Rezilion Prioritize Datasheet

Security is a difficult balancing act in a DevOps environment, apply it too strictly and nothing gets released, too passively and risk quickly accumulates. This leads to constant back and forth between devops and security - identifying vulnerabilities, prioritizing patches, and finally passing builds for production release. Worse yet, oftentimes vulnerabilities are flagged in components that will never be loaded to memory and therefore pose no risk.

Additionally, production environments are full of legacy code and backlogs of vulnerabilities that would take years to patch. Security and DevOps teams need an understanding of their actual attack surface consisting only of exploitable vulnerabilities.

Rezilion Prioritize allows Security teams to keep pace with DevOps and reduces work for both teams in the process. Traditional scanning solutions create work, Prioritize reduces it by allowing users to focus solely on exploitable vulnerabilities. Leveraging existing Security and DevOps tools and processes, Rezilion Prioritize eliminates the manual work required to protect applications from vulnerabilities and threats.

Key Benefits:

- **Patch Up to 70% Less**
  Save countless hours and dollars by focusing time and resources on vulnerabilities that actually pose a risk.
- **Release Faster**
  Less unplanned work means releases happen faster without introducing risk into production.
- **Improve Relationships Between Security and DevOps**
  Unplanned work and false positives strain relationships, Prioritize eliminates both and brings teams together.
- **Remove Code Bloat**
  Unloaded packages can be removed from images for more maintainable code. This also allows you to harden images for security and compliance requirements.
- **Reduce Tech Debt**
  Identify unused legacy components that can be removed from the environment without downstream risk
- **Measure the Actual Attack Surface**
  Demonstrate week over week progress remediating exploitable vulnerabilities and remove distractions.

Key Features:

- **Patent Pending Vulnerability Validation Technology**
  Validation is an automated vulnerability analysis that reverse-engineers cloud workloads and applications. It automatically creates an inventory of all artifacts in dev and prod, and maps dependencies, connections, code provenance, memory and runtime execution flows. This inventory is compared to your scan results and reduces patching efforts by 70% by identifying exploitable vulnerabilities in code loaded to memory.

- **Full Stack Visibility**
  Prioritize aggregates and validates vulnerabilities across the entire workload. Vulnerabilities in the application, container, and the infrastructure layers are displayed in a single pane of glass view that can be customized based on the user's needs.

- **Package Level Remediation Information**
  DevOps practitioners don't speak in CVEs and Prioritize doesn't require them to. Prioritize captures package level information and ranks them in order of risk so teams know exactly what changes need to be made to have the greatest impact.

- **No Tuning Required**
  Vulnerability validation is fully deterministic and doesn't rely on tuning policies or creating baselines. This allows Prioritize to maintain accuracy whether you're pushing code once a month or several times a day.

- **Flexible Deployment in as Little as 15 Minutes**
  Prioritize can be deployed in CI, Staging, or Production environments in a variety of architectures including on-premise, hybrid, cloud, containers, and kubernetes. Deployment options include a CI plugin, sidecar container, YAML, or a continuously running script.

- **Seamless Integration with Existing Scanning Technologies**
  Prioritize is not a scanner but it's also not meant to disrupt existing workflows or displace existing technology. Highly extensible APIs allow Prioritize to integrate with any of the leading SCA, and infrastructure scanning tools. Alternatively, Prioritize can be packaged with an open source scanning solution in environments that don't currently have tools in place.