Web-Browser based Remote Access to Embedded Devices

A new technology from Applied Informatics enables easy and secure remote access to the built-in web server of an embedded device, even if the device is located behind a NAT router or a firewall and does not have a public IP address.

Web-based user interfaces are state-of-the-art in network-based embedded systems. These web-based user interfaces makes configuration, control and monitoring of a device from every PC, smart phone or tablet device running a web browser possible. Thanks to advanced web browsers, JavaScript and Ajax technologies, modern web-based user interfaces are powerful, visually attractive and easy to use. Since their only requirement is a HTTP(S) connection between the web browser and the web server running on the device, they are perfectly fitted for remote access scenarios. However, for this to work, the web browser must be able to create a network connection to the device's web server. This is only possible if the embedded device is located in the same network as the device running the web browser, if the networks containing the client and server are linked, or if the embedded device can be directly reached over the internet. Unfortunately, this is rarely the case in practice. Embedded devices in the field are often connected to private networks behind NAT routers or firewalls. This is especially true for consumer electronics devices like set-top-boxes, home automation/networking devices or smart metering devices, which are typically located behind a NAT broadband router. Even devices connected to a mobile network such as GSM/GPRS or UMTS in most cases do not have public IP addresses and thus are not directly reachable. This means that while these devices can open connections to servers on the internet, it is not possible to access the device's web server from the outside, unless additional measures are taken.

INTERNET-BASED REMOTE ACCESS TO DEVICES WITH REVERSE HTTP

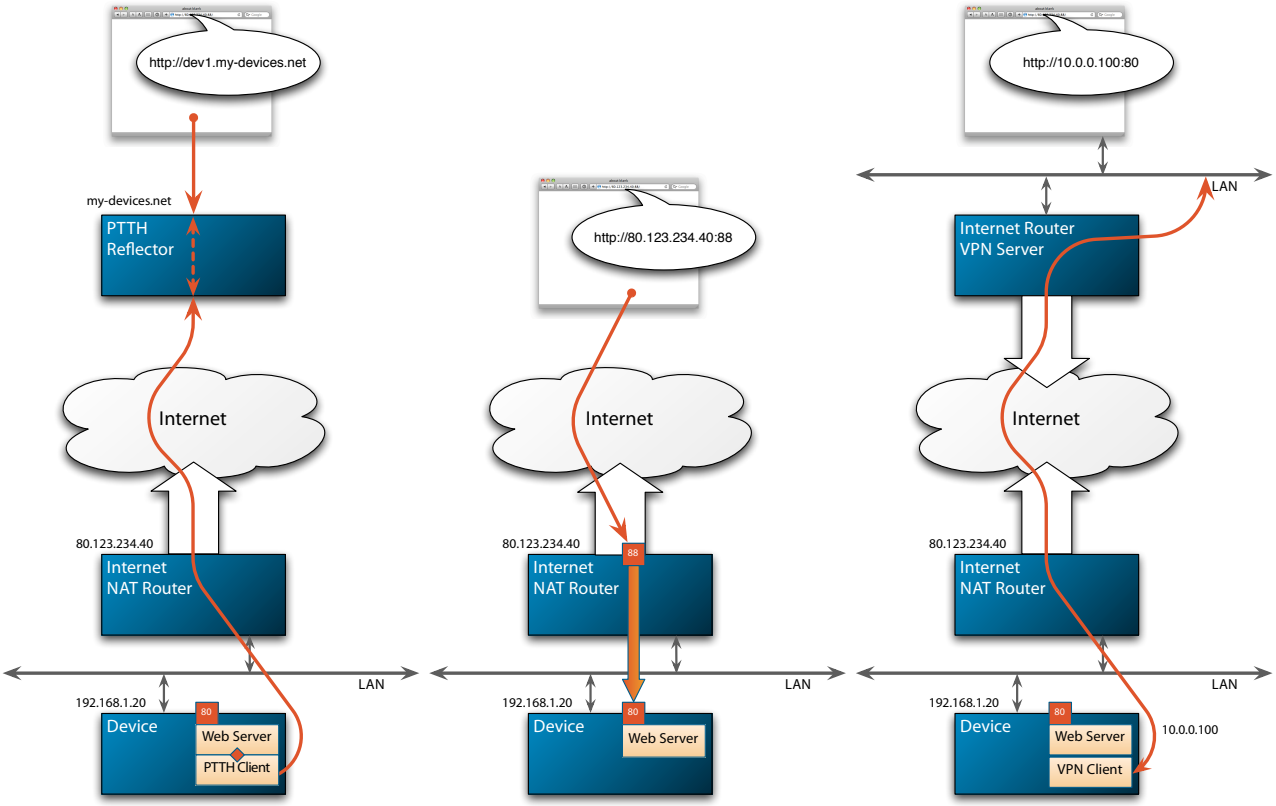
Port forwarding and Virtual Private Network (VPN) are well-known and established technologies for enabling internet-based remote access to computers and network devices behind NAT routers or firewalls. However, as detailed in the table at the end of this article, both technologies have severe drawbacks when being used with embedded systems. For this reason, Applied Informatics has created a new technology that is a great alternative to port forwarding and VPN. Reverse HTTP enables easy and secure remote access to the web server of a device, even if the device is located in a private or mobile network behind a NAT router or firewall. How this new technology works will be explained in the following.

Application Scenarios

- Remote access to data loggers and monitoring devices, e.g. in renewable energy (photovoltaics and wind energy plants), environmental monitoring, traffic and transport, etc.
- Smart metering (remote access to smart power meters or smart metering gateways)
- Remote access to mobile devices for data acquisition, tracking, fleet management, etc.
- Remote maintenance and servicing of consumer electronics, home/building automation and HVAC devices
- Remote maintenance and servicing of machines and industrial equipment
- Remote access to ticketing and access control systems

HOW REVERSE HTTP WORKS

Reverse HTTP is based on a simple modification of the well-known and proven HTTP protocol that drives the internet. The only difference to HTTP is who is setting up the network connection which is used for sending HTTP requests and receiving their responses. In HTTP, the client (web browser) is responsible for opening a connection to the web server, over which it then sends the requests. In Reverse HTTP, it's the server that sets up the connection. Since the Reverse HTTP server does not know its clients, and would not even be able to create a direct network connection to each client (as clients are usually behind a NAT router or firewall), the Reverse HTTP server opens a connection to a special server called the Reverse HTTP Reflector. For this to work, the Reverse HTTP Reflector must be accessible over the internet. Once the connection between the device and the reflector server has been established, the reflector uses this connection to send HTTP requests to the device. Where do this HTTP requests come from? The Reverse HTTP Reflector also contains a standard HTTP server, which accepts requests from



Reverse HTTP (left), NAT Port Forwarding (center) und VPN (right) in comparison

clients (web browsers). These requests are then simply forwarded to the device, using the device's Reverse HTTP connection. Setting up the initial Reverse HTTP connection between the device and the reflector server is almost always possible as long as the device can access the internet. It even works through a HTTP proxy server.

REVERSE HTTP IN PRACTICE

In a typical usage scenario, more than one device will be connected to a reflector server. Therefore, when the reflector receives a HTTP request from a client, it needs to find out to which device the request must be forwarded. There are two ways to make this work. The first one is via the URL sent from the client to the reflector (e.g., *http://dev1.my-devices.net*). This requires setting up a wildcard DNS record in the DNS server which resolves all requests for **.my-devices.net* to the reflector server *www.my-devices.net*. The reflector server can then use the Host header in the HTTP request together with an internal table to associate the request with a device. Alternatively, the reflector server can set a cookie in the client after logging in to the reflector server and selecting a target device. This cookie is then sent with every request from the client to the reflector and allows the reflector server to forward the request to the appropriate device. There are multiple options for running the reflector server. It can be ran on a server in a private datacenter, or it can be ran on a virtual server provided by a cloud service provider such as Amazon (EC2) or Rackspace. Running the reflector server can also be outsourced to a dedicated service provider.

SECURITY AND PRIVACY GUARANTEED

Since the reflector server only transparently forwards HTTP requests, but does not store any data passed through it, Reverse HTTP does not introduce any additional data security and privacy risks – especially if the reflector server is operated in a private data center. Of course, both the connection between the device and the reflector server, as well as the connection between the client (web browser) and the reflector server can be encrypted with SSL or TLS. A single reflector server instance can easily handle thousands of devices, with up to 100 or more simultaneous browser sessions. A great advantage of this technology is that it is inherently secure. Since the device does not need to have any open ports to the internet, there is no danger of denial-of-service or other attacks against the device. Requests to the device can only be sent through the reflector server, and the reflector server requires proper authentication of the client before forwarding requests to the device. Also, devices must authenticate themselves against the reflector server when setting up the Reverse HTTP connection. Authentication can be done through a shared secret (challenge-response authentication), or by presenting a digital certificate when using a SSL or TLS connection.

WORKS FOR WEB SERVICES AS WELL

Reverse HTTP is not just for accessing web pages. Every HTTP-based protocol can also be used over a Reverse HTTP connection, including web services based on SOAP, JSON or REST technologies. This makes Reverse HTTP a great foundation for automated device management applications.

EASY INTEGRATION

The software necessary for integrating Reverse HTTP into a device, as well as the reflector server is provided by Applied Informatics. For devices where the necessary modification of the firmware is not possible, a special gateway device can be used to connect the device to a reflector server. The gateway is located in the same local area network as the device, and forwards requests from the reflector server to the device's web server.

Reverse HTTP is a great alternative to technologies like NAT port forwarding and virtual private networks to enable easy and secure remote access to the built-in web server of a device. The technology can be used without touching the existing network infrastructure. The necessary reflector server can be operated in "the cloud", and devices can be easily integrated, either by updating the firmware or by using a special gateway device.

CONTACT US FOR MORE INFORMATION

Applied Informatics Software Engineering GmbH
Maria Elend 96/4
9182 Maria Elend
Austria

T +43 4253 32596 **F** +43 4253 32096
info@appinf.com | www.appinf.com

LIVE DEMONSTRATION

A live demonstration of Applied Informatics' Reverse HTTP technology is available at http://my-devices.net.

Technology	Advantages	Disadvantages
Reverse HTTP	> simple extension of proven HTTP > can be used without changes to the existing network infrastructure > supports secure, encrypted connections > the necessary reflector server can be operated in the cloud > high scalability, up to thousands of devices per reflector server instance	> a special reflector server must be operated > Reverse HTTP software must be integrated into device, or a gateway device must be used to integrate legacy devices > only HTTP access to device possible, no SSH or FTP
Port Forwarding	> simple and widely supported by NAT routers > allows access to any network service provided by the device, not just HTTP	> NAT router configuration for port forwarding can be complex, especially if multiple devices must be accessible (every device needs a unique public port number) > a Dynamic DNS service is needed if the NAT router does not have a static public IP address > the device is directly exposed to the internet – danger of denial-of-service or other attacks
Virtual Private Network	> the device is directly integrated into a remote network using a secure tunnel through the internet > secure, encrypted connection > proven, standardized technology	> VPNs may be blocked by network provider > necessary network infrastructure is difficult to setup and to maintain, especially if many devices must be integrated