01 Deployment Architecture

MWRASP Quantum Defense System

Generated: 2025-08-24 18:15:10

TOP SECRET//SCI - HANDLE VIA SPECIAL ACCESS CHANNELS

MWRASP DEPLOYMENT ARCHITECTURE

Complete Implementation Blueprint for Defense & Enterprise

TABLE OF CONTENTS

REFERENCE ARCHITECTURES

1. SMALL TACTICAL UNIT (Squad/Platoon Level)

Deployable in 30 minutes on single ruggedized server

```
           TACTICAL OPERATIONS CENTER

       MWRASP Tactical Server
       - Dell EMC XR12 Rugged
       - 128GB RAM, 24 cores
       - 10 Active Agents

  [SATCOM]        [Local Network]

  [Field         [Warfighter
   Assets]        Devices]

Specifications:
- Coverage: 50 warfighters
- Response Time: <50ms
- Power: 500W (battery/generator capable)
- Weight: 45 lbs
- Cost: $45,000
```

2. FORWARD OPERATING BASE (Battalion Level)

High-availability cluster with 500+ user support

```
                FOB NETWORK OPERATIONS

    Load Balancer (F5 BIG-IP)

      MWRASP-1 MWRASP-2 MWRASP-3
      Primary  Secondary Tertiary
```

```
          Shared Storage (SAN)

  [SIPR Net]  [NIPR Net]   [Coalition]

Specifications:
- Nodes: 3x HPE ProLiant DL380 Gen11
- RAM: 512GB per node
- Storage: 50TB SAN
- Agents: 50-75 active
- Users: 500-1000
- Uptime: 99.99%
- Cost: $250,000
```

3. ENTERPRISE HEADQUARTERS (Division/Corps)

Multi-site, geo-redundant architecture

```
             GLOBAL DEFENSE NETWORK

   PRIMARY SITE                 SECONDARY SITE
   Washington DC                Colorado Springs

     MWRASP        MWRASP
     Cluster A       WAN Link     Cluster B
     (10 nodes)      Encrypted    (10 nodes)

      TS/SCI                        TS/SCI
      Network                       Network

               EDGE LOCATIONS (50+)

           Edge    Edge    Edge    Edge

Specifications:
- Total Nodes: 20+ across 2 sites
- Agents: 500+ (auto-scaling)
- Users: 10,000+
- Bandwidth: 100Gbps interconnect
- Latency: <5ms intra-site, <50ms inter-site
- Cost: $5M initial, $500K/year operational
```

HARDWARE SPECIFICATIONS

Minimum Requirements by Deployment Size

Deployment	CPU Cores	RAM	Storage	Network	Agents	Users
Tactical	16	64GB	2TB	1Gbps	10	50
Small Business	24	128GB	4TB	10Gbps	20	100
FOB/Medium	64	512GB	20TB	25Gbps	75	1000
Enterprise	256	2TB	100TB	100Gbps	200	5000
Global	1024+	8TB+	1PB+	400Gbps	500+	50000+

Recommended Hardware Vendors

Tactical/Ruggedized

- **Dell EMC XR Series**: MIL-STD certified
- **Crystal Group RS Servers**: Battlefield proven
- **Systel RuggedServers**: Airborne certified

Enterprise

- **HPE ProLiant Gen11**: Best price/performance
- **Dell PowerEdge R750**: High density compute
- **Lenovo ThinkSystem SR650**: Reliability leader

High Performance

- **NVIDIA DGX Systems**: AI acceleration

- **IBM Power10**: Quantum-ready architecture
- **Supermicro Ultra**: Maximum density

NETWORK TOPOLOGY

Security Zone Architecture

```
                 DMZ (Zone 0)
           Internet-Facing Services
        [Web Portal] [API Gateway]

                       Firewall

             Application Zone (Zone 1)
                MWRASP Core Services
       [MWRASP Nodes] [Load Balancers]

                       Firewall

              Data Zone (Zone 2)
            Sensitive Data Processing
        [Fragmentation] [Encryption]

                       Air Gap

           Classified Zone (Zone 3)
              TS/SCI Operations
          [Classified MWRASP Instance]
```

Network Segmentation Rules

```
firewall_rules:
  dmz_to_app:
    - protocol: HTTPS
      port: 443
      direction: inbound
      rate_limit: 10000/sec

  app_to_data:
    - protocol: TCP
```

```
      port: 8443
      direction: bidirectional
      encryption: required

  data_to_classified:
    - protocol: NONE
      description: "Air-gapped, no network connection"
      transfer: "Automated data diode only"
```

CLOUD NATIVE DEPLOYMENT

Kubernetes Architecture

```
apiVersion: v1
kind: Namespace
metadata:
  name: mwrasp-system
---
apiVersion: apps/v1
kind: StatefulSet
metadata:
  name: mwrasp-core
  namespace: mwrasp-system
spec:
  serviceName: mwrasp
  replicas: 5
  selector:
    matchLabels:
      app: mwrasp
  template:
    metadata:
      labels:
        app: mwrasp
    spec:
      containers:
      - name: mwrasp
        image: mwrasp/quantum-defense:v2.0
        resources:
          requests:
            memory: "32Gi"
            cpu: "8"
            ephemeral-storage: "100Gi"
          limits:
            memory: "64Gi"
            cpu: "16"
        env:
        - name: MWRASP_MODE
          value: "QUANTUM_DEFENSE"
        - name: AGENT_COUNT
          value: "50"
        - name: FRAGMENT_LIFETIME_MS
          value: "100"
        volumeMounts:
        - name: config
          mountPath: /etc/mwrasp
        - name: ephemeral
          mountPath: /tmp/fragments
      volumes:
      - name: config
        configMap:
          name: mwrasp-config
      - name: ephemeral
        emptyDir:
          sizeLimit: 100Gi
---
apiVersion: v1
kind: Service
metadata:
  name: mwrasp-service
  namespace: mwrasp-system
spec:
  type: LoadBalancer
  ports:
  - port: 443
    targetPort: 8443
    protocol: TCP
    name: https
  - port: 8080
    targetPort: 8080
    protocol: TCP
    name: metrics
  selector:
    app: mwrasp
---
apiVersion: autoscaling/v2
kind: HorizontalPodAutoscaler
metadata:
  name: mwrasp-autoscaler
  namespace: mwrasp-system
spec:
  scaleTargetRef:
    apiVersion: apps/v1
    kind: StatefulSet
    name: mwrasp-core
  minReplicas: 5
  maxReplicas: 100
  metrics:
```

```
  - type: Resource
    resource:
      name: cpu
      target:
        type: Utilization
        averageUtilization: 50
  - type: Resource
    resource:
      name: memory
      target:
        type: Utilization
        averageUtilization: 70
```

Helm Chart Structure

```
mwrasp-chart/
 Chart.yaml
 values.yaml
 templates/
     deployment.yaml
     service.yaml
     configmap.yaml
     secret.yaml
     ingress.yaml
     hpa.yaml
     networkpolicy.yaml
     serviceaccount.yaml
 charts/
     postgresql/
     redis/
     prometheus/
 tests/
     test-connection.yaml
```

EDGE COMPUTING ARCHITECTURE

Edge Node Deployment

```
class MWRASPEdgeNode:
    """
    Lightweight edge deployment for distributed operations
    """

    def __init__(self, location, parent_cluster):
        self.location = location
        self.parent = parent_cluster
        self.local_agents = 5  # Minimal agent count
        self.cache_size = "10GB"
        self.sync_interval = 60  # seconds

    def deploy_configuration(self):
        return {
            "mode": "edge",
            "features": {
                "quantum_detection": True,
                "temporal_fragmentation": True,
                "behavioral_auth": True,
                "legal_barriers": False,  # Centralized only
                "agent_spawning": False  # Limited at edge
            },
            "resources": {
                "cpu": "4 cores",
                "ram": "16GB",
                "storage": "500GB SSD",
                "network": "1Gbps"
            },
            "sync": {
                "parent": self.parent,
                "interval": self.sync_interval,
                "priority_data_only": True
            }
        }
```

Edge Locations

- **Aircraft Carriers**: 1 edge node per carrier
- **Embassies**: 1 edge node per embassy
- **Forward Bases**: 1 edge node per base
- **Satellites**: Micro-nodes for space assets
- **Submarines**: Isolated nodes with delayed sync

CLASSIFIED NETWORK DEPLOYMENT

Cross-Domain Solution Integration

```
            UNCLASSIFIED (NIPR)
          MWRASP Public Instance

              Cross-Domain
              Solution (CDS)
              [Forcepoint]

              SECRT (SIPR)
         MWRASP Secret Instance

               Raise CDS
              [Owl DualDiode]

            TOP SECRET (JWICS)
       MWRASP TS/SCI Instance
```

Security Controls for Classified

- **Physical**: SCIFs, two-person control
- **Network**: Air-gapped, TEMPEST certified
- **Crypto**: NSA Type 1 encryption
- **Access**: PKI certificates, clearance verification
- **Audit**: 100% logging, 7-year retention

SCALING STRATEGIES

Horizontal Scaling

```
def calculate_scaling_requirements(metrics):
    """
    Dynamic scaling algorithm for MWRASP
    """

    scaling_decision = {
        'scale_up': False,
        'scale_down': False,
        'new_nodes': 0
    }

    # Scale up triggers
    if metrics['cpu_usage'] > 70:
        scaling_decision['scale_up'] = True
        scaling_decision['new_nodes'] =
math.ceil((metrics['cpu_usage'] - 70) / 10)

    if metrics['threat_level'] == 'CRITICAL':
        scaling_decision['scale_up'] = True
        scaling_decision['new_nodes'] += 5

    if metrics['response_time'] > 100:  # ms
        scaling_decision['scale_up'] = True
        scaling_decision['new_nodes'] += 3

    # Scale down triggers
    if metrics['cpu_usage'] < 30 and metrics['threat_level'] == 'LOW':
        scaling_decision['scale_down'] = True
        scaling_decision['new_nodes'] = -2

    return scaling_decision
```

Vertical Scaling

Metric	Threshold	Action
Memory Usage	>80%	Add 64GB RAM
Agent Count	>100	Upgrade CPU (2x cores)
Fragment Rate	>10K/sec	Add NVMe storage
Network I/O	>80%	Upgrade to 100Gbps

DISASTER RECOVERY

Backup and Recovery Strategy

```
 backup_strategy:
  configuration:
    frequency: every_change
    retention: unlimited
    replication: 3_sites

  operational_data:
    frequency: continuous
    retention: 30_days
    method: streaming_replication

  audit_logs:
    frequency: real_time
    retention: 7_years
    compliance: NIST_800-53

recovery_targets:
  rto: 15_minutes   # Recovery Time Objective
  rpo: 5_minutes    # Recovery Point Objective

failover_process:
  automatic: true
  detection_time: 10_seconds
  failover_time: 30_seconds
  data_validation: cryptographic_hash
```

Disaster Scenarios and Responses

Scenario	Detection	Response	Recovery
Site Failure	<10s	Auto-failover	15 min
Cyber Attack	<1ms	Fragmentation	Immediate
EMP/Nuclear	N/A	Hardened backup	1 hour

Scenario	Detection	Response	Recovery
Insider Threat	<100ms	Isolation	5 min
Natural Disaster	Predictive	Pre-migration	0 min

DEPLOYMENT AUTOMATION

Infrastructure as Code

```
# main.tf - MWRASP Infrastructure

provider "aws" {
  region = var.aws_region
}

module "mwrasp_cluster" {
  source = "./modules/mwrasp"

  cluster_name = "mwrasp-production"
  node_count   = 10
  node_type    = "c6i.8xlarge"

  network = {
    vpc_cidr = "10.0.0.0/16"
    private_subnets = ["10.0.1.0/24", "10.0.2.0/24", "10.0.3.0/24"]
    public_subnets  = ["10.0.101.0/24", "10.0.102.0/24",
"10.0.103.0/24"]
  }

  security = {
    enable_encryption = true
    kms_key_id        = aws_kms_key.mwrasp.id
    enable_flow_logs = true
  }

  monitoring = {
    enable_cloudwatch = true
    enable_prometheus = true
    alert_email       = "soc@organization.mil"
  }
}
```

```
output "mwrasp_endpoints" {
  value = {
    api_endpoint = module.mwrasp_cluster.api_endpoint
    dashboard    = module.mwrasp_cluster.dashboard_url
    metrics      = module.mwrasp_cluster.metrics_endpoint
  }
}
```

Ansible Playbooks

```
 # deploy-mwrasp.yml
---
- name: Deploy MWRASP Quantum Defense System
  hosts: mwrasp_nodes
  become: yes

  tasks:
    - name: Install system dependencies
      package:
        name:
          - python3.9
          - python3-pip
          - docker
          - nginx
        state: present

    - name: Deploy MWRASP containers
      docker_container:
        name: mwrasp-core
        image: mwrasp/quantum-defense:latest
        state: started
        restart_policy: unless-stopped
        ports:
          - "8443:8443"
        env:
          MWRASP_MODE: "PRODUCTION"
          AGENT_COUNT: "{{ agent_count }}"

    - name: Configure monitoring
      template:
        src: prometheus.yml.j2
        dest: /etc/prometheus/prometheus.yml
      notify: restart prometheus
```

DEPLOYMENT CHECKLIST

Pre-Deployment

- [ ] Hardware provisioned and tested
- [ ] Network connectivity verified
- [ ] Security clearances confirmed
- [ ] Backup systems operational
- [ ] Monitoring infrastructure ready

Deployment

- [ ] Base OS hardened (STIG compliance)
- [ ] MWRASP software installed
- [ ] Certificates deployed
- [ ] Initial configuration applied
- [ ] Agent network initialized

Post-Deployment

- [ ] Health checks passing
- [ ] Performance baselines established
- [ ] Security scan completed
- [ ] Documentation updated
- [ ] Team training completed

Go-Live

- [ ] Threat detection active
- [ ] Response times validated
- [ ] Failover tested

- [ ] 24/7 monitoring active
- [ ] Incident response team ready

ARCHITECTURE DECISION RECORDS

ADR-001: Microservices vs Monolithic

Decision: Hybrid approach with monolithic core and microservice extensions **Rationale**: Critical response times require tight integration

ADR-002: Container Orchestration Platform

Decision: Kubernetes for cloud, Docker Swarm for edge **Rationale**: K8s for scale, Swarm for simplicity at edge

ADR-003: Message Queue Technology

Decision: Redis Streams for internal, Kafka for external **Rationale**: Redis for speed, Kafka for durability

ADR-004: Database Strategy

Decision: In-memory primary, PostgreSQL for audit only **Rationale**: 100ms data expiration makes persistence unnecessary

---

Document Version: 2.0 **Classification**: UNCLASSIFIED // FOUO **Last Updated**: February 2024 **Next Review**: Quarterly