Project StartLLM: UC-01: Pentesting Insights Acceleration (PIA)

by Dinis Cruz and Claude 3.7", 2025/03/01

Use Case (UC) Overview

Title: Pentesting Insights Acceleration (PIA)

Reference: This document details Use Case UC-01 as part of the Project StartLLM initiative outlined in the main proposal.

Senior Stakeholders: Head of Security Operations and Chief Risk Officer, or Security consulting companies that want to create briefs for them

Brief Description: Leverage GenAI to transform penetration testing outputs and security assessments into standardized risk materials that can be effectively utilized by non-security teams.

Current Situation Analysis

Existing Process Description:

Security teams conduct penetration tests and security assessments on company systems and applications, capturing findings in technical reports primarily written for security experts.

These reports typically range from 20-50 pages in PDF or Word format and are stored in the company's Confluence workspace under the Security team space. Critical security insights often remain siloed within these technical documents or, worse, exist only as undocumented knowledge in the minds of

security professionals.

Risk teams and other business units currently spend 4-8 hours per assessment manually translating these technical findings into actionable risk information.

Current Workflow:

The existing workflow involves security professionals completing assessments and documenting findings in technical formats using the company's standard pentesting templates. These reports are stored in Confluence with limited distribution permissions. Risk team members must schedule meetings with security staff or spend hours combing through technical documentation to extract relevant information for their risk registers and executive reports. The business context and implications of security findings are frequently diluted or misinterpreted during this manual translation process, leading to misaligned priorities and delayed remediation efforts.

Pain Points:

The current process requires significant time investment from both security and risk teams, with an average of 6 hours spent on translation activities per assessment. Valuable contextual information often goes uncaptured, particularly the security team's professional judgment about business impact and exploitation likelihood. The communication gap between security and risk teams leads to inconsistent prioritization, and security professionals waste valuable time repeatedly explaining the same concepts to different stakeholders throughout the organization.

Use Case Requirements

Primary Objective: Develop a simple GenAI solution using an LLM (like Claude 3.7) that captures, processes, and transforms penetration testing outputs into standardized risk materials with minimal disruption to existing security workflows.

Principles and Constraints:

- **Minimum Friction**: Solution must not add significant overhead to existing security workflows
- **GitHub-Centric**: Use GitHub as the primary interface for document handling and workflow

- **Security First**: Ensure all security data remains protected in private repositories
- **Simplicity**: Focus on straightforward implementation using existing tools rather than complex new systems
- **Rapid Implementation**: Deployable within 3 days (working in 2-hour blocks)
- **Expert Review**: Outputs must be reviewable by security professionals before dissemination

Specific Requirements:

The solution must accept existing pen test reports (PDF/DOCX) and text notes with minimal additional effort from security professionals. It should integrate easily with the existing Confluence documentation workflow.

The GenAI component needs to extract key security findings and vulnerabilities, identify business impact and risk implications, categorize findings according to standard risk frameworks, and maintain technical accuracy while translating into business language that non-security teams can understand and act upon.

The system should generate multiple output formats tailored to different audiences: executive summaries for leadership (2-3 pages), detailed risk assessment documents for risk teams (5-10 pages), presentation materials (PowerPoint format) for stakeholder communication, and remediation roadmaps with prioritized actions.

Constraints:

The solution must not require significant changes to how penetration testers work, should be deployable within 3 days (working in 2-hour blocks), must handle sensitive security information appropriately within private GitHub repositories, and must ensure outputs are reviewable by security professionals before dissemination to other teams.

Implementation Assumptions

Existing Materials and Infrastructure:

- Penetration test reports exist in PDF and DOCX formats, typically 20-50 pages, following internal templates
- Reports are currently stored in Confluence under the Security team space
- All reports follow a similar structure with sections for findings, impact, and recommendations
- Company uses CVSS for vulnerability scoring and has an internal risk categorization framework
- Security and risk teams both have access to the company GitHub Enterprise instance

Technical Architecture:

- GitHub will serve as the main workflow interface for the entire solution
- All source documents (reports and notes) will be stored in a dedicated private GitHub repository
- Security professionals will upload reports to a designated "input" folder in the repository
- GitHub Actions will be used for workflow orchestration and to trigger API calls to Claude 3.7
- Processed outputs will be stored in an "output" folder in the same repository
- Authentication and access control will be managed through GitHub's existing permissions system
- Repository-based notifications will alert stakeholders when new materials are available
- No complex data pipelines will be created at this stage

Data Privacy and Security:

- All penetration testing data will remain within private GitHub repositories
- Repository access will be strictly limited to authorized security and risk team members
- API calls to Claude will use company's existing API keys and security protocols
- No customer data is included in the pentesting reports
- Reports only contain information about internal systems that is already shared between security and risk teams

---

Technical Solution Design

Proposed Technical Approach:

We'll implement a straightforward GenAI solution using Claude 3.7, focusing on its strong capabilities in understanding technical documentation and converting it to different audience-appropriate formats. The solution will use GitHub as the central platform for document management, workflow orchestration, and access control, with Claude 3.7 providing the core intelligence.

We'll use Claude directly with custom prompts designed for security context extraction from the pentesting reports and notes. We'll develop a set of carefully engineered prompts that instruct Claude to identify critical security findings, translate technical details, prioritize based on business impact, and format outputs according to predefined templates.

Processing Flow:

A security professional will upload a penetration test report to the designated "input" folder in the GitHub repository. This will automatically trigger a GitHub Action workflow that processes the document. The workflow will extract the content and send it to Claude 3.7 with tailored prompts based on the input type. Claude will process the input and generate draft outputs in multiple formats according to predefined templates.

The workflow will then store these draft outputs in a "review" folder and notify the security professional. The security professional can review the outputs directly in GitHub, and when satisfied, approve them by moving them to the "output" folder (either manually or via a simple approval workflow). Upon approval, GitHub notifications will alert risk team members that new materials are available in the output folder.

Implementation Plan

Day 1: Setup and Initial Configuration (4 hours total)

First Block (2h): Meet with key security and risk stakeholders to gather specific requirements and examples of existing reports. Set up the GitHub repository structure with input, review, and output folders. Configure repository permissions to ensure proper access control. Begin creating GitHub Action workflow templates for document processing.

Second Block (2h): Develop initial set of Claude prompts based on example reports. Test basic extraction capabilities with 1-2 sample reports. Create templates for the different output formats (executive summary, risk assessment, presentation). Configure GitHub notifications and document the upload process.

Day 2: Development and Testing (6 hours total)

First Block (2h): Complete the GitHub Actions workflows for document processing. Create scripts to extract content from different document types. Implement the Claude API integration and test with actual report content.

Second Block (2h): Develop and test the end-to-end workflow with sample documents. Refine Claude prompts based on initial results. Implement the document review and approval process within GitHub using appropriate branch and folder structures.

Third Block (2h): Process 2-3 existing penetration test reports as test cases. Review outputs with security and risk representatives. Further refine prompts and output templates based on feedback.

Day 3: Refinement and Deployment (6 hours total)

First Block (2h): Address feedback from testing and make final adjustments to prompts and templates. Create user documentation including a README in the GitHub repository with clear instructions for all users.

Second Block (2h): Deploy the final workflow in the production GitHub repository. Ensure all permissions are correctly set. Conduct a 30-minute orientation session with security team members who will be using the system. Process the first live penetration test report.

Third Block (2h): Gather initial feedback from users. Make quick adjustments as needed to GitHub Actions workflows or prompts. Document lessons learned and plan for potential enhancements through GitHub Issues for future development.

Success Criteria and Measurement

Key Performance Indicators:

The primary success metric will be time efficiency, with the goal of reducing the time to create risk materials from 4-8 hours to less than 1 hour per assessment, with security team time investment reduced to 30 minutes or less per assessment.

Quality will be measured by ensuring risk materials accurately reflect security findings with correct prioritization, maintain technical accuracy while improving accessibility for non-technical audiences, and provide complete information compared to the manual process.

Adoption will be tracked through the percentage of penetration tests using the new process, security team satisfaction ratings, and risk team satisfaction with output quality.

Measurement Method:

We'll track time spent on the first five processed assessments compared to historical baselines. Both security and risk team representatives will conduct quality reviews of the outputs. We'll distribute stakeholder surveys after the first two weeks of use and track the number of clarification requests from the risk team to the security team, which should decrease if the solution is effective.

Integration and Sustainability

The solution is designed to fit into existing penetration testing workflows with minimal disruption. Security professionals will continue to create reports as they normally do, with the PIA system serving as an additional post-processing step that they trigger simply by uploading documents to GitHub. All security controls are maintained through GitHub's existing permissions system, ensuring that generated materials remain protected with the same level of security as the source documents.

We'll identify a solution administrator within the security or risk team who will maintain the GitHub repository, Actions workflows, and Claude prompts going forward. The repository will include detailed documentation on the prompt engineering approaches used, enabling future refinement. GitHub Issues will provide a simple feedback mechanism for tracking enhancement requests and bugs, supporting continuous improvement of the system.

Expected Benefits

For Security Teams: Security professionals will spend significantly less time translating technical findings for business audiences, leading to more time for actual security work. The solution will provide better communication of security concerns to decision-makers, improved knowledge transfer efficiency, and more consistent representation of security findings across assessments.

For Risk Teams: Risk management staff will gain faster access to security insights in appropriate risk language, more comprehensive understanding of technical security issues, better prioritization data for remediation planning, and improved ability to communicate security needs to business stakeholders.

For the Organization: The organization will benefit from accelerated risk response to security findings, better alignment between security investments and business risk priorities, more efficient use of specialized security resources, and enhanced security knowledge sharing across teams.

Next Steps

To begin implementation next week, we require:

1. Access to 3-5 sample penetration test reports (redacted if necessary)
2. Introduction to key security and risk team members
3. Access to create a private repository in the company's GitHub Enterprise instance
4. Company's Claude API credentials
5. Permissions to create GitHub Actions workflows in the organization

Upon approval of this use case, we'll schedule the kick-off meeting and begin implementation according to the outlined plan.