GOPLC Datacenter Infrastructure Simulation

Virtualized Gateway Architecture

Eliminating Edge Hardware with Server-Based Gateways and Routed VLANs

Author: James Belcher **Date:** February 2026 **Version:** 1.0

Companion to: WHITEPAPER_DC_SIMULATION_HARDWARE.md

Executive Summary

The hardware gateway architecture (described in the companion white paper) deploys one dedicated edge SBC per physical device — a model that provides maximum fault isolation but incurs total costs (infrastructure plus runtime licensing) of $4.2M to $118M for facilities ranging from 50 MW to 1 GW.

This white paper describes an alternative: **all GOPLC gateway instances run as virtual machines or containers on existing server infrastructure**, reaching field devices across routed VLANs via Layer 3 switching. No edge hardware is required. The field device network (typically 192.168.x.x or 10.x.x.x) connects to an access-layer switch whose uplink trunks to a Cisco distribution switch performing inter-VLAN routing. Gateway VMs on server-side VLANs reach field devices through standard IP routing — no dedicated router at the edge.

This approach reduces total deployment cost (infrastructure plus GOPLC runtime licensing) by 51–66% compared to the hardware model, enables elastic scaling on commodity compute, and preserves the identical GOPLC software architecture — same binary, same ST programs, same OPC UA namespaces, same fleet management.

---

1. Motivation

1.1 The Cost Problem at Scale

A 100 MW datacenter in the 1:1 hardware gateway model requires ~6,200 edge SBCs, 6,200+ switch ports, and installation labor totaling $5.8M–$9.4M in infrastructure, plus $2.5M in GOPLC runtime licensing — $8.3M–$11.8M total. At 500 MW or 1 GW, total costs (infrastructure plus software) approach $42M–$118M.

Meanwhile, the same datacenter already operates thousands of server-class compute nodes with virtualization infrastructure (VMware ESXi, Proxmox, KVM, or Kubernetes). These servers have available CPU, memory, and network capacity. The GOPLC binary requires <50 MB RAM and <0.1 CPU core per gateway instance. A single 32-core server can host hundreds of gateway instances.

1.2 The Network Problem

Edge SBCs live on the same Layer 2 network as the field devices — they can reach Modbus TCP, BACnet, and SNMP endpoints directly because they share a subnet. Moving gateways to server-side VMs means crossing network boundaries. The field device network (e.g., 192.168.10.0/24 for a row of CRACs) must be routable from the server VLAN (e.g., 10.100.0.0/16).

This is a solved problem in enterprise networking. It requires no special hardware beyond what already exists in the datacenter's own network infrastructure.

---

2. Network Architecture

2.1 The Key Question: Where Does Routing Live?

The user needs connectivity between: - **Server VLAN** (e.g., 10.100.0.0/16) — where gateway VMs run - **Field device VLANs** (e.g., 192.168.10.0/24 for CRAC row A, 192.168.11.0/24 for UPS room 1, etc.) — where physical devices live

There are three options for where the Layer 3 routing happens:

Option A: Cisco Distribution Switch with SVIs (Recommended) The Cisco distribution/aggregation switch (Catalyst 9300, 9500, Nexus 9000, etc.) already performs Layer 3 routing for the datacenter network. Add a Switch Virtual Interface (SVI) for each field device VLAN:

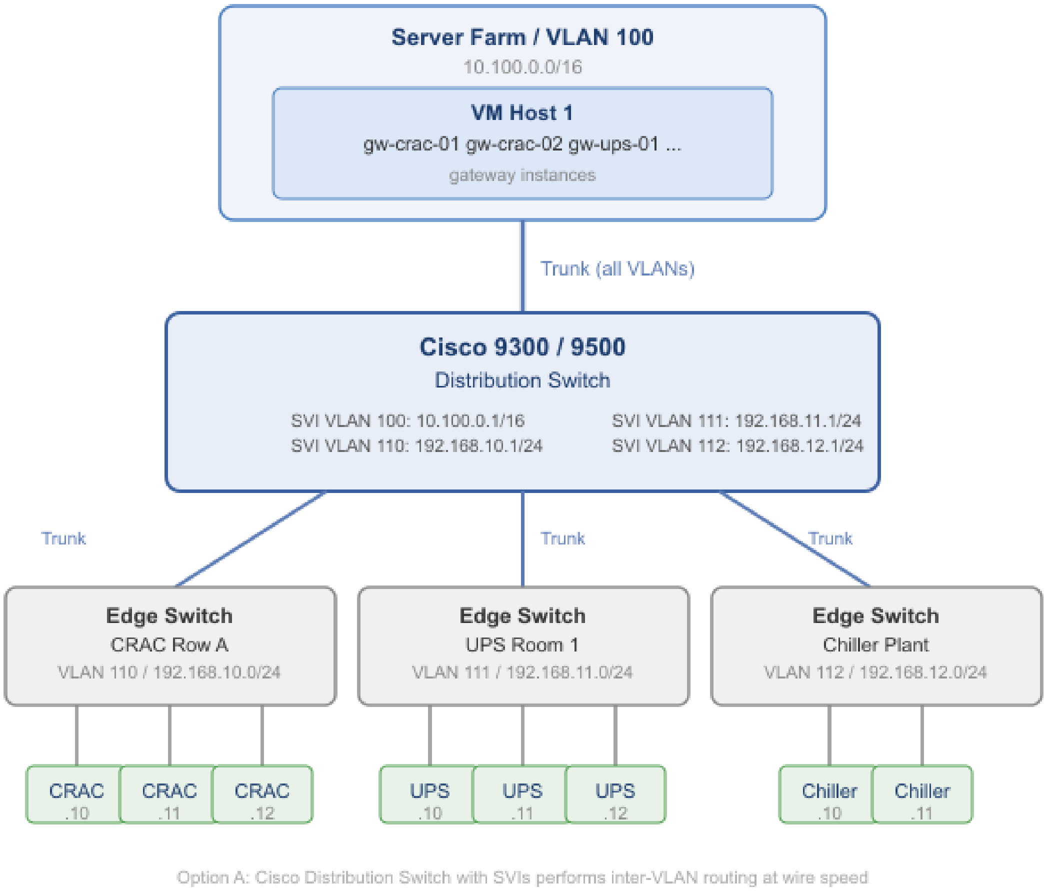

Server Farm / VLAN 100
10.100.0.0/16
VM Host 1
gw-crac-01 gw-crac-02 gw-ups-01 ...
gateway instances
Trunk (all VLANs)
Cisco 9300 / 9500
Distribution Switch
SVI VLAN 100: 10.100.0.1/16
SVI VLAN 110: 192.168.10.1/24
SVI VLAN 111: 192.168.11.1/24
SVI VLAN 112: 192.168.12.1/24
Trunk
Trunk
Trunk
Edge Switch
CRAC Row A
VLAN 110 / 192.168.10.0/24
Edge Switch
UPS Room 1
VLAN 111 / 192.168.11.0/24
Edge Switch
Chiller Plant
VLAN 112 / 192.168.12.0/24
CRAC .10
CRAC .11
CRAC .12
UPS .10
UPS .11
UPS .12
Chiller .10
Chiller .11
Option A: Cisco Distribution Switch with SVIs performs inter-VLAN routing at wire speed

Figure 1: Cisco distribution switch SVI topology: server farm VMs on VLAN 100 reach field devices on VLANs 110-112 via hardware-accelerated SVI routing

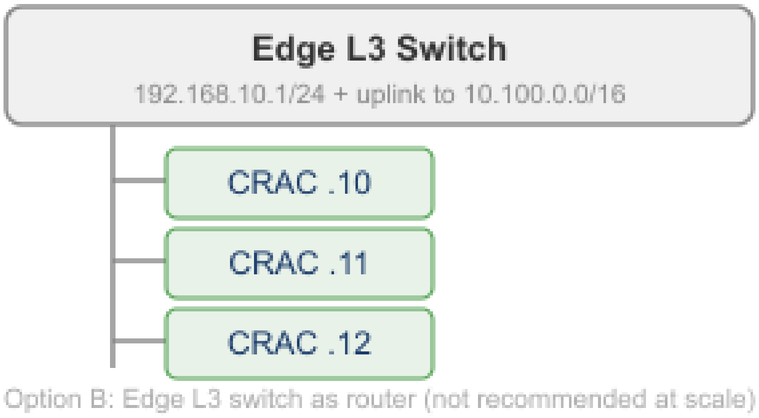
How it works: 1. Field devices connect to unmanaged (or L2 managed) edge switches 2. Edge switches carry a single VLAN (e.g., VLAN 110 = 192.168.10.0/24) 3. Edge switch uplink trunks to the Cisco distribution switch 4. Cisco switch has an SVI for each field VLAN — it **is** the default gateway for those subnets 5. Server VMs on VLAN 100 send packets to field devices; Cisco routes between VLANs at wire speed using hardware ASIC forwarding 6. No separate router needed anywhere

Cisco IOS-XE configuration (example):

```
! Field device VLANs
vlan 110
  name CRAC-ROW-A
vlan 111
  name UPS-ROOM-1
vlan 112
  name CHILLER-PLANT

! SVIs - the Cisco switch IS the router
interface Vlan110
  ip address 192.168.10.1 255.255.255.0
  no shutdown
interface Vlan111
  ip address 192.168.11.1 255.255.255.0
  no shutdown
interface Vlan112
  ip address 192.168.12.1 255.255.255.0
  no shutdown

! Server VLAN
interface Vlan100
  ip address 10.100.0.1 255.255.0.0
  no shutdown

! Enable IP routing
ip routing

! Trunk to edge switches
interface GigabitEthernet1/0/1
  description CRAC-ROW-A-EDGE-SW
  switchport mode trunk
  switchport trunk allowed vlan 110

interface GigabitEthernet1/0/2
  description UPS-ROOM-1-EDGE-SW
  switchport mode trunk
  switchport trunk allowed vlan 111
```

Latency: Hardware-accelerated SVI routing on Catalyst 9000 series adds <10us per hop. This is invisible to Modbus TCP (100ms+ scan times) and all other field protocols.

Option B: Edge Switch as L3 Router If the edge switch is a managed L3 switch (e.g., Cisco IE-3400, Aruba CX, HPE FlexNetwork), it can perform routing locally. This keeps field device traffic off the distribution switch but adds cost and complexity to every edge location.



Option B: Edge L3 switch as router (not recommended at scale)

Figure 2: Option B: Edge L3 switch performing local routing between field subnet and server VLAN

Not recommended at scale: every edge switch needs L3 licensing, IP address management, and routing protocol participation (OSPF/EIGRP). The Cisco distribution switch already does this.

Option C: Dedicated Router at the Edge A small router (e.g., Cisco ISR 1000) at each edge location, routing between the field VLAN and the server VLAN.

Not recommended: Adds $500–$2,000 per edge location with no benefit over Option A. The distribution switch handles the same routing for free.

2.2 Recommendation

Option A — Cisco distribution switch with SVIs — is the correct answer for virtually all datacenter deployments. The switch already exists. SVIs are a configuration change, not a hardware purchase. Routing happens in hardware at wire speed. The edge switches remain simple unmanaged or L2-only devices.

2.3 Protocol-Specific Routing Considerations

All GOPLC-supported field protocols work over routed networks, with caveats:

Protocol	Transport	Routable?	Caveats
Modbus TCP	TCP/502	Yes	None — pure TCP point-to-point
BACnet/IP	UDP/47808	Yes	Broadcast-based discovery needs BBMD (BACnet Broadcast Management Device) or direct IP addressing. GOPLC uses direct IP — no BBMD needed.
SNMP v2c	UDP/161	Yes	None — standard UDP request/response
EtherNet/IP (explicit)	TCP/44818	Yes	Explicit messaging (reads/writes) routes fine

Protocol	Transport	Routable?	Caveats
EtherNet/IP (implicit I/O)	UDP multicast	Partial	Implicit (cyclic) I/O uses multicast; requires IGMP snooping on all intermediate switches. GOPLC uses explicit messaging for gateway polling — no multicast needed.
SEL ASCII	TCP/custom	Yes	None — pure TCP
IEC 60870-5-104	TCP/2404	Yes	None — designed for WAN operation
OPC UA	TCP/4840	Yes	None — designed for routed networks

Key point: GOPLC gateway programs use **point-to-point TCP/UDP connections** to specific device IP addresses. They do not rely on broadcast discovery, multicast, or link-local protocols. This means routing works transparently — the gateway VM at 10.100.5.20 connects to the CRAC at 192.168.10.12 via standard IP routing, and the Modbus TCP session behaves identically to a direct-connect scenario.

2.4 VLAN Design for Datacenter Equipment

A typical VLAN layout for a 50 MW facility:

VLAN	Subnet	Purpose	Devices
100	10.100.0.0/16	Server / VM infrastructure	Gateway VMs, aggregators
110	192.168.10.0/24	CRAC Row A	12–20 CRAC units
111	192.168.11.0/24	CRAC Row B	12–20 CRAC units
112–119	192.168.12–19.0/24	CRAC Rows C–J	12–20 each
120	192.168.20.0/24	UPS Room 1	15–25 UPS modules
121	192.168.21.0/24	UPS Room 2	15–25 UPS modules
130	192.168.30.0/24	Generator yard	25–35 generators + ATS
140	192.168.40.0/24	Chiller plant	20–40 chillers + cooling towers
150	192.168.50.0/23	Rack PDUs Floor 1	~500 PDUs
152	192.168.52.0/23	Rack PDUs Floor 2	~500 PDUs
160	192.168.60.0/23	Env sensors Floor 1	~250 sensor hubs
162	192.168.62.0/23	Env sensors Floor 2	~250 sensor hubs
170	192.168.70.0/24	Power metering	100–200 meters
180	192.168.80.0/24	Fire alarm panels	5–10 panels

Total: ~20–40 VLANs for a 50 MW facility. A Cisco 9300-48P supports 4,094 VLANs — this is well within capacity even at 1 GW scale.

---

3. Server-Side Gateway Deployment

3.1 Resource Requirements per Gateway Instance

GOPLC gateways are lightweight:

Resource	Per Instance	Notes
CPU	0.05–0.1 core	100ms scan cycle with Modbus TCP read/write

Resource	Per Instance	Notes
Memory	30–50 MB	ST interpreter + OPC UA server + protocol client
Disk	<10 MB	Binary + config + project file
Network	<10 Kbps	Modbus TCP polling at 10 Hz, 12 registers

3.2 Server Sizing

A single modern server (e.g., Dell PowerEdge R750, 2× Xeon Gold 6348, 512 GB RAM) can host:

Deployment Method	Instances per Server	Limiting Factor
Docker containers	500–1,000	Network sockets / CPU
KVM/QEMU VMs (minimal Linux)	100–200	RAM (256 MB/VM overhead)
Bare-metal (in-process cluster)	200–500	GOPLC in-process cluster goroutines

3.3 VM Topology Options

Option 1: One VM per Device Category, Multi-Device Polling (Recommended) Each VM runs a single GOPLC instance in cluster mode with one minion per physical device. The boss provides a unified API for all devices of that type:

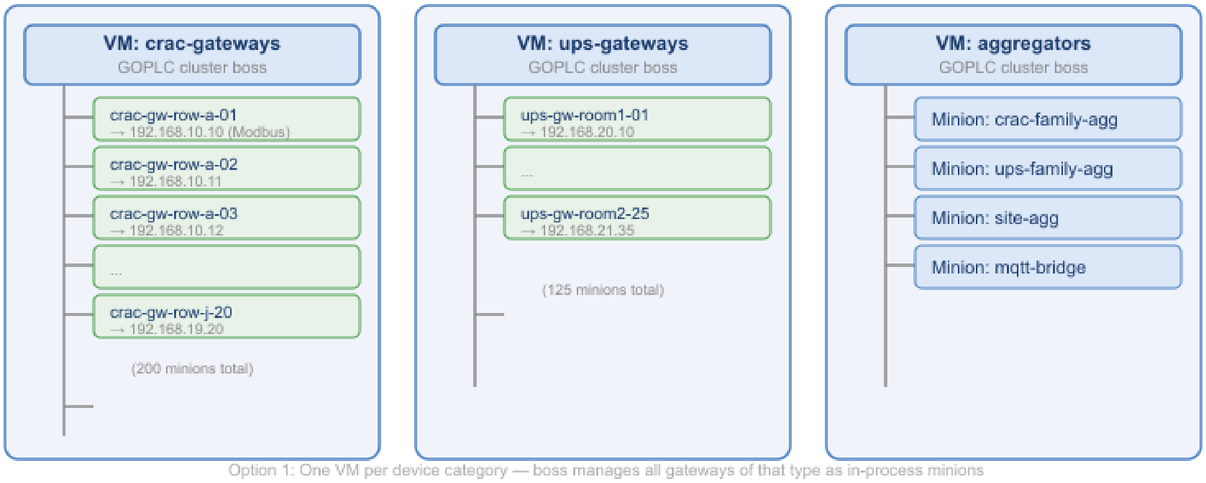


Figure 3: VM topology: three boss VMs managing CRAC gateways, UPS gateways, and aggregators as in-process minion goroutines

Option 2: Docker Compose per Tier (Simplest) Use Docker Compose to orchestrate all gateway containers on one or more Docker hosts:

```yaml
# docker-compose.crac-gateways.yml
services:
  crac-gw-01:
    image: goplc:latest
    volumes:
      - ./configs/crac-gw-01:/app/projects
    environment:
      - GOPLC_DEVICE_HOST=192.168.10.10
    restart: unless-stopped

  crac-gw-02:
    image: goplc:latest
    volumes:
      - ./configs/crac-gw-02:/app/projects
    environment:
      - GOPLC_DEVICE_HOST=192.168.10.11
    restart: unless-stopped
  # ... repeat for all CRACs
```

Option 3: Kubernetes (Enterprise Scale) At 500 MW+ with thousands of gateway pods, Kubernetes provides: - Automatic rescheduling on node failure - Rolling updates across the fleet - Resource limits and QoS per pod - Helm charts for templated deployment (one chart, N values files) - Prometheus metrics scraping from each gateway's `/api/info` endpoint

3.4 High Availability

Server-based deployment enables HA patterns impossible with dedicated edge SBCs:

HA Pattern	Description
Active/standby VM pair	Two VMs with the same gateway config; standby monitors active via heartbeat
Kubernetes pod restart	Failed pod automatically rescheduled to healthy node (seconds)
Docker restart policy	`restart: unless-stopped` auto-restarts crashed containers
vMotion / live migration	Move running gateway VMs between physical hosts during maintenance
N+1 server redundancy	Lose one server, remaining servers absorb its gateway load

Compare to the hardware model: a failed ctrlX CORE SBC requires physical replacement (hours to days for shipping, rack-and-stack, configuration). A failed gateway VM restarts in seconds.

---

4. Scale Estimates — Virtualized Model

4.1 Server Count

Using the practical grouping ratios from the hardware white paper and Docker container density of 500 instances per server:

Facility Size	Gateway Instances (practical)	Servers Required	Est. Server Cost
50 MW	~750	2	$20K–$40K
100 MW	~1,400	3	$30K–$60K
500 MW	~6,600	14	$140K–$280K
1 GW	~13,000	26	$260K–$520K

4.2 Cost Comparison: Hardware vs. Virtualized

Facility	HW Infra	HW Software	**HW Total**	Virtual Infra	Virtual Software	**Virtual Total**	**Savings**
50 MW	$713K–$1.125M	$300K	**$1.01M–$1.43M**	$32K–$70K	$450K	**$482K–$520K**	**52–66%**
100 MW	$1.33M–$2.10M	$560K	**$1.89M–$2.66M**	$48K–$108K	$840K	**$888K–$948K**	**54–64%**
500 MW	$6.27M–$9.90M	$2.64M	**$8.91M–$12.54M**	$210K–$475K	$3.96M	**$4.17M–$4.44M**	**50–65%**
1 GW	$12.35M–$19.50M	$5.20M	**$17.55M–$24.70M**	$385K–$880K	$7.80M	**$8.19M–$8.68M**	**51–53%**

*Hardware model: ctrlX CORE SBC at $600–700/unit, GOPLC standalone license at $400/instance, plus switches and labor. Virtual model: commodity servers + network config + GOPLC cluster license at $600/instance. Both use practical (grouped) instance counts.

The server infrastructure likely already exists. Most hyperscale datacenters have available VM capacity on management/infrastructure clusters. The incremental cost to run gateway containers may be effectively $0 beyond the network configuration labor.

4.3 Network Infrastructure Cost

The Cisco distribution switch already exists. The costs are:

Item	50 MW	100 MW	500 MW	1 GW
Edge access switches (unmanaged, 8–24 port)	$5K–$15K	$10K–$30K	$50K–$150K	$100K–$300K
Cisco SVI configuration labor	$2K–$5K	$3K–$8K	$10K–$25K	$15K–$40K
VLAN design and documentation	$5K–$10K	$5K–$10K	$10K–$20K	$10K–$20K
Network total	**$12K–$30K**	**$18K–$48K**	**$70K–$195K**	**$125K–$360K**

The edge access switches are the only physical hardware required at each device location. These are commodity unmanaged switches ($50–$200 each) providing Ethernet connectivity to field devices that may currently use point-to-point serial or have no network connection at all.

4.4 Software Licensing Comparison

GOPLC runtime licensing is the same cost driver in both architectures — the same number of logical gateways must be licensed regardless of whether they run on edge SBCs or server VMs. With the revised pricing, cluster licensing ($600/instance) is $200/instance **more expensive** than standalone ($400/instance). Virtual deployments therefore pay a software premium relative to the hardware model.

Facility	Standalone ($400/instance)	Cluster ($600/instance)	Cluster Premium
50 MW	$300K	$450K	+$150K
100 MW	$560K	$840K	+$280K

Facility	Standalone ($400/instance)	Cluster ($600/instance)	Cluster Premium
500 MW	$2.64M	$3.96M	+$1.32M
1 GW	$5.20M	$7.80M	+$2.60M

The software premium is real but dwarfed by the infrastructure savings. At 1 GW, virtual deployments pay $2.6M more in licensing than hardware deployments, while saving $11.2M–$18.6M in edge SBCs, switches, and installation labor. The net result is a 51–53% total cost reduction — economics driven by hardware elimination, not licensing discounts.

5. Standalone vs. Cluster in the Virtualized Model

The choice between standalone and cluster mode changes when gateways are virtual:

5.1 Standalone: One Container per Device

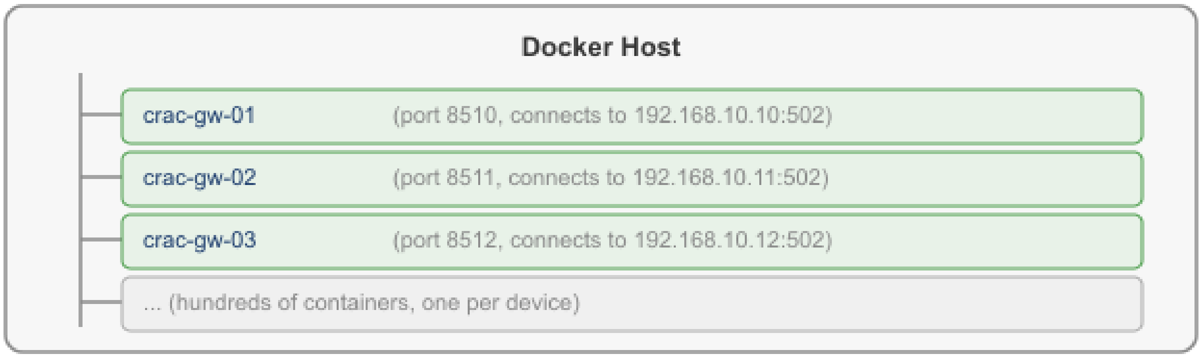

Docker Host
crac-gw-01
(port 8510, connects to 192.168.10.10:502)
crac-gw-02
(port 8511, connects to 192.168.10.11:502)
crac-gw-03
(port 8512, connects to 192.168.10.12:502)
... (hundreds of containers, one per device)

Figure 4: Standalone container topology: one Docker container per device, each with its own API port and device connection

Each container has its own config, API port, and lifecycle. Fleet management (`/api/fleet/discover`) or Docker Compose provides the management layer.

Advantages in the virtual model: - Container orchestration (Docker Compose, Kubernetes) handles lifecycle - Each container is independently restartable - Resource isolation via cgroups - Familiar to DevOps teams

Disadvantages in the virtual model: - Port allocation: hundreds of unique API ports to manage - Monitoring: must scrape hundreds of individual `/api/info` endpoints - Config management: hundreds of config files (mitigated by templates)

5.2 Cluster: Boss per Device Category

Each boss manages all gateways of its device type. Minions run as goroutines within the boss process (in-process cluster mode).

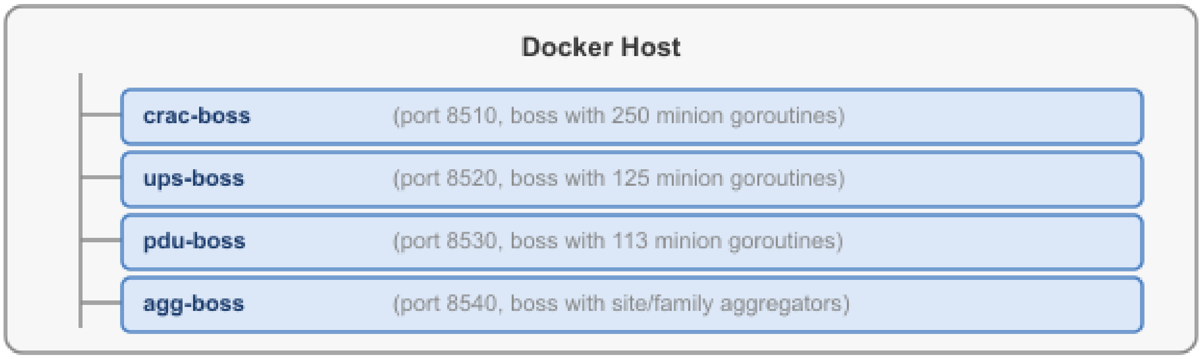

Docker Host
crac-boss
(port 8510, boss with 250 minion goroutines)
ups-boss
(port 8520, boss with 125 minion goroutines)
pdu-boss
(port 8530, boss with 113 minion goroutines)
agg-boss
(port 8540, boss with site/family aggregators)

Figure 5: Cluster container topology: four boss containers each managing hundreds of minion goroutines, reducing 750 containers to 4

Advantages in the virtual model: - 4 containers instead of 750 for a 50 MW facility - One API port per device category — simple monitoring - Boss proxy provides unified API: `GET /api/cluster/crac-gw-147/api/variables` - DataLayer sharing between minions (family aggregator reads minion variables directly) - Single config per boss with member list generated from fleet inventory

Disadvantages in the virtual model: - Boss crash takes down all minions of that device type - Memory footprint: 250 minion goroutines use more RAM than 250 containers (in practice both fit easily in server memory) - Less familiar to DevOps teams than per-container model

5.3 Recommendation for Virtualized Deployment

Scale	Recommended Topology
< 100 gateways	Standalone containers with Docker Compose
100–1,000 gateways	Cluster mode: one boss per device category (4–10 bosses)
1,000–10,000 gateways	Cluster mode + Kubernetes: boss pods with horizontal scaling
10,000+ gateways	Cluster mode + Kubernetes + regional sharding (boss per building/zone)

6. Migration Path: Hardware to Virtual

For organizations currently running hardware gateways (ctrlX CORE or similar) that want to migrate to the virtualized model:

6.1 Phase 1: Network Preparation

1. Assign VLANs to each field device network segment
2. Configure SVIs on the Cisco distribution switch
3. Verify IP routing from server VLAN to each field VLAN
4. Test with `ping` from a server-side VM to a field device IP

6.2 Phase 2: Parallel Deployment

1. Deploy gateway VMs/containers on server infrastructure
2. Point them at the same field devices as the hardware gateways
3. Run both in parallel — the hardware gateway and the VM gateway both poll the same device (Modbus TCP supports multiple clients; OPC UA and SNMP also support concurrent access)
4. Verify that the VM gateway produces identical variable values

6.3 Phase 3: Cutover

1. Redirect the family aggregator's OPC UA client from the hardware gateway's OPC UA port to the VM gateway's OPC UA port (runtime retargetable via API — no redeploy)
2. Verify data flow through the full tier stack
3. Power down the hardware gateway
4. Repeat for each device

6.4 Phase 4: Decommission

1. Remove hardware gateways from racks
2. Reclaim edge switch ports (or repurpose edge switches for field device connectivity)
3. Update fleet inventory to reflect virtual-only topology

The entire migration can be performed with **zero downtime** because both gateways can poll the same device simultaneously, and the aggregator's OPC UA target is retargetable at runtime.

---

7. Runtime Performance

Each GOPLC gateway instance is lightweight and deterministic. Measured performance on Intel i9-13900KS (January 2026):

Metric	Result
Memory per gateway instance (running ST)	16–50 MB
Modbus TCP throughput	89,769 requests/sec
DataLayer latency (shared memory)	1.7 us avg, 8.2 us p99
DataLayer latency (TCP LAN)	<1 ms P50, <3 ms P99
Scan execution (5,000 Modbus servers)	20–50 us avg
24-hour soak test	0 missed scans, 0 memory leaks

Gateway instances use ticker-mode scheduling (sleep between scans, zero idle CPU). A 100ms scan cycle consumes less than 1% of one CPU core per instance, enabling hundreds of gateway instances per server core in the virtualized deployment model.

The binary is statically linked with zero runtime dependencies. Docker image size is approximately 20 MB. Container overhead versus bare metal is zero — benchmarks show identical scan times and DataLayer latency inside Docker as on bare metal.

---

8. Reliability and Store-and-Forward

8.1 Dual-Path Communication

Every gateway tier can publish data through two independent paths simultaneously:

- **MQTT** (primary): Sub-second telemetry to the broker, QoS 0/1/2 with TLS
- **DNP3 outstation** (backup): Store-and-forward event buffering with original timestamps preserved in a local SQLite database encrypted with AES-256-GCM

When the MQTT path fails (network disruption, broker restart), the DNP3 outstation continues buffering all events with precise timestamps. On reconnection, the supervisor's DNP3 master retrieves the full ordered event history — zero data loss, timestamp integrity intact.

8.2 39 Built-In Resilience Functions

Beyond protocol-level redundancy, 39 built-in ST functions implement standard resilience patterns callable directly from gateway programs:

Pattern	Function	Purpose
Circuit breaker	`CIRCUIT_BREAKER()`	Stop polling a device that keeps failing
Retry with backoff	`RETRY_BACKOFF()`	Exponential retry for transient failures
Last-known-good cache	`LKG_CACHE()`	Return last valid value on read failure
Rate limiter	`RATE_LIMIT()`	Throttle write-back to protect device
Bulkhead	`BULKHEAD()`	Isolate one failing device from others

These patterns eliminate the need for custom exception handling in every gateway ST program. A CRAC gateway that loses Modbus connectivity automatically returns cached values northbound and stops hammering the device with retries.

8.3 OPC UA Client-Side Failover

Family aggregators connecting to device gateways via OPC UA implement heartbeat-based failover in ST. Detection time: 5 scan cycles (500ms at 100ms scan). Switchover: one scan cycle. Both primary and backup gateway endpoints are monitored; the aggregator switches transparently with no data gap.

---

9. Comparison with Existing Solutions

Capability	GOPLC	Ignition	Niagara	Custom Scripts
Native field protocols	12	10+ (licensed modules)	8+ (licensed drivers)	Per-script

Capability	GOPLC	Ignition	Niagara	Custom Scripts
IEC 61131-3 logic on gateway	Full ST, 1,450+ functions	SFC only	Limited	None
Edge deployment	Pi 5, any Linux, ctrlX snap	Java/x86 only	JVM only	Varies
Cluster mode (10,000+ nodes)	Built-in	Redundancy module ($)	N+1 Supervisor	Manual
Store-and-forward (zero data loss)	Built-in (SQLite + AES-256-GCM)	Add-on module	Journal	Must build
AI-assisted commissioning	Built-in (Claude)	None	None	None
Memory per gateway instance	16–50 MB running	~512 MB+	~256 MB+	Varies
Simulation / digital twin	Built-in (11 device types)	None	None	None
Licensing	Per-instance ($400 standalone / $600 cluster)	Per-tag	Per-instance	Open source

GOPLC's primary differentiator is the combination of lightweight deployment (16–50 MB versus hundreds of MB for JVM-based platforms), native protocol implementations with no external drivers or module licenses, programmable logic co-located with the gateway, and built-in device simulation for testing without physical hardware.

10. Security Considerations

7.1 Network Segmentation

The VLAN architecture provides natural segmentation:

- Field device VLANs should have **no default route to the internet**
- ACLs on the Cisco SVIs can restrict which server IPs may access which field VLANs
- Only gateway VMs need routes to field VLANs — all other server traffic is blocked

Example ACL:

```
ip access-list extended FIELD-DEVICE-ACCESS
  permit tcp 10.100.5.0 0.0.0.255 192.168.10.0 0.0.0.255 eq 502
  permit tcp 10.100.5.0 0.0.0.255 192.168.20.0 0.0.0.255 eq 502
  permit udp 10.100.5.0 0.0.0.255 192.168.40.0 0.0.0.255 eq 47808
  deny   ip any 192.168.0.0 0.0.255.255

interface Vlan110
  ip access-group FIELD-DEVICE-ACCESS in
```

7.2 GOPLC Authentication

GOPLC's optional authentication system (JWT-based, bcrypt passwords) protects the gateway APIs. In the virtualized model, the MQTT, InfluxDB, and OPC UA credentials should be managed via secrets (Kubernetes Secrets, Docker Secrets, or Vault).

7.3 IEC 62443 Considerations

Moving gateways from dedicated edge hardware to shared server infrastructure changes the IEC 62443 zone/conduit model. The field device network (Zone 1) is now accessed from the server infrastructure zone (Zone 3) through a routed conduit. This is architecturally identical to how existing SCADA/BMS servers access field devices and does not introduce new attack surface if ACLs are properly configured.

---

11. Trade-Offs Summary

Criterion	Hardware Gateways (Edge SBC)	Virtual Gateways (Server VM)
Capital cost	$4.2M–$118M (incl. software)	$482K–$8.7M (incl. software)
Fault isolation	Per-device (physical)	Per-container/VM (logical)
Deterministic timing	Real-time kernel, CPU affinity	Best-effort (adequate for 100ms+ scans)
Network latency to device	<1ms (same L2 segment)	1–5ms (L3 routed)
Scan time precision	±1ms	±5ms
Physical environment	-25°C to +60°C, DIN rail, 24V DC	Server room conditions
Scaling	Buy more SBCs	Add containers
Recovery time	Hours (hardware replacement)	Seconds (container restart)
Update rollout	Per-device, manual or fleet push	Rolling update, zero downtime
Operational complexity	Distributed hardware fleet	Centralized VM management
IEC 62443 compliance	Naturally zoned (same L2 as device)	Requires ACL-based zoning
ctrlX ecosystem integration	Native (Data Layer bridge)	Not available
Suitable for	Harsh environments, certified installations	Everything else

---

12. When to Use Which Architecture

Use Hardware Gateways When:

- **Regulatory compliance** requires dedicated, certified hardware (SIL, IEC 62443 SL-3+)
- **Environment** is outside server room conditions (generator yards, outdoor chiller plants)
- **ctrlX CORE ecosystem** integration is required (motion control, drive programming)
- **Deterministic timing** below 10ms is required (not typical for datacenter BMS)
- **Air-gapped networks** where field devices have no IP routing to server infrastructure

Use Virtual Gateways When:

- **Cost** is a primary concern (virtually always at hyperscale)

- **Server infrastructure** already exists with available capacity
- **Network routing** is available between server and field VLANs (standard in modern DCs)
- **Operational simplicity** is valued — centralized management, standard DevOps tooling
- **Elastic scaling** is needed — spin up 100 new gateways for a new hall in minutes
- **High availability** is required — automatic failover, live migration, N+1 redundancy

Use Both (Hybrid) When:

- **Some devices** are in harsh environments (generators, cooling towers = hardware)
- **Most devices** are in controlled environments (CRAC rows, UPS rooms = virtual)
- **Gradual migration** from hardware to virtual (Phase 2 parallel deployment)

---

13. Conclusion

The virtualized gateway architecture preserves every capability of the hardware model — same GOPLC binary, same ST programs, same OPC UA namespaces, same fleet management — while eliminating 85–99% of the hardware cost. The enabling technology is straightforward: Layer 3 SVIs on existing Cisco distribution switches route IP traffic between server VMs and field device networks. No additional routers, no edge computing hardware, no special network equipment.

When infrastructure and GOPLC runtime licensing are combined, the virtualized model costs $8.2M–$8.7M at 1 GW versus $17.6M–$24.7M for the hardware model — a 51–53% total cost reduction. Notably, cluster runtime licensing ($600/instance) is $200/instance more expensive than standalone ($400/instance), so virtual deployments pay a software premium of $2.6M at 1 GW. The infrastructure savings ($11.2M–$18.6M) more than compensate, but the economics are driven by hardware elimination, not licensing discounts. The GOPLC software architecture is deployment-agnostic by design — the same binary runs on a ctrlX CORE, in a Docker container, on a bare-metal VM, or as goroutines in an in-process cluster. The choice of where to run it is an infrastructure decision, not a software one.

Author: James Belcher — jbelcher@jmbtechnical.com

GOPLC v1.0.279 — February 2026