Virtual Channel SDK Gude

1.1

Table of Contents

Introduction

The PCoIP Virtual Channel Software Development Kit (SDK) enables developers to build custom PCoIP Virtual Channel plug-ins for PCoIP sessions. You can implement PCoIP Virtual Channel functionality as a plug-in to send encrypted data between servers and client endpoints during an active PCoIP session.

The PCoIP Virtual Channel Application Programming Interface (API) is available as an optional add-on to solution developers who want to extend the types of traffic flowing through the PCoIP session, such as clipboard redirection, local printing, and customised device support.

Virtual Channel SDK Documentation

The Virtual Channel SDK documentation is available on the HP Anyware Documentation Portal.

Overview of the PCoIP Virtual Channel

The PCoIP Virtual Channel Software Development Kit (SDK) enables developers to build custom PCoIP Virtual Channel plug-ins for PCoIP sessions. You can implement PCoIP Virtual Channel functionality as a plug-in to send encrypted data between servers and client endpoints during an active PCoIP session.

The PCoIP Virtual Channel Application Programming Interface (API) is available as an optional add-on to solution developers who want to extend the types of traffic flowing through the PCoIP session, such as clipboard redirection, local printing, and custom device support.

About the PCoIP Virtual Channel API/SDK

Solution developers can write plug-ins that stream data between the agent and the client using a secure PCoIP session. For example, the PCoIP copy-and-paste function uses a virtual channel to transport clipboard data between a remote desktop client and the clipboard on a local PC, as shown next.

An example use case of the PCoIP Virtual Channel

PCoIP-Virtual-Channel

Virtual channel traffic travels over the PCoIP session, and is therefore secure since it is encrypted and authenticated like other PCoIP traffic.

The virtual channel does not set up a new socket or port number and the channel can be opened and closed dynamically.

Both reliable and unreliable transport options are available. If PCoIP Virtual Channel traffic can be compressed and still meet WAN criteria, the API will automatically apply lossless compression. The API periodically checks if traffic can be compressed. Plug-ins can explicitly disable compression.

During the session, the PCoIP Virtual Channel API works to:

- Transfer data in either direction using streaming or datagram transfers,
- Receive notifications of relevant events via callbacks, and
- Log messages in the standard PCoIP session logs.

This SDK includes sample plug-ins, which demonstrate how to implement plug-ins and use the PCoIP Virtual Channel API. These sample plug-ins use the PCoIP Virtual Channel API to transfer data between client and server endpoints:

- plugin_vchan_sample_stream uses the reliable PCoIP Virtual Channel streaming data API.
- plugin_vchan_sample_dgram uses the reliable PCoIP Virtual Channel datagram API.
- plugin_vchan_sample_u_echo uses the unreliable PCoIP Virtual Channel API to send unreliable datagrams in parallel with the reliable data transfer.

Building PCoIP Virtual Channel Plug-In Samples

Use CMake (Cross-Platform Make) to build the sample plug-ins for Windows, macOS, or Linux. CMake is an open-source, cross-platform family of tools designed to build, test, and package software.

You can download the latest version of CMake for platform specific installation file from www.cmake.org.

Info

Microsoft Visual Studio is required for building the plug-ins.

You can use CMake to configure and generate platform and compiler specific build files and build the target plug-ins on the chosen platforms.

Building the plug-ins for other platforms requires the SCons software construction tool. SCons requires Python and GCC compiler or a corresponding toolchain that supports the `pthreads` library (for the operating system primitives used in the plug-ins).

Using CMake to Configure and Generate Platform and Compiler Build Files

The following image shows the configuration information for CMake with Microsoft Visual Studio on Windows:

CMake

You can configure other platforms in a similar way to take the source codes and generate the build files in the designated binaries folder. It is possible to configure CMake on the command-line user interface. For more information, see the CMake documentation.

Completing the CMake process

You must first **Configure**, and then **Generate** a two-pass routine to complete the CMake process.

To build the target plug-ins on your chosen platform, build the plug-in with the designated compiler from the output build files folder.

Binary compatibility for Windows plug-ins

For Windows Clients, ensure that the PCoIP Virtual Channel plug-ins are built as 64-bit `.dlls` with Visual Studio Code 2015 or later.

For Windows Agents, if the PCoIP Virtual Channel plug-ins are built with a version of Visual Studio earlier than 2015, ensure that the required re-distributables are installed.

Binary compatibility for macOS plug-ins

To maintain binary compatibility, ensure that PCoIP Virtual Channel plug-ins on macOS are built as 64-bit `dylibs` with Xcode 5.0 or above.