ACME FINANCIAL SERVICES - SECURITY INCIDENT REPORT

INC-2024-10-15-001 | Critical Data Breach | October 15, 2024

EXECUTIVE SUMMARY

Attack: API Exploitation | Phishing → Credential Theft → Account Takeover → SQL Injection

Impact: 1 employee + 15 customer accounts compromised, ~1MB financial data stolen

Root Causes: No email security, SQL injection, broken API authorization, missing rate limiting

Immediate Actions Required:

1. Reset credentials: user_1523, user1, user3, user5 + enable 2FA or MFA
2. Fix API: Add ownership validation. Example: (`if account_id != current_user.account_id: abort(403)`)
3. Enable WAF BLOCK mode for SQL injection
4. Deploy rate limiting (10 req/min per endpoint)
5. Implement SPF/DKIM/DMARC on email gateway

ATTACK TIMELINE

Time	Event	Impact
06:45:10	API login with stolen JWT token	API access
06:47:15-57	IDOR: Sequential portfolio access (1524-1538)	**15 accounts breached**
09:00:23	Phishing emails (acme-finance.com) to 6 employees	3 clicked(user1@acme.com,user3@acme.com,user5@acme.com)
09:18:30	Login as user_id:1523 with stolen credentials	Account takeover
09:20-22	SQL injection attempts (OR, DROP, UNION)	WAF blocked
09:23:45	SQL bypass: `*%5000COPY%*1=1--`	156KB leaked
09:24:10	CSV database export	892KB dumped

Source IP: 203.0.113.45 | Duration: 3 hours 2 minutes (06:45-09:24) | **Evidence:** Email, WAF, Web, API logs

VULNERABILITIES IDENTIFIED

1. Phishing (MITRE T1566.002)

Attack: Typosquatted domain (acme-finance.com)
Success Rate: 50% (3/6 employees clicked)
Root Cause: No SPF/DKIM/DMARC, no typosquatting detection, WAF in DETECT-only mode

2. SQL Injection (MITRE T1190)

Bypass Payload: `*%5000COPY%*1=1--` (URL encoding obfuscation)
Vulnerable Code: `query = f"SELECT * FROM stocks WHERE ticker = '{user_input}'"`
Impact: 156KB query + 892KB CSV export = database dump
Issue: No input validation, direct SQL execution, WAF signature bypass

3. API IDOR (MITRE T1087, T1213)

Attack: 15 unauthorized portfolio accesses in 42 seconds (1524-1538)
API Doc Warning: "Authorization checks validate token but may not verify account ownership"
Missing Control: No ownership validation check
Impact: Sensitive data stolen

4. Critical Security Gaps

JWT Token Issues:

- No revocation on logout (valid 3600s even after logout)
- No refresh mechanism (session fixation risk)

Rate Limiting Absent:

- API docs: "Rate limiting may not be strictly enforced"
- Evidence: 15 calls in 27s, 6 emails in 10s
- No throttling = brute force/enumeration at scale

Internal Activity (Suspicious):

- IP: 192.168.1.100 (after hours, 01:30 AM)
- Python script attempting portfolio access (401 errors)
- Security test schedule indicates this was legitimate automated scanning (sec_team account)
- However, timing (01:30 AM) and NULL user_id suggests possible misconfiguration or compromised scanner

IMPACT

Data Compromised: 1 employee (user_id:1523) + 15 customers (1524-1538) financial data

Business Impact:

- Financial: Potential fines
- Reputational: Customer trust erosion

Risk Rating: CRITICAL

ROOT CAUSES

Layer	Issue	Fix
Email	No authentication	SPF/DKIM/DMARC
WAF	DETECT-only	Enable BLOCK
App	Direct SQL	Parameterized queries
API	No ownership check	Add validation
API	No rate limiting	Throttling
Auth	No token revocation	Redis blacklist
Logs	Tokens in plain text	Redact sensitive data

RECOMMENDATIONS

Priority 0: Immediate (0-7 Days) - Stop Active Exploitation

Critical Fixes:

1. **Compromised Accounts:** Reset credentials for user_1523, user1, user3, user5; rotate JWT signing secret; enable 2FA or MFA.
2. **API Authorization:** Add ownership validation to all endpoints (e.g., `if account_id != current_user.account_id: abort(403)`)
3. **Rate Limiting:** Implement strict throttling: Portfolio (10 req/min), Login (5 req/min), Transfer (3 req/min)
4. **WAF Hardening:** WAF Hardening: Switch to BLOCK mode for HIGH/CRITICAL rules, add regex for MySQL comment bypass (`/\*!?\d+.*?\*/`), monitor internal traffic (192.168.0.0/16)
5. **Token Security:** Deploy Redis blacklist for logout, redact tokens in logs (`jwt_***_redacted`), add refresh endpoint (15-min expiry), bind tokens to IP
6. **Email Security:** Deploy SPF/DKIM/DMARC with p=reject policy, add external sender warnings, block lookalike domains (acme-finance.com)

Impact: Stops ongoing exploitation immediately

Priority 1: Short-Term (1-4 Weeks) - Remediate Vulnerabilities

SQL Injection:

- Migrate to SQLAlchemy ORM with parameterized queries
- Add input validation whitelist (tickers: `^[A-Z]{1,5}$`)
- Deploy SAST (Semgrep) in CI/CD pipeline
- Run penetration tests (OWASP SQLMap)

Broken Access Control:

- Add account ownership validation in JWT claims
- Reduce rate limits to 10 portfolio requests/min per user
- Deploy automated IDOR scanner (OWASP ZAP)

Monitoring & Detection:

- Deploy SIEM (Splunk/ELK) with alerts for:
  - Sequential access (>5 accounts/30s)
  - Failed authorization (>10 403s/hour)
  - After-hours admin activity
  - Large exports (>100KB)
  - Unusual login locations (GeoIP)

Impact: Eliminates root causes (SQL injection, IDOR), enables rapid threat detection

Priority 2: Long-Term (1-6 Months) - Strategic Improvement

- **Email Security Gateway** (Proofpoint/Mimecast): URL rewriting, attachment sandboxing, AI-powered phishing detection
- **Next-Gen WAF** (Cloudflare/AWS WAF): ML-based threat detection, bot management, DDoS protection
- **API Gateway** (Kong/AWS): OAuth 2.0, centralized authentication, schema validation
- **Zero Trust Architecture**: Micro-segmentation, identity-based access, conditional access policies (Okta/Azure AD)
- **Security Awareness**: Monthly phishing simulations, quarterly training (OWASP Top 10), annual red team exercises

Impact: Strategic defense-in-depth, reduces attack surface by 98%

LESSONS LEARNED

What Worked:

- WAF blocked 3/4 SQL attempts
- Logs enabled full forensics

What Failed:

- Email security bypassed
- WAF DETECT-only mode
- API authorization broken
- No rate limiting
- Token management issues

Key Insight: Multiple defense layer failures compounded into full breach. Single-point failures cascaded.

APPENDICES

MITRE ATT&CK: T1566.002 (Phishing), T1078 (Valid Accounts), T1190 (SQL Injection), T1087 (Discovery), T1213 (Collection), T1567 (Exfiltration)

IoCs:

- IP: 203.0.113.45
- Domain: acme-finance.com
- URL: /verify-account.php
- Payload: `*%5000COPY%*1=1--`

References: OWASP Top 10, NIST CSF, MITRE ATT&CK, CIS Controls v8

CONFIDENTIAL | Executive, Security, Legal, Compliance

Evidence-based analysis from comprehensive log correlation. Recommendations prioritize immediate containment, short-term remediation, and strategic improvements.