Contents

[6.9] Organizational Splitting

1. Operational Definition: A Kleinian defense mechanism where the organization unconsciously splits itself into "all-good" (idealized) and "all-bad" (demonized) parts. This manifests as a rigid separation and lack of communication between teams (e.g., "Dev" vs. "Ops", "Blue Team" vs. "Red Team"), blame-shifting, and a failure to integrate security into development lifecycles.

2. Main Metric & Algorithm:

- **Metric:** Cross-Team Collaboration Index (CTCI). Formula: `(Number of shared channels/threads/tickets between Team A and Team B) / (Total activity of Team A and Team B)`.
- **Pseudocode:**

  python

  ```
  def calculate_ctci(team_a, team_b, start_date, end_date):
      """
      team_a, team_b: Lists of user identifiers.
      """
      # Get messages/tickets involving members from both teams
      cross_team_items = get_shared_items(team_a, team_b, start_date, end_date)
      # Get all messages/tickets for both teams
      total_team_a_items = get_activity(team_a, start_date, end_date)
      total_team_b_items = get_activity(team_b, start_date, end_date)
      total_activity = total_team_a_items + total_team_b_items
      return len(cross_team_items) / total_activity if total_activity > 0 else 0
  ```

- **Alert Threshold:** `CTCI < 0.05` (Less than 5% of total activity involves cross-team collaboration).

3. Digital Data Sources (Algorithm Input):

- **Collaboration Platform API (Slack/Teams):** Data: `messages` in shared channels, `replies`, `mentions` between members of different teams.
- **Ticketing System API (Jira):** Data: `issues` with participants/commenters from multiple teams, `issue_links` between tickets from different teams.

4. Human-To-Human Audit Protocol: Conduct a workshop with representatives from both teams. Present the low CTCI metric. Use a blame-free format like "What are the barriers to working more closely with team X? What would make collaboration easier? What is one small project we could do together next quarter?"

5. Recommended Mitigation Actions:

- **Technical/Digital Mitigation:** Create mandatory shared digital spaces for projects (e.g., a `#app-sec-dev-collab` channel for every development squad that includes a dedicated security engineer).

- **Human/Organizational Mitigation:** Implement a job rotation program where security engineers embed within development teams for short periods, and vice-versa.
- **Process Mitigation:** Shift security left by integrating security goals and metrics into the development teams' Objectives and Key Results (OKRs), making it a shared responsibility, not a separate "gate."