Generate a secure policy for API rate limiting . Ensure security best practices.

**API Rate Limiting Policy**

**1. Purpose and Scope**

This policy establishes standardized rate limiting controls for all API endpoints to protect system resources, ensure fair usage, prevent abuse, and maintain service availability for legitimate users.

**Scope**: All public-facing and internal APIs, including REST, GraphQL, and WebSocket endpoints.

**2. Rate Limiting Strategy**

**2.1 Multi-Tier Approach**

* **Global Rate Limits**: System-wide protection against traffic spikes
* **Per-User Limits**: Individual user quotas based on authentication
* **Per-IP Limits**: Network-level protection against anonymous abuse
* **Per-Endpoint Limits**: Resource-specific protection for critical operations

**2.2 Rate Limiting Algorithms**

* **Token Bucket**: Primary algorithm for burst traffic handling
* **Fixed Window**: Secondary algorithm for strict periodic limits
* **Sliding Window**: For precise rate calculations over time periods

**3. Rate Limit Tiers and Quotas**

**3.1 User Authentication Tiers**

**Anonymous Users**

* **Requests per minute**: 60
* **Requests per hour**: 1,000
* **Requests per day**: 10,000
* **Concurrent connections**: 5

**Authenticated Users (Basic)**

* **Requests per minute**: 300
* **Requests per hour**: 10,000
* **Requests per day**: 100,000
* **Concurrent connections**: 20

**Premium Users**

* **Requests per minute**: 1,000
* **Requests per hour**: 50,000
* **Requests per day**: 1,000,000
* **Concurrent connections**: 100

**Enterprise Users**

* **Requests per minute**: 5,000
* **Requests per hour**: 200,000
* **Requests per day**: 5,000,000
* **Concurrent connections**: 500

**3.2 Endpoint-Specific Limits**

**High-Impact Operations**

* **User registration**: 5 per hour per IP
* **Password reset**: 3 per hour per user
* **File uploads**: 10 per minute per user
* **Data export**: 2 per hour per user

**Authentication Endpoints**

* **Login attempts**: 5 per minute per IP
* **Token refresh**: 10 per minute per user
* **Account lockout**: After 10 failed attempts in 15 minutes

**4. Implementation Requirements**

**4.1 Technical Implementation**

* Rate limiting must be implemented at the API gateway level
* Distributed rate limiting using Redis or similar shared cache
* Rate limit counters must be atomic and consistent across instances
* Support for both synchronous and asynchronous rate limiting

**4.2 Rate Limit Headers**

All API responses must include standard rate limit headers:

X-RateLimit-Limit: [quota]

X-RateLimit-Remaining: [remaining requests]

X-RateLimit-Reset: [reset timestamp]

X-RateLimit-Retry-After: [seconds until retry allowed]

**4.3 Error Responses**

When rate limits are exceeded, return HTTP 429 (Too Many Requests) with:

{

"error": "rate\_limit\_exceeded",

"message": "Rate limit exceeded. Try again in [time]",

"retry\_after": [seconds],

"limit": [current limit],

"reset\_time": "[ISO 8601 timestamp]"

}

**5. Security Controls**

**5.1 IP-Based Protection**

* Implement IP allowlists for trusted sources
* Geographic blocking for high-risk regions when appropriate
* Progressive penalties for repeat violators
* Automatic temporary IP blocking for severe abuse

**5.2 Authentication Security**

* Rate limits must not reveal whether user accounts exist
* Implement exponential backoff for failed authentication attempts
* Log all rate limit violations for security monitoring
* Integrate with fraud detection systems

**5.3 DDoS Protection**

* Implement circuit breakers for critical endpoints
* Automatic scaling triggers based on traffic patterns
* Emergency rate limit reduction capabilities
* Coordination with CDN and DDoS protection services

**6. Monitoring and Alerting**

**6.1 Metrics Collection**

* Track rate limit hit rates by user tier and endpoint
* Monitor average response times during high traffic
* Measure false positive rates for legitimate users
* Track geographic distribution of rate-limited requests

**6.2 Alert Thresholds**

* **Warning**: Rate limit hit rate >5% for any tier
* **Critical**: Rate limit hit rate >15% for any tier
* **Emergency**: System-wide rate limiting activated
* **Security**: Coordinated attack patterns detected

**6.3 Reporting Requirements**

* Daily rate limiting summary reports
* Weekly trend analysis and capacity planning
* Monthly policy effectiveness review
* Quarterly security incident correlation analysis

**7. Exemptions and Overrides**

**7.1 Approved Exemptions**

* Health check endpoints (monitoring systems)
* Emergency administrative access
* Approved batch processing operations
* Critical business partner integrations

**7.2 Override Procedures**

* Temporary limit increases require security team approval
* Emergency overrides limited to 24-hour duration
* All overrides must be logged and reviewed
* Post-incident analysis required for all emergency overrides

**8. User Experience Considerations**

**8.1 Grace Periods**

* New users receive 24-hour grace period with 2x limits
* Post-maintenance 15-minute grace period for all users
* Holiday and peak usage temporary limit increases

**8.2 User Communication**

* Proactive notification when approaching limits (90% threshold)
* Clear upgrade paths for users exceeding free tier limits
* Self-service rate limit monitoring dashboard
* API documentation with clear limit explanations

**9. Compliance and Governance**

**9.1 Policy Enforcement**

* Automated enforcement with manual override capabilities
* Regular compliance audits and penetration testing
* Integration with existing security incident response procedures
* Coordination with legal team for abuse handling

**9.2 Data Privacy**

* Rate limiting logs must comply with data retention policies
* PII must be masked in rate limiting analytics
* User consent required for detailed usage tracking
* Regular data purging of rate limiting historical data

**10. Policy Maintenance**

**10.1 Review Schedule**

* **Monthly**: Operational metrics and threshold adjustments
* **Quarterly**: Policy effectiveness and user impact assessment
* **Annually**: Complete policy review and security audit
* **As-needed**: Emergency policy updates for security incidents

**10.2 Change Management**

* All policy changes require security team approval
* Minimum 48-hour notice for non-emergency changes
* Rollback procedures for problematic policy updates
* Version control for all policy configurations

**11. Incident Response**

**11.1 Rate Limiting Failures**

* Automatic failover to backup rate limiting systems
* Emergency manual override procedures
* Incident commander notification protocols
* Service degradation communication plans

**11.2 Abuse Response**

* Automated temporary blocking for severe violations
* Escalation procedures for coordinated attacks
* Evidence preservation for legal proceedings
* Coordination with law enforcement when required

**12. Training and Awareness**

**12.1 Required Training**

* Annual security awareness training including rate limiting
* Developer training on proper API usage patterns
* Operations team training on rate limiting system management
* Customer support training on rate limiting user issues

**12.2 Documentation Requirements**

* Maintain current API documentation with rate limits
* Provide SDK examples showing proper retry logic
* Create troubleshooting guides for common issues
* Regular updates to developer best practices

**Policy Version**: 1.0  
**Effective Date**: [Insert Date]  
**Review Date**: [Insert Date + 1 Year]  
**Policy Owner**: API Security Team  
**Approved By**: [CISO/CTO Signature]