Category 3: Social Influence Vulnerabilities

Contents

This directory contains detailed implementation schemas for all 10 indicators in the Social Influence vulnerability category.

Overview

Social influence vulnerabilities exploit human tendencies toward conformity, social proof, peer pressure, and trust in social networks.

Indicators

1. **[3.1] Social Proof Exploitation** - Following others' risky behaviors
2. **[3.2] Peer Pressure Security Bypass** - Conformity to group norms over policy
3. **[3.3] Trust Network Exploitation** - Leveraging social relationships

4. [**3.4**] **Consensus Bias in Decisions** - Deferring to group consensus
5. [**3.5**] **Social Validation Seeking** - Need for approval overriding security
6. [**3.6**] **In-Group Trust Bias** - Reduced scrutiny of in-group members
7. [**3.7**] **Reciprocity Exploitation** - Quid pro quo social contracts
8. [**3.8**] **Social Identity Threat** - Identity-based manipulation
9. [**3.9**] **Celebrity/Influencer Exploitation** - Leveraging social status
10. [**3.10**] **Network Effect Cascades** - Viral spread of risky behaviors

Implementation Schema

Each indicator follows the **OFTLISRV** framework with focus on social graph analysis.

Key Metrics

Social Proof Compliance Rate

```
SPCR = N_followed_risky / N_exposed_to_risky
```

Measures tendency to follow others' risky behaviors.

Trust Network Exploitation Score

```
TNES = Σ(Trust_weight × Risk_behavior) / N_relationships
```

Cascade Propagation Factor

```
CPF = N_influenced / N_initial_actors
```

Measures viral spread of behaviors through network.

Key Data Sources

- **Social Graphs**: Organizational chart, email networks, collaboration tools
- **Communication Platforms**: Slack, Teams, email interaction patterns
- **SIEM**: Correlated events across social connections
- **HR Systems**: Department, team, reporting relationships
- **Badge/Access**: Physical proximity patterns

Detection Approach

Social Graph Analysis

```python
# Build trust network
G = build_social_graph(email_data, slack_data)

# Identify risky behavior propagation
for user in G.nodes:
    peer_behaviors = [G.neighbors(user).behaviors]
    user_behavior = user.behaviors

    if user_behavior == peer_behaviors.mode():
        social_proof_triggered = True
```

Cascade Detection

```
# Track behavior spread
cascade = detect_cascade(
    initial_event=policy_bypass,
    time_window=24_hours,
    network=social_graph
)

if len(cascade.nodes) > threshold:
    alert_cascade_detected()
```

Baseline Establishment

Social indicators require: - 90-day social graph construction - Normal communication patterns - Peer influence baselines - Trust network mapping

Common Event Types

- `peer_action_observed` → 3.1, 3.4
- `in_group_request` → 3.6
- `reciprocal_favor` → 3.7
- `influencer_post` → 3.9
- `cascade_propagation` → 3.10

Risk Levels

- **Low** (0-0.33): Independent decision-making maintained
- **Medium** (0.34-0.66): Some peer influence, verification still occurs
- **High** (0.67-1.00): Systematic conformity, reduced critical thinking

Related Resources

- **Dense Foundation**: `/foundation docs/core/en-US/` - Social influence models
- **Dashboard**: `/dashboard/soc/` - Social network visualization
- **Graph Analysis**: NetworkX-based social graph tools