

OPC UA Troubleshooting & Integration Checklist

A field-tested diagnostic flow for connecting any PLC — network, ports, certificates, namespace and status codes — plus a Python (asyncua) client template.

Diagnostic flow (top to bottom)

- **Network** — can you ping/route to the server host? Same VLAN or firewall rule in place?
- **Port** — default is 4840 (TCP). Open it end-to-end; confirm the endpoint URL: `opc.tcp://host:4840`.
- **Endpoint & security policy** — list endpoints; match security policy (None / Basic256Sha256) and mode (Sign / SignAndEncrypt).
- **Certificates** — exchange and trust client/server certs; place them in the trusted store. Most 'BadSecurityChecksFailed' errors are trust issues.
- **Authentication** — anonymous vs username/password vs certificate; confirm the server allows your method.
- **Namespace & NodeIds** — namespace index can change between server versions; resolve NodeIds by browse name, don't hard-code ns index.
- **Session/subscription** — check session timeout and publishing interval; watch for `BadTooManySessions`.
- **Status codes** — read the code: `BadConnectionRejected` (port/firewall), `BadCertificateUntrusted` (trust), `BadIdentityTokenRejected` (auth), `BadNodeIdUnknown` (namespace).

Common errors → fix

Symptom	Likely cause	Fix
Connection refused / timeout	Port 4840 blocked; wrong host	Open firewall; verify endpoint URL
<code>BadCertificateUntrusted</code>	Cert not trusted	Trust client cert on server (and vice versa)
<code>BadSecurityChecksFailed</code>	Policy/mode mismatch	Match security policy & message mode
<code>BadIdentityTokenRejected</code>	Auth method not allowed	Use allowed user/anon/cert method
<code>BadNodeIdUnknown</code>	Namespace index shifted	Resolve NodeId by browse name
Data stale / no updates	Publishing interval / sampling	Tune subscription & sampling interval

Python client template (asyncua)

```
import asyncio
from asyncua import Client

URL = "opc.tcp://192.168.0.10:4840"
async def main():
    async with Client(url=URL) as c:
        # c.set_security(...) for Sign&Encrypt
        node = await c.nodes.root.get_child(["0:Objects", "2:MyDevice", "2:Temperature"])
        print(await node.read_value())
    asyncio.run(main())
```