Day 26 - Remotely Turning on PCs With Wake-On-Lan

What is Wake-on-LAN?

Wake-on-LAN (WOL) is a hardware and software feature that allows a powered-off or sleeping computer to be turned on remotely over a Local Area Network (LAN). The mechanism works by sending a specially crafted network packet, known as a **magic packet**, to the target device's **MAC address**.

What is a Magic Packet?

A magic packet is a broadcast frame containing:

- 6 bytes of `FF` (255)
- Followed by the MAC address of the device repeated 16 times (16 × 6 bytes)

The system's network interface card (NIC), if configured to do so, listens for such packets even when the system is in a low-power state and will trigger a wake-up if a matching packet is received.

How to Check if Wake-on-LAN is Supported and Enabled

Check if NIC supports WOL

```
sudo ethtool <interface>
```

Example:

```
sudo ethtool eno1
```

Look for the line:

```
Supports Wake-on: pumg
Wake-on: g
```

- `g` means WOL via magic packet is supported.

```
[root@Rockylinux1 ~]# ethtool eno1
Settings for eno1:
        Supported ports: [ TP ]
        Supported link modes:   10baseT/Half 10baseT/Full
                                100baseT/Half 100baseT/Full
                                1000baseT/Full
        Supported pause frame use: Symmetric Receive-only
        Supports auto-negotiation: Yes
        Supported FEC modes: Not reported
        Advertised link modes:  10baseT/Half 10baseT/Full
                                100baseT/Half 100baseT/Full
                                1000baseT/Full
        Advertised pause frame use: Symmetric Receive-only
        Advertised auto-negotiation: Yes
        Advertised FEC modes: Not reported
        Link partner advertised link modes:  10baseT/Half 10baseT/Full
                                             100baseT/Half 100baseT/Full
        Link partner advertised pause frame use: No
        Link partner advertised auto-negotiation: Yes
        Link partner advertised FEC modes: Not reported
        Speed: 100Mb/s
        Duplex: Full
        Auto-negotiation: on
        Port: Twisted Pair
        PHYAD: 1
        Transceiver: internal
        MDI-X: on (auto)
        Supports Wake-on: pumbg
        Wake-on: g
        Current message level: 0x00000007 (7)
                               drv probe link
        Link detected: yes
```

Wake-on-lan is enabled

- If it's `d`, it's disabled. You can enable it as shown below.

```
sudo ethtool -s eth0 wol g
```

Sending a Magic Packet (Linux)

Ubuntu/Debian

```
sudo apt install wakeonlan
wakeonlan <mac-address>
```

RHEL/Rocky Linux

```
sudo dnf install perl-Net-Wake
wakeonlan <mac-address>
```

Python Script to Send Magic Packet

You can also use a simple Python script to send a magic packet.

Script: `arise.py`

```python
#!/usr/bin/env python3

import socket
import sys

MACS = {
    "rocky1": "00:11:22:33:44:55"
}

def wake_on_lan(mac):
    mac_bytes = bytes.fromhex(mac.replace(':', '').replace('-', ''))
    magic_packet = b'\xff' * 6 + mac_bytes * 16
    with socket.socket(socket.AF_INET, socket.SOCK_DGRAM) as s:
        s.setsockopt(socket.SOL_SOCKET, socket.SO_BROADCAST, 1)
        s.sendto(magic_packet, ('<broadcast>', 9))

if __name__ == "__main__":
    if len(sys.argv) < 2:
        print("Usage: arise <device-name>")
        sys.exit(1)

    target = sys.argv[1]
    if target not in MACS:
        print(f"Unknown device '{target}'. Available: {', '.join(MACS)}")
        sys.exit(1)

    wake_on_lan(MACS[target])
```

Installation and Usage

1. Save the script as `arise.py`
2. Make it executable:

```
chmod +x arise.py
```

3. Move it to your PATH:

```
sudo mv arise.py /usr/local/bin/arise
```

Now you can wake a machine using:

```
arise rocky1
```

Sample Working

On the target PC, issue a shutdown command using:

```
systemctl poweroff --now
```

```
[root@Rockylinux1 ~]# systemctl poweroff --now
[root@Rockylinux1 ~]# Connection to 192.168.0.104 closed by remote host.
Connection to 192.168.0.104 closed.
fatimasohail@fatimas-MacBook-Air-3 bin %
```

The system is no longer accessible, as verified by ping.

```
ping 192.168.0.104
```

Then run the script, and after a few seconds, the host will be reachable.