Product Requirements Document

Temporal Dimension Enhancement: 6-Tuple Contextual Integrity Framework

Version: 1.0
Date: January 2025
Author: Privacy Engineering Team
Stakeholder: Security Operations, Compliance, Clinical Emergency Response

---

Executive Summary

Enhance the Privacy Firewall's Contextual Integrity framework from 5-tuple to 6-tuple by adding a **temporal dimension**, enabling situation-aware privacy decisions that adapt to time-based contexts, emergencies, and dynamic access windows, reducing inappropriate access denials by 67% while maintaining security.

Problem Statement

The current 5-tuple framework **(data type, data subject, data sender, data recipient, transmission principle)** is temporally blind, causing:

- **Emergency Access Failures**: legitimate after-hours medical emergency accesses blocked
- **Stale Permissions**: Temporary roles/projects persist indefinitely (zombie permissions)
- **Context Ignorance**: Earnings data were equally restricted before and after public release
- **Compliance Violations**: Cannot enforce time-windowed regulations (GDPR 72-hour rules)
- **Alert Fatigue**: false positives/week from legitimate time-shifted work

Solution: 6-Tuple Framework

Add **temporal context** as the sixth tuple element:

```
@dataclass
class EnhancedContextualIntegrityTuple:
   data_type: str              # What information
   data_subject: str            # About whom
```

```
    data_sender: str           # From whom
    data_recipient: str        # To whom
    transmission_principle: str # Under what agreement
    temporal_context: TemporalContext  # WHEN (NEW)
```

Temporal Context Components

```
@dataclass
class TemporalContext:
    # Absolute time
    timestamp: datetime
    timezone: str

    # Relative context
    business_hours: bool
    emergency_override: bool

    # Time windows
    access_window: TimeWindow  # Valid from/to
    data_freshness: timedelta  # Age of data

    # Situational flags
    situation: Enum[
        NORMAL,
        EMERGENCY,
        MAINTENANCE,
        INCIDENT_RESPONSE,
        AUDIT,
        LEGAL_HOLD
    ]

    # Temporal relationships
    temporal_role: Optional[str]  # "on-call", "acting-manager"
    event_correlation: Optional[str]  # Related to specific event/incident
```

Key Situation-Aware Scenarios

Scenario	5-Tuple Decision	6-Tuple Decision
ER doctor accessing patient records at 2 AM	❌ BLOCKED (outside hours)	✅ ALLOWED (emergency + on-call)

Manager viewing team salaries during review period	❌ BLOCKED (sensitive data)	✅ ALLOWED (review window active)
Auditor accessing 2-year-old financial records	❌ BLOCKED (historical)	✅ ALLOWED (audit situation + legal requirement)
DevOps accessing production data during incident	❌ BLOCKED (production data)	✅ ALLOWED (incident_response + 2hr window)
Contractor accessing project data after contract end	✅ ALLOWED (has role)	❌ BLOCKED (access_window expired)

Temporal Privacy Rules

```
- id: EMRG-001
  name: Emergency Medical Override
  tuples:
    data_type: medical_record
    data_sender: emergency_physician
    data_recipient: patient_care_team
    temporal_context:
      situation: EMERGENCY
      temporal_role: on-call
  action: ALLOW_WITH_AUDIT
  duration: 24_hours

- id: FIN-001
  name: Earnings Embargo
  tuples:
    data_type: earnings_data
    temporal_context:
      before_event: earnings_release
      buffer: -48_hours
  action: BLOCK
  exception_roles: [CFO, CEO, Investor_Relations]

- id: GDPR-001
  name: GDPR Breach Notification Window
  tuples:
    data_type: breach_details
    temporal_context:
      after_event: breach_detected
      window: 72_hours
    transmission_principle: regulatory_requirement
  action: EXPEDITE

- id: TEMP-001
  name: Acting Role Permissions
  tuples:
    data_sender: "{acting_role}"
    temporal_context:
      temporal_role: acting_*
      access_window:
        from: role_assignment_date
        to: role_end_date
  action: INHERIT_PERMISSIONS
```

Implementation Architecture

```
┌─────────────────────────────┐
│  Incoming Request           │
└──────────────┬──────────────┘
               ▼
┌─────────────────────────────┐
│ Temporal Context            │
│ Enrichment Service          │
├─────────────────────────────┤
│ • Current time/zone         │
│ • Business calendar         │
│ • On-call schedules         │
│ • Active incidents          │
│ • Temporal roles            │
└──────────────┬──────────────┘
               ▼
┌─────────────────────────────┐
│ 6-Tuple Construction        │
├─────────────────────────────┤
│ Standard 5 tuples +         │
│ temporal_context            │
└──────────────┬──────────────┘
               ▼
┌─────────────────────────────┐
│ Temporal Rule Engine        │
├─────────────────────────────┤
│ • Time window checks        │
│ • Situation detection       │
│ • Emergency overrides       │
│ • Expiry validation         │
```

```
└──────────────────┬──────────────────┘
        ▼
┌─────────────────────────────────────┐
│ Privacy Decision │
└─────────────────────────────────────┘
```

Temporal Data Sources

Source	Data	Update Frequency	Purpose
PagerDuty	On-call schedules	Real-time	Identify emergency responders
ServiceNow	Active incidents	Real-time	Incident response context
Workday	Temporal roles, PTO	5 min	Acting roles, coverage
Business Calendar API	Hours, holidays	Daily	Business context
Compliance Calendar	Regulatory deadlines	Daily	Time-window requirements
Project Management	Project timelines	Hourly	Time-bounded access

Time-Based Privacy Patterns

```
class TemporalPrivacyPatterns:
    # Emergency override with decay
    emergency_access = {
        "immediate": (0, timedelta(hours=1)),    # Full access
        "urgent": (1, timedelta(hours=4)),       # Degraded access
        "follow_up": (4, timedelta(hours=24)),   # Read-only
        "expired": (24, None)                    # No access
    }

    # Data sensitivity decay
    financial_data_sensitivity = {
        "embargo": (-48_hours, "earnings_release"),  # Highly restricted
        "public": ("earnings_release", +∞),          # Public info
    }

    # Consent expiration
```

```
consent_windows = {
    "medical_records": timedelta(days=365),
    "marketing_data": timedelta(days=90),
    "biometric_data": timedelta(days=30),
}
```