BANK

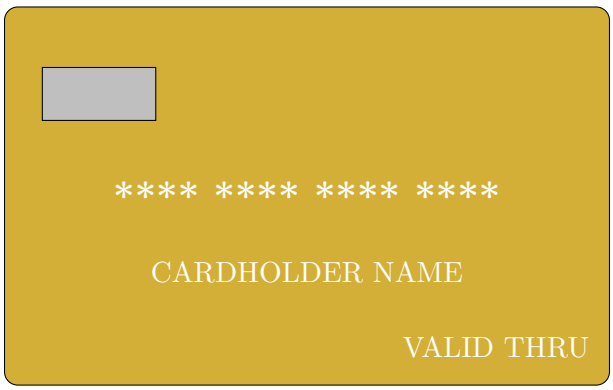
SECURITY

Data Loss Prevention

Bank Card Printing Scenario

SOC Analytics Diploma

Project 4: DLP Implementation



Prepared By:

Abdelrahman Wael

December 24, 2025

Contents

List of Figures

List of Tables

Chapter 1

Introduction to DLP in Banking

1.1 Overview

Data Loss Prevention (DLP) is a critical security technology designed to detect and prevent unauthorized transmission of sensitive data. In the banking sector, particularly in card printing operations, DLP plays a vital role in protecting customer financial information.

What is DLP?

Data Loss Prevention (DLP) is a set of tools and processes used to ensure that sensitive data is not lost, misused, or accessed by unauthorized users. DLP software detects potential data breaches and prevents them by monitoring, detecting, and blocking sensitive data while in use, in motion, and at rest.

1.2 Importance in Banking Sector

The banking industry handles vast amounts of sensitive data including:

- **Primary Account Numbers (PAN)** - Credit/Debit card numbers
- **Card Verification Values (CVV)** - Security codes
- **Personal Identification Numbers (PIN)** - Access codes
- **Customer Personal Information** - Names, addresses, SSN
- **Card Expiration Dates** - Validity information
- **Magnetic Stripe Data** - Track 1 and Track 2 data

1.3 Card Printing Environment

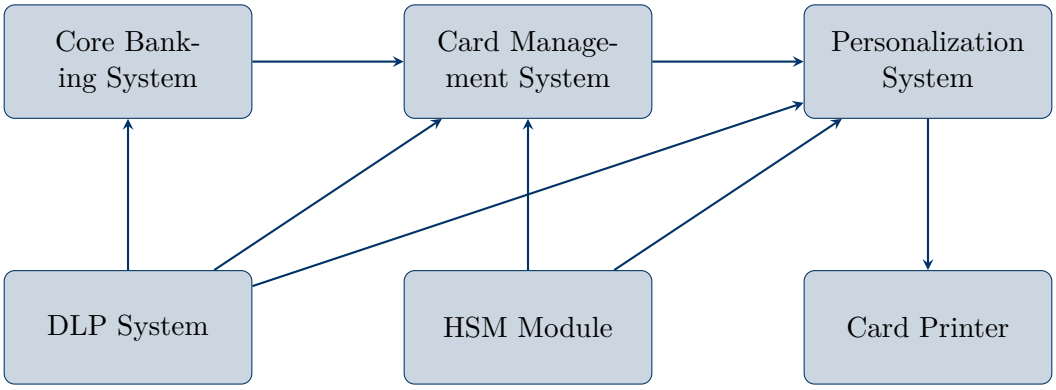


Figure 1.1: Card Printing Environment Architecture

1.4 Regulatory Compliance Requirements

Table 1.1: Key Compliance Standards for Card Printing

bankblue!20 **Standard**	**Version**	**Relevance to Card Printing**
PCI DSS	4.0	Payment Card Industry Data Security Standard
ISO 27001	2022	Information Security Management
GDPR	2018	Personal Data Protection
SOX	2002	Financial Data Integrity
CBE Guidelines	2023	Central Bank of Egypt Regulations

Chapter 2

Threat Landscape Analysis

2.1 Threat Actors

Understanding potential threat actors is essential for effective DLP implementation:

Internal Threat Actors

1. **Malicious Insiders** - Employees with authorized access
2. **Negligent Employees** - Unintentional data exposure
3. **Compromised Accounts** - Legitimate credentials misused
4. **Third-Party Contractors** - External personnel with access

External Threat Actors

1. **Cybercriminal Groups** - Organized crime targeting financial data
2. **Nation-State Actors** - Advanced persistent threats
3. **Hacktivists** - Ideologically motivated attackers
4. **Competitors** - Corporate espionage

2.2 Attack Vectors

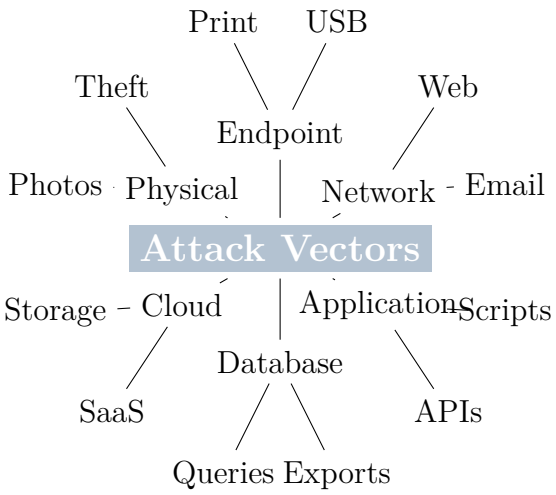


Figure 2.1: Attack Vector Mind Map

2.3 Risk Assessment Matrix

Table 2.1: Card Printing DLP Risk Assessment

bankblue!20 **Risk Scenario**	**Impact**	**Likelihood**	**Risk Level**	**Priority**
Unauthorized card data export	Critical	Medium	securityred!30High	P1
Email containing PAN data	High	High	securityred!30High	P1
USB copy of card files	Critical	Low	alertorange!30Medium	P2
Screen capture of card data	Medium	Medium	alertorange!30Medium	P2
Print of sensitive reports	Medium	Low	safegreen!30Low	P3
Cloud upload of card data	Critical	Low	alertorange!30Medium	P2

2.4 Data Classification for Card Printing

Data Classification Levels

Level 4 - Restricted: Full PAN, CVV, PIN, Track Data

Level 3 - Confidential: Masked PAN, Customer PII

Level 2 - Internal: Card batch information, printing schedules

Level 1 - Public: General card program information

Chapter 3

DLP Architecture Design

3.1 Three-Tier DLP Model

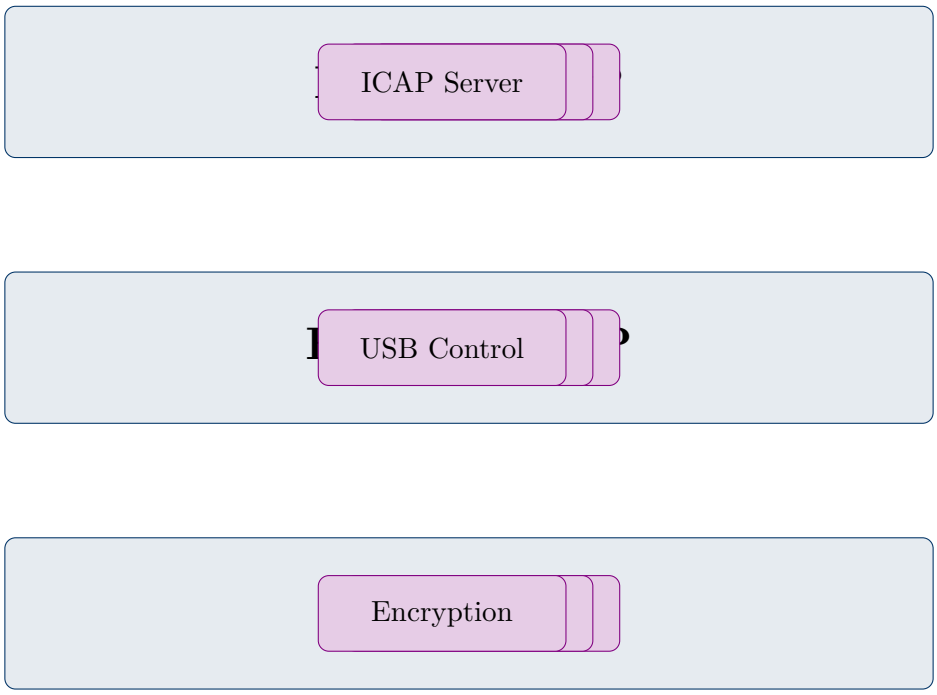


Figure 3.1: Three-Tier DLP Architecture

3.2 Network DLP Components

3.2.1 Email DLP Gateway

```
# Email DLP Policy for Card Data
policy_name: "Card_Data_Email_Prevention"
priority: 1
enabled: true

conditions:
  - type: content_match
    patterns:
```

```
      - name: "PAN_Pattern"
        regex: '\b(?:4[0-9]{12}(?:[0-9]{3})?|5[1-5][0-9]{14})\b'
        description: "Visa/Mastercard PAN Detection"

      - name: "Track_Data"
        regex: '%B[0-9]{13,19}\^[A-Z\s]{2,26}\^[0-9]{4}'
        description: "Track 1 Magnetic Stripe Data"

actions:
  - type: block
    notification: true
    quarantine: true
    log_level: critical
```

Listing 3.1: Email DLP Policy Configuration

3.2.2 Web DLP Proxy

```
# Web Upload Prevention Rules
rule web_upload_block {
    name: "Block_Card_Data_Upload"

    match_conditions:
        http_method: [POST, PUT]
        content_type: [multipart/form-data, application/json]

    content_inspection:
        enable_ocr: true
        enable_deep_inspection: true

    detection_rules:
        - credit_card_number
        - cvv_pattern
        - expiry_date_pattern

    action: BLOCK
    alert_severity: CRITICAL
    notify: [soc_team, dlp_admin, ciso]
}
```

Listing 3.2: Web DLP Proxy Rules

3.3 Endpoint DLP Components

3.3.1 Agent Architecture

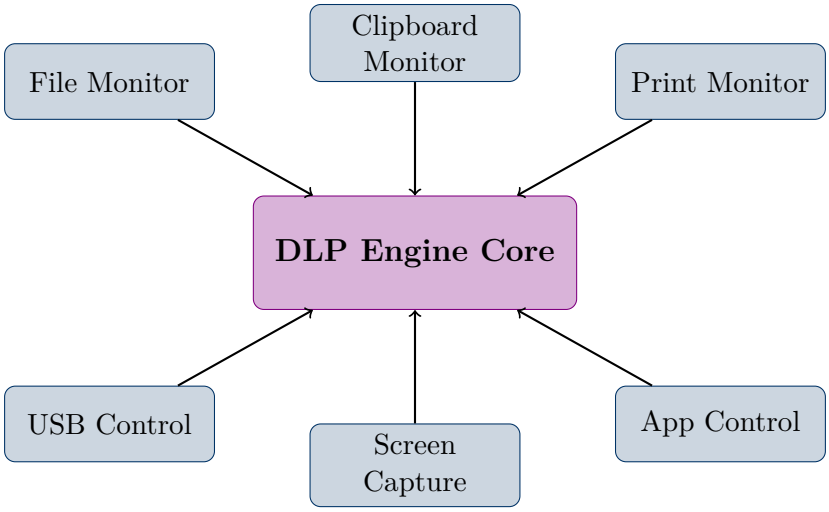


Figure 3.2: Endpoint DLP Agent Architecture

3.3.2 Print Monitoring Configuration

```
# Print Monitoring for Card Printing Workstations
print_policy:
  name: "Card_Print_Room_Policy"
  scope:
    - OU=CardPrinting,DC=bank,DC=local

  printers:
    allowed:
      - "CARD_PRINTER_01"
      - "CARD_PRINTER_02"
    blocked:
      - "*"  # Block all other printers

  content_rules:
    - name: "Block_Card_List_Print"
      pattern: "Card.*List|PAN.*Report"
      action: block_and_alert

    - name: "Block_Bulk_Card_Data"
      condition: "card_count > 10"
      action: block_and_alert

  watermark:
    enabled: true
    content: "${username} - ${datetime} - ${hostname}"
```

Listing 3.3: Print Monitor Policy

3.4 USB Device Control

Table 3.1: USB Device Control Matrix

bankblue!20 **Device Type**	**Card Room**	**IT Admin**	**General**
USB Storage	securityred!30Blocked	alertorange!30Audit	securityred!30Blocked
USB Keyboard	safegreen!30Allowed	safegreen!30Allowed	safegreen!30Allowed
USB Mouse	safegreen!30Allowed	safegreen!30Allowed	safegreen!30Allowed
Mobile Device	securityred!30Blocked	securityred!30Blocked	securityred!30Blocked
Camera	securityred!30Blocked	securityred!30Blocked	securityred!30Blocked

Chapter 4

Detection Rules and Patterns

4.1 Regular Expression Patterns

Credit Card Detection Patterns

```
# Visa Card Pattern
VISA_PATTERN = r'\b4[0-9]{12}(?:[0-9]{3})?\b'

# Mastercard Pattern
MC_PATTERN = r'\b5[1-5][0-9]{14}\b'

# American Express Pattern
AMEX_PATTERN = r'\b3[47][0-9]{13}\b'

# Generic Card Pattern with Luhn Validation
GENERIC_CARD = r'\b(?:\d{4}[-\s]?){3}\d{4}\b'

# CVV Pattern
CVV_PATTERN = r'\b[0-9]{3,4}\b'

# Expiry Date Pattern
EXPIRY_PATTERN = r'\b(0[1-9]|1[0-2])\/([0-9]{2}|[0-9]{4})\b'

# Track 1 Data Pattern
TRACK1_PATTERN = r'%B[0-9]{13,19}\^[A-Z\s
   /]{2,26}\^[0-9]{4}[0-9]*\?'

# Track 2 Data Pattern
TRACK2_PATTERN = r';[0-9]{13,19}=[0-9]{4}[0-9]*\?'
```

Listing 4.1: Card Number Detection Regex

4.2 Luhn Algorithm Validation

Listing 4.2: Luhn Algorithm Implementation

```
def luhn_checksum(card_number):
```

```
    """
    Validate credit card number using Luhn algorithm
    Used by DLP to reduce false positives
    """
    def digits_of(n):
        return [int(d) for d in str(n)]

    digits = digits_of(card_number)
    odd_digits = digits[-1::-2]
    even_digits = digits[-2::-2]

    checksum = sum(odd_digits)
    for d in even_digits:
        checksum += sum(digits_of(d * 2))

    return checksum % 10

def is_valid_card(card_number):
    """
    Returns True if card number passes Luhn check
    """
    # Remove spaces and dashes
    card_number = card_number.replace(' ', '').replace('-', '')

    if not card_number.isdigit():
        return False

    if len(card_number) < 13 or len(card_number) > 19:
        return False

    return luhn_checksum(card_number) == 0

# Example usage in DLP rule
def dlp_card_detection(text):
    """
    Detect valid credit card numbers in text
    """
    import re

    pattern = r'\b(?:\d{4}[-\s]?){3}\d{4}\b'
    matches = re.findall(pattern, text)

    valid_cards = []
    for match in matches:
        clean_number = match.replace(' ', '').replace('-', '')
        if is_valid_card(clean_number):
            valid_cards.append({
                'original': match,
                'cleaned': clean_number,
                'card_type': identify_card_type(clean_number)
            })
```

```
53
54      return valid_cards
```

4.3 Fingerprinting Rules

```
# Document Fingerprinting for Card Printing Documents
fingerprint_policy:
  name: "Card_Document_Fingerprints"

  document_sources:
    - path: "\\\\fileserver\\CardPrinting\\Templates"
      recursive: true
      file_types: [xlsx, docx, pdf, csv]

    - path: "\\\\fileserver\\CardPrinting\\Reports"
      recursive: true
      file_types: [xlsx, pdf]

  fingerprint_settings:
    algorithm: SHA256
    chunk_size: 1024
    similarity_threshold: 0.85

  indexed_documents:
    - name: "Card_Production_Template"
      path: "Card_Production_Report.xlsx"
      sensitivity: critical

    - name: "PIN_Mailer_Template"
      path: "PIN_Mailer_Format.docx"
      sensitivity: critical

    - name: "Card_Batch_Export"
      path: "Batch_Export_*.csv"
      sensitivity: critical

  actions:
    on_match:
      - action: block
      - action: alert
        recipients: [dlp_team, card_ops_manager]
      - action: log
        level: critical
```

Listing 4.3: Document Fingerprinting Configuration

4.4 Machine Learning Detection

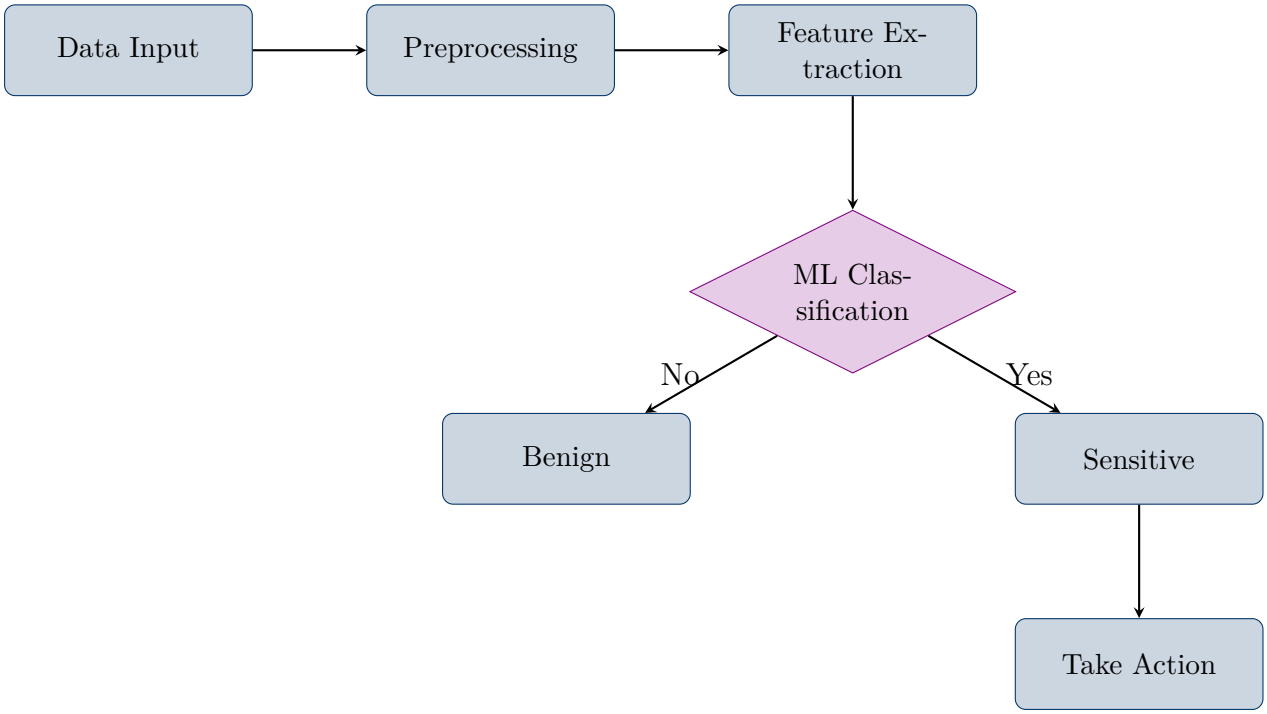


Figure 4.1: ML-Based Detection Flow

Chapter 5

SIEM Integration and Monitoring

5.1 Log Collection Architecture

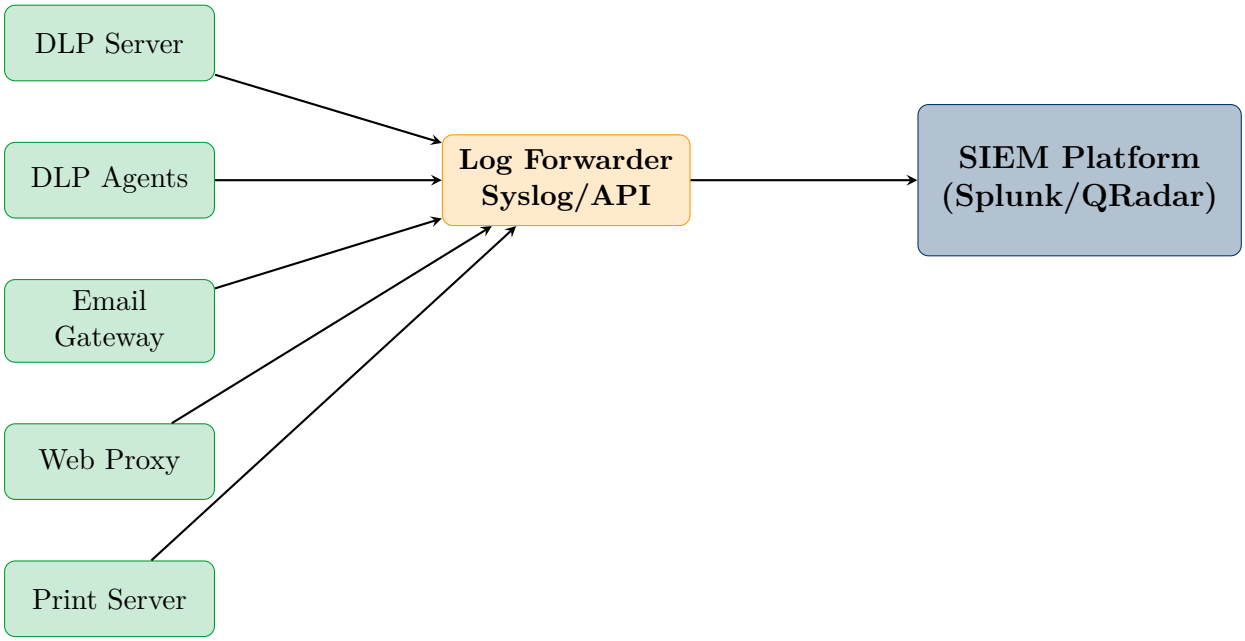


Figure 5.1: DLP Log Collection to SIEM

5.2 Splunk Queries for DLP Events

Listing 5.1: Splunk DLP Alert Queries

```
# Query 1: High Severity DLP Alerts
index=dlp sourcetype=dlp:alerts severity=high OR severity=
    critical
| stats count by user, policy_name, action, dest_ip
| where count > 5
| sort -count

# Query 2: Card Data Exfiltration Attempts
index=dlp sourcetype=dlp:events
```

```
policy_name="*card*" OR policy_name="*PAN*"
action=blocked
| timechart span=1h count by user

# Query 3: USB Block Events in Card Printing Room
index=dlp sourcetype=dlp:endpoint
event_type=usb_blocked
src_host=CARDPRINT*
| stats count by user, device_type, device_serial
| table user, device_type, device_serial, count

# Query 4: Email DLP Violations
index=dlp sourcetype=dlp:email
action=blocked
| rex field=subject "(?<card_pattern>\d{4}[- ]?\d{4}[- ]?\d{4}[- ]?\d{4})"
| stats count by sender, recipient, policy_violated

# Query 5: Print Attempts of Sensitive Documents
index=dlp sourcetype=dlp:print
document_classification=sensitive OR document_classification=restricted
| stats count by user, printer_name, document_name
| where count > 3
```

5.3 QRadar AQL Queries

```
-- AQL Query 1: DLP Events Summary
SELECT
    username,
    CATEGORYNAME(category) as category,
    COUNT(*) as event_count
FROM events
WHERE LOGSOURCENAME(logsourceid) = 'DLP_Server'
    AND CATEGORYNAME(category) ILIKE '%data%loss%'
GROUP BY username, category
HAVING COUNT(*) > 10
ORDER BY event_count DESC
LAST 24 HOURS

-- AQL Query 2: Card Data Detection Events
SELECT
    sourceip,
    username,
    UTF8(payload) as event_detail,
    DATEFORMAT(starttime, 'yyyy-MM-dd HH:mm:ss') as event_time
FROM events
WHERE LOGSOURCENAME(logsourceid) ILIKE '%DLP%'
    AND UTF8(payload) ILIKE '%credit%card%'
```

```
    OR UTF8(payload) ILIKE '%PAN%detected%'
LAST 7 DAYS

-- AQL Query 3: Endpoint DLP Blocks
SELECT
    sourceip,
    destinationip,
    username,
    PROTOCOLNAME(protocolid) as protocol,
    COUNT(*) as block_count
FROM events
WHERE devicetype = 'DLP_Endpoint'
    AND eventdirection = 'blocked'
GROUP BY sourceip, destinationip, username, protocol
LAST 24 HOURS
```

Listing 5.2: QRadar AQL for DLP Monitoring

5.4 Custom Correlation Rules

```
# Correlation Rule 1: Multiple DLP Violations by Same User
rule "DLP_Multiple_Violations_Same_User" {
    meta:
        description = "Detect multiple DLP violations by same
           user"
        severity = "high"
        category = "data_exfiltration"

    condition:
        count(dlp_violation WHERE user = $user) > 5
        within 1 hour

    action:
        create_offense(
            name = "Multiple DLP Violations - " + $user,
            severity = 8,
            assign_to = "DLP_Team"
        )
        send_email(
            to = "soc@bank.com",
            subject = "ALERT: Multiple DLP Violations"
        )
}

# Correlation Rule 2: After Hours Card Data Access
rule "DLP_AfterHours_CardData_Access" {
    meta:
        description = "Card data access outside business hours"
        severity = "critical"

```

```
    condition:
        dlp_event.policy_name contains "card"
        AND (hour(timestamp) < 7 OR hour(timestamp) > 20)
        AND dayofweek(timestamp) NOT IN [6, 7]

    action:
        create_offense(severity = 9)
        send_sms(to = "on_call_analyst")
        block_user_session()
}

# Correlation Rule 3: Data Exfiltration Pattern
rule "DLP_Exfiltration_Pattern" {
    meta:
        description = "Detect data exfiltration patterns"
        severity = "critical"

    sequence:
        A: large_file_access(user = $user)
        B: usb_mount_attempt(user = $user) within 10 min
        C: dlp_block_event(user = $user) within 5 min

    action:
        create_high_priority_incident()
        isolate_endpoint($source_host)
}
```

Listing 5.3: SIEM Correlation Rules for DLP

Chapter 6

DLP Use Cases for Card Printing

6.1 Use Case 1: Unauthorized Card Data Export

Use Case Description

Scenario: An employee attempts to export card holder data including PAN numbers to a USB drive or external storage.
Risk Level: Critical
Regulatory Impact: PCI DSS Violation, potential fine up to $500,000

6.1.1 Detection Logic

Listing 6.1: Card Data Export Detection

```
class CardDataExportDetector:
    """
    Detects unauthorized export of card data
    """

    def __init__(self):
        self.card_patterns = [
            r'\b4[0-9]{12}(?:[0-9]{3})?\b',  # Visa
            r'\b5[1-5][0-9]{14}\b',           # Mastercard
            r'\b3[47][0-9]{13}\b'             # Amex
        ]
        self.threshold = 10  # Alert if more than 10 cards

    def analyze_file(self, file_content, destination):
        """
        Analyze file for card data before transfer
        """
        detected_cards = []

        for pattern in self.card_patterns:
            matches = re.findall(pattern, file_content)
            for match in matches:
                if self.validate_luhn(match):
```

```
                    detected_cards.append(match)

        if len(detected_cards) > 0:
            return {
                'alert': True,
                'card_count': len(detected_cards),
                'severity': self.calculate_severity(len(
                    detected_cards)),
                'action': self.determine_action(destination, len(
                    detected_cards))
            }

        return {'alert': False}

    def calculate_severity(self, card_count):
        if card_count > 100:
            return 'critical'
        elif card_count > 50:
            return 'high'
        elif card_count > 10:
            return 'medium'
        return 'low'

    def determine_action(self, destination, card_count):
        if 'USB' in destination or 'removable' in destination.
            lower():
            return 'block_and_alert'
        if card_count > self.threshold:
            return 'block_and_alert'
        return 'audit_only'
```

6.1.2 Response Workflow

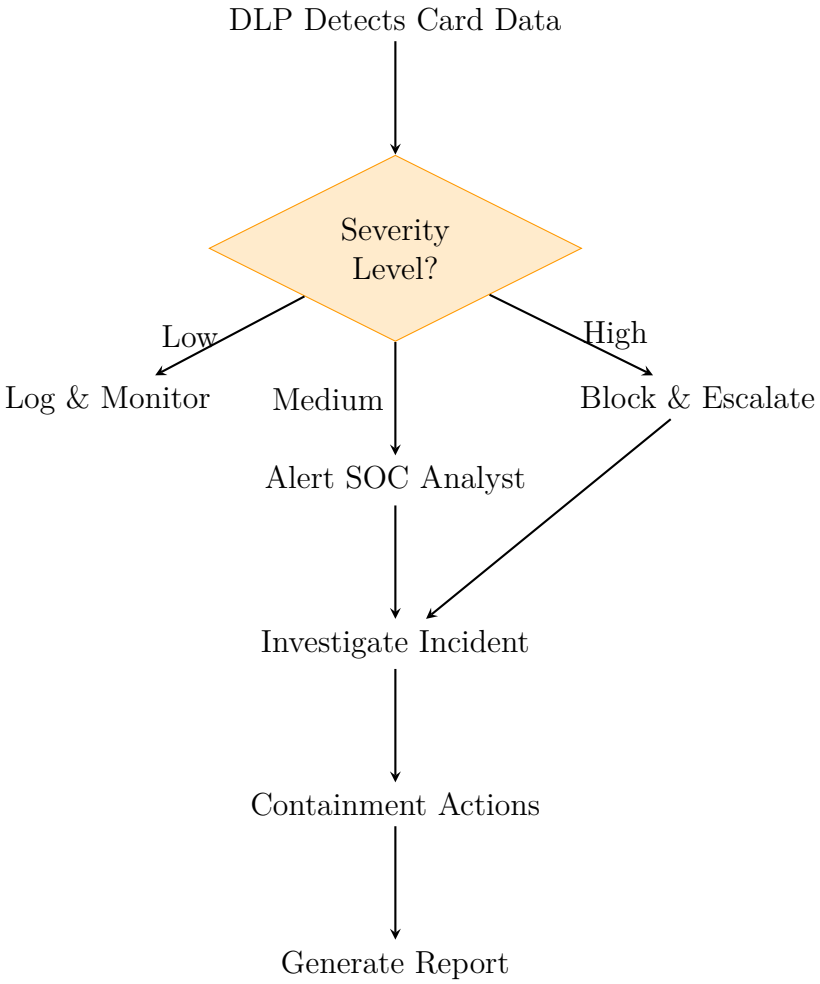


Figure 6.1: DLP Response Workflow

6.2 Use Case 2: Email Containing Card Data

Scenario Details

Scenario: Employee sends email with attachment containing PAN data to external recipient.
Detection Method: Email gateway DLP scanning
Response: Block email, quarantine, alert security team

```
# Email DLP Rule for Card Data
email_rule:
  name: "Block_Card_Data_Email"
  priority: 1
  enabled: true

  scope:
    direction: outbound
```

```
    domains:
      exclude: ["@bank.com", "@trusted-partner.com"]

  conditions:
    any:
      - content_match:
          type: regex
          patterns:
            - '\b4[0-9]{12}(?:[0-9]{3})?\b'
            - '\b5[1-5][0-9]{14}\b'
          validate_luhn: true
          min_matches: 1

      - attachment_match:
          file_types: [xlsx, csv, pdf, txt]
          content_scan: true
          fingerprint_match:
            - "Card_Production_Report"
            - "Card_Batch_Export"

  exceptions:
    - user_group: "Card_Operations_Managers"
      condition: "encrypted_attachment"
      action: "audit_only"

  actions:
    primary:
      - action: block
        message: "Email blocked: Contains card data"
    secondary:
      - action: quarantine
        retention_days: 90
      - action: alert
        recipients:
          - soc@bank.com
          - dlp-admin@bank.com
      - action: notify_sender
        template: "dlp_block_notification"
```

Listing 6.2: Email DLP Rule Configuration

6.3 Use Case 3: Unauthorized Printing

Table 6.1: Print DLP Scenarios and Responses

bankblue!20 **Scenario**	**Risk**	**Detection Method**	**Action**
Print card list to non-secure printer	Critical	Printer whitelist check	Block
Bulk print of cardholder data	High	Document content scan	Block + Alert
Print to personal printer	High	Network printer detection	Block
Screenshot of card data	Medium	Screen capture monitor	Block + Log
Print encrypted report	Low	Encryption verification	Allow + Audit

6.4 Use Case 4: Cloud Upload Prevention

```
# Cloud Upload Prevention Policy
cloud_dlp_policy:
  name: "Prevent_Card_Data_Cloud_Upload"

  monitored_services:
    - Google Drive
    - OneDrive
    - Dropbox
    - Box
    - iCloud
    - WeTransfer
    - Generic File Sharing

  detection:
    inline_inspection: true
    ssl_inspection: true

    content_rules:
      - name: "PAN_Detection"
        enabled: true
        action: block

      - name: "Card_Document_Fingerprint"
        enabled: true
        action: block

      - name: "Bulk_PII_Upload"
        threshold: 100
        action: block

```

```
  exceptions:
    approved_cloud_storage:
      - service: "Bank_Approved_Cloud"
        condition: "encrypted"
        action: allow_with_logging

  user_notification:
    enabled: true
    message: |
      Your upload has been blocked by DLP policy.
      Uploading card data to cloud services is prohibited.
      Contact IT Security if you need assistance.
```

Listing 6.3: Cloud DLP Policy

Chapter 7

Incident Response Procedures

7.1 DLP Incident Classification

Table 7.1: DLP Incident Severity Classification

bankblue!20 **Level**	**Description**	**Example**	**SLA**
securityred!30P1 - Critical	Confirmed data breach, large volume of card data	1000+ cards exfiltrated	15 min
securityred!20P2 - High	Attempted exfiltration blocked, policy violation	USB block with card data	30 min
alertorange!30P3 - Medium	Policy violation, no data loss confirmed	Email with PAN blocked	2 hours
safegreen!30P4 - Low	Minor policy violation, false positive review	Single card number in email	8 hours

7.2 Response Playbook

Algorithm 1 DLP Incident Response Algorithm

Input: DLP Alert with severity level
Output: Incident resolution
if Alert received **then**
    Acknowledge alert within SLA
    Collect initial evidence
    Determine if true positive or false positive
end if
if True Positive **then**
    Escalate based on severity
    Contain the threat (block user/endpoint if needed)
    Preserve evidence for forensics
    Notify stakeholders per matrix
    Begin detailed investigation
    Document all findings
    Implement remediation steps
    Close incident with lessons learned
else
    Document false positive
    Tune DLP rules if needed
    Close alert as false positive
end if

7.3 Escalation Matrix

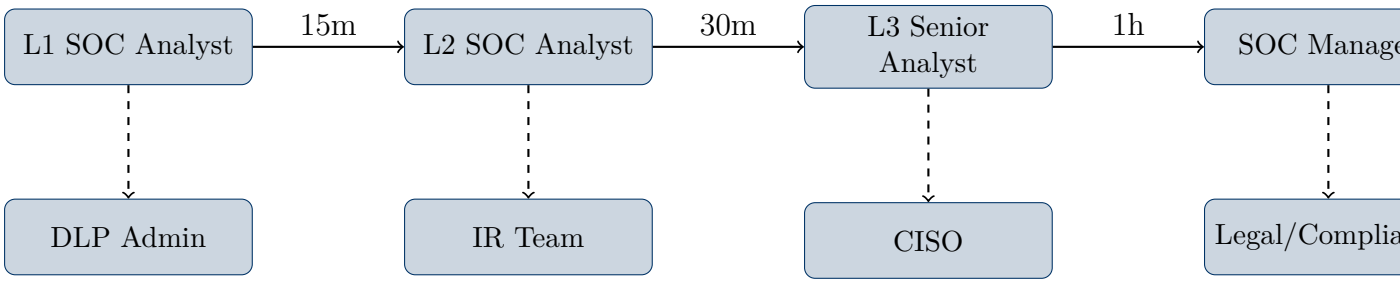


Figure 7.1: DLP Incident Escalation Matrix

7.4 Evidence Collection

Listing 7.1: Evidence Collection Script

```
#!/usr/bin/env python3
"""
DLP Incident Evidence Collection Script
SOC Analytics Diploma - Abdelrahman Wael
"""
```

```

import os
import json
import hashlib
import datetime
from pathlib import Path

class DLPEvidenceCollector:
    """
    Collect and preserve evidence for DLP incidents
    """

    def __init__(self, incident_id, analyst_name):
        self.incident_id = incident_id
        self.analyst = analyst_name
        self.timestamp = datetime.datetime.now().isoformat()
        self.evidence_path = Path(f"/evidence/dlp/{incident_id}")
        self.chain_of_custody = []

    def create_evidence_folder(self):
        """Create secure evidence folder structure"""
        self.evidence_path.mkdir(parents=True, exist_ok=True)

        folders = ['logs', 'screenshots', 'memory', 'network', '
            reports']
        for folder in folders:
            (self.evidence_path / folder).mkdir(exist_ok=True)

        self.log_custody("Evidence folder created")

    def collect_dlp_logs(self, source_server):
        """Collect DLP server logs"""
        log_types = [
            'dlp_alerts.log',
            'dlp_events.log',
            'dlp_policy_violations.log',
            'dlp_audit.log'
        ]

        for log in log_types:
            self.copy_and_hash(
                source=f"//{source_server}/logs/{log}",
                dest=self.evidence_path / 'logs' / log
            )

        self.log_custody(f"DLP logs collected from {source_server
            }")

    def collect_endpoint_evidence(self, hostname):
        """Collect evidence from endpoint"""
        # Collect running processes
```

```
        self.capture_process_list(hostname)

        # Collect network connections
        self.capture_network_connections(hostname)

        # Collect clipboard history
        self.capture_clipboard_history(hostname)

        # Collect recent file access
        self.capture_file_access_history(hostname)

        self.log_custody(f"Endpoint evidence collected from {
            hostname}")

    def calculate_hash(self, file_path):
        """Calculate SHA256 hash for integrity"""
        sha256 = hashlib.sha256()
        with open(file_path, 'rb') as f:
            for chunk in iter(lambda: f.read(4096), b''):
                sha256.update(chunk)
        return sha256.hexdigest()

    def log_custody(self, action):
        """Log chain of custody entry"""
        entry = {
            'timestamp': datetime.datetime.now().isoformat(),
            'analyst': self.analyst,
            'action': action,
            'incident_id': self.incident_id
        }
        self.chain_of_custody.append(entry)

    def generate_report(self):
        """Generate evidence collection report"""
        report = {
            'incident_id': self.incident_id,
            'collection_timestamp': self.timestamp,
            'analyst': self.analyst,
            'evidence_items': self.list_evidence(),
            'chain_of_custody': self.chain_of_custody
        }

        report_path = self.evidence_path / 'reports' / '
            collection_report.json'
        with open(report_path, 'w') as f:
            json.dump(report, f, indent=2)

        return report_path
```

Chapter 8

Reporting and Metrics

8.1 Key Performance Indicators

Table 8.1: DLP Program KPIs

bankblue!20 **KPI**	**Target**	**Current**	**Status**
False Positive Rate	$< 15\%$	12%	safegreen!30On Target
Mean Time to Detect	< 5 min	3 min	safegreen!30On Target
Mean Time to Respond	< 30 min	25 min	safegreen!30On Target
Policy Violation Rate	$< 5\%$	4.2%	safegreen!30On Target
Incidents Prevented	$> 95\%$	97%	safegreen!30On Target
User Awareness Score	$> 80\%$	75%	alertorange!30Needs Improvement

8.2 Dashboard Visualization

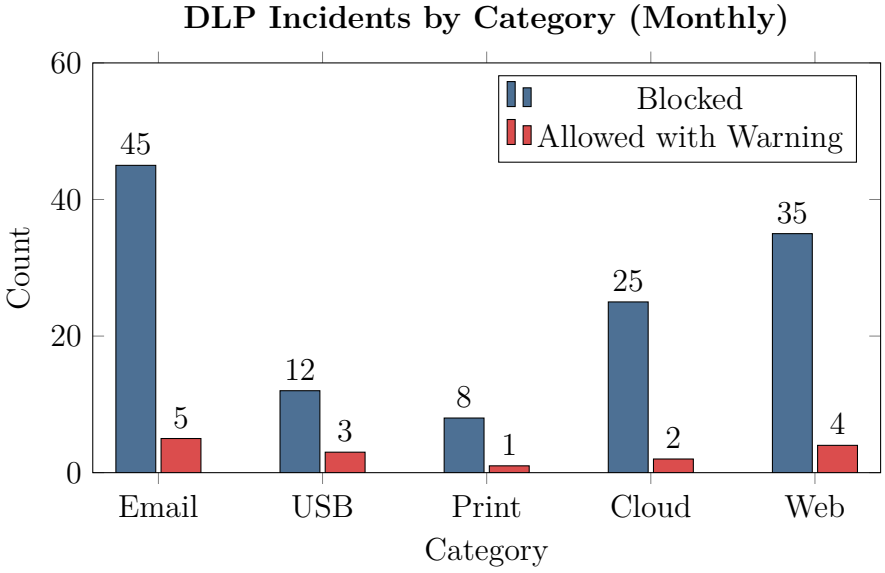


Figure 8.1: DLP Incidents by Category

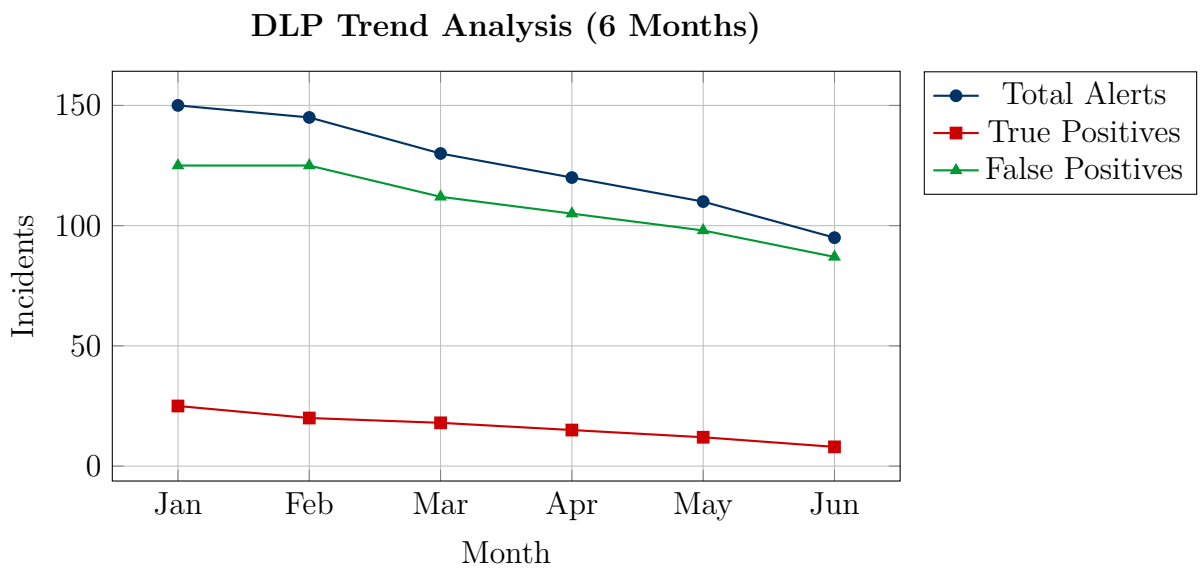


Figure 8.2: DLP Trend Analysis

8.3 Executive Summary Template

Monthly DLP Executive Summary

Reporting Period: June 2024
Key Highlights:

- Total DLP events: 2,450
- Critical incidents blocked: 15
- Successful prevention rate: 99.2%
- Zero confirmed data breaches
- Card data protection: 100% effective

Top Violation Categories:

1. Email containing PAN data (45%)
2. Web upload attempts (25%)
3. USB copy attempts (15%)
4. Cloud storage uploads (10%)
5. Print violations (5%)

Recommendations:

- Enhance user awareness training
- Review email DLP policies
- Implement additional cloud app controls

Chapter 9

Best Practices and Recommendations

9.1 Policy Design Best Practices

1. **Start with Discovery Mode**
    - Monitor before blocking
    - Understand data flow patterns
    - Identify false positive sources
2. **Implement Layered Protection**
    - Network DLP for data in motion
    - Endpoint DLP for data in use
    - Storage DLP for data at rest
3. **Use Appropriate Detection Methods**
    - Regex for structured data (PAN, SSN)
    - Fingerprinting for documents
    - ML for unstructured content
4. **Define Clear Exceptions**
    - Document all exceptions
    - Require manager approval
    - Review exceptions quarterly

9.2 Operational Recommendations

Table 9.1: DLP Operational Recommendations

bankblue!20 #	**Recommendation**	**Implementation**
1	Regular policy review	Monthly review cycle
2	False positive tuning	Weekly analysis and adjustment
3	User awareness training	Quarterly training sessions
4	Integration testing	Before each policy change
5	Incident review meetings	Weekly SOC meetings
6	Compliance audits	Quarterly internal audits

9.3 Card Printing Specific Controls

Card Printing Room Security Controls

1. **Physical Controls**
   - Biometric access control
   - CCTV monitoring
   - No mobile devices allowed
   - Secure card storage
2. **Technical Controls**
   - Dedicated network segment
   - All USB ports disabled
   - DLP agents on all workstations
   - Encrypted connections to core systems
3. **Administrative Controls**
   - Background checks for all staff
   - Dual control for sensitive operations
   - Regular access reviews
   - Separation of duties

Chapter 10

Conclusion

10.1 Summary

This document has presented a comprehensive DLP implementation strategy for bank card printing operations. Key areas covered include:

- Complete DLP architecture design
- Detection rules and patterns for card data
- SIEM integration for monitoring and alerting
- Use cases specific to card printing
- Incident response procedures
- Reporting and metrics framework
- Best practices and recommendations

10.2 Future Enhancements

1. **AI/ML Integration** - Advanced threat detection using machine learning
2. **UEBA Integration** - User behavior analytics for insider threat detection
3. **Zero Trust Architecture** - Enhanced access controls
4. **Cloud-Native DLP** - Protection for cloud workloads
5. **Automated Response** - SOAR integration for faster response

10.3 Final Notes

Project Completion

Project: DLP Scenario for Bank Card Printing
Author: Abdelrahman Wael
Program: SOC Analytics Diploma
Status: Completed
Date: December 24, 2025

Appendix A

DLP Policy Templates

A.1 Complete Email DLP Policy

```yaml
# Complete Email DLP Policy for Card Operations
policy:
  metadata:
    name: "Card_Operations_Email_DLP"
    version: "2.0"
    author: "Abdelrahman Wael"
    last_updated: "2024-01-15"

  scope:
    apply_to:
      - OU=CardOperations
      - OU=CardPrinting
    direction: [inbound, outbound]

  rules:
    - name: "Block_Outbound_PAN"
      priority: 1
      conditions:
        direction: outbound
        content_match:
          patterns:
            - type: credit_card
              validate: luhn
              count: ">= 1"
      actions:
        - block
        - quarantine
        - notify: [sender, manager, soc]
        - log: critical

    - name: "Block_Card_List_Attachment"
      priority: 2
      conditions:
        attachment:
          fingerprint_match: "Card_List_*"
```

```
        recipient: external
      actions:
        - block
        - alert: critical

  - name: "Audit_Internal_Card_Email"
    priority: 3
    conditions:
      content_match:
        patterns:
          - type: credit_card
      recipient: internal
    actions:
      - allow
      - log: info
      - audit_trail: true
```

Listing A.1: Complete Email DLP Policy Template

Appendix B

SIEM Integration Scripts

B.1 Splunk App Configuration

Listing B.1: Splunk DLP App Configuration

```
# inputs.conf
[monitor:///var/log/dlp/*.log]
sourcetype = dlp:events
index = dlp
disabled = false

# props.conf
[dlp:events]
TIME_PREFIX = timestamp=
TIME_FORMAT = %Y-%m-%dT%H:%M:%S
SHOULD_LINEMERGE = false
LINE_BREAKER = ([\r\n]+)
TRUNCATE = 0

# transforms.conf
[dlp_extract_fields]
REGEX = user=([^\s]+)\s+action=([^\s]+)\s+policy=([^\s]+)
FORMAT = user::$1 action::$2 policy::$3

# savedsearches.conf
[DLP Critical Alerts]
search = index=dlp severity=critical | stats count by user,
   policy
cron_schedule = */5 * * * *
alert.severity = 5
action.email = 1
action.email.to = soc@bank.com
```

Appendix C

Glossary

DLP Data Loss Prevention

PAN Primary Account Number

CVV Card Verification Value

HSM Hardware Security Module

PCI DSS Payment Card Industry Data Security Standard

SIEM Security Information and Event Management

SOC Security Operations Center

UEBA User and Entity Behavior Analytics

SOAR Security Orchestration, Automation and Response