Python Scripte

- VLAN ID aus Ethernetframe extrahieren
- Python Script Portchecker

VLAN ID aus Ethernetframe extrahieren

Beschreibung:

Ein kleines python Script das von einer entzwerkkarte empfangende Pakete die VLAN ID ausliste

Das Programm:

Abhängigkeiten installieren

```
apt install python3-args python3-scapy
```

Quelltext:

```
import argparse
from scapy.all import *

def extract_vlan_id(packet):
    if packet.haslayer(Dot1Q):
        vlan_id = packet[Dot1Q].vlan
        print(f"VLAN ID found: {vlan_id}")
    else:
        print("No VLAN found")

def main():
    parser = argparse.ArgumentParser(description="Extract VLAN ID from Ethernet frames.")
    parser.add_argument("-d", "--device", type=str, required=True, help="Network interface to listen on")
    args = parser.parse_args()

    print(f"Listening on interface: {args.device}")
    sniff(iface=args.device, prn=extract_vlan_id)
```

```
if __name__ == "__main__":
    main()
```

Python Script Portchecker

Beschreibung:

Ein Script das zu einem port verbindet und schaut ober der offen ist.

Das Script

```
#!/usr/bin/env python3
"""
Check_MK Spoolserver Connection Test
Tests TCP connection to a Check_MK spoolserver on port 6555
"""

import socket
import sys
import time

def test_connection(host, port=6555, timeout=5):
    """
    Tests TCP connection to the specified host and port

    Args:
        host: Hostname or IP address
        port: Port number (default: 6555)
        timeout: Connection timeout in seconds (default: 5)

    Returns:
        True if connection successful, False otherwise
    """
    print(f"Testing connection to {host}:{port}...")
    print(f"Timeout: {timeout} seconds")
    print("-" * 50)

    start_time = time.time()

    try:
        # Create socket
```

```
        sock = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
        sock.settimeout(timeout)

        # Try to connect
        result = sock.connect_ex((host, port))

        elapsed_time = time.time() - start_time

        if result == 0:
            print(f"✓ SUCCESS: Connection established in {elapsed_time:.2f} seconds")
            print(f"  Remote address: {sock.getpeername()}")
            sock.close()
            return True
        else:
            print(f"✗ FAILED: Could not connect (Error code: {result})")
            print(f"  Time elapsed: {elapsed_time:.2f} seconds")
            sock.close()
            return False

    except socket.timeout:
        print(f"✗ TIMEOUT: Connection timed out after {timeout} seconds")
        return False
    except socket.gaierror as e:
        print(f"✗ DNS ERROR: Could not resolve hostname: {e}")
        return False
    except ConnectionRefusedError:
        print(f"✗ CONNECTION REFUSED: Target actively refused the connection")
        return False
    except Exception as e:
        print(f"✗ ERROR: {type(e).__name__}: {e}")
        return False
    finally:
        try:
            sock.close()
        except:
            pass

def main():
    if len(sys.argv) < 2:
        print("Usage: python3 check_spoolserver.py <hostname_or_ip> [port] [timeout]")
```

```
        print("\nExamples:")
        print("  python3 check_spoolserver.py monitoring.example.com")
        print("  python3 check_spoolserver.py 192.168.1.100")
        print("  python3 check_spoolserver.py monitoring.example.com 6555 10")
        sys.exit(1)

    host = sys.argv[1]
    port = int(sys.argv[2]) if len(sys.argv) > 2 else 6555
    timeout = int(sys.argv[3]) if len(sys.argv) > 3 else 5

    success = test_connection(host, port, timeout)

    print("-" * 50)
    if success:
        print("Result: Firewall rules appear to be correct ✓")
        sys.exit(0)
    else:
        print("Result: Connection failed - check firewall rules ✗")
        sys.exit(1)

if __name__ == "__main__":
    main()
```