UW-Madison Badger Build Fest

CLOUD NATIVE MADISON

Participant Guildelines

Secure dApp: Open-Source Focus

This distilled guide organizes essential security practices into two primary areas, prioritizing widely adopted, open-source tools to build a secure cloud-native dApp.

Area 1: Pre-Deployment (Build, Test, and Artifacts)

Focus: Securing your code and container images *before* deployment. We emphasize open-source tooling for vulnerability scanning, static analysis, and supply chain integrity.

Sub-Category	Concept	Open-Source Tools	Importance
Container Image & Artifacts	Secure base dependencies and verify image contents.	**Trivy** (Vulnerability Scanning), **Harbor** (Container Registry), Minimal Base Images (Alpine, Distroless).	**Scan images early.** An insecure base or dependency is the weakest link.
CI/CD Pipeline	Automate security testing into your build process.	**Argo CD** or **Flux** (Orchestration), **Bandit** or **Semgrep** (SAST), **pip-audit** (Dependency Scanning).	Catching bugs in CI/CD is the most cost-effective approach. Pipeline should **block** vulnerable deployments.
Software Distribution	Ensure the deployed image is verified as the one built and signed.	**Sigstore/Cosign** (Signing), **OIDC** (Identity-based signing standard).	Use open standards to prove the container's authenticity and prevent attacks.

Area 2: Runtime Security (Deployment, Policy, and Monitoring)

Focus: Securing the application while it runs inside the Kubernetes cluster, handling sensitive data, and monitoring behavior using open-source platforms.

Sub-Category	Concept	Open-Source Tools	Importance
Orchestration & Definition	Securely manage secrets (keys, tokens) and application configuration.	**HashiCorp Vault** (Secrets Management), **Kubernetes/Helm**.	**Never hardcode keys.** Secrets must be injected securely at runtime via managed tools.
Observability & Analysis	Create a searchable audit trail of application behavior.	**structlog** (Structured Logging), **Loki** (Log Aggregation), **Grafana** (Visualization/Dashboards).	Secure what you cannot see. Structured logging is essential for real-time analysis.
Service Proxy & Mesh	Protect the "front door" of services from public network threats.	**NGINX Ingress** + **ModSecurity** (WAF), **Cilium/Istio** (Service Mesh with Policy).	Enforce boundary policies (WAF, rate limits, security headers) at the ingress/mesh layer.
Networking, Policy, & Security	Enforce real-time security policies and detect threats within the cluster.	**Falco** (Runtime Detection), **Open Policy Agent (OPA)** (Policy Enforcement), **JWT/OIDC** (Auth Standards).	Detect suspicious actions (e.g., exec in container). Use **OPA** to enforce access rules based on identity.