VAULTMESH

INFRASTRUCTURE CATALOG

Version 1.0

Sovereign mesh network providing secure, cryptographically-verified infrastructure across distributed nodes using Tailscale, per-node SSH keys, and proof-based attestation.

VaultMesh Technologies
Dublin, Ireland

Global Catalog Index

Complete inventory of VaultMesh infrastructure capabilities and cross-references.

ID	Capability	Pages
VM-001	Sovereign mesh network via Tailscale MagicDNS (story-ule.ts.net)	1, 2, 3
VM-002	Per-node ed25519 SSH keys with IdentitiesOnly isolation	1, 2, 4
VM-003	Cryptographic proof system with Merkle tree receipts	1, 5, 6
VM-004	Multi-tier node architecture (Mine/Gate/VM/Mobile)	1, 2, 3
VM-005	libvirt/KVM virtualization on brick server	2, 3
VM-006	WireGuard exit nodes for privacy routing	2, 3
VM-007	Cross-platform support (Arch, Debian, Android/Termux, iOS)	2, 4
VM-008	Lawchain compliance ledger integration	5, 6
VM-009	Oracle reasoning engine with tactical chains	5, 7
VM-010	Shield defensive monitoring system	7, 8
VM-011	AppSec toolchain (Nuclei, Trivy, Semgrep, TruffleHog)	7, 8
VM-012	Proof anchoring (local, OTS, ETH, mesh attestation)	5, 6
VM-013	Braid protocol for foreign ledger interop	5, 6
VM-014	MCP server integration for AI orchestration	7, 8
VM-015	Cockpit/VNC console access for VMs	3, 4
VM-016	SSH ControlMaster multiplexing for performance	1, 4
VM-017	Automated key deployment via mesh-sync	2, 4
VM-018	LAN fallback addressing when Tailscale unavailable	1, 2
VM-019	Federation metrics and telemetry emission	5, 8
VM-020	Agent task automation with scheduled triggers	7, 8

1. Infrastructure Overview

VaultMesh is a sovereign mesh network providing secure, cryptographically-verified infrastructure across distributed nodes using Tailscale, per-node SSH keys, and proof-based attestation.

Key Findings

- Zero-trust architecture with per-node key isolation
- MagicDNS naming (*.story-ule.ts.net) for human-readable addressing
- Automatic peer discovery via Tailscale coordination
- SSH ControlMaster multiplexing for connection reuse
- LAN fallback IPs when Tailscale unavailable

Core Components

- Tailscale mesh network (story-ule.ts.net tailnet)
- Per-node ed25519 SSH keypairs
- SSH config with IdentitiesOnly enforcement
- Known hosts with MagicDNS entries

Workflow

```
ssh {node} → Tailscale MagicDNS resolution → per-node key auth → ControlMaster multiplexing
```

Security Notes

- IdentitiesOnly prevents key leakage across hosts
- StrictHostKeyChecking accept-new for TOFU model
- HashKnownHosts disabled for auditability
- AddKeysToAgent with UseKeychain for macOS integration

2. Node Topology

Multi-tier node architecture spanning home infrastructure, cloud gates, virtual machines, and mobile endpoints.

Mine Nodes — Primary Infrastructure

Node	Hostname	OS	Role
gamma	gamma.story-ule.ts.net	Arch Linux	Home Primary
beta	beta.story-ule.ts.net	Arch Linux	Backup Node
brick	brick.story-ule.ts.net	Debian	Dell Server
w3	w3.story-ule.ts.net	Raspbian	Raspberry Pi

Gate Nodes — Edge/Exit Infrastructure

Node	Hostname	OS	Role
v1-nl-gate	v1-nl-gate.story-ule.ts.net	Debian	Netherlands Gate
gate-vm	gate-vm.story-ule.ts.net	Debian	Gateway VM

VM Nodes — Virtual Machines (on brick)

Node	Hostname	OS	Role
debian-golden	debian-golden.story-ule.ts.net	Debian	Golden Image
shield-vm	shield-vm.story-ule.ts.net	Debian	Android VM

Mobile Nodes

Node	Hostname	OS	Port
shield	shield.story-ule.ts.net	Android/Termux	22
bank-mobile	bank-mobile.story-ule.ts.net	iOS	8022

LAN Fallbacks

Node	LAN IP
gamma	192.168.0.191
brick	192.168.0.119
beta	192.168.0.236

3. Virtualization Layer

libvirt/KVM virtualization on brick server hosting Debian VMs with Cockpit management and VNC console access.

Key Findings

- brick server runs libvirt/KVM hypervisor
- Three primary VMs: debian-golden, gate-vm, shield-vm
- VNC access via SSH tunnel forwarding
- Cockpit web interface for VM management
- VMs use NAT networking (192.168.122.x subnet)

VM Network Layout

VM	NAT IP	VNC Port
debian-golden	192.168.122.187	5900
gate-vm	192.168.122.236	5901
shield-vm	192.168.122.73	5902

Workflows

- VM Management: Cockpit → https://brick:9090 → Virtual Machines
- VNC Access: ssh -L 590x:localhost:590x brick → vnc://localhost:590x
- Key Deploy: brick SSH → VM NAT IP → authorized_keys

Security Notes

- VMs isolated in NAT subnet
- VNC only accessible via SSH tunnel
- Tailscale provides mesh access post-boot
- Per-VM SSH keys prevent lateral movement

Dependencies

- libvirt, qemu-kvm on brick
- Cockpit with cockpit-machines
- TigerVNC viewer on client
- SSH tunnel for VNC forwarding

4. SSH Key Architecture

Per-node ed25519 SSH key system with IdentitiesOnly isolation, ControlMaster multiplexing, and secure key management.

Key Findings

- One keypair per destination node (id_gamma, id_brick, etc.)
- IdentitiesOnly prevents key probing across hosts
- ControlMaster enables connection multiplexing
- AddKeysToAgent with UseKeychain for macOS
- StrictHostKeyChecking accept-new for TOFU

Key Inventory

Key File	Target Node	Algorithm
id_gamma	gamma	ed25519
id_beta	beta	ed25519
id_brick	brick	ed25519
id_w3	w3	ed25519
id_v1-nl-gate	v1-nl-gate	ed25519
id_gate-vm	gate-vm	ed25519
id_debian-golden	debian-golden	ed25519
id_shield-vm	shield-vm	ed25519
id_shield	shield	ed25519
id_bank-mobile	bank-mobile	ed25519

External Service Keys

Key File	Service
id_ed25519_github	GitHub
id_ed25519_gitlab	GitLab

Security Notes

- ed25519 keys (256-bit security, small signatures)
- IdentitiesOnly blocks key enumeration attacks
- ControlPath sockets auto-cleaned on disconnect
- No password authentication; key-only

5. Cryptographic Proof System

Merkle tree-based proof system with receipts, anchoring, and cross-ledger attestation via the Braid protocol.

Key Findings

- All significant actions generate cryptographic receipts
- Receipts batched into Merkle trees for efficient verification
- Multiple anchoring strategies: local, OTS (Bitcoin), ETH, mesh
- Braid protocol enables foreign ledger interoperability
- Lawchain integration for compliance tracking

Proof Lifecycle

1. Action occurs (oracle answer, tactical decision, etc.)
2. proof_generate creates signed receipt with action hash
3. Receipts accumulate until batch threshold
4. proof_batch creates Merkle tree from receipt hashes
5. proof_anchor_* writes root to durable storage
6. proof_verify confirms integrity at any time

Anchoring Strategies

Type	Method	Durability
local	File in data/anchors/	Node-local
ots	OpenTimestamps → Bitcoin	Blockchain
eth	Calldata or contract → ETH	Blockchain
mesh	Cross-attest via Tailscale	Multi-node

MCP Tools

- proof_generate, proof_verify, proof_batch, proof_chain
- proof_anchor_local, proof_anchor_ots, proof_anchor_eth, proof_anchor_mesh
- proof_braid_import, proof_braid_emit, proof_braid_status
- proof_stats, proof_root

Security Notes

- Blake3 hashing for speed and security
- Ed25519 signatures for authenticity
- Merkle trees enable efficient inclusion proofs
- Multiple anchors provide defense in depth

6. Lawchain Compliance Ledger

Compliance-focused ledger tracking regulatory obligations, oracle answers, and audit trails with cryptographic receipts.

Key Findings

- Tracks compliance artifacts across frameworks (GDPR, AI Act, NIS2)
- Oracle answers validated against schema before recording
- Hash-backed receipts ensure answer integrity
- Federation metrics emitted for cross-node sync
- Policy evaluation with JSON input/output

Compliance Frameworks Tracked

- GDPR (General Data Protection Regulation)
- EU AI Act (Artificial Intelligence Act)
- NIS2 (Network and Information Security Directive)
- Custom framework extensions supported

Workflow

```
Question → RAG Search → Context Blocks → Oracle Prompt → LLM → Validate → Hash → Receipt → Ledger
```

Security Notes

- Answer hash computed: blake3(json.dumps(answer, sort_keys=True))
- Receipt stored with answer_hash for tamper detection
- Gaps and insufficient_context flags enable honest uncertainty
- Citations must reference real document_ids

7. Oracle Engine & Shield Defense

AI reasoning engine with tactical decision chains and defensive monitoring system for threat detection and response.

Oracle Tools

Tool	Purpose
oracle_status	Node status and capabilities
oracle_reason	Analyze situation, recommend actions
oracle_decide	Make autonomous decision
oracle_tactical_chain	End-to-end reason→decide→act

Shield Monitor Vectors

Vector	Detection Capability
network	Port scans, unusual traffic
wifi	Rogue APs, deauth attacks
bluetooth	Device enumeration, exploits
usb	Mass storage, HID attacks
process	Suspicious execution
file	Unauthorized modifications

Shield Response Levels

Level	Action
log	Record event only
alert	Notify operator
block	Prevent access/execution
isolate	Quarantine affected resource
counter	Active defense measures

Security Notes

- Dry-run mode for tactical chains (default: true)
- Risk tolerance levels: minimal → low → medium → high → maximum
- All decisions include constraints array
- Proofs chain to previous for audit trail

8. AppSec Toolchain

Integrated application security scanning suite with vulnerability detection, secret scanning, SBOM generation, and IaC analysis.

Tool Capabilities

Tool	Target Types	Output
nuclei	URLs, IPs, domains	Findings by severity
trivy	Images, dirs, repos, SBOMs	CVEs, secrets, misconfigs
semgrep	Source code directories	Security findings
trufflehog	Git, GitHub, GitLab, S3, GCS	Verified secrets
gitleaks	Git repos, filesystems	Secret locations
checkov	Terraform, K8s, Helm, Docker	Misconfigurations
syft	Images, dirs, archives	CycloneDX/SPDX SBOM
grype	Images, dirs, SBOMs	Vulnerability list

Severity Levels

- Nuclei: info, low, medium, high, critical
- Trivy: UNKNOWN, LOW, MEDIUM, HIGH, CRITICAL
- Semgrep: INFO, WARNING, ERROR

Workflows

1. SBOM Generation: syft → CycloneDX JSON
2. Dependency Audit: grype against SBOM
3. Secret Scan: gitleaks pre-commit + trufflehog CI
4. IaC Review: checkov on Terraform/K8s
5. Web Vuln Scan: nuclei with CVE templates

MCP Tools

- offsec_appsec_nuclei_scan
- offsec_appsec_trivy_scan
- offsec_appsec_semgrep_scan
- offsec_appsec_trufflehog_scan
- offsec_appsec_gitleaks_scan
- offsec_appsec_checkov_scan
- offsec_appsec_syft_sbom
- offsec_appsec_grype_scan

Security Notes

- Rate limiting available for nuclei (default 150 rps)
- Only verified secrets flag reduces false positives
- Baseline files exclude known findings
- Timeout configs prevent runaway scans

VAULTMESH

Earth's Civilization Ledger

Solve et Coagula

vaultmesh.org • offsecshield.com • vaultmesh.earth