboración Propia

Se valida nuevamente en la carpeta output el archivo generado con las marcaciones de los nodos que se encuentran en producción, pero en este caso se muestra únicamente el nodo que se usa para pruebas ATP debido a temas de confidencialidad

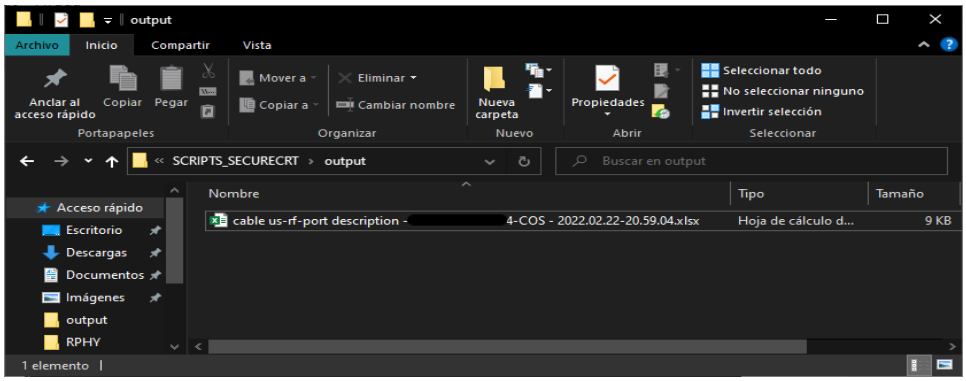


Figura 95. Salida ejecución script sh_running_config_nombresNodos.vbs

Fuente: Elaboración Propia

En el archivo Excel se puede validar que el resultado del comando está escrito en la columna A de la Hoja 1 correctamente.

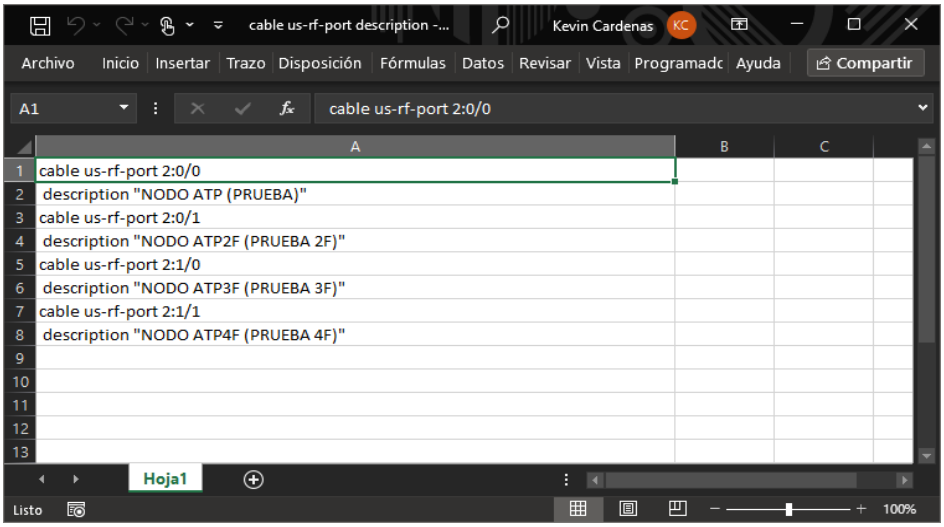


Figura 96. Archivo Excel generado - sh_running_config_nombresNodos.vbs

Fuente: Elaboración Propia

8. RESULTADOS

Los principales resultados que se obtuvieron son los siguientes:

- El principal resultado que se obtuvo fue la reducción del tiempo necesario para ejecutar algunas actividades en el proceso de segmentaciones de Remote PHY, como se muestra a continuación:

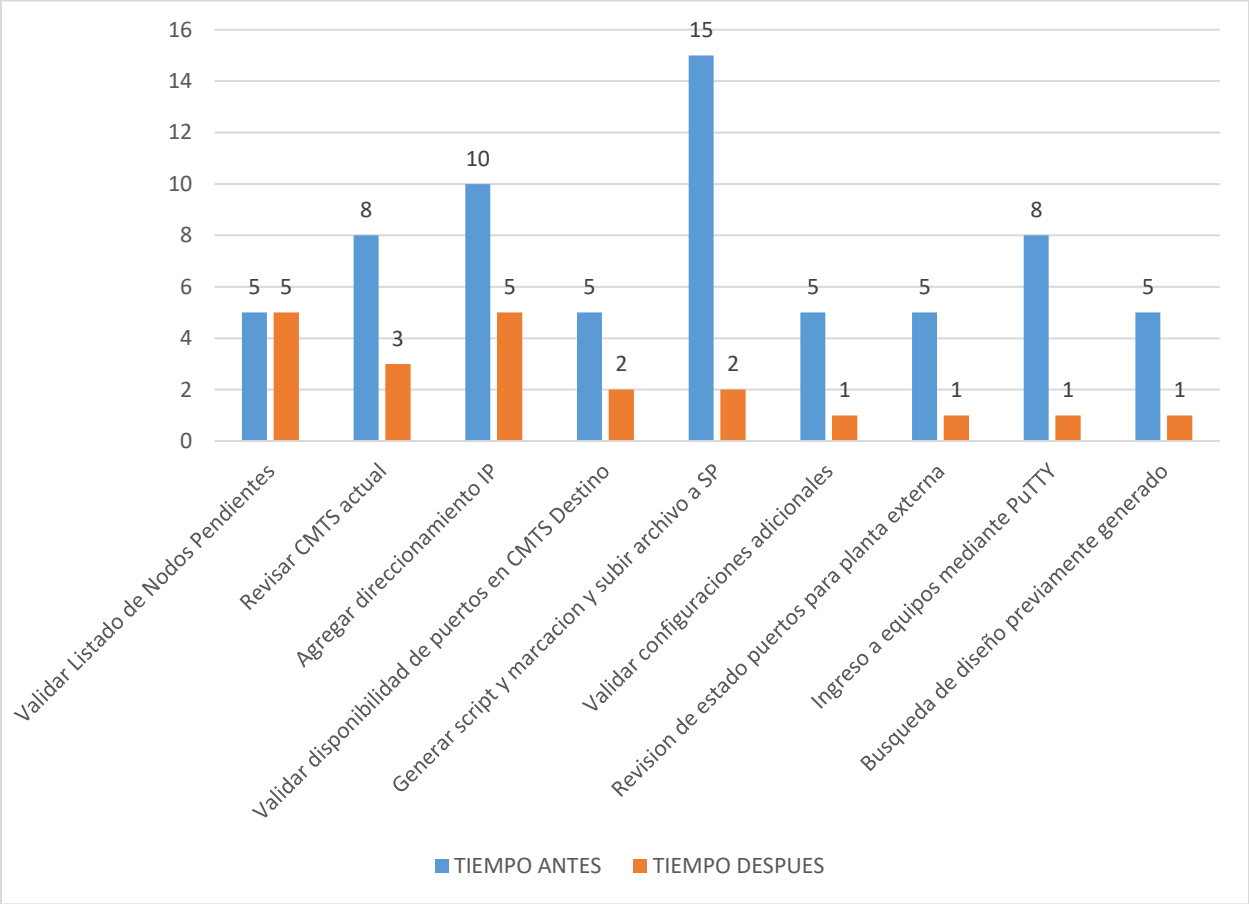


Figura 97. Tiempos antes vs Tiempos Después de desarrollo de automatizaciones

Fuente: Elaboración Propia

- Gracias al cambio del cliente SSH PuTTY por el cliente SecureCRT se puede mejorar la manera en la que se accede a los equipos de Remote PHY, ya que esto se realiza de manera automática y además permite guardar algunos comandos que se usan con frecuencia en botones que se encuentran ubicados en la parte inferior del programa.
- De igual manera, debido al desarrollo del archivo de IPs RPHY.txt que se encuentra directamente en el servidor Linux Bash, se puede obtener las IPs de

manera más fácil, utilizando simplemente un comando de filtrado, algo especialmente útil para aquellos que no requieran acceder tan frecuentemente a los equipos, por ejemplo, aquellos que necesiten las IPs para agregarlas a algún sistema de gestión de la compañía, o también para el caso de los que prefieran seguir usando el cliente PuTTY.

- Así mismo, se mejoró la manera en la que el grupo de segmentaciones y la operación en general pueden acceder a los diseños y scripts de Remote PHY, separándolos por carpeta, ya que anteriormente todos los archivos se encontraban en una carpeta correspondiente al sitio donde se encuentra el vCCAP, lo cual hacía que buscar un diseño específico fuera una tarea engorrosa.
- El script de Python permite generar los diseños y scripts de manera más fácil y rápida, ya que anteriormente era necesario crear dichos archivos manualmente, una tarea que resulta desgastante, especialmente cuando dichos diseños son de equipos diferentes y los puertos en estos equipos no son consecutivos. Gracias al script se redujo el tiempo de entrega de dichos documentos de 3 días a 1 día.
- Los scripts de VBA que permiten obtener las descripciones del lado del vCMTS y Switch de distribución son especialmente útiles para que el área de segmentaciones pueda tener un status de los puertos que se encuentran libres y cuáles están ocupados, ya que anteriormente esta tarea se debía hacer de manera manual, a menudo mediante pantallazos o mediante archivos de Excel.
- El script que obtiene el nombre de los puertos de cada nodo es especialmente útil para el área de gestión, ya que permite obtener automáticamente el nombre de los nodos que ya están en producción, es decir, con clientes, ya que dichos equipos también deben agregarse a los sistemas de gestión de la compañía.
- El último script, que toma toda la configuración del equipo, permite obtener dicha configuración de manera automática, y al ser un comando casi estándar (excepto por los switches Juniper que usan un comando distinto), este script se puede aplicar en una buena cantidad de equipos, algo que es necesario en caso de requerir hacer comparaciones entre las configuraciones de varios equipos para validar configuraciones, etc.

9. RECOMENDACIONES

Algunas de las recomendaciones pertinentes son:

- Es importante que se haga un análisis más exhaustivo de los procesos que se pueden automatizar, ya sea mediante Python o mediante otras herramientas
- Los equipos que se usen para estas tareas deben tener la suficiente potencia computacional y de velocidad de almacenamiento para poder cumplir con los requerimientos de las automatizaciones
- Es necesario hacer una revisión general del proceso de implementación y del proceso de segmentaciones, para optimizar dichos procesos y evitar reprocesos
- Adicional, es importante que se tengan todos los prerrequisitos al día antes de comenzar con la implementación de un nuevo sitio (capacidad de tráfico, espacio, refrigeración y energía)
- A nivel de los scripts, es posible mejorar aún más la automatización de la generación de los scripts, como, por ejemplo, generando un script que acceda a todos los equipos para obtener un comando o realizar una búsqueda específica.
- Así mismo, en general, es necesario mantener un orden en todo el proceso, por ejemplo, a nivel de los diseños, de los direccionamientos asignados a cada equipo, entre otros, ya que este proyecto tiene un margen de crecimiento con el tiempo, especialmente teniendo en cuenta la necesidad que se tiene de mejorar la calidad de servicio de los clientes.
- Una buena alternativa para el desarrollo del proyecto, es el uso de metodologías ágiles, ya que actualmente este proyecto se maneja de manera tradicional, y usualmente en las reuniones de seguimiento se hace un barrido completo de los pendientes, usando tiempo que puede ser vital para desarrollar otras tareas

10. CONCLUSIONES

- En general, la arquitectura Remote PHY provee varios beneficios que hacen que sea viable su implementación vs. la arquitectura HFC tradicional, especialmente en estos casos en los que no es viable hacer una transición de HFC a FTTH, ya que permite mantener parte de la infraestructura y además ofrece la gran ventaja de ofrecer más velocidad y calidad por menos Capex y Opex, especialmente en lo que a energía y espacio se refiere, ya que un vCMTS ocupa mucho menos que un CMTS tradicional, y el consumo de energía es inferior..
- Falta mejorar un poco más la automatización en los procesos internos de la compañía, ya que todavía hay tareas repetitivas que se podrían automatizar de una manera u otra, y que permitan aprovechar mejor el tiempo. Esto es complicado teniendo en cuenta la cantidad de áreas dentro de la compañía
- Se logró demostrar que a través de scripting se puede automatizar tareas del día a día, haciendo dichas tareas mucho más fácil, e incluso haciendo que los tiempos de entrega de estas actividades se minimicen, un buen ejemplo puede ser la generación de los diseños y scripts que antes tomaban aproximadamente tres días a un solo día, evitando posibles errores en cuanto a la marcación.
- En términos generales, Python es un lenguaje de programación más amigable y con curva de aprendizaje más rápida que VBA, y por lo tanto es más recomendable usar Python en tareas de automatización, por lo que puede permitir programar tareas más complejas de manera más ágil.
- La implementación de esta arquitectura representa algunos retos para la compañía en general, ya que al distribuir parte de algo que era centralizado, se va a requerir más apoyo de parte de planta externa
- Debido a alcance limitado que se tuvo, al ser proveedor externo de Claro, no fue posible realizar todas las soluciones planteadas, principalmente debido a temas de confidencialidad de la compañía
- Se logro reducir el tiempo necesario para desarrollar las actividades de generación de diseños y script de los nodos a segmentar, de ½ hora por cada nodo a solo 5 minutos, lo cual es una mejora notable
- Adicionalmente, otras partes del proceso de implementación se pueden automatizar mediante scripts, como se aplicó en los scripts de obtener la salida del comando “show running-config” y “show running-config cable us-rf-port description”, los cuales como tal no son requeridos de manera constante sino que en casos puntuales ayudan a recopilar información de manera más rápida.

11. BIBLIOGRAFÍA

1. Chapman, J. T. (2013). DOCSIS Remote Phy. *White Paper, Society of Cable Telecommunications Engineers*.
2. Romero Barrantes, Y. (2020). Diseño de Remote PHY para la comercialización de servicios de Internet residencial por parte de Grupo ICE, para octubre 2020.
3. Espinosa Villarreal, M. G. (2008). Análisis de la integración de redes HFC con *tecnologías inalámbricas, bajo el estándar DOCSIS* (Bachelor's thesis, QUITO/EPN/2008).
4. Quizhpi Mejía, M. G. (2017). Estudio de factibilidad técnico económico para que operadores de servicios mediante cable modems opten por migrar al estándar docsis 3.1 (Master's thesis, Espol).
5. Alvarez, Michael. (2020) SecureCRT para Ingenieros de Redes - Coding Networks Blog. Retrieved April 18, 2022, from https://codingnetworks.blog/es/securecrt-para-ingenieros-de-redes/
6. VIAVI Solutions. Arquitectura de acceso distribuido (DAA). Desafíos y ventajas de la arquitectura DAA. Retrieved April 18, 2022, from https://www.viavisolutions.com/es-es/arquitectura-de-acceso-distribuido
7. Migelle. (2022) Tipos de SFP y diferencias entre SFP, SFP+, QSFP, SFP28 y QSFP28 | FS comunidad. Retrieved April 19, 2022, from https://community.fs.com/es/blog/sfp-vs-sfp-vs-sf-p28-vs-qsfp-vs-qsf-p28-what-are-the-differences.html
8. Pascal, Jorge. (2018) DOCSIS 3.1 vs. FTTH: Is That the Real Question? - Intraway Blog | Think Incredible. Retrieved April 19, 2022, from https://www.intraway.com/blog/DOCSIS-FTTH-Is-That-the-Real-Question
9. Arquitecturas de acceso distribuido. (2021, September 16). Viavisolutions.com. https://www.viavisolutions.com/es-mx/arquitectura-de-acceso-distribuido
10. Distributed Access Architecture. (2021, March 3). CableLabs. https://www.cablelabs.com/technologies/distributed-access-architecture
11. DOCSIS 3.1. (2022, January 14). Viavisolutions.com. https://www.viavisolutions.com/es-es/docsis-31
12. Driving 5G and HFC Convergence with Multi-Tenancy at the Edge. (2022, May 5). CableLabs. https://www.cablelabs.com/blog/multi-tenancy-at-the-edge
13. García Bish, J. R. (2019). EVOLUCIÓN DE LA TECNOLOGÍA HFC HACIA ARQUITECTURAS DISTRIBUIDAS. https://encuentrosregionales.com/2019/descargas/EncRegTel2019-EvoluciondeHFC.pdf
14. Inducción a las Redes HFC. (2014, February 13). Monografias.com. https://www.monografias.com/trabajos99/induccion-redes-hfc/induccion-redes-hfc

15. La historia de DOCSIS - Intraway Blog | Think Incredible. (2020, October 28). Intraway Blog | Think Incredible. https://www.intraway.com/blog/la-historia-de-docsis/
16. Redes HFC – Entre Redes y Servidores. (2009, December 10). Entre Redes Y Servidores; Entre Redes y Servidores. https://alexalvarez0310.wordpress.com/category/redes-hfc/
17. Software para Monitoreo, Gestión y Pruebas. (2022). Promptlink.com. https://www.promptlink.com/technology/docsis/index.esp.html#:~:text=Las%20Especificaciones%20de%20Interfaz%20de,para%20la%20industria%20del%20cable.

12. ANEXOS

14.1. PROCESO DE INSTALACIÓN DE SECURECRT

- Acceder a la página de descargas de SecureCRT https://www.vandyke.com/products/securecrt/, y dar clic en Download:

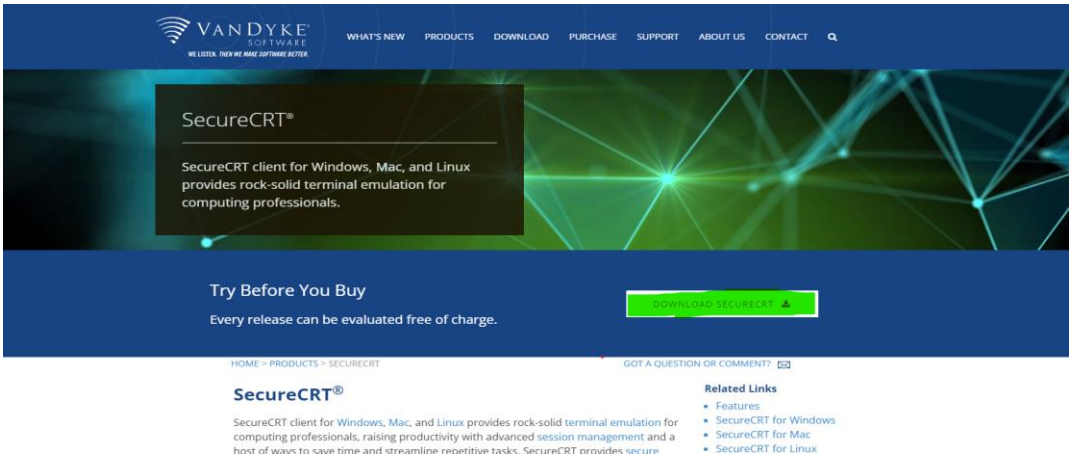


Figura 98. Pagina inicial SecureCRT

Fuente: Elaboración Propia

- Seleccionar la versión de 64 bits:

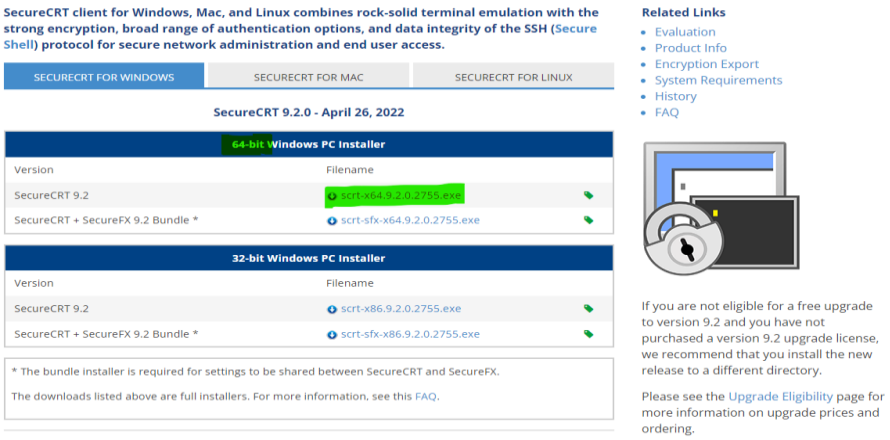


Figura 99. Selección versión SecureCRT

Fuente: Elaboración Propia

- Crear una cuenta nueva, en caso de que no la tenga:

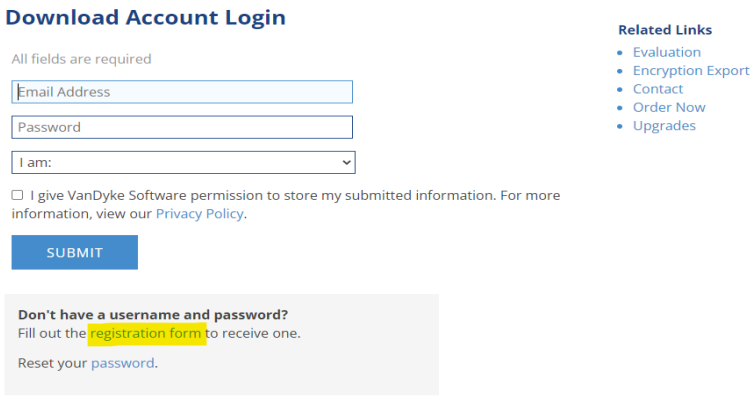


Figura 100. Registro para descargar el programa

Fuente: Elaboración Propia

- Diligenciar los campos solicitados:

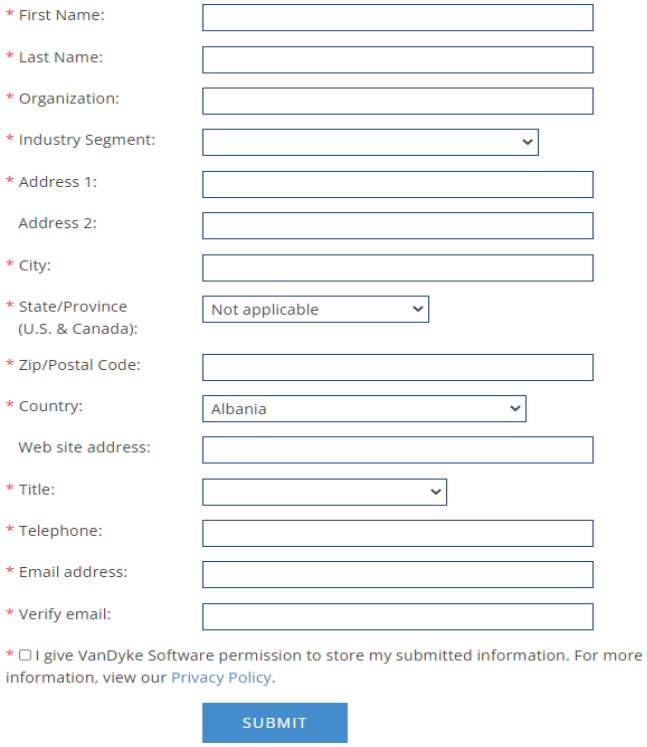


Figura 101. Formulario de registro

Fuente: Elaboración Propia

- Una vez registrado, al correo llegará el enlace de descarga, el cual abrirá la siguiente página e iniciará la descarga:

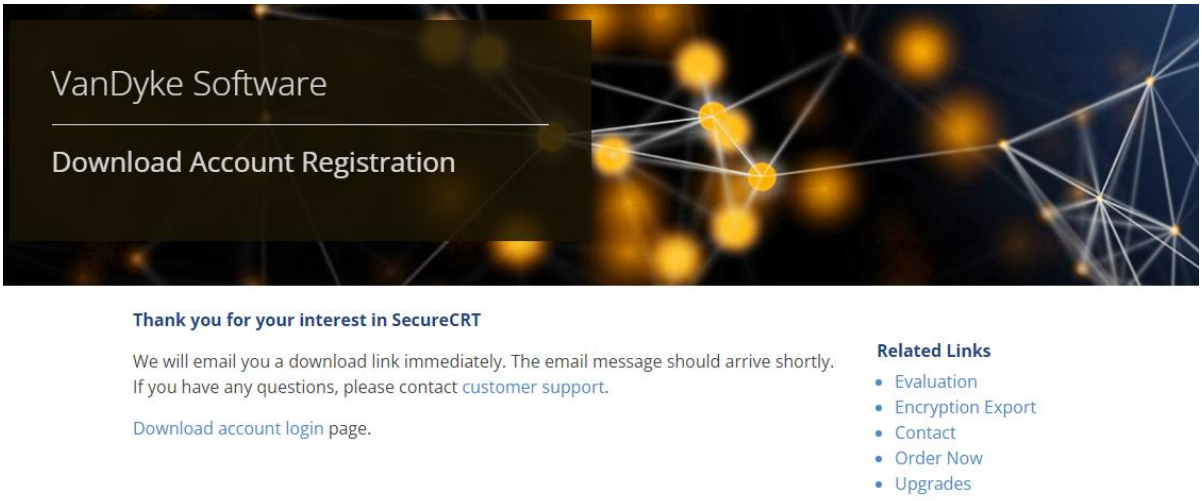


Figura 102. Página de confirmación registro

Fuente: Elaboración Propia

- Una vez descargado, abrir el archivo:

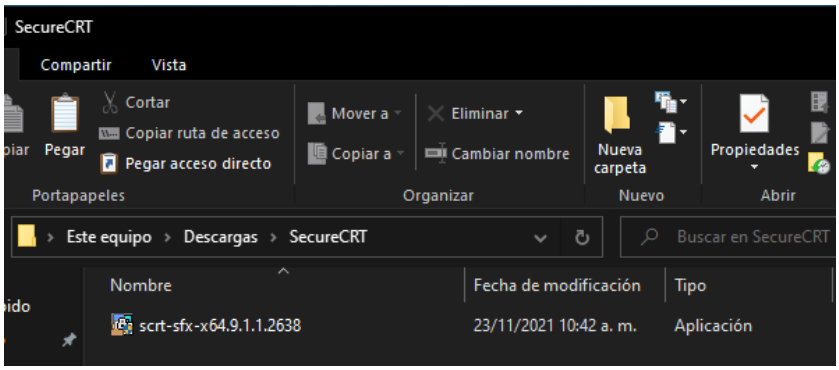


Figura 103. Archivo descargado

Fuente: Elaboración Propia

- Clic en Next:

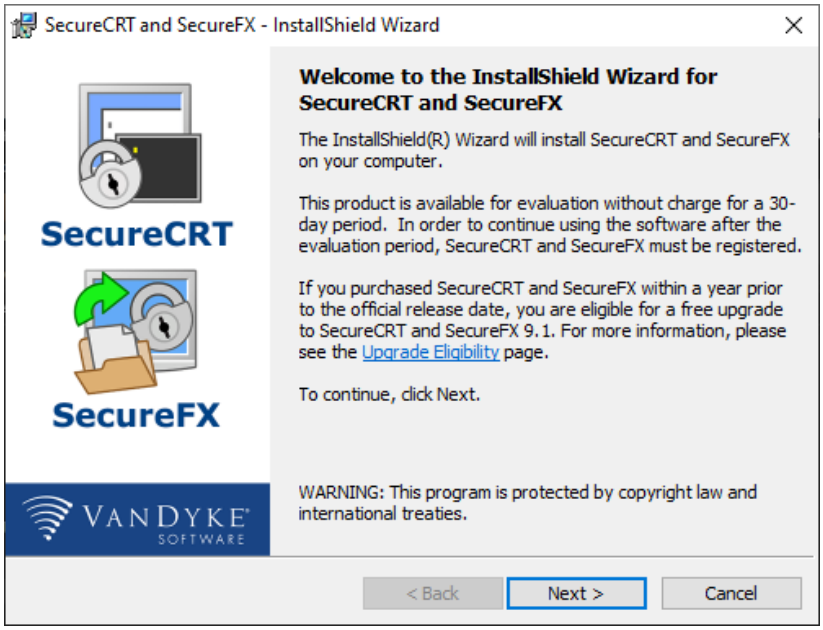


Figura 104. Instalación SecureCRT #1

Fuente: Elaboración Propia

- Aceptar el acuerdo de licencia, luego dar clic en Next:

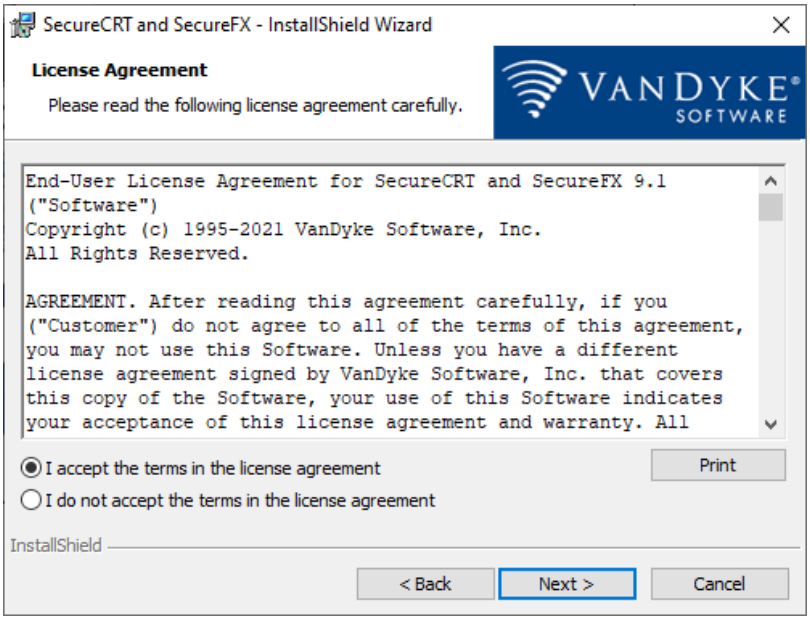


Figura 105. Instalación SecureCRT #2

Fuente: Elaboración Propia

- Elegir la primera opción, para que la instalación sea sobre todos los usuarios del equipo:

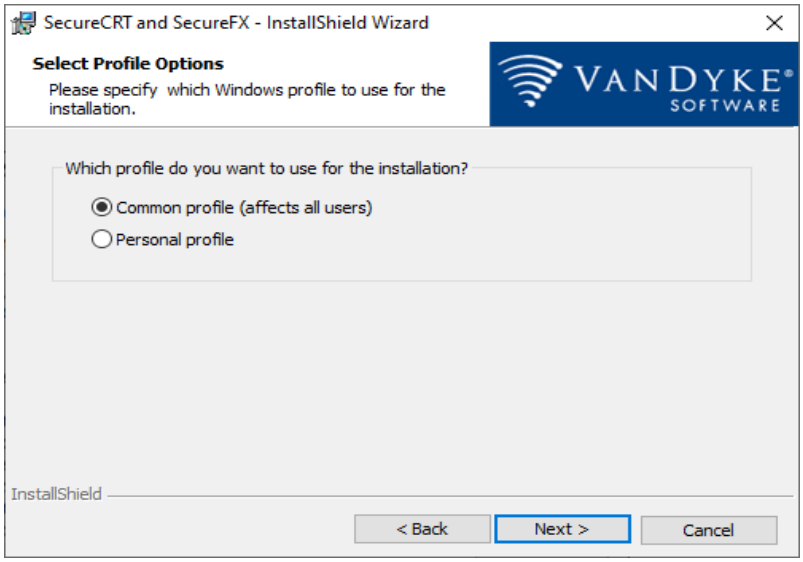


Figura 106. Instalación SecureCRT #3

Fuente: Elaboración Propia

- Seleccionar instalación completa:

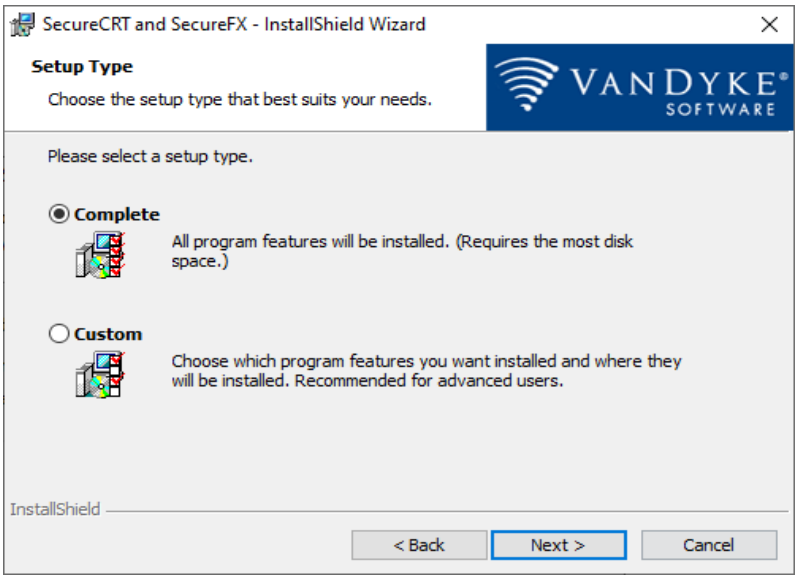


Figura 107. Instalación SecureCRT #4

Fuente: Elaboración Propia

- Dejar las opciones por defecto:

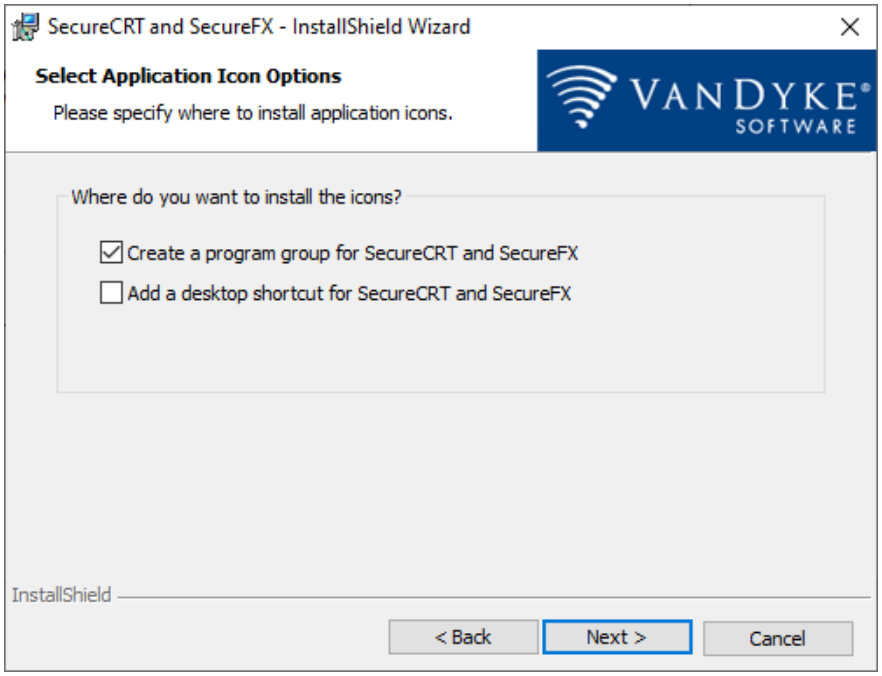


Figura 108. Instalación SecureCRT #5

Fuente: Elaboración Propia

- Clic en Install y esperar a que la instalación se complete:

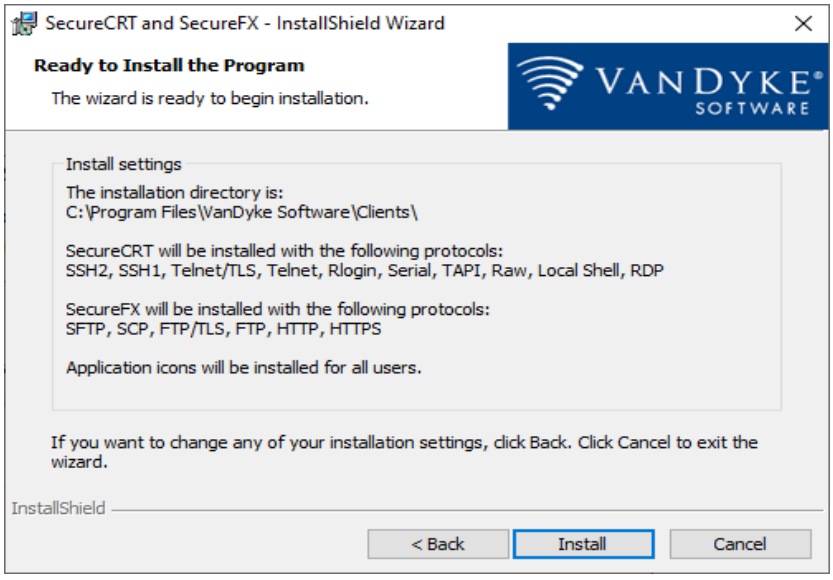


Figura 109. Instalación SecureCRT #6

Fuente: Elaboración Propia

- Clic en Finish:

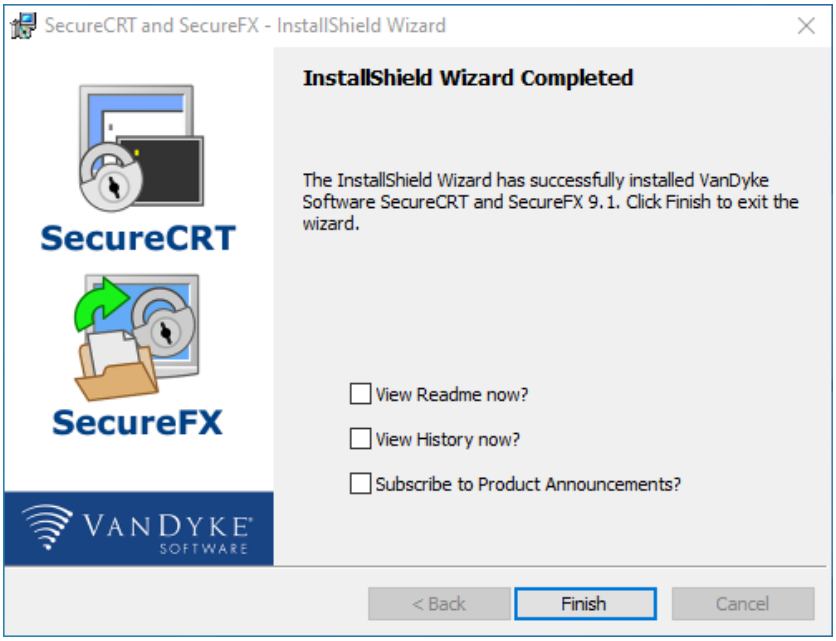


Figura 110. Instalación SecureCRT #7

Fuente: Elaboración Propia

- Buscar el programa SecureCRT en Windows:

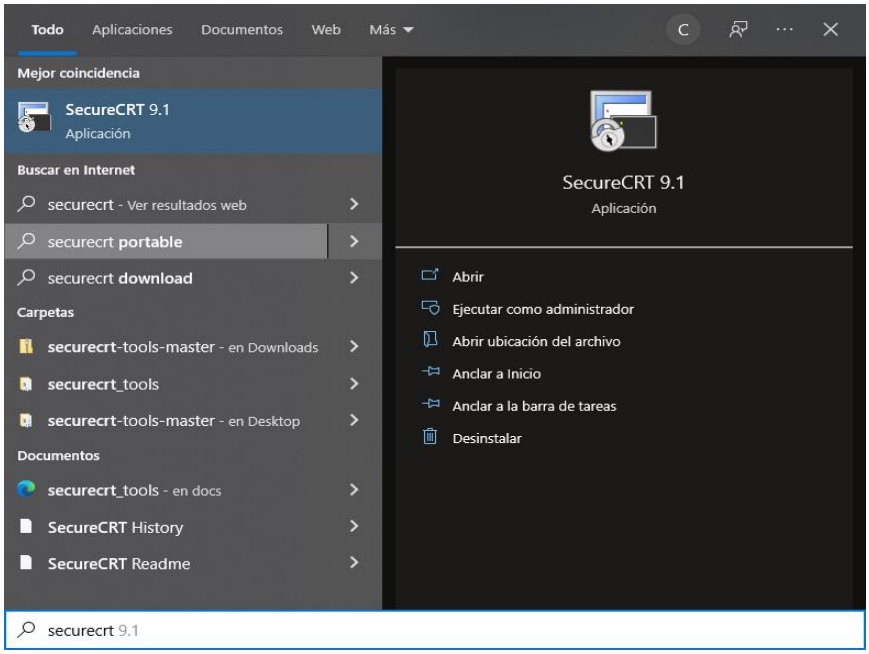


Figura 111. Búsqueda programa en Menú Inicio

Fuente: Elaboración Propia

- Dejar la opción por defecto, este aviso solo saldrá la primera vez que se abra el programa:

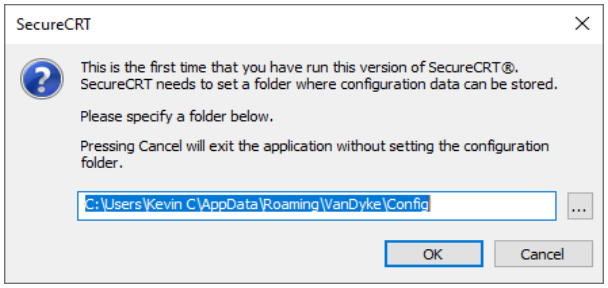


Figura 112. Selección carpeta de configuración personal

Fuente: Elaboración Propia

- Una vez abierto el programa, ya se puede conectar a la sesión SSH que requiera:

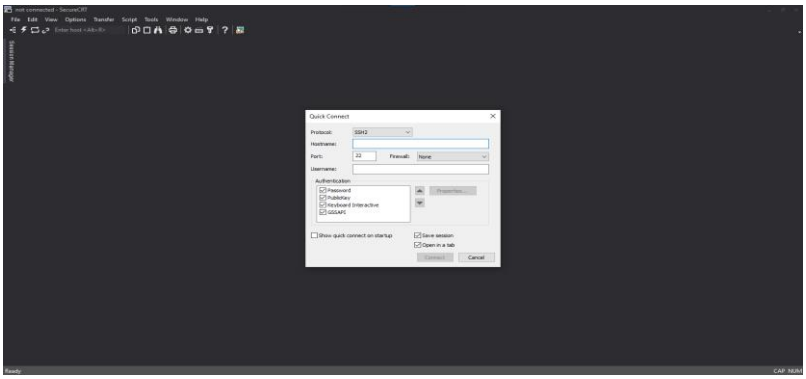


Figura 113. Ventana principal SecureCRT

Fuente: Elaboración Propia

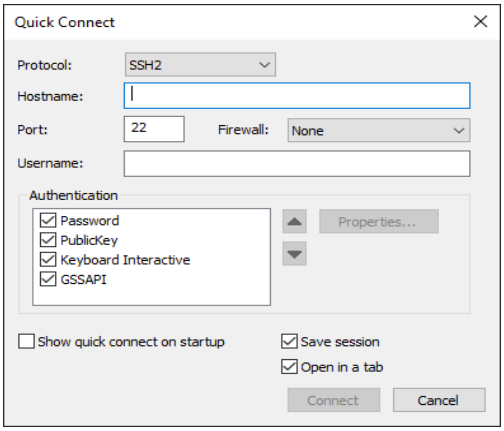


Figura 114. Ventana nueva sesión SecureCRT

Fuente: Elaboración Propia