Contents

[3.6] Sfruttamento del Principio di Unità

1. Definizione Operativa: La tendenza a favorire le richieste da individui percepiti come parte del proprio "gruppo interno" (ad esempio, stesso team, azienda o background), portando a uno scrutinio di sicurezza ridotto basato su un falso senso di identità condivisa.

2. Metrica Principale e Algoritmo:

- **Metrica: Indice di Trattamento Preferenziale del Gruppo Interno (IPTI)**. Formula: `IPTI = (N_bypass_interno / N_richieste_interno) / (N_bypass_esterno / N_richieste_esterno)`.
- **Pseudocodice:**

  python

```
def calculate_ipti(access_logs, request_logs, hr_data):
    """
    Confronta i tassi di bypass per le richieste interno vs. esterno.
    """
    # 1. Definisci il gruppo interno (ad esempio, stesso dipartimento) e il gruppo esterno
    user_departments = get_department_mapping(hr_data)

    in_group_requests = 0; in_group_bypass = 0
    out_group_requests = 0; out_group_bypass = 0

    for request in request_logs:
        requester = request.requester_id
        grantor = request.grantor_id
        # Verifica se il richiedente e chi concede l'accesso sono nello stesso gruppo inte
        if user_departments[requester] == user_departments[grantor]:
            in_group_requests += 1
            if request.was_bypassed: # ad esempio, nessun ticket, eccezione policy
                in_group_bypass += 1
        else:
            out_group_requests += 1
            if request.was_bypassed:
                out_group_bypass += 1

    bypass_rate_in = in_group_bypass / in_group_requests if in_group_requests > 0 else 0
    bypass_rate_out = out_group_bypass / out_group_requests if out_group_requests > 0 else

    IPTI = bypass_rate_in / bypass_rate_out if bypass_rate_out > 0 else float('inf')
    return IPTI
```

- **Soglia di Allerta:** `IPTI > 2.0` (Il tasso di bypass per le richieste interno è il doppio di quelle esterno).

3. Fonti di Dati Digitali (Input dell'Algoritmo):

- **API Database HR:** Per determinare le appartenenze ai gruppi (dipartimento, team, ubicazione). Campi: `user_id`, `department_id`, `team_id`.
- **Log Richieste di Accesso (ServiceNow, Jira, IAM personalizzato):** Per ottenere un record delle richieste e il loro stato di approvazione. Campi: `requester`, `grantor`, `approval_status`, `timestamp`, `policy_exception_flag`.
- **Log di Accesso (come sopra):** Per dedurre i bypass non acquisiti nei log di richiesta.

4. Protocollo di Audit Umano-Umano: Presenta agli auditor una serie di scenari di richiesta di accesso ipotetica da richiedenti "interno" e "esterno". Misura la differenza nella legittimità percepita della richiesta e nella probabilità di concederla. In alternativa, analizza i ticket storici di approvazione dell'accesso per la correlazione tra il dipartimento di chi concede/richiedente e la velocità di approvazione.

5. Azioni di Mitigazione Consigliate:

- **Mitigazione Tecnica/Digitale:** Implementa il controllo di accesso basato su attributi (ABAC) dove possibile, automatizzando le decisioni in base agli attributi dell'utente e alla sensibilità della risorsa, rimuovendo il bias del gruppo interno dell'equazione umana.
- **Mitigazione Umana/Organizzativa:** Condurre formazione sul bias del gruppo interno, sensibilizzando i team sulla tendenza e incoraggiandoli ad applicare gli stessi standard di sicurezza a tutti, indipendentemente dall'affiliazione.
- **Mitigazione del Processo:** Nascondi al concedente certi attributi del richiedente (come il dipartimento) durante il processo di approvazione per determinate richieste a basso rischio, forzando un focus sul merito della richiesta.