Cyber Risk Quantification & Residual Risk- QuantifyXR

Acronym & Synonym

Acronym	Full Form	Definition / Synonyms
CRQ	Cyber Risk Quantification	The process of measuring and assessing cyber risk exposure quantitatively
FAIR	Factor Analysis of Information Risk	A risk analysis framework for understanding, measuring, and analyzing information risk in financial terms
CVE	Common Vulnerabilities and Exposures	Publicly disclosed cybersecurity vulnerabilities and exposures
CVSS	Common Vulnerability Scoring System	Standardized framework for rating the severity of security vulnerabilities
IAM	Identity and Access Management	Framework for managing digital identities and controlling access to resources
ERP	Enterprise Resource Planning	Integrated system that manages core business processes
SCADA	Supervisory Control and Data Acquisition	Industrial control system used for monitoring and controlling industrial processes
NVD	National Vulnerability Database	U.S. government repository of vulnerability management data
DARPA	Defense Advanced Research Projects Agency	U.S. agency that develops emerging technologies for national security
NSL-KDD	Network Security Laboratory Knowledge Discovery and Data Mining	Dataset for evaluating intrusion detection systems
CAIDA	Center for Applied Internet Data Analysis	Research organization providing datasets and tools for Internet data analysis
MAWI	Measurement and Analysis on the WIDE Internet	Dataset project for analyzing Internet traffic
CERT	Computer Emergency Response Team	Organization that handles computer security incidents
HPA	Horizontal Pod Autoscaler	Kubernetes component for automatically scaling pods based on load
GNN	Graph Neural Network	Neural network architecture designed for graph-structured data
JWT	JSON Web Token	Compact token format for secure data transmission between parties
OCR	Optical Character Recognition	Technology for converting images of text into

		machine-readable text
NLP	Natural Language Processing	AI discipline focused on interaction between computers and human language
KPI	Key Performance Indicator	Measurable value that indicates success of a process or activity
SOC	Security Operations Center	Facility for monitoring and managing security incidents
ISO	International Organization for Standardization	International standards body (ISO 27001 relevant to information security)
NIST	National Institute of Standards and Technology	U.S. standards body providing cybersecurity guidelines
PCI DSS	Payment Card Industry Data Security Standard	Security standard for protecting payment card data
POC	Proof of Concept	Prototype to demonstrate feasibility of a concept or approach
API GW	API Gateway	Server that acts as a single entry point for APIs and manages routing, authentication, and rate limiting
ETL	Extract, Transform, Load	Process of moving data from source to destination after transforming it
KPI	Key Performance Indicator	Metric used to evaluate performance or success of a process
SOC	Security Operations Center	Centralized unit for monitoring, detecting, and responding to security incidents
RBAC	Role-Based Access Control	Security method for restricting system access based on roles

Elevator pitch

- QuantifyXR turns noisy scanner results and asset inventories into business-centric, monetary risk estimates and prioritized remediation actions.
- Using FAIR-style Monte Carlo simulations, Bayesian attack graphs, and explainable ML, QuantifyXR shows Expected Annual Loss (EAL) per asset, the residual risk after controls, and the financial benefit of proposed mitigations — enabling security and risk teams to make fast, defensible decisions and report to executives.

Problem statement

Security tooling generates huge volumes of telemetry (vulnerabilities, alerts, logs) but lacks business context. Security teams struggle to prioritize remediation:

- Which patch or control reduces the company’s financial exposure the most?
- How much residual risk remains after applying controls?
- Without quantitative answers, remediation is inefficient and risk acceptance is ad-hoc.

Product vision & goals

- Provide an auditable, repeatable method to convert technical findings into monetary risk estimates.
- Model attack paths and control effectiveness to compute residual risk.
- Make recommendations that are actionable and explainable to technical teams and execs.
- Integrate with common enterprise systems (Qualys/Nessus/Tenable, ServiceNow, SIEMs).

Key success metrics (example):

- Time-to-prioritize (scan → prioritized risk list) < 1 hour.
- Reduction in high-EAL assets within 3 months (customer measured).
- Executive risk-report adoption (monthly board report generated automatically).

Target users & personas

- CISO / Risk Officer — wants numeric enterprise exposure and board-ready reporting.
- Security Operations Lead — needs prioritized remediation tasks and control efficacy metrics.
- SecOps / Patch Team — needs concrete tickets and business-backed prioritization.
- Risk/Compliance Auditor — requires audit trail and reasoning for residual-risk acceptance.

Core features (MVP + roadmap)

1. Asset inventory ingestion & mapping to owners.
2. Vulnerability ingestion (Nessus/Qualys/Tenable CSV/JSON).
3. CVE/CVSS enrichment (NVD feed).
4. FAIR-style risk engine (Monte Carlo) producing Expected Annual Loss per asset with confidence intervals.
5. Control registry + residual risk calculation (apply control reduction factors).
6. Dashboard: Top-20 risks, distribution charts, what-if simulation (apply control → new EAL).
7. Exportable report & ticket creation (Jira/ServiceNow integration stub).

Post-MVP roadmap:

- Bayesian attack-graph engine for multi-step probabilities.
- ML models: exploitability / breach-likelihood predictors; GNN for attack-path ranking.
- Real-time stream ingestion (Kafka/Kinesis).
- Policy-driven automated remediation playbooks.
- Multi-tenant SaaS & RBAC for enterprises.

Architecture (textual diagram)

Its a batch inference, so I am going for Data Fabric architecture, because of following reasons

- Data comes from many sources: CMDB, vulnerability scanners (Tenables, Qualys), endpoint telemetry, policy databases, external threat intel.
- You need unified normalization, enrichment, correlation → exactly what a data fabric provides.
- FAIR Monte Carlo simulations, Bayesian attack graphs, GNNs need centralized, consistent data.
- Real-time APIs can then query this fabric layer for up-to-date risk calculations.

Data Sources

Category	Subcategory / Description
Cyber Risk Data Sources	
Network Related	Ethernet, wireless, SCADA, cloud
Application	Database, IAM, ERP, mail, web, thick client, middleware
Firmographic Data	Company size, geography, business segments
External Cybersecurity Datasets	DARPA, NSL-KDD, CAIDA, MAWI, CERT
Endpoint	Server, network devices, workstations, mobile, IoT
Other Internal Security Data	Controls list/library, server security configurations, routing tables, network node-level information, system logs, user activity, endpoint usage stats
Policy and Regulatory	SEC requirements, ISO 27001:2013, NIST SP 800-53 R4, PCI DSS v3.2.1
External Historic Cyber Breach Information	Event details, attack type, target, country, incident triggers, damages, resolution
Other External Data	Vulnerability/configuration management platforms, malicious IP addresses, threat intelligence feeds, vulnerability analysis reports, social media feeds, tracker communities, deep/dark/surface web
Data Warehouse	Structured/unstructured data aggregation, cleansing, normalization, parsing, correlation using NLP and big data capabilities

Details:

Layer	Description	AWS Services	APIs for Layer
Data Pipeline			
Data Ingestion Layer (Bronze Layer)	Collect raw data from multiple cybersecurity data sources: CMDB, vulnerability scanners (Tenable, Qualys), endpoint telemetry, threat intelligence feeds, controls registry. Data is ingested in raw format (JSON, CSV, log streams).	- AWS Kafka → real-time stream ingestion. - AWS Glue / AWS Lambda → event-driven ingestion. - AWS S3 → store raw datasets (Bronze Layer).	- /ingest/vulnerabilities → ingest vuln scan JSON. - /ingest/assets → ingest CMDB asset CSV. - /ingest/controls → ingest controls registry. - APIs to trigger ingestion pipelines.
Data Cleaning & Normalization Layer (Silver Layer)	Process raw data to remove noise, handle missing values, unify formats, standardize fields. This layer converts raw JSON/CSV logs into structured datasets for modeling.	- AWS Glue → ETL with PySpark. - AWS EMR → large-scale Spark jobs. - dbt → schema transformation & normalization. - Great Expectations → data quality checks. - AWS Step Functions / Airflow (Amazon MWAA) → orchestration pipelines. - Output stored in cleaned format in S3 Silver Layer.	- /transform/normalize → run ETL job. - /quality/check → trigger Great Expectations checks. - /metadata/update → update catalog entries.
Data Enrichment & Gold Layer	Enrich normalized datasets with derived features, threat intelligence, CVE scoring, asset tagging, vulnerability severity enrichment. Data here is ready for consumption by analytics and ML.	- dbt → transformations for gold datasets. - AWS Glue / EMR → enrichment pipelines. - AWS Athena → query and transform gold datasets. - S3 → store Gold Layer datasets.- AWS Redshift / Snowflake → serve gold datasets for modeling.	- /enrich/vulns → enrich CVE data with external feeds. - /generate/gold → create gold-layer datasets. - /query/gold → Athena/Redshift queries.
Metadata & Schema Layer	Centralized catalog and schema registry to track data definitions, lineage, and quality across layers. Enables	- AWS Glue Data Catalog → metadata & schema registry. - DataHub / Amundsen → open-source data	- /metadata/catalog → retrieve dataset metadata. - /metadata/schema → retrieve schema definitions. - /metadata/lineage → retrieve data

	governance and unified access.	catalogs. - AWS Lake Formation → access control. - AWS Glue Schema Registry → schema versioning for streaming data.	lineage.
Model Building Pipeline			
Risk Modeling Layer	FAIR Monte Carlo engine, Bayesian networks, GNN models, and supervised models (XGBoost, Logistic Regression) to generate cyber risk scores.	- Amazon SageMaker → train and host ML models. - AWS Batch → batch model execution. - AWS Lambda → lightweight scoring functions. - AWS Step Functions → orchestrate Monte Carlo simulations.	- /model/run_monte_carlo → trigger FAIR Monte Carlo simulation. - /model/predict_risk → run risk scoring. - /model/explain → return model explainability results.
Explainability Layer	Generate human-interpretable explanations (SHAP, LIME) for model predictions, produce remediation summaries.	- SageMaker Clarify → model explainability. - AWS Lambda → execute explainability jobs. - Amazon Comprehend → generate natural language summaries. - S3 / DynamoDB → store explainability reports.	- /explain/shap → return SHAP plots. - /explain/lime → return LIME results. - /explain/nlg → return natural language justification.
Inference Pipeline			
Serving & API Layer	API gateway layer to expose risk scores, reports, and dashboards. This layer provides real-time scoring and batch results to dashboards and external systems.	- AWS API Gateway → expose REST APIs. - AWS Lambda → API compute layer. - Amazon AppSync → GraphQL API. - AWS Cognito → auth & RBAC. - Amazon CloudFront → API caching.	- /api/risk_score → return real-time risk score. - /api/report → download risk report. - /api/dashboard → return dashboard data.
MLOps			
Feature Store	Centralized repository for storing, versioning, and serving ML features	Amazon SageMaker Feature Store	SageMaker Feature Store API, boto3 SDK

	consistently across training and inference.		
Experimentation / Model Development	Build, train, and validate ML models with versioning, notebooks, and containerized environments.	Amazon SageMaker Studio, SageMaker Training Jobs, AWS Deep Learning AMIs	SageMaker API, boto3 SDK, Docker SDK
Model Registry	Store, track, and version trained ML models for reuse, governance, and deployment approval workflows.	Amazon SageMaker Model Registry, AWS CodeCommit (for versioned artifacts)	SageMaker Model Registry API, boto3 SDK, CodeCommit API
Model Deployment & Serving	Deploy models for batch or real-time inference with autoscaling, monitoring, and rollback.	Amazon SageMaker Endpoints, SageMaker Serverless Inference, Amazon ECS/Fargate, EKS	SageMaker Inference API, REST endpoints, gRPC APIs
Monitoring & Drift Detection	Continuously track model performance, drift, data quality, and trigger alerts for retraining.	SageMaker Model Monitor, CloudWatch, AWS CloudTrail	SageMaker Monitor API, CloudWatch API, boto3 SDK
Pipeline Orchestration	Automate ML workflows (data prep → training → evaluation → deployment) with CI/CD integration.	Amazon SageMaker Pipelines, AWS Step Functions, Apache Airflow (MWAA), AWS CodePipeline	SageMaker Pipelines API, Step Functions API, Airflow API, CodePipeline API
Security & Governance	Ensure compliance, access control, and auditability across MLOps lifecycle.	AWS IAM, AWS KMS, Amazon Macie, AWS Secrets Manager	IAM API, KMS API, Macie API, Secrets Manager API
Explainability & Responsible AI	Explain predictions, ensure fairness, bias detection, and compliance reporting.	SageMaker Clarify, Amazon A2I (Augmented AI)	Clarify API, A2I API, boto3 SDK
Observability			
Observability & Governance Layer	Track pipeline health, model performance, audit logs, compliance.	- Amazon CloudWatch → logs & metrics. - AWS X-Ray → request tracing. - AWS Config → resource compliance. - AWS Lake Formation → data access governance.	- /metrics/pipeline → return pipeline health. - /metrics/model → return model performance. - /audit/logs → return access logs.

Dashboard & Reporting Layer	Visualization of risk scores, attack paths, residual risk, compliance levels, and remediation recommendations.	- Amazon QuickSight → dashboards. - Grafana → metrics dashboards. - Tableau / Superset → advanced analytics. - S3 → reports storage.- AWS Athena / Redshift → query gold data for dashboard.	- /dashboard/top_risks → return top asset risks. - /dashboard/trend → risk score trends. - /dashboard/compliance → compliance reports.

Dummy Data Model for ML model building

assets.csv

Field	Type	Description
asset_id	string	Unique asset identifier
asset_name	string	Human-readable name of the asset
owner	string	Person or team responsible
business_unit	string	Department or unit owning the asset
business_value_usd	number	Proxy for annual revenue/criticality
environment	enum (prod/dev)	Operating environment
ip_address	string	IPv4/IPv6 address
hostname	string	System hostname
tags	string (semi-colon list)	Classification tags (e.g., critical;sensitive)

Vulns.csv

Field	Type	Description
vuln_id	string	Scanner-specific vulnerability ID (e.g., Tenable ID)
cve_id	string	CVE identifier (CVE-YYYY-XXXX)

asset_id	string	Foreign key to asset inventory
scan_date	ISO8601 datetime	Date of vulnerability scan
cvss_v3	float	CVSS v3 base score
cvss_vector	string	CVSS vector string
exploit_available	boolean	Whether a public exploit is available
evidence_links	list (string)	Links to PoCs, advisories, references
remediation_status	enum (open/mitigated/wontfix)	Current remediation status

Controls.csv

Field	Type	Description
control_id	string	Unique control identifier
control_name	string	Control name (e.g., MFA, WAF)
asset_scope	list (asset_ids or tags)	Assets covered by control
control_type	enum (MFA, firewall, WAF, patching)	Control category
effectiveness_score	float (0.0–1.0)	Measured/estimated effectiveness
last_tested	ISO8601 datetime	Last validation/test date

Risk_results.csv

Field	Type	Description
risk_id	string	Unique risk record identifier
asset_id	string	Related asset
scenario_id	string	Scenario or threat case identifier
inherent_likelihood	float (0–1)	Likelihood without controls
residual_likelih	float (0–1)	Likelihood after controls

ood		
impact_usd_distribution_summary	object {mean, p10, p50, p90}	Monte Carlo distribution summary
expected_annual_loss_usd	float	Calculated EAL
controls_applied	list (string)	Controls applied to asset
timestamp	ISO8601 datetime	Record creation timestamp

AI / ML & statistical techniques

FAIR (Factor Analysis of Information Risk) is a framework that structures cyber risk in terms of Loss Event Frequency (LEF) and Loss Magnitude (LM).
FAIR explicitly addresses uncertainty and variability, which fits perfectly here.

Monte Carlo simulation allows us to model uncertainty by simulation rather than fixed-point estimates:

Benefit	Reason in Cyber Risk
Handles uncertainty	Probabilities and impacts in cyber risk are estimates — Monte Carlo samples from probability distributions rather than using fixed numbers.
Models multiple scenarios	Cybersecurity threats vary — Monte Carlo creates thousands of simulated scenarios to see the range of possible outcomes.
Produces probability distributions	Instead of one number for Expected Annual Loss (EAL), we get P10, P50, P90 estimates — giving risk managers a richer view of risk.
Allows sensitivity analysis	We can see how changes in vulnerability scores, control effectiveness, or asset value affect risk outcomes.
Supports decision-making	Enables prioritizing risk mitigation efforts based on potential financial loss and uncertainty rather than guesswork.

- **Monte Carlo simulation** (FAIR) for frequency × magnitude distributions → EAL.
- **Bayesian networks / attack graphs** to model multi-step compromise probabilities.
- **Supervised models** (XGBoost / logistic regression) to predict whether a CVE will be exploited in production given features (CVSS, exploit, asset exposure, historical signals).
- **Graph Neural Networks (GNNs)** for attack-path ranking and node compromise scoring.
- **Calibration & explainability**: isotonic/Platt calibration, SHAP feature attributions, natural-language summaries for remediation justification.

Implementation plan & timeline (8–10 week example)

Week(s)	Activities & Deliverables
Week 0	- Project kickoff & data access- Stakeholder interviews- Select 5 critical assets for pilot- Obtain vulnerability scan exports and sample CMDB
Weeks 1–2	- Build data connectors- Define normalized schema- Implement NVD enrichment job-Stand up storage (Postgres + S3 + Neo4j dev instance)
Weeks 3–4	- Implement FAIR Monte Carlo risk engine as a microservice- Build minimal React dashboard to display top risks and distribution plots
Weeks 5–6	- Add controls registry- Implement control effectiveness logic and what-if simulation-Prototype ticketing integration (Jira/ServiceNow) for remediation tasks
Weeks 7–8	- Implement Bayesian attack graph POC for 2–3 scenarios- Compare outputs with Monte Carlo engine- Conduct SME calibration sessions- Backtest on historical incidents
Weeks 9–10	- Harden system: observability, logging, basic auth/RBAC- Create exportable board report template- User training and runbook delivery

Team & roles

- Product Manager — scope, stakeholder alignment, roadmap.
- Tech Lead / Architect — overall architecture, infra decisions.
- Backend Engineers (2) — ingestion, risk engine services.
- Data Engineer (1) — ETL, normalization, storage.
- ML Engineer (1) — Monte Carlo, Bayesian/ML models, model ops.
- Frontend Engineer (1) — dashboard, visualizations.
- Security SME / Risk Analyst (1) — FAIR priors, calibration and validation.
- QA / SRE (shared) — testing, deployment, SLOs.

Validation, monitoring & governance

- Calibration workshops with SMEs to tune priors and distribution ranges.
- Backtesting with historical vulnerability/outage/incidents to measure calibration (predicted vs observed).
- Drift detection for ML models and input signal freshness checks.
- Audit trail: every residual-risk acceptance is stored with owner, timestamp, and rationale.

Productionisation

Stage	Task	Description	Tools / Technologies
1. Containerization	Dockerize Services	Package all microservices (Risk Engine, Bayesian Graph, Supervised Models, GNN, Calibration, Explainability, API Gateway) into Docker containers for portability and isolation.	Docker, Docker Compose
2. Container Registry	Store images	Push Docker images to a registry for deployment.	Docker Hub, AWS ECR, GitHub Container Registry
3. Orchestration	Kubernetes Cluster Setup	Deploy containerized microservices in a Kubernetes cluster for scalability, fault tolerance, and orchestration.	Kubernetes, Minikube (dev), AWS EKS / GKE / Azure AKS (prod)
4. Service Management	Microservice Deployment	Deploy each module as a Kubernetes Deployment with proper resource requests/limits and ReplicaSets.	kubectl, Helm charts
5. API Gateway	Central entry point	Route requests to microservices, handle authentication, rate limiting, and API versioning.	Kong, Ambassador, NGINX Ingress, AWS API Gateway
6. Load Balancing	Traffic Distribution	Balance traffic across replicas for high availability and optimal performance.	Kubernetes Service LoadBalancer, Istio, NGINX
7. CI/CD Pipeline	Automate build, test, deploy	Automatically build Docker images, run tests, push to registry, and deploy to Kubernetes.	GitHub Actions, GitLab CI/CD, Jenkins, ArgoCD
8. Configuration Management	Manage configs & secrets	Store and manage configuration and secrets securely.	Kubernetes ConfigMaps, Secrets, HashiCorp Vault
9. Monitoring & Observability	Track performance, logs, metrics	Implement observability to detect anomalies and performance issues.	Prometheus, Grafana, ELK Stack (Elasticsearch, Logstash, Kibana), Jaeger
10. Security & Access Control	Protect APIs & services	Implement authentication, authorization, RBAC, network security.	OAuth2 / JWT, Kubernetes RBAC, Istio mTLS, API Gateway policies
11. Auto-scaling	Dynamically scale services	Automatically scale pods based on demand.	Kubernetes Horizontal Pod Autoscaler (HPA), Cluster Autoscaler
12. Disaster Recovery	Backups & failover	Implement backup and disaster recovery strategies for data & services.	Velero, AWS Backup, GCP Backup
13. Documentation & Runbook	Operational readiness	Maintain runbooks, onboarding docs, troubleshooting guides.	Markdown docs, Confluence, ReadTheDocs
14. Production Rollout	Deploy system to production	Deploy all components, validate performance, run integration tests, monitor.	Helm charts, kubectl apply, automated smoke tests
15. Continuous Improvement	Feedback loop	Monitor logs, performance, security events; refine models and system based on new data.	Observability dashboards, automated retraining pipelines

Production checklist

- Data contracts defined and monitored.
- Model versioning & CI for model training.
- RBAC & encryption for sensitive fields.
- SLOs & runbooks for incidents.
- Regular SME calibration cadence (quarterly).

Risks & mitigations

- **Data quality gaps**: Mitigate by building robust validation rules and fallback conservative priors.
- **Stakeholder trust**: Provide explainability, SME calibration, and audit trails.
- **Label scarcity for ML**: Use probabilistic/Bayesian methods first; add supervised models only when sufficient labeled events exist.

—---

V2 — Agentic AI Architecture

High-Level Concept

In V2, your system becomes an **Agentic Cyber Risk Quantification & Mitigation AI (CyberRisk-Agent)**.
It combines:

Layer / Component	Purpose & Function	Key Technologies / Modules
Perception Layer	Ingests and normalizes data from multiple sources. Gathers vulnerability feeds, asset inventory, control effectiveness metrics, and historical security events.	API connectors (NVD, CVE), ETL pipelines, database ingestion scripts
Reasoning Layer	Processes ingested data to quantify risk. Builds attack graphs, runs Bayesian inference, predicts exploit likelihood, and ranks attack paths.	Bayesian Networks (pgmpy), FAIR Monte Carlo simulator, Logistic Regression, XGBoost, Graph Neural Networks (PyTorch Geometric), numpy, pandas
Decision Layer	Prioritizes risk mitigation actions and selects the next course of action autonomously. Uses scoring to decide patching, control adjustments, or escalation.	Rule-based logic, Reinforcement Learning (Q-Learning, Policy Gradient), custom decision engine
Execution Layer	Automates or coordinates mitigation	Ansible, SCCM, Jira API,

	actions. Interfaces with orchestration tools, patch management, ticketing systems, and notification channels.	ServiceNow API, custom action executors
Learning Layer	Continuously improves models based on new data and events. Supports retraining pipelines and active learning for better risk prediction.	Online Learning frameworks, Active Learning, Transfer Learning, CI/CD pipelines
Explainability Layer	Provides transparency and trust by explaining risk predictions and decisions. Generates visual and natural language reports for stakeholders.	SHAP, LIME, matplotlib, Natural Language Generation tools
Integration / API Layer	Provides endpoints for all agent functions: risk assessment, graph building, prediction, action execution, retraining, and explanation. Supports orchestration & UI.	FastAPI / Flask, REST APIs, WebSocket APIs
Agent Control Layer	Central orchestration of the agentic system. Coordinates perception, reasoning, decision-making, execution, learning, and explainability layers autonomously.	Workflow orchestration frameworks (Apache Airflow, Prefect), internal agent controller, event bus
Data Storage Layer	Stores raw and processed data for analysis, retraining, and audit. Supports historical risk analysis and regulatory compliance.	Relational DB (PostgreSQL), NoSQL DB (MongoDB), Object storage (S3), Data lake (Delta Lake)

AIML Techniques for Agentic AI

Layer	Techniques	Proposed Models	uggested Implementation Approac
Reasoning Layer	Bayesian Networks, Monte Carlo Simulation, Graph Neural Networks (GNN)	- pgmpy.DiscreteBayesianNetwork for attack progression- FAIR Monte Carlo Risk Simulator (NumPy/Pandas)- PyTorch Geometric / DGL GNN for attack-path ranking	- Use pgmpy for DAG modeling- Containerize Monte Carlo microservice- Train GNN on simulated + CVE graph data, deploy via TorchServe
Prediction Layer	Logistic Regression, XGBoost	- scikit-learn LogisticRegression for exploit prediction- xgboost.XGBClassifier for CVE exploitation likelihood	- Train on CVSS + exploit-db features- Wrap as REST API with FastAPI- Automate retraining in CI/CD pipeline
Decision Layer	Reinforcement Learning (Policy Gradients, Q-Learning)	- Stable Baselines3 PPO/DQN for adaptive control optimization- Custom Q-learning for patch prioritization	- Simulate environments (attack-defense)- Deploy RL agents via Ray RLlib- Integrate into risk engine API for decision support
Learning Layer	Online Learning, Active Learning, Transfer Learning	- River for online anomaly detection- modAL for active vulnerability triage- Node2Vec/GloVe embeddings	- Deploy online learners on streaming logs- Use active learning loop with SME feedback- Fine-tune

		fine-tuned for asset-vuln graphs	embeddings on graph DB (Neo4j)
Explainability	SHAP, LIME, Natural Language Generation (NLG)	- shap.TreeExplainer for attribution- lime.lime_tabular for local interpretability- LLMs (OpenAI GPT, Anthropic Claude) for remediation summaries & board-level reports	- Combine SHAP/LIME outputs with LLM prompt templates- Restrict LLM use to reporting/justification, not raw prediction- Host LLM integration behind API Gateway for auditability

MVP for V2 Agentic AI

Minimum viable version includes:

1. Bayesian attack graph inference API
2. Supervised CVE exploitation prediction API
3. Simple decision engine to rank risks and suggest mitigation
4. Execution agent that logs actions (can later connect to orchestration tools)
5. Calibration + explainability API

Production Plan

- **Phase 1 (Prototype)**: Implement APIs for Bayesian graphs, supervised learning, GNN, calibration + explainability.
- **Phase 2 (Agentic Layer)**: Build the decision engine and integrate with perception & execution modules.
- **Phase 3 (Automation)**: Connect to patching tools (Ansible, SCCM), ticketing systems (Jira, ServiceNow), and SOC dashboards.
- **Phase 4 (Learning)**: Implement continuous retraining pipelines and active learning feedback loops.
- **Phase 5 (Explainability)**: Add natural-language report generation for executive summaries.

Ref:

1. https://arxiv.org/abs/2405.03513#:~:text=To%20bridge%20this%20gap%2C%20we%20introduce%20QBER%20approach,existing%20cybersecurity%20measures%2C%20and%20provides%20thorough%20cost%20assessments.
2. https://arxiv.org/pdf/2206.11586
3. https://www.infosys.com/iki/perspectives/cybersecurity-risk-management.html