Appendix A – Code Gates/Security Functions Summary

Department of Defense (DoD) Chief Information Officer DoD Enterprise DevSecOps Reference Design Version 1.0, 12 August 2019 depicts the phases in the DevSecOps process where the security functions and gates should be. DevSecOps is about including security functions at each phase. Gates are for testing code as it traverses the pipeline before it resides and functions on a production system. The same security functions can exist across multiple phases as shown in column B. The Gates and feedback loops presented are a framework and may vary based on the pipeline construction. The listing below although referring to everything as "Gates", includes the security functions that need to be part of the DevSecOps Pipeline and would be looked at by an AO. *This chart introduces the term “categories” and then maps the gates and functions to the DevSecOps Phases.	DevSecOps Phases 1. Plan 2. Develop 3. Build 4. Test 5. Release/Deliver 6. Deploy 7. Operate 8. Monitor
1. Category One Gates: Compliance with the DoD Enterprise DevSecOps Ref Design	All phases
1.1. Does the environment fully comply and if not where are the differences? What is being used as the Orchestrator?	
2. Category Two Gates: GitOps/Infrastructure as Code	
2.1. Changes are code in Git, no changes in production. This assumes the use of Git, if not using Git indicate what is being used? What is the process for controlling where code changes are being made and preventing changes in production?	Phases 1-5
2.1.1. Git becomes the source of truth, the desired state. That includes networking (firewall...), configuration changes. Is Git or "chosen" tool the source of truth? Are code base images kept up to date and hardened so they deploy securely? Is the environment managed using Configuration as Code, Compliance as Code, Security as Code, Infrastructure as Code?	
2.2. Staging/Production environment deployment:	Phases 6-8
2.2.1. Pull from Git” (no push allowed). Are pulls from production rather than push to production to move code being performed and if not, what is the logic for doing a push?	
2.3. Disaster Recovery:	All Phases
2.3.1. Automatic Backing up GIT repo and databases provide full disaster recovery (Resiliency Factor). What is the backup and disaster recovery process and schedules?	
3. Category Three Gates: Change management	
3.1. Segmentation of code (Need to know / Least privilege): Do you have separate repositories for each of the categories listed? How is separation configured and controlled?	All Phases
3.1.1. Dedicated/separate repositories for IaC/Kubernetes management.	

3.1.2. Dedicated/separate repository for Service Mesh (for whitelist rules etc.).	
3.1.3. Dedicated/separate repositories per container.	
3.2. Separation of duties: How many eyes are on each subject? Are the people different from subject to subject? If there is overlap specify where and why.	
3.2.1. Number of set of eyes for code change for IaC/Kubernetes management: Minimum of two.	
3.2.2. Number of set of eyes for code change for Networking/Service Mesh: Minimum of two.	
3.2.3. Number of set of eyes for code change for containers: Minimum of two.	
4. Category Four Gates: Cybersecurity	
4.1. Static Code Analysis	Phases 2-3
4.1.1. Based on severity of findings, can either break the build or pass the build with X days to fix. Findings can be whitelisted by X set of eyes to pass the pipeline when no mitigation can be done. Explain the process for performing static code analysis and the security procedures that govern what passes and fails?	
4.2. Dynamic Code Analysis	Phases 4-5
4.2.1. Based on severity of findings, can either break the build or pass the build with X days to fix. Findings can be whitelisted by X set of eyes to pass the pipeline when no mitigation can be done. Explain the process for performing dynamic code analysis and the security procedures that govern what passes and fails? If not performing dynamic code analysis, state the justification for why?	
4.3. Bill of Material / SW Supply Chain	Phases 1-3
4.3.1. Produce list of dependencies and their security findings. Based on severity of findings, can either break the build or pass the build with X days to fix. Findings can be whitelisted by X set of eyes to pass the pipeline when no mitigation can be done. How does your pipeline generate the BoM and where/what is being used for SW supply chain?	
4.4. Container Security Scanning: Two scanners minimum	Phases 4,6, and 8
4.4.1. Based on severity of findings, can either break the build or pass the build with X days to fix. Findings can be whitelisted by X set of eyes to pass the pipeline when no mitigation can be done. How many container security scanners are being used? Where in the pipeline are they performing their functions? What is the process for examining and mitigating findings?	
5. Category Five Gates: Testing	
5.1. Percentage of unit tests: What is the percentage of unit tests?	Phases 3-4
5.2. Percentage of Integration test coverage: What is the percentage of integration test coverage?	Phases 4-5
5.3. Percentage of integration tests: What is the percentage of integration tests?	
5.4. Percentage of End-to-end testing coverage: What is the percentage of end-to-end testing coverage?	Phases 5-6

6. Category Six Gates: Continuous Monitoring	
6.1. Real Time Dashboards Reporting: What is being used to provide real time dashboards for alerting? Who is developing the dashboards? What for tools and reports are reporting to the dashboard tool?	This generally refers to the monitoring phase, although best practice is to apply to as many phases as possible. For example, dashboards can be utilized with many of the tools through the pipeline. Zero Trust, alerting and centralized logging should apply across the width and length of the pipeline.
6.2. Sidecar Container Security Stack: Describe the implementation and flow into and out of the sidecar container security stack?	
6.3. Container Behavior Detection:	
6.3.1. Detection mode or preventive mode. Preventive mode kills containers when bad behavior is detected after X days of learning behavior. Explain the implementation, detection and/or preventive and if applicable the days allow for learning behavior?	
6.4. Zero trust enforcement:	
6.4.1. mTLS tunnel using strong identifies (x509 certs etc.) with deny all by default and whitelist of traffic by Service Mesh team. Is zero trust being utilized? What is the authn/authz process?	
6.5. Alerting:	
6.5.1. Automated alerting with SIEM/Alerting system etc. with the right level of alerting based on severity either email or page. What for SIEM tool is being used? What for tools/software/hardware is reporting to the SIEM? Is the SIEM providing real time alerts on its own, is it reporting to another tool, or is it the central dashboard tool?	
6.6. Centralized logging and telemetry:	
6.6.1. EFK is mandated to consolidated logs and telemetry across the cluster: What tool is being used for centralized logging and is all components reporting to the tool? If components are not reporting to the tool those components need listed.	
6.7. CVE scanning:	Phases 7-8
6.7.1. Continuously scan CVEs, including in registry and runtime. What tool is being used to continuously scan for CVEs and is it covering the registry and runtime environments? What is the process/time frame to remediate/mitigate findings?	
7. Category Seven Gates: Penetration testing	
7.1. Continuous vulnerability scanning/pen test Frequency: How is pen testing implemented and how often? Explain the scope and process.	Should be done at various phases randomly.
7.2. Continuous assessment of RBAC / identities to enforce need to know / least privilege with frequency of the assessment. How is he subjects implemented, enforced, and assessed?	