Instant VPN, Remote Access

Vedivi Free Trial.
No setup fee. No monthly payments. No credit cards. No hassle.

Sign up now >

Configure your firewall in C++

User Rating: / 6
Poor Best **Rate**

I did work recently on configuring Firewalls programmatically within an application, so I thought I should share

Challenge

Most computers run a **Personal Firewall** these days (Windows Firewall, PC-Cillin, Norton, AVG...). So in ma
deal with getting their application configured in their customer's firewall.

You could either choose to tell your users how they should configure their firewall (in a help document) or do it

Some application only require a static configuration, i.e a configuration that will never change. For instance "
"*exceptions*" or trusted applications. This can be performed during the installation process and is normally su

But in other cases, the configuration depends on the user configuration and needs to be done after the install
application. So let's see how to do it in C++

Scope

To keep it simple, we will consider the following usages:

Check if Windows Firewall is enabled

See if Windows Firewall (Windows XP & Vista) is installed and enabled.

Port Enable/Disable

Check if a specific port is enabled and eventually enable it if it exists.

Configure the Firewall in C++

Windows Firewall API is available through COM, so we will need the usual COM initialize/uninitialize.

CFirewallUtil.h

The class described is called ***CFirewallUtil*** and has the following header (.h):

```
class CFirewallUtil
{
public:
    CFirewallUtil(void);
    ~CFirewallUtil(void);

    HRESULT IsFirewallEnabled(INetFwProfile* fwProfile, BOOL * pEnabled)
    HRESULT IsPortEnabled(INetFwProfile* fwProfile, BOOL *pEnabled, int
    HRESULT SetPortEnabled(INetFwProfile* fwProfile, int iPort, bool bTC

    HRESULT Initialize(INetFwProfile** fwProfile);
    void Uninitialize(INetFwProfile* fwProfile);

private:
    INetFwMgr* m_pFwMgr;
    INetFwPolicy* m_pFwPolicy;
};
```

Initialize

Initialize COM, instanciate the FirewallManager object (NetFwMgr) and loads the firewall profile (INetFwProfi

```
HRESULT CFirewallUtil::Initialize(INetFwProfile** fwProfile)
{
    HRESULT hr = S_OK;

    // Initialize COM.
    hr = CoInitializeEx(0, COINIT_APARTMENTTHREADED);

    // Ignore RPC_E_CHANGED_MODE; this just means that COM has already b
    // initialized with a different mode. Since we don't care what the m
    // We'll just use the existing mode.
    if (hr == RPC_E_CHANGED_MODE)
    {
        hr = S_OK;
    }

    INetFwMgr* fwMgr = NULL;
    if(SUCCEEDED(hr))
    {

        // Create an instance of the firewall settings manager.
        hr = CoCreateInstance(
                __uuidof(NetFwMgr),
                NULL,
                CLSCTX_INPROC_SERVER,
                __uuidof(INetFwMgr),
                (void**)&fwMgr
                );
    }

    INetFwPolicy* fwPolicy = NULL;
    if(SUCCEEDED(hr))
    {
        // Retrieve the local firewall policy.
        hr = fwMgr->get_LocalPolicy(&fwPolicy);
    }

    // Retrieve the firewall profile currently in effect.
    *fwProfile = NULL;
    if(SUCCEEDED(hr))
    {
        hr = fwPolicy->get_CurrentProfile(fwProfile);
    }

    if(FAILED(hr))
    {
       if(fwPolicy != NULL)
            fwPolicy->Release();

        if(fwMgr != NULL)
            fwMgr->Release();
    }
    else
    {
        m_pFwPolicy = fwPolicy;
        m_pFwMgr = fwMgr;
    }

    return hr;
}
```

Unitialize

Free the profile and any initialized object and unitialize COM

```
void CFirewallUtil::Uninitialize(INetFwProfile* fwProfile)
{
    // Release the firewall profile.
    if (fwProfile != NULL)
    {
        fwProfile->Release();
    }

    if(m_pFwPolicy != NULL)
    {
        m_pFwPolicy->Release();
        m_pFwPolicy = NULL;
    }

    if(m_pFwMgr != NULL)
    {
        m_pFwMgr->Release();
        m_pFwMgr = NULL;
    }


    CoUninitialize();
}
```

IsFirewallEnabled

Checks if Windows firewall is enabled for the given profile.

```
HRESULT CFirewallUtil::IsFirewallEnabled(INetFwProfile* fwProfile, BOOL
{
    HRESULT hr = S_OK;
    *pEnabled = FALSE;
    *pEnabled = FALSE;

    VARIANT_BOOL fwEnabled;

    // Get the current state of the firewall.
    hr = fwProfile->get_FirewallEnabled(&fwEnabled);
    if(SUCCEEDED(hr))
    {
        *pEnabled = fwEnabled != VARIANT_FALSE;
    }

    return hr;
}
```

IsPortEnabled

Checks if the given port has an existing rule allowing it for the given protocol (TCP or UDP). It basically loops
matching configuration exists and is enabled.

```
HRESULT CFirewallUtil::IsPortEnabled(INetFwProfile* fwProfile, BOOL *pEn
{
    HRESULT hr = S_OK;
    *pEnabled = FALSE;

    INetFwOpenPort* fwOpenPort = NULL;
    INetFwOpenPorts* fwOpenPorts = NULL;

    // Retrieve the globally open ports collection.
    hr = fwProfile->get_GloballyOpenPorts(&fwOpenPorts);

    if(SUCCEEDED(hr))
    {
        NET_FW_IP_PROTOCOL ipProtocol = NET_FW_IP_PROTOCOL_TCP;
        if(!bTCP)
            ipProtocol = NET_FW_IP_PROTOCOL_UDP;
        // Attempt to retrieve the globally open port.
        hr = fwOpenPorts->Item(iPort, ipProtocol, &fwOpenPort);
        if (SUCCEEDED(hr))
        {
            VARIANT_BOOL fwEnabled;
            // Find out if the globally open port is enabled.
            hr = fwOpenPort->get_Enabled(&fwEnabled);

            if(SUCCEEDED(hr))
            {
                *pEnabled = fwEnabled != VARIANT_FALSE;
            }
        }
        else
        {
            // The globally open port was not in the collection.
            hr = S_OK;
        }
    }

    // Release the globally open port.
    if (fwOpenPort != NULL)
        fwOpenPort->Release();

    // Release the globally open ports collection.
    if (fwOpenPorts != NULL)
        fwOpenPorts->Release();

    return hr;
}
```

SetPortEnabled

Checks if the given port has an existing rule allowing it for the given protocol (TCP or UDP) and if so, enables
exceptions). It basically loops through all the open ports and if a matching configuration exists, enables or dis

```
HRESULT CFirewallUtil::SetPortEnabled(INetFwProfile* fwProfile, int iPor
{
    HRESULT hr = S_OK;

    INetFwOpenPort* fwOpenPort = NULL;
    INetFwOpenPorts* fwOpenPorts = NULL;

    // Retrieve the globally open ports collection.
    hr = fwProfile->get_GloballyOpenPorts(&fwOpenPorts);

    if(SUCCEEDED(hr))
    {
        NET_FW_IP_PROTOCOL ipProtocol = NET_FW_IP_PROTOCOL_TCP;
        if(!bTCP)
            ipProtocol = NET_FW_IP_PROTOCOL_UDP;
        // Attempt to retrieve the globally open port.
        hr = fwOpenPorts->Item(iPort, ipProtocol, &fwOpenPort);
        if (SUCCEEDED(hr))
        {
            VARIANT_BOOL fwEnabled = bEnable;
            // Find out if the globally open port is enabled.
            hr = fwOpenPort->put_Enabled(fwEnabled);
        }
    }

    // Release the globally open port.
    if (fwOpenPort != NULL)
        fwOpenPort->Release();

    // Release the globally open ports collection.
    if (fwOpenPorts != NULL)
        fwOpenPorts->Release();

    return hr;
}
```

Putting it together

Here it how you would use the class to check for example if the Windows Firewall is enabled:

```
    HRESULT hr;
    BOOL bEnabled = FALSE;
    INetFwProfile* fwProfile = NULL;

    CFirewallUtil pUtil;
    hr = pUtil.Initialize(&fwProfile);
    if(SUCCEEDED(hr))
    {
        hr = pUtil.IsFirewallEnabled(fwProfile, &bEnabled);
    }

    pUtil.Uninitialize(fwProfile);
```

Conclusion

This is a small subset of what you can do with the firewall API, basically any configuration you can do through
interfaces makes it quite straightforward to use.

Download the complete example class

MSDNWindows Firewall: Internet Connection Sharing and Internet Connection Firewall (MSDN)

Last Updated on Tuesday, 01 June 2010 14:36

Discover Vedivi

Vedivi Business is the latest generation of secure remote access solution, it combines a VPN with Remote D

With Vedivi you can:

- Setup a VPN and provide secure remote access for your business in minutes
- Offer full network connectivity to your office network to collaborators wherever they connect from
- Allow users to connect (Remote Desktop) to their computer or a Windows Server session from any web b
- Provide Remote Assistance to any client or user from a web browser without software installation

Get started with Vedivi 30-Day free trial so you can see for yourself why so many businesses trust Vedivi for V

Home |