TECH BRIEF

NETWORK SECURITY WITH WYRESTORM'S NETWORKHD

NETWORK SECURITY WITH WYRESTORM'S NETWORKHD

In today's world, the majority of ProAV devices, whether it be an HDBaseT matrix, presentation switcher or an AV over IP solution, exist on the network. While it may be easy to connect a device via an ethernet cable to a switch or router, it's critical to understand how that device is capable of being secure.

At WyreStorm, we understand that security is paramount and take it very seriously. We want you and your customers to feel confident that when a WyreStorm product is on the network, it will not easily fall prey to malicious activity. To this end, we have made significant improvements and changes to our key technologies.

Our NetworkHD controller supports SSH protocol for both internal and external APIs. Not only does this encrypted key prevent unauthorized access, but you can also change the SSH communication port from the default 22 for more flexible control.

The NetworkHD controller also uses HTTPS (Transport Layer Security) to encrypt data shared between the NHD-000-CTL/ NHD-CTL-PRO and client devices like computers using PKI (Public Key Infrastructure).

Furthermore, we have implemented security on our encoders and decoders in the form of SSH and HTTPS which ensure secure communication with the NHD-000-CTL/ NHD-CTL-PRO.

Additionally, NetworkHD supports encryption of audio and video streams over the network using AES 128-bit encryption methods, so only intended viewers can view the content. Lastly, 802.1x ensures that only authenticated devices can exist on a network based on credentials and policies using Active Directory/LDAP.

At WyreStorm, we value your peace of mind when it comes to security and strive to make sure our products and technologies are always up to date with industry standards.

SECURITY PROTOCOLS

Certain communication protocols can be enabled or disabled on individual NetworkHD components. By default, all series of products use Telnet and HTTP for various aspects of the operation. SSH and TLS (SSL) protocols can be enabled in addition or in lieu of Telnet and HTTP to provide encrypted communication methods.

Enabling SSH/TLS on the NHD-CTL provides the following:

- Encrypted internal API communication between TX and RX devices.
- Encrypted external API communication between CTL and 3rd-party control system.
- Encrypted access of the web interface
- Prevents unauthorized access to root privileges by requiring authenticated credentials.

Enabling SSH/TLS on encoders/decoders provides the following:

- Encrypted internal API communication to CTL
- Encrypted access to web servers (preview streams)
- Prevents unauthorized access to root privileges by requiring authenticated credentials.

In addition to the above-mentioned security protocols, NetworkHD encoders and decoders utilize AES 128-bit encryption of multicast audio and video streams over the network.

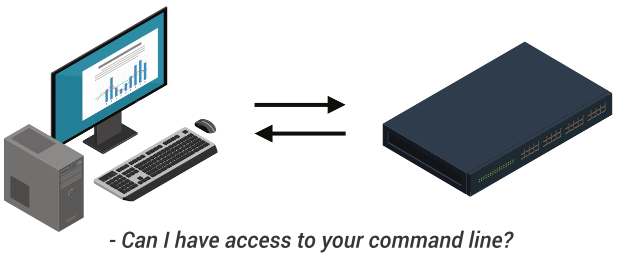
TELNET- PORT 23 (TERMINAL EMULATION)

Telnet is one of the oldest Internet standards, developed way back in 1969! It's the ultimate chameleon of protocols, specializing in terminal emulation. It allows users on a remote client machine, the Telnet client, to access the resources of another machine, the Telnet server, and gain access to a command-line interface.

Telnet does this by making the client machine look like a terminal directly connected to the local network. This projection is actually a software image, a virtual terminal that can interact with the chosen remote host.

Unfortunately, there are no encryption techniques available within the Telnet protocol, so everything must be sent in plain text, including passwords!

Check out the example below to see a Telnet client trying to connect to a Telnet server.



- Can I have access to your command line?
- Okay... Go ahead, configure me
- I will send everything in clear text with passwords,I don't do encryption

These text-mode emulated terminals are capable of executing pre-defined procedures, such as displaying menus that allow users to select options and gain access to applications on the duplicated server. A Telnet session is initiated by running the Telnet client software and then logging into the Telnet server. Telnet utilizes an 8-bit, byte-oriented data connection over TCP, which makes it highly efficient. Despite the fact that all data is transmitted in plain text, it is still used today due to its simplicity and low overhead; however, it is not recommended for production use.

Telnet is used for an open and insecure method of communication to the API channel on the NHD-CTL-PRO. Connecting to the NHD-CTL-PRO on port 23 will allow access to send and receive API-related actions, such as a matrix switch or to 2-way feedback from peripheral equipment.

Telnet cannot be used to access encoders and decoders directly. Any communication to an endpoint is strictly performed via a proprietary connection to the NHD-CTL-PRO.

Telnet can be enabled/disabled as needed to meet application security requirements. Telnet port usage can also be changed from the default 23 to any port you wish via the NHD-CTL-PRO web interface.

TELNET OVER TLS -PORT 992

Telnet over TLS works in a similar way to Telnet, the main difference being an encrypted and authenticated connection. Using TLS will provide a secure connection to the NHD-CTL-PRO's API channel. Telnet over TLS operates on port 992.

Telnet over TLS cannot be used to access encoders and decoders directly. Any communication to an endpoint is strictly performed via a proprietary connection to the NHD-CTL-PRO. Telnet over TLS port usage can also be changed from the default 992 to any port you wish via the NHD-CTL-PRO web interface.

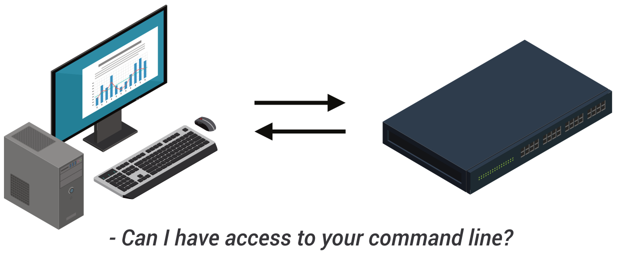
SECURE SHELL (SSH) -PORT 22 (10022)

The Secure Shell (SSH) protocol is utilized to establish a secure session, similar to Telnet, across a standard TCP/IP connection. This protocol is employed for tasks such as logging into systems, executing programs on remote systems, and transferring files from one system to another, all while maintaining an encrypted connection.

As illustrated in Figure below, a SSH client is attempting to connect to a SSH server, wherein the data must be transmitted in an encrypted state.

When you use SSH, you establish a secure encrypted connection between your computer and a remote computer, which makes it difficult for others to intercept or eavesdrop on your communication.

This encrypted connection also ensures that any data you transmit, such as login credentials or sensitive information, is protected from prying eyes.



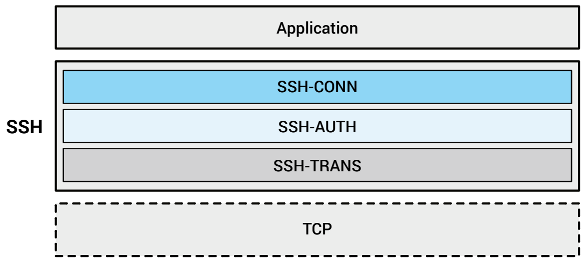
- Can I have access to your command line?
- I accept only encrypted data!
- Here is my encrypted username, password and key: a€@aslkgj!f2HVm
- Here is my response: eSgkh2#%kjka12s

SSH provides a secure channel over an unsecured network in a client-server architecture, connecting an SSH client application with an SSH server. The protocol specification distinguishes between two major versions, referred to as SSH-1 and SSH-2.

The standard TCP port for SSH is 22. The first version, SSH-1, is now deprecated because of security flaws in it. In this paragraph, we discuss only SSH-2. SSH is an incredibly powerful tool that provides a secure and flexible way to remotely access and manage networked devices.

Not only does it support various authentication mechanisms, such as password-based authentication and public key-based authentication, but public key-based authentication is considered more secure since it doesn't require sending passwords over the network and can provide stronger protection against certain types of attacks.

SSH can be used as a 3rd method for accessing the NHD-CTL-PRO's API channel. Similar to Telnet over TLS, SSH offers an encrypted and authentication connection. SSH operates on port 10022 and usage can also be changed from the default 10022 to any port you wish via the NHD-CTL-PRO web interface. SSH also exists via port 22 on encoders, decoders, and the NHD-CTL-PRO, however, this connection is locked for WyreStorm use only in the case of accessing device diagnostics or advanced troubleshooting.



SSH TRANSPORT-LAYER PROTOCOL (SSH-TRANS)

Because TCP is not a secured transport-layer protocol, SSH first uses a protocol that creates a secured channel on top of the TCP. This new layer is an independent protocol referred to as SSH-TRANS. When the procedure implementing this protocol is called, the client and server first use TCP to establish an insecure connection. Then they exchange several security parameters to establish a secure channel on top of the TCP.

The SSH-TRANS protocol is responsible for several tasks, including:

1. Encryption and integrity protection: SSH-TRANS negotiates encryption algorithms and keys to secure the connection between the client and server. It also provides integrity protection by including a message authentication code (MAC) with each packet.
2. Compression: SSH-TRANS can optionally compress data before encrypting and transmitting it between the client and server to reduce network bandwidth usage.
3. Key exchange: SSH-TRANS uses a key exchange algorithm to securely exchange session keys between the client and server.
4. Reliable data transmission: SSH-TRANS provides a reliable data transmission mechanism by implementing a sequence number and acknowledgement mechanism to ensure that data packets are delivered in order and without errors.

SSH AUTHENTICATION PROTOCOL (SSH-AUTH)

Once the secure channel between the client and the server is established, the server can be authenticated for the client using another procedure.

The client authentication process in SSH is quite similar to the process used in Secure Socket Layer (SSL). This layer provides various authentication tools, similar to those used in SSL, to authenticate the client for the server.

The authentication process begins with the client sending a request message to the server which includes the username, server name, authentication method and required data. The server then responds either with a success message confirming that the client is successfully authenticated or a failed message indicating that the authentication process needs to be repeated with a new request message. With this secure and efficient authentication procedure, SSH ensures that only authorized persons can access the system.

The SSH-AUTH protocol is responsible for several tasks, including:

1. Authentication methods: SSH-AUTH supports various authentication methods, including password-based authentication, public-key authentication, and host-based authentication.
2. User authentication: SSH-AUTH verifies the identity of the client by requesting and verifying the user's credentials, such as a password or public key.
3. Server authentication: SSH-AUTH also provides mechanisms for verifying the identity of the server, such as using the server's public key to authenticate the server during the initial connection setup.
4. Key management: SSH-AUTH provides key management functions, such as generating and storing user keys, and managing access to those keys.

SSH CONNECTION PROTOCOL (SSH-CONN)

Once the secure channel is established, and the server and client have been mutually authenticated, the SSH protocol calls a piece of software known as SSH-CONN, which provides a variety of services. One of these services is multiplexing, which allows the client to create multiple logical channels over the same secure channel. Each of these channels can then be used for distinct purposes, such as remote login or file transfer.

The SSH-CONN protocol thus significantly enhances the capability of SSH by providing an efficient way to use a single secure connection for multiple tasks.

The SSH-CONN protocol is responsible for several tasks, including:

1. Authentication: SSH-CONN authenticates the client to the server using a variety of authentication methods, including passwords, public-key authentication, and host-based authentication.
2. Encryption: SSH-CONN negotiates encryption algorithms and keys to secure the connection between the client and server.
3. Channel multiplexing: SSH-CONN allows multiple channels to be multiplexed over a single SSH connection, enabling multiple simultaneous sessions to be established between the client and server.
4. Connection management: SSH-CONN manages the connection between the client and server, including handling disconnections and reconnections.

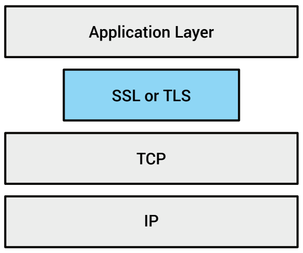
HTTP over TLS/SSL (HTTPS) – Port 443

Web servers are found in most NetworkHD devices. The NHD-CTL-PRO contains a web server which hosts an interface for the management, configuration and maintenance of encoders and decoders. With the exception of the 600 series, all other encoders utilize a web server to host an MJPEG preview stream which can be accessed via a web browser, the WyreStorm NetworkHD Touch app or 3rd-party control panels.

The Secure Sockets Layer (SSL) Protocol and the Transport Layer Security (TLS) Protocol are the two dominant protocols that provide security at the transport layer today.

The SSL protocol has evolved into three versions—SSL 1.0, SSL 2.0, and SSL 3.0—to become what is currently known as the TLS protocol. TLS is an IETF-defined version of SSL and is used widely for secure communication over the internet.

Both of these protocols ensure data integrity, confidentiality, and authentication for communication between two endpoints. SSL provides the encryption component implemented within the TCP/IP.



SSL protocol does not strictly speaking, operate at the transport layer. Instead, it operates at an intermediate layer between the transport layer and the application layer. When a client makes a request to a server over HTTPS, the server responds by sending a digital certificate, which includes a public key. The client verifies the certificate and generates a symmetric key for encrypting and decrypting data transmitted between the client and server.

The encryption used in HTTPS is based on the Transport Layer Security (TLS) or Secure Sockets Layer (SSL) protocol, which provides a secure communication channel by encrypting all data transmitted between the client and server.

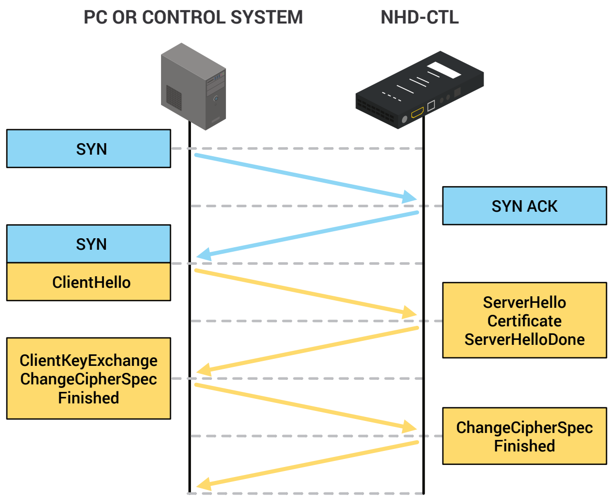
SSL Architecture

Secure Sockets Layer (SSL) is a networking protocol designed to provide security and compression services for data generated from the application layer. It is commonly used with the Hypertext Transfer Protocol (HTTP) to ensure that data transmitted over the network is secure.

SSL works by compressing (optionally) the data received from the application, signing it and encrypting it before transmission using a reliable transport-layer protocol like Transmission Control Protocol (TCP). The SSL architecture consists of several layers, each responsible for specific tasks in the secure communication process.

1. SSL Record Protocol: The SSL Record Protocol is the lowest level of the SSL architecture and is responsible for fragmenting, compressing (if enabled), encrypting and decrypting data. The SSL Record Protocol also provides message authentication and message integrity using a Message Authentication Code (MAC).
2. SSL Handshake Protocol: The SSL Handshake Protocol is responsible for negotiating the cryptographic algorithms to be used during the session, authenticating the server (using digital certificates), and establishing a shared secret key between the client and server. The shared secret key is used to encrypt and decrypt data transmitted between the client and server during the session.
3. SSL Change Cipher Spec Protocol: The SSL Change Cipher Spec Protocol is responsible for signaling that the encryption algorithm and keys should be changed for future data transmissions.
4. SSL Alert Protocol: The SSL Alert Protocol is responsible for signaling errors or abnormal conditions in the SSL connection.

In addition to these core protocols, SSL offers a range of powerful cryptographic algorithms and security mechanisms. From public key cryptography to digital certificates and message authentication codes (MACs), these mechanisms work in tandem to create a secure web of communication, ensuring the confidentiality, integrity, and authenticity of data transmitted over the network.

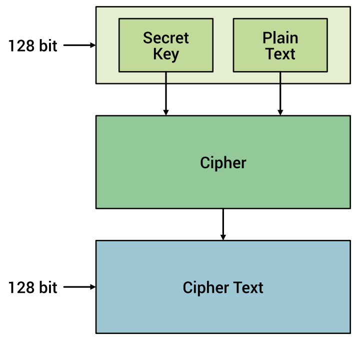


AES-128

AES-128 is an incredibly powerful and secure encryption algorithm, used to protect important data from unwanted access. It is a symmetric encryption algorithm established by the US National Institute of Standards and Technology (NIST) that utilizes a 128-bit key.

This key is used to transform plaintext into ciphertext, and vice versa, through a substitution-permutation network, operating on 128-bit blocks of data. It is widely used across the NetworkHD series to encrypt A/V and control signal streams, as well as to protect users from brute-force attacks.

Furthermore, AES encryption does not impede on the video preview streams, which are generated via MJPEG through HTTP(s) protocol. AES-128 is an essential component in protecting data and ensuring that only authorized individuals can access it.



802.1x & LDAP

Conformance to additional Network security standards

802.1x is an authentication protocol that secures access to a network, such as Wi-Fi or Ethernet. It requires a user to provide valid credentials, including a username and password, to gain access. T

he authentication process involves three components: the supplicant (the client device trying to connect), the authenticator (the network device, such as a switch or access point), and the authentication server (a separate server that verifies the credentials). The process starts when the supplicant requests access to the network.

The authenticator then sends an Extensible Authentication Protocol (EAP) message to the supplicant, prompting it to provide credentials. The supplicant sends the credentials to the authentication server, which verifies them and sends an authentication response to the authenticator. If the credentials are valid, the authenticator permits the device to connect to the network.

With 802.1x authentication, users can securely access networks using valid credentials.
802.1x supports two authentication protocols: EAP-MSCHAPV2 and EAP-TLS.

The former requires a username and password to authenticate a device, while the latter uses digital certificates to do the same. This is a secure and reliable way to protect data and create a safe environment for users.

With 802.1x, you can rest assured that your data is safe from unauthorized access. This protocol can be used for the NHD-CTL-PRO and individual encoders/decoders, ensuring they are permitted to gain access to the network.

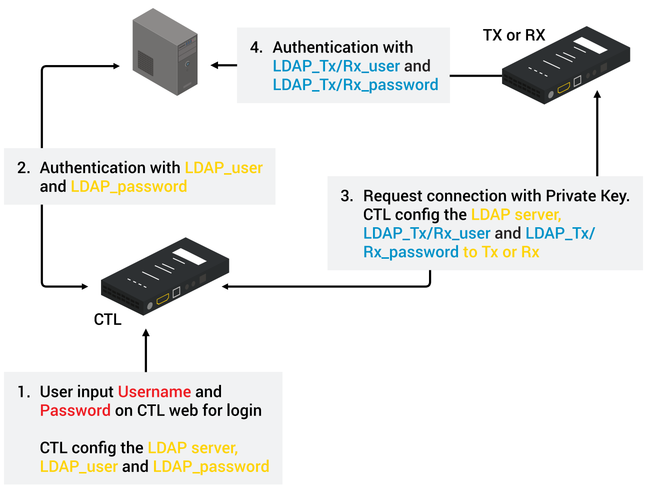
Compatibility with enhanced user access protocols

Lightweight Directory Access Protocol (LDAP) is a protocol used to access and manage information directories. An LDAP directory is a hierarchical structure that stores information about users, groups, and other network resources. LDAP is often used in enterprise environments to manage user authentication, authorization, and other network services.

LDAP is a powerful client-server protocol for managing and accessing information stored in directories. It's widely used in enterprise environments and is supported by various directory services software and tools.

With LDAP, users can authenticate, search, add, modify, delete, access control, replicate, refer and perform other operations. It typically uses ports 389 (unencrypted) or 636 (SSL/TLS encrypted) over TCP/IP for secure communication. LDAP is an essential tool in keeping large-scale networks secure, reliable and efficient.

It also offers administrators the ability to manage user access to resources on a large scale, making it a highly valuable asset for organizations of any size.


4. Authentication with LDAP_Tx/Rx_user and LDAP_Tx/Rx_password
TX or RX
2. Authentication with LDAP_user and LDAP_password
3. Request connection with Private Key. CTL config the LDAP server, LDAP_Tx/Rx_user and LDAP_Tx/Rx_password to Tx or Rx
CTL
1. User input Username and Password on CTL web for login
CTL config the LDAP server, LDAP_user and LDAP_password

Multicast IP Addresses & Ports

The following information outlines multicast addresses, protocols, and ports utilization across multiple NetworkHD series products.

NetworkHD 400 & 500s

B.CD.EF shown in the multicast addresses below should be replaced by the last 20 bits of a device's MAC address, which is then converted from hexadecimal to decimal. For example, if calculating a device's video stream MC address where the device has a MAC address of E4:CE:02:A2:34:4F, the calculated MC address would be 234.34.52.79.

Application	Address	Host -> Client Host <- Client
Heartbeat/ Multicast/ UDP	video 234.32+B.CD.EF	X → 59002 59002 ← X
	audio 234.48+B.CD.EF	X → 59003 59003 ← X
	ir 234.64+B.CD.EF	X → 59004 59004 ← X
	serial 234.80+B.CD.EF	X → 59005 59005 ← X
	audio return 234.96+B.CD.EF	X → 59006 59006 ← X
	usb 234.112+B.CD.EF	X → 59007 59007 ← X
node_query → node_response	225.1.0.1	X → 59101
node_response → node_query	UDP	X → 59100
VideoIP Host encode	234.32+B.CD.EF	X → 59200 59201 ← X X → 59204 59204 ← X
AudioIP Host encode	234.48+B.CD.EF	X → 59300
IRoIP Host encode	234.64+B.CD.EF	X → 59400 59400 ← X
SoIP Type 3	234.80+B.CD.EF	X → 59500 59500 ← X
Audio Return Decoder	234.96+B.CD.EF	59301 ← X
USBoIP	TCP	59700 ← X
KMoIP	234.112+B.CD.EF	X → 59702 59703 ← X
node_list → node_reply	225.1.0.0	X → 3333
node_reply → node_list	UDP	X → 3334
CTL → TX/RX	226.1.0.0 228.1.0.0	X → 1234
TX/RX → CTL	226.2.0.0	X → 10000 X → 20002
HTTP Service	TCP	80
HTTPS Service	TCP	443

NetworkHD 100 & 200s

For details on how to calculate NHD-100/200 series multicast addresses please reach out to WyreStorm Technical Support.

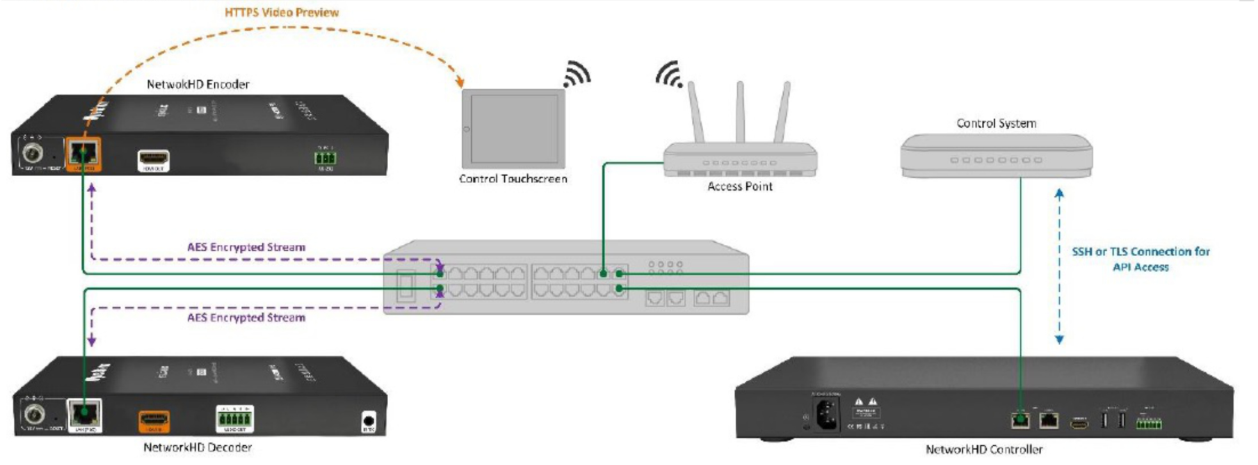
Application	Protocol	IP-Port	Relevant Device
Main video stream port	UDP	F(0x1)[1]:12345	NHD-100, 200, 110, 140 & 250
Main audio stream port	UDP	F(0x1) [1]:12346	NHD-100, 200, 110, 140 & 250
Sub stream port	UDP	F(0x2) [1]:12345	NHD-100, 200, 110, 140 & 250
Pure audio stream port	UDP	F(0x3) [1]:12346	NHD-100, 200, 110, 140 & 250
RS232 TX	UDP	F(0x4) [1]:12400	NHD-110
RS232 RX	UDP	F(0x4) [1]:12401	NHD-110
IR TX	UDP	F(0x5) [1]:12410	NHD-110
IR RX	UDP	F(0x5) [1]:12411	NHD-110
USB TX	UDP	F(0x6) [1]:12420	NHD-110
USB RX	UDP	F(0x6) [1]:12421	NHD-110
Searching devices listening port	UDP	225.1.0.0[3]:3333	NHD-100, 200, 110, 140 & 250
I frame unified switch to RX information listening port	UDP	226.2.0.0[3]:4321	NHD-100, 200, 110, 140 & 250
I frame unified switch order listening port	UDP	226.2.0.1[3]:4322	NHD-100, 200, 110, 140 & 250
CTL Notify information listening port	UDP	226.2.0.0:10000	NHD-100, 200, 110, 140 & 250
Mjpeg http browser port	TCP	0.0.0.0:80	NHD-100, 200, 110, 140 & 250
Mjpeg https browser port	TCP	0.0.0.0:443	NHD-100, 200, 110, 140 & 250

NHD-CTL & Console Software

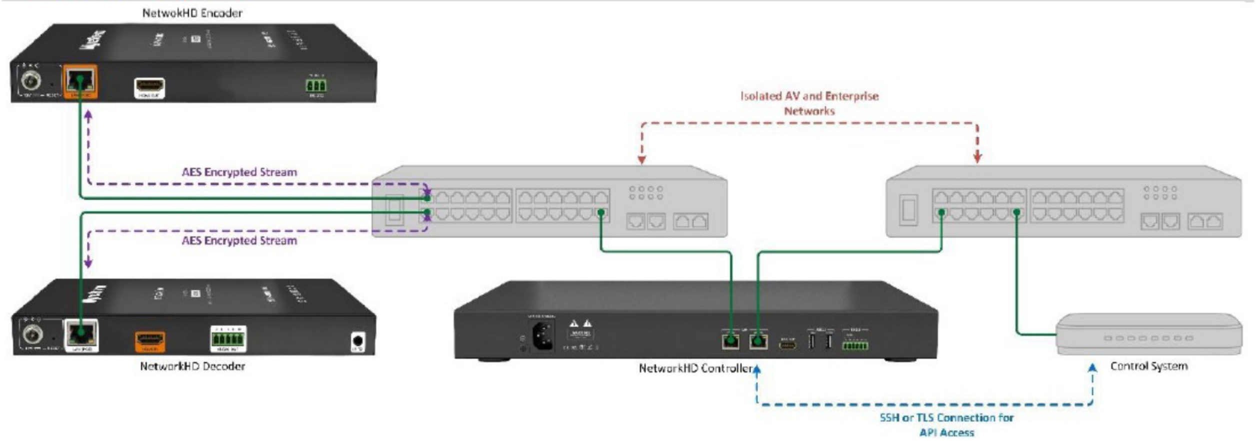
Application	Protocol	IP-Port	Relevant Device
Search protocol used for devices searching or searched by PC software.	UDP/MC	225.1.0.0	CTL->TX/RX or PC->CTL
Batch control event channel for CTL and Console software.	UDP/MC	226.1.0.0	CTL->TX/RX
Matrix switch command to TX address via Console	UPD/MC	226.2.0.1	CTL->TX/RX
Serial report and video lost/ found Notification via Console.	UPD/MC	226.2.0.0	CTL->TX/RX
Search protocol used by CTL	UDP/MC	225.1.0.0	X->3333
Search protocol reply	UDP		3334<-X
Matrix switch command to RX	UDP/MC	226.1.0.0	X->1234
Matrix switch command to TX. (Only valid for H.264 products)	UDP/MC	226.2.0.1	X->4322
Serial report and video lost/ found notification.	UDP/MC	226.2.0.0	10000<-X
Telnet API channel	TCP		23 <- X
Telnet API channel for NHD Touch	TCP		23000 <- X
SSH API channel for NHD Touch	TCP		10022 <- X
Web service	TCP		80 <- X

Secure System Designs

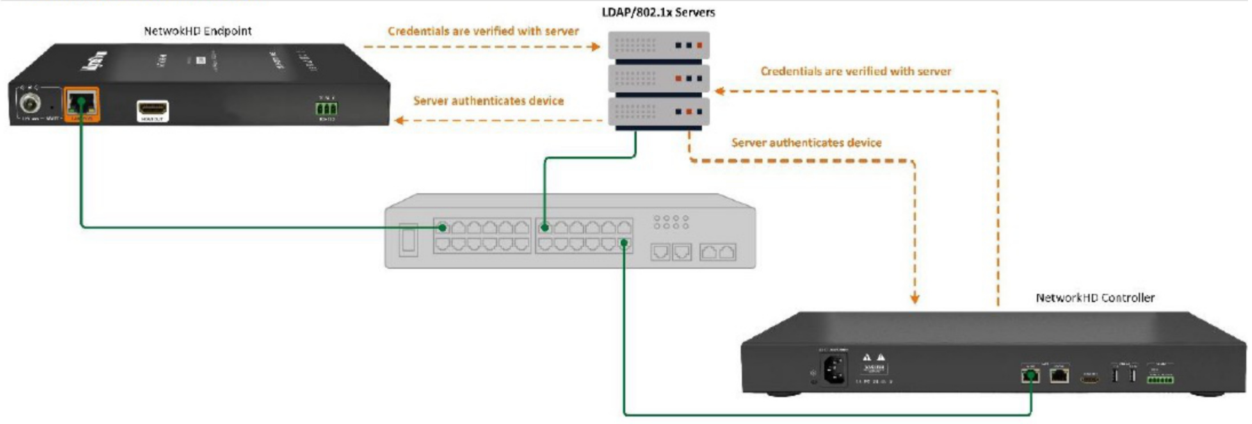
Integrated AV Network



Isolated AV Network



LDAP/802.1x Network


NetwokHD Endpoint
LDAP/802.1x Servers
Credentials are verified with server
Server authenticates device
Credentials are verified with server
Server authenticates device
NetworkHD Controller

WyreStorm Technologies
Rest of the World
Tech Valley Park
23 Wood Road
Round Lake, NY 12151 USA
Tel: +1 518-289-1293

WYRESTORM.COM

Follow us on social media to stay up to date with all things WyreStorm.