Defense in Depth Table for Application Security

The goal is for you to fill this table out with your team. Each section may contain people, process or technology for deploying defense in depth. For a more exhaustive list of possible controls to add to your defense in depth table see the NIST Cybersecurity Framework

DEFENSE LAYER	SECURITY CONTROLS			
Create Secure Applications ● Controls that help developers create secure code	**Education, Awareness, and Secure Code Training.** ● Code reviews ● Annual secure code training ● Communities of practice ● First-class events ● Office hours			
Analyze Applications ● Assess each code change for new and existing vulnerabilities	**Static Analysis** Automatically can our source code for known vulnerabilities ● Rails: Brakeman ● Elixir: sobelow ● PHP: Veracode, semgrep ● Python: Bandit	**Dynamic Analysis** Automatically examine our live running application for vulnerabilities ● Rapid7	**Third-party Code Analysis** Tool(s) scan our third-party libraries for known vulnerabilities ● dependabot ● safety ● snyk	**Manual Analysis** Testing application for issues that cannot be identified automatically: ● Abuse of functionality vulnerabilities ● Privilege escalation vulnerabilities ● Business logic flaws

				Uses manual tools like burp suite, secure code checklists and browser plugins.
	External Code & Architecture Reviews ● Ad-hoc threat modeling exercises ● Security champion boards (ie. standard releases, pull request template, sprint planning risk assessments ● Regulatory gap analyses			
	Third Party Penetration Testing ● External application testing ● Responsible disclosure programs and bug bounties			
Protect Applications Detect and block attacks occurring against deployed code	● Web Application Firewall			
Response Processes Identify and eliminate any successful attacks against cmm assets	● Incident response procedure ● SIEM logging & monitoring			