Contents

[1.1] Conformità incondizionata all'autorità apparente

1. Definizione operativa: Una vulnerabilità in cui il personale di sicurezza si conforma automaticamente alle richieste di individui percepiti come figure di autorità, senza eseguire i controlli di verifica necessari, aumentando il rischio di attacchi di social engineering riusciti come la frode dell'amministratore delegato.

2. Metrica principale e algoritmo:

- **Metrica:** Tasso di richiesta di autorità non verificato (UARR). Formula: `UARR = (Numero di richieste non verificate ad alto impatto da "autorità") / (Numero totale di richieste ad alto impatto da "autorità")`.
- **Pseudocodice:**

  python

```
def calculate_uarr(access_logs, comms_data, start_date, end_date):
    """
    access_logs: Lista di log da sistemi di controllo accesso/rete
    comms_data: Lista di messaggi da piattaforme email/chat
    """
    # 1. Identificare potenziali figure di autorità (ad es., livello C, VP) da dati HR o m
    authority_users = identify_authority_figures(comms_data)

    # 2. Estrarre tutte le richieste ad alto impatto da questi utenti (ad es., reset passw
    authority_requests = [
        req for req in extract_requests(comms_data, start_date, end_date)
        if req.sender in authority_users and req.impact == 'high'
    ]

    # 3. Fare riferimento incrociato con i log di accesso per verificare la verifica (ad e
    unverified_count = 0
    for req in authority_requests:
        # Verificare se la richiesta è stata seguita da un'azione senza un ticket collegat
        corresponding_actions = find_corresponding_actions(access_logs, req)
        if not was_request_verified(corresponding_actions):
            unverified_count += 1

    # 4. Calcolare UARR
    total_requests = len(authority_requests)
    UARR = unverified_count / total_requests if total_requests > 0 else 0
    return UARR
```

- **Soglia di avviso:** `UARR > 0.2` (Più del 20% delle richieste di autorità ad alto impatto viene eseguito senza verifica)

3. Fonti di dati digitali (input dell'algoritmo):

- **Piattaforme di posta/chat (Microsoft Graph API, Slack API):** Per estrarre richieste contenenti parole chiave ad alto impatto (`"ho bisogno di accesso a"`, `"reset password per"`, `"urgente"`) da mittenti identificati come figure di autorità.
- **Log IAM/Active Directory:** Per verificare se un'azione richiesta è stata eseguita.
- **API del sistema di ticketing (Jira, ServiceNow):** Per verificare se un'azione era collegata a un ticket convalidato e approvato.

4. Protocollo di audit da umano a umano: Condurre un esercizio di phishing simulato mirato ad analisti SOC con un messaggio che impersona un noto dirigente aziendale (ad es., CISO) richiedendo un'azione ad alto impatto come la disabilitazione di un controllo di sicurezza. Misurare la percentuale di analisti che si conformano senza cercare verifica secondaria tramite un canale diverso (ad es., una chiamata telefonica).

5. Azioni di mitigazione consigliate:

- **Mitigazione tecnica/digitale:** Implementare un flusso di lavoro tecnico obbligatorio che blocchi le azioni ad alto impatto (ad es., concessioni di accesso privilegiato) richieste tramite chat/email a meno che non siano collegate a un numero di ticket approvato nel sistema.
- **Mitigazione umana/organizzativa:** Condurre una formazione mirata su bias di autorità, concentrandosi su procedure di verifica stabilite per tutte le richieste, indipendentemente dalla seniority percepita del richiedente.
- **Mitigazione dei processi:** Formalizzare un principio "4 occhi" nel playbook SOC per qualsiasi azione richiesta al di fuori del sistema di ticketing, richiedendo conferma obbligatoria da un supervisore di turno.