SSH Tunnel Management System - Deployment Guide

Overview

The SSH Tunnel Management System is a comprehensive network administration tool that enables automatic establishment and management of SSH tunnels between remote stations and a central parent server. This system is designed for network administrators who need secure, reliable remote access to distributed systems.

System Architecture

Components

1. **SSH Tunnel Client** - PowerShell-based client that automatically establishes reverse SSH tunnels
2. **SSH Tunnel Server** - Central management server that coordinates client connections
3. **Web Management Dashboard** - React-based web interface for monitoring and managing tunnels
4. **Network Discovery Module** - Automatic discovery of tunnel servers and network configuration

Key Features

- **Automatic Tunnel Establishment**: Clients automatically connect to parent servers
- **Reverse SSH Tunneling**: Secure tunnels from clients back to central server
- **Auto-Reconnection**: Automatic recovery from connection failures
- **Network Discovery**: Multiple methods for discovering tunnel servers
- **Web-Based Management**: Professional dashboard for monitoring all clients
- **Service Integration**: Runs as Windows service for persistent operation
- **Implied Consent**: Installation implies administrative consent for tunnel establishment

Prerequisites

Server Requirements

- **Operating System**: Windows Server 2016+ or Linux with PowerShell Core
- **PowerShell**: Version 5.1 or higher
- **SSH Server**: OpenSSH Server or compatible SSH daemon
- **Network**: Static IP address or FQDN
- **Ports**: SSH port (22 or custom) and web interface port (8080 or custom)
- **Resources**: Minimum 2GB RAM, 10GB disk space

Client Requirements

- **Operating System**: Windows 10/11 or Windows Server 2016+
- **PowerShell**: Version 5.1 or higher
- **SSH Client**: OpenSSH Client (included in Windows 10 1809+)
- **Network**: Outbound connectivity to tunnel server
- **Permissions**: Administrator privileges for service installation

Network Requirements

- **Firewall**: Allow outbound SSH connections from clients
- **DNS**: Optional SRV records for automatic discovery
- **Routing**: Network connectivity between clients and server
- **Security**: SSH key-based authentication recommended

Server Installation

Step 1: Prepare the Server Environment

1. **Install PowerShell** (if not already installed): ```powershell # Windows Server Enable-WindowsOptionalFeature -Online -FeatureName Microsoft-Windows-Subsystem-Linux

Or download PowerShell Core from GitHub releases ```

1. **Install OpenSSH Server**: `powershell # Windows 10/Server 2019+ Add-WindowsCapability -Online -Name OpenSSH.Server~~~~0.0.1.0 Start-Service sshd Set-Service -Name sshd -StartupType 'Automatic'`

2. **Configure Windows Firewall**: `powershell New-NetFirewallRule -Name sshd -DisplayName 'OpenSSH Server (sshd)' -Enabled True -Direction Inbound -Protocol TCP -Action Allow -LocalPort 22 New-NetFirewallRule -Name ssh-tunnel-web -DisplayName 'SSH Tunnel Web Interface' -Enabled True -Direction Inbound -Protocol TCP -Action Allow -LocalPort 8080`

Step 2: Install SSH Tunnel Server

1. **Download and Extract** the SSH Tunnel Management System to `C:\Program Files\SSHTunnelServer`

2. **Create SSH User Account**: `powershell # Create dedicated user for tunnel clients $password = ConvertTo-SecureString "ComplexPassword123!" -AsPlainText -Force New-LocalUser -Name "tunnel-client" -Password $password -Description "SSH Tunnel Client Account" Add-LocalGroupMember -Group "Users" -Member "tunnel-client"`

3. **Configure SSH Server**: ```powershell # Edit C:\ProgramData\ssh\sshd_config Add-Content -Path "C:\ProgramData\ssh\sshd_config" -Value @"

SSH Tunnel Configuration Match User tunnel-client AllowTcpForwarding yes GatewayPorts yes X11Forwarding no PermitTunnel no AllowAgentForwarding no ForceCommand /bin/false "@

Restart-Service sshd ```

1. **Generate SSH Keys**: ```powershell # Create SSH key pair for tunnel authentication ssh-keygen -t rsa -b 4096 -f "C:\Program Files\SSHTunnelServer\keys\tunnel_key" -N ""

Set up authorized_keys for tunnel-client $sshDir = "C:\Users\tunnel-client.ssh" New-Item -Path $sshDir -ItemType Directory -Force Copy-Item "C:\Program Files\SSHTunnelServer\keys\tunnel_key.pub" "$sshDir\authorized_keys" ```

Step 3: Configure and Start Server Services

1. **Configure Server Settings**: ```powershell # Copy and edit server configuration Copy-Item "C:\Program Files\SSHTunnelServer\config\server.conf.example" "C:\Program Files\SSHTunnelServer\config\server.conf"

Edit configuration as needed notepad "C:\Program Files\SSHTunnelServer\config\server.conf" ```

1. **Start SSH Tunnel Server**: `powershell cd "C:\Program Files\SSHTunnelServer\src\server" .\WebAPI.ps1 -Port 8080`

2. **Install as Windows Service** (optional): ```powershell # Create service wrapper script $servicePath = "C:\Program Files\SSHTunnelServer\service\TunnelServerService.ps1"

Install using NSSM or create scheduled task schtasks /create /tn "SSH Tunnel Server" /tr "powershell.exe -File '$servicePath'" /sc onstart /ru SYSTEM ```

Client Installation

Step 1: Prepare Client Systems

1. **Verify Prerequisites**: ```powershell # Check PowerShell version $PSVersionTable.PSVersion

Check SSH client availability ssh -V

Verify administrator privileges ([Security.Principal.WindowsPrincipal][Security.Principal.WindowsIdentity]::GetCurrent()).IsInRole([Security.Principal.WindowsBuiltInRole]"Administrator") ```

Step 2: Install SSH Tunnel Client

1. **Download Client Package** from the server web interface or copy installation files
2. **Run Installation Script**: `powershell # Run as Administrator Set-ExecutionPolicy -ExecutionPolicy RemoteSigned -Scope CurrentUser .\Install-SSHTunnelClient.ps1 -ServerHost "tunnel.company.com" -AutoStart`
3. **Manual Installation** (if needed): ```powershell # Create installation directory New-Item -Path "C:\Program Files\SSHTunnelClient" -ItemType Directory -Force

Copy client files Copy-Item -Path ".\src\client*" -Destination "C:\Program Files\SSHTunnelClient\bin\" -Recurse Copy-Item -Path ".\src\common*" -Destination "C:\Program Files\SSHTunnelClient\bin\" -Recurse

Copy SSH private key Copy-Item -Path ".\keys\tunnel_key" -Destination "C:\Program Files\SSHTunnelClient\keys\" ```

Step 3: Configure Client

1. **Create Configuration File**: ```powershell # Copy and edit client configuration Copy-Item "C:\Program Files\SSHTunnelClient\config\client.conf.example" "C:\Program Files\SSHTunnelClient\config\client.conf"

Update server settings $config = Get-Content "C:\Program Files\SSHTunnelClient\config\client.conf" -Raw $`config =` $config -replace "tunnel.company.com", "your-server.domain.com" Set-Content "C:\Program Files\SSHTunnelClient\config\client.conf" -Value $config ```

1. **Test Connection**: `powershell cd "C:\Program Files\SSHTunnelClient\bin" Import-Module .\SSHTunnelClient.psm1 Test-SSHTunnelConnection -ConfigFile "..\config\client.conf"`

Step 4: Install as Service

1. **Install Windows Service**: `powershell cd "C:\Program Files\SSHTunnelClient\bin" .\SSHTunnelService.ps1 -Action Install`
2. **Start Service**: ```powershell # Using PowerShell Start-Service -Name "SSHTunnelClient"

Or using Service Manager services.msc ```

1. **Verify Service Status**: `powershell Get-Service -Name "SSHTunnelClient" Get-EventLog -LogName Application -Source "SSHTunnelClient" -Newest 10`

Network Discovery Configuration

DNS-Based Discovery

1. **Configure DNS SRV Records**: `dns _ssh-tunnel._tcp.company.com. 300 IN SRV 10 5 22 tunnel.company.com. _ssh-tunnel._tcp.company.com. 300 IN SRV 20 5 443 backup-tunnel.company.com.`
2. **Test DNS Discovery**: `powershell Resolve-DnsName -Name "_ssh-tunnel._tcp.company.com" -Type SRV`

Broadcast Discovery

1. **Configure Server Broadcast Response**: `powershell # Enable broadcast discovery on server $config = @{ BroadcastDiscovery = @{ Enabled = $true Port = 9999 ResponsePort = 22 } }`
2. **Test Broadcast Discovery**: `powershell # From client Import-Module .\NetworkDiscovery.ps1 Find-ServersByBroadcast -Timeout 10`

Web Dashboard Access

Accessing the Dashboard

1. **Open Web Browser** and navigate to: `http://your-server:8080`
2. **Dashboard Features**:
3. **Clients Tab**: View all connected clients and their status
4. **Tunnels Tab**: Monitor active tunnels and port usage
5. **Monitoring Tab**: View connection history and system health

6. **Deployment Tab**: Download client installation packages

Dashboard Configuration

1. **Enable HTTPS** (recommended for production): ```powershell # Generate SSL certificate New-SelfSignedCertificate -DnsName "tunnel.company.com" -CertStoreLocation "cert:\LocalMachine\My"

Configure HTTPS in server settings $`config.WebInterface.SSL =` $true $config.WebInterface.Port = 8443 ```

1. **Configure Authentication** (optional): `powershell # Add basic authentication $config.WebInterface.Authentication = @{ Enabled = $true Type = "Basic" Users = @{ "admin" = "hashed_password" } }`

Troubleshooting

Common Issues

1. **Client Cannot Connect to Server**: ```powershell # Test network connectivity Test-NetConnection -ComputerName "tunnel.company.com" -Port 22

Test SSH authentication ssh -i "C:\Program Files\SSHTunnelClient\keys\tunnel_key" tunnel-client@tunnel.company.com

Check firewall rules Get-NetFirewallRule -DisplayName "*SSH*" ```

1. **Tunnels Not Establishing**: ```powershell # Check SSH server configuration Get-Content "C:\ProgramData\ssh\sshd_config" | Select-String -Pattern "tunnel-client" -Context 5

Verify port forwarding settings ssh -i "key" -R 10022:localhost:22 tunnel-client@server ```

1. **Service Not Starting**: ```powershell # Check service status Get-Service -Name "SSHTunnelClient" | Format-List *

View service logs Get-EventLog -LogName Application -Source "SSHTunnelClient" -Newest 20

Test service manually .\SSHTunnelService.ps1 -Action Test ```

Log Files

- **Client Logs**: `C:\Program Files\SSHTunnelClient\logs\`
- **Server Logs**: `C:\Program Files\SSHTunnelServer\logs\`
- **Windows Event Log**: Application log, source "SSHTunnelClient"

Diagnostic Commands

```
# Test client configuration
Import-Module SSHTunnelClient
Test-ClientConfiguration

# Test network discovery
Import-Module NetworkDiscovery
Find-SSHTunnelServers -NetworkRange "192.168.1.0/24"

# Check tunnel status
Get-SSHTunnelStatus

# View active connections
netstat -an | findstr :22
```

Security Considerations

Authentication

- **SSH Key Authentication**: Use strong RSA keys (4096-bit minimum)
- **Key Rotation**: Regularly rotate SSH keys
- **User Isolation**: Use dedicated user account for tunnel clients
- **Access Control**: Limit SSH user permissions

Network Security

- **Firewall Rules**: Restrict access to SSH and web interface ports
- **VPN Integration**: Consider VPN for additional security layer
- **Network Segmentation**: Isolate tunnel traffic where possible
- **Monitoring**: Log and monitor all tunnel connections

System Security

- **Service Account**: Run services with minimal required privileges
- **File Permissions**: Secure configuration and key files
- **Updates**: Keep SSH software and PowerShell updated
- **Auditing**: Enable audit logging for SSH connections

Maintenance

Regular Tasks

1. **Monitor System Health**: ```powershell # Check service status Get-Service -Name "SSHTunnelClient", "SSHTunnelServer"

Review logs Get-EventLog -LogName Application -Source "SSHTunnelClient" -After (Get-Date).AddDays(-1) ```

1. **Update Client Software**: ```powershell # Stop service Stop-Service -Name "SSHTunnelClient"

Update files Copy-Item -Path ".\new-version*" -Destination "C:\Program Files\SSHTunnelClient\" -Recurse -Force

Start service Start-Service -Name "SSHTunnelClient" ```

1. **Backup Configuration**: `powershell # Backup client configuration Copy-Item "C:\Program Files\SSHTunnelClient\config\*" "\\backup-server\ssh-tunnel-configs\$(Get-Date -Format 'yyyy-MM-dd')\"`

Performance Monitoring

- **Connection Count**: Monitor number of active tunnels
- **Resource Usage**: Track CPU and memory usage
- **Network Bandwidth**: Monitor tunnel traffic
- **Error Rates**: Track connection failures and retries

Advanced Configuration

Load Balancing

Configure multiple tunnel servers for high availability:

```
$config.ParentServers = @(
    @{ Host = "tunnel1.company.com"; Port = 22; Priority = 1 }
    @{ Host = "tunnel2.company.com"; Port = 22; Priority = 2 }
    @{ Host = "tunnel3.company.com"; Port = 443; Priority = 3 }
)
```

Custom Port Mappings

Configure specific port mappings for services:

```
$config.PortMappings = @{
    22 = 10022      # SSH
    3389 = 13389    # RDP
    80 = 10080      # HTTP
    443 = 10443     # HTTPS
    5985 = 15985    # WinRM
}
```

Proxy Configuration

Configure proxy settings for environments with restricted internet access:

```
$config.ProxySettings = @{
    Enabled = $true
    Type = "HTTP"
    Host = "proxy.company.com"
    Port = 8080
    Username = "proxy-user"
    Password = "proxy-password"
}
```

This completes the deployment guide for the SSH Tunnel Management System. The system provides a robust, secure, and manageable solution for network administrators who need reliable remote access to distributed systems.