BÁO CÁO ĐÁNH GIÁ BẢO MẬT

BÁO CÁO ĐÁNH GIÁ MÃ NGUỒN
ỨNG DỤNG E-COM

KHÁCH HÀNG: XX

MỤC LỤC

1. THÔNG TIN DỰ ÁN

1.1. PHIÊN BẢN TÀI LIỆU

STT	NGÀY CẬP NHẬT	PHIÊN BẢN	LOẠI	NGƯỜI CẬP NHẬT
1	20/08/20xx	V.1.0	Draft	xx
2	23/08/20xx	V.1.1	Final	xx

1.2. THỜI GIAN TRIỂN KHAI

- Thời gian đánh giá mã nguồn: 08/07/20xx – 14/07/20xx

1.3. NHÂN SỰ TRIỂN KHAI

STT	NHÂN SỰ	VAI TRÒ
1	xx	Project Manager
2	xx	Pentester
3	xx	Pentester
4	xx	Pentester

1.4. PHẠM VI THỰC HIỆN

- Đánh giá mã nguồn ứng dụng E-Com, bao gồm các thành phần Web và Mobile app (Android và iOS).
- Xác định các lỗ hổng, điểm yếu trong mã nguồn ứng dụng nhằm ngăn chặn các khai thác có thể xảy ra.

1.5. ĐỐI TƯỢNG ĐÁNH GIÁ

- Ứng dụng Web
- Ứng dụng Mobile Android: phiên bản 1.x
- Ứng dụng Mobile iOS: phiên bản 1.x

1.6. TIÊU CHUẨN THỰC HIỆN

Phương pháp đánh giá dựa trên OWASP Secure Coding Best Practices nhằm phát hiện ra các vấn đề có thể dẫn đến các lỗ hổng phổ biến trên ứng dụng Web và Mobile.

Danh sách Top 10 lỗ hổng phổ biến đối với ứng dụng Web:

- A1 Injection
- A2 Broken Authentication
- A3 Sensitive Data Exposure
- A4 XML External Entities (XXE)
- A5 Broken Access Control
- A6 Security Misconfiguration
- A7 Cross-Site Scripting XSS
- A8 Insecure Deserialization

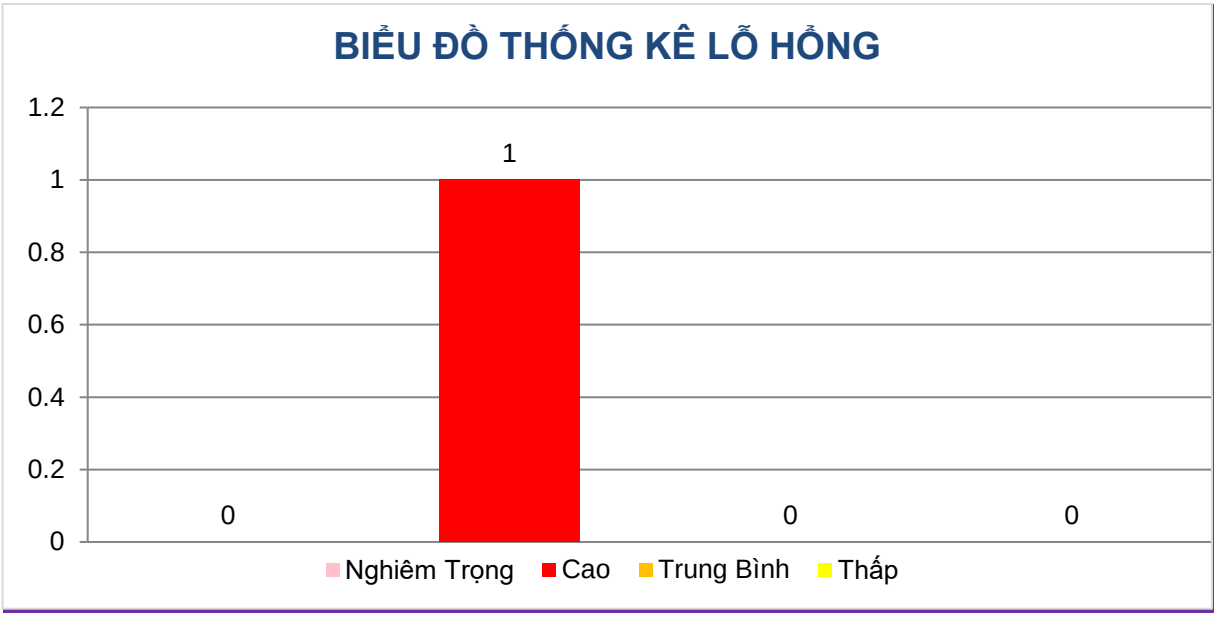
- A9 Using Components with Known Vulnerabilities
- A10 Insufficient Logging & Monitoring

Danh sách Top 10 lỗ hổng phổ biến đối với ứng dụng Mobile:

- M1: Improper Platform Usage
- M2: Insecure Data Storage
- M3: Insecure Communication
- M4: Insecure Authentication
- M5: Insufficient Cryptography
- M6: Insecure Authorization
- M7: Client Code Quality
- M8: Code Tampering
- M9: Reverse Engineering
- M10: Extraneous Functionality

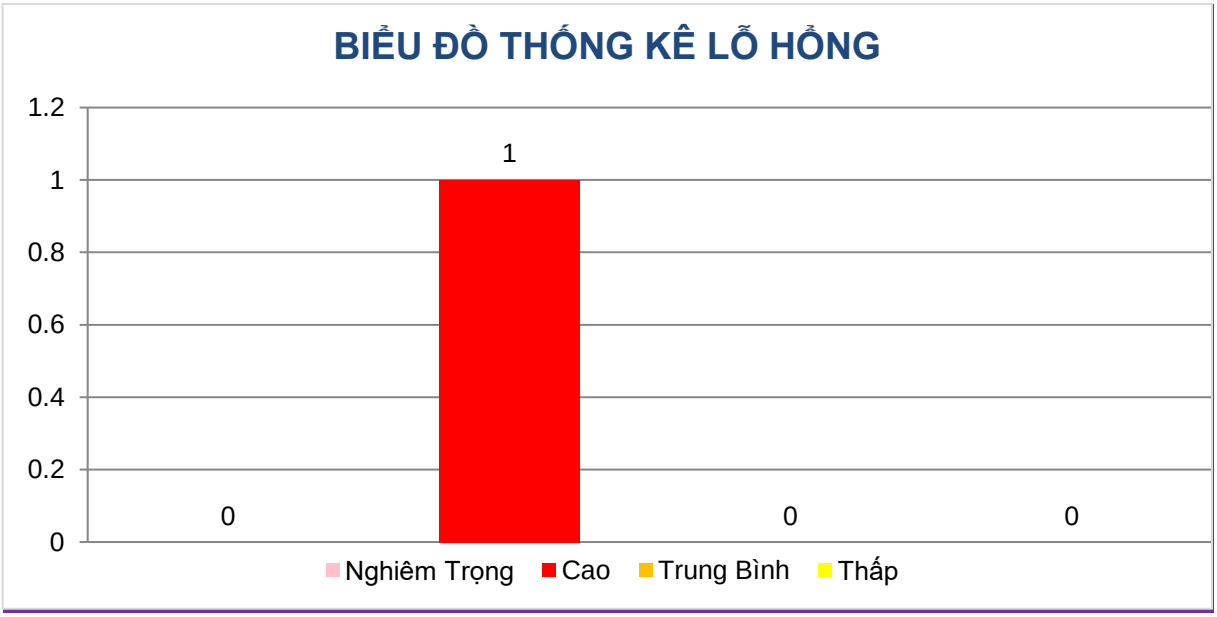
2. BÁO CÁO TỔNG QUÁT

2.1. DANH SÁCH CÁC LỖ HỔNG TRÊN MÃ NGUỒN ỨNG DỤNG WEB



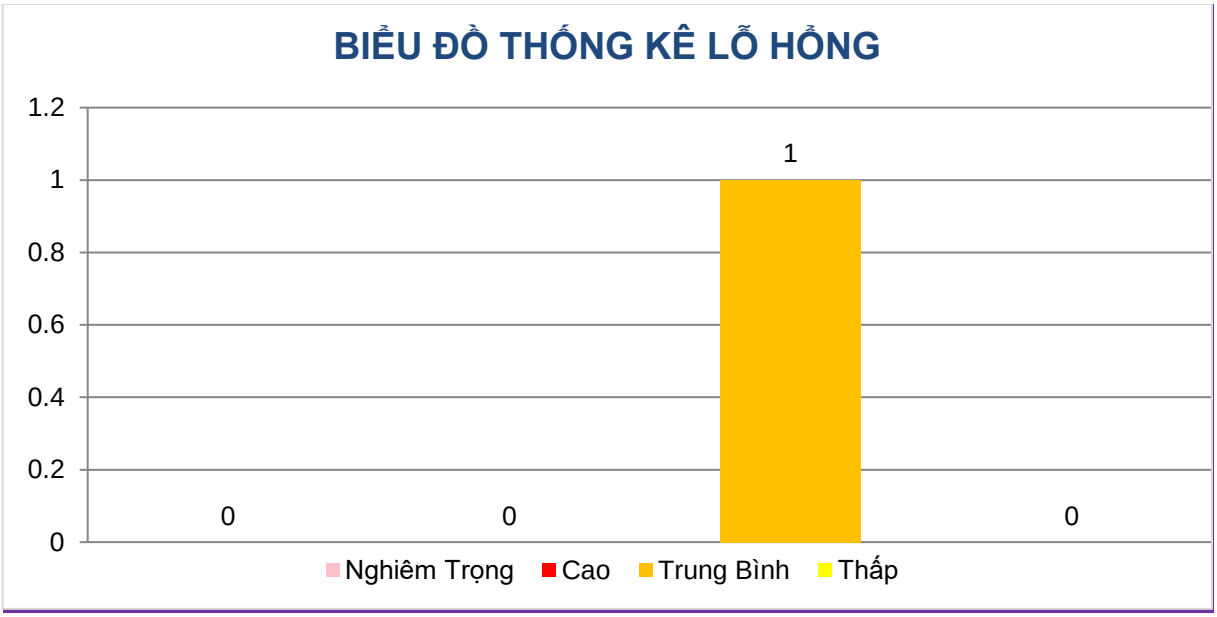
STT	MỨC ĐỘ	LỖ HỔNG
1	CAO	Lack of Centralized Malicious Input Validation

2.2. DANH SÁCH CÁC LỖ HỔNG TRÊN MÃ NGUỒN ỨNG DỤNG ANDROID



STT	MỨC ĐỘ	LỖ HỔNG
1	CAO	Insecure Logging

2.3. DANH SÁCH CÁC LỖ HỔNG TRÊN MÃ NGUỒN ỨNG DỤNG IOS



STT	MỨC ĐỘ	LỖ HỔNG
1	TRUNG BÌNH	App Transport Security Disabled

2.4. KHUYẾN NGHỊ

Trong quá trình thực hiện đánh giá/kiểm thử xâm nhập ứng dụng. Chúng tôi có một số tổng hợp/nhận xét:

- Ứng dụng web cần áp dụng cơ chế sàng lọc dữ liệu chung và đảm bảo mọi dữ liệu đầu vào từ phía người dùng đều được xử lý qua cơ chế này.
- Đối với ứng dụng Android cần review lại các nội dung log được thực hiện qua hàm console.log(). Hạn chế sử dụng hàm ghi log này để tránh lộ thông tin qua nhạy quả qua Logcat của thiết bị.
- Đối với ứng dụng iOS cần sử dụng cơ chế mã hóa kênh cho ứng dụng khi giao tiếp với máy chủ để đảm bảo các kết nối được an toàn.

3. BÁO CÁO CHI TIẾT CHO MÃ NGUỒN ỨNG DỤNG WEB

3.1. LACK OF CENTRALIZED MALICIOUS INPUT VALIDATION

<table>
<tr><th colspan="4">THÔNG TIN LỖ HỔNG</th></tr>
<tr><td>MÔ TẢ</td><td colspan="3">Ứng dụng hiện chưa có cơ chế sàng lọc, xử lý các nội dung độc trong tham số từ HTTP Request người dùng gửi lên.

Các tham số được lấy thông qua hàm getParam() hay getPostValue() được truyền thẳng vào các hàm xử lý sau đó mà không qua kiểm tra. Việc này có thể dẫn đến các lỗ hổng liên quan đến Input Validation như XSS, SQL Injection, Command Injection, …

Việc khai thác các lỗ hổng này được thực hiện bằng các chèn các ký tự đặc biệt trong tham số đầu vào nhằm phá vỡ các cấu trúc xử lý như các câu truy vấn SQL, nội dung HTML hiển thị phía người dùng, …</td></tr>
<tr><td>MỨC ĐỘ</td><td colspan="3">CAO</td></tr>
<tr><td>ẢNH HƯỞNG</td><td>CAO</td><td>KHẢ NĂNG</td><td>TRUNG BÌNH</td></tr>
<tr><td>KHUYẾN NGHỊ</td><td colspan="3">Ứng dụng cần có hàm kiểm tra, sàng lọc dữ liệu cho các tham số đầu vào được gửi lên từ người dùng và đảm bảo mọi form dữ liệu đều được xử lý. Thực hiện sàng lọc các ký tự đặc biệt như ‘ “ \ / > <

Nội dung mẫu hàm xử lý dữ liệu:

public function sanitizeParams()

{
foreach (Mage::app()->getRequest()->getPost() as $key => $value) {
$_POST[$key] = filter_var($value, FILTER_SANITIZE_SPECIAL_CHARS);
}
}

Có thể sử dụng event controller_action_predispatch trong tập tin config.xml:

<?xml version="1.0"?>
<config>
<frontend>
[...]
<events>
<controller_action_predispatch>
<observers>
<something_meaningful_and_unique>
<class>yourmodule/observer</class>
<method>sanitizeParams</method>
</something_meaningful_and_unique></td></tr>
</table>

	</observers> *</controller_action_predispatch>* *</events>* *[...]* *</frontend>* *</config>*
CHI TIẾT LỖ HỔNG	
CHỨC NĂNG	N/A
TẬP TIN ẢNH HƯỞNG	Nhiều tập tin
THAM SỐ	N/A
ĐIỀU KIỆN	N/A

Một số mã nguồn không an toàn:

Mã nguồn app/code/Smart/SurveyForm/Controller/Adminhtml/Form/Save.php: dữ liệu gửi lên từ POST request được đưa vào lưu trong model mà không được sàng lọc

```
$resultRedirect = $this->resultRedirectFactory->create();
    $data = $this->getRequest()->getParams()['general'];
    if ($data) {
        $model = $this->formModelFactory->create();
        $date = $data['start_date'];
        if (!$date) {
            $date = $this->timezone->date()->format('d-m-Y');
        }
        $id = isset($data['id']) ? $data['id'] : null;
        if (!$id) {
            $model->setActive($data['active']);
            $model->setTitle($data['title']);
            $model->setThumbnailImage($data["thumbnail_image"][0]['name']);
            $model->setDescription($data['description']);
            $model->setStartDate($date);
            $model->setEndDate($data['end_date']);
            $model->setThPoint($data['th_point']);
            try {
                $model->save();
                $this->messageManager->addSuccess(__('Insert Record Successfully!'));
                $this->dataPersistor->clear('smart_survey_form_list');
```

```
            } catch (Exception $e) {
                $this->messageManager->addError($e->getMessage());
            }
        }
```

Mã nguồn app/code/Smart/Customer/Controller/Account/EditPost.php: dữ liệu chỉnh sửa thông tin người dùng được lấy thẳng từ request mà không được sàng lọc

```
if ($validFormKey && $this->getRequest()->isPost()) {
        $currentCustomerDataObject = $this->getCustomerDataObject($this->session->getCustomerId());
        $customerCandidateDataObject = $this->populateNewCustomerDataObject(
            $this->_request,
            $currentCustomerDataObject
        );
```

Mã nguồn app/code/Smart/SalesRule/Controller/Adminhtml/Promo/Quote/Customer.php: dữ liệu import từ file CSV được lưu thẳng vào CSDL mà không qua sàng lọc

```
if (isset($_FILES['customers_file']['name'])) {
        try {
            $uploader = $this->uploaderFactory->create(['fileId' => 'customers_file']);
            $workingDir = $this->varDirectory->getAbsolutePath('importexport/');
            $insertResult = [];
            $insertErrors = [];
            $result = $uploader->save($workingDir);
            $csvData = $this->csv->getData($result['path'] . '/' . $result['name']);
            $numberOfRows = 0;
            $numberOfExisteds = 0;
            $numberOfSuccess = 0;
            $numberOfFailed = 0;
            foreach ($csvData as $index => $row) {
                if ($index > 0 && count($row)) {
                    if ($row[0][0] !== '0') {
                        $row[0] = '0' . $row[0];
                    }
                    $numberOfRows++;
                    if (!$this->isExistedCustomerCoupon($ruleId, $row)) {
                        $insertResult[] = $insertCustomerCouponResult = $this->insertCustomerCoupon($ruleId, $row);
                        if ($insertCustomerCouponResult['success']) {
                            $numberOfSuccess++;
```

```
                } else {
                    $numberOfFailed++;
                    $insertErrors[] = [
                        'row'       => $row,
                        'message'   => $insertCustomerCouponResult['message'],
                        'code'      => $insertCustomerCouponResult['code'],
                    ];
                }
            } else {
                $numberOfExisteds++;
            }
        }
    }
```

4. BÁO CÁO CHI TIẾT CHO MÃ NGUỒN ỨNG DỤNG ANDROID

4.1. INSECURE LOGGING

THÔNG TIN LỖ HỔNG			
MÔ TẢ	Theo nội dung mã nguồn trong tập tin index.android.bundle, ứng dụng thực hiện ghi log lại kết quả thực hiện của nhiều chức năng thông qua hàm *console.log*. Hàm *console.log* sẽ thực hiện ghi các nội dung này trong Logcat của thiết bị, có thể được truy xuất và đọc bởi bất kỳ ứng dụng nào khác. Việc sử dụng hàm *console.log* sẽ có thể dẫn đến việc lộ các thông tin nhạy cảm không mong muốn.		
MỨC ĐỘ	**CAO**		
ẢNH HƯỞNG	**CAO**	**KHẢ NĂNG**	**TRUNG BÌNH**
KHUYẾN NGHỊ	Hạn chế việc ghi log ở ứng dụng client phía người dùng để tránh lộ các thông tin không mong muốn. Chọn lọc các thông tin cần thiết để ghi log trong trường hợp cần debug ứng dụng và thực hiện ghi log ra file thay vì Logcat.		
CHI TIẾT LỖ HỔNG			
CHỨC NĂNG	N/A		
TẬP TIN ẢNH HƯỞNG	index.android.bundle		
THAM SỐ	N/A		
ĐIỀU KIỆN	N/A		

Một số mã nguồn không an toàn: một số nội dung được ghi log lại có thể để lộ thông tin nhạy cảm

index.android.bundle – dòng 50370:

u = l.sent, console.log('get_customer_token:', u), u && (s = "Bearer " + u), t.headers.Authorization = s || c.default.defaults.headers.Authorization;

index.android.bundle – dòng 50413:

if (401 === n && f._resource && f._resource.Type === _.default.Admin) return console.log('acess_token_logggerr:', f._accessToken), f._accessToken

index.android.bundle – dòng 224629:

var o = n.getParam('itemId');

console.log('voucherId:::', o), t.props.getRedeemRewardDetail(o)

index.android.bundle – dòng 219776:

if (console.log('LOGIN'), l._username && l._password) l.props.loginUser(l._username, l._password);

5. BÁO CÁO CHI TIẾT CHO MÃ NGUỒN ỨNG DỤNG iOS

5.1. APP TRANSPORT SECURITY DISABLED

THÔNG TIN LỖ HỔNG			
MÔ TẢ	Ứng dụng vô hiệu hóa tính năng ATS cho tất cả các kết nối, điều này có thể dẫn đến các kết nối không an toàn từ ứng dụng đến các máy chủ (đặc biệt là các máy chủ của bên thứ 3).		
MỨC ĐỘ	**TRUNG BÌNH**		
ẢNH HƯỞNG	**TRUNG BÌNH**	**KHẢ NĂNG**	**TRUNG BÌNH**
KHUYẾN NGHỊ	ATS nên được sử dụng trên bất kỳ kết nối ứng dụng đến máy chủ nào, đặc biệt là các máy chủ ứng dụng của bên thứ ba. App Transport Security (ATS) là một tập hợp các kiểm tra bảo mật mà hệ điều hành thực thi khi tạo kết nối với NSURLConnection, NSURLSession và CFURL với các tên máy chủ công cộng. ATS được bật theo mặc định cho các ứng dụng được xây dựng trên iOS SDK 9 trở lên.		
CHI TIẾT LỖ HỔNG			
CHỨC NĂNG	N/A		
TẬP TIN ẢNH HƯỞNG	Info.plist		
THAM SỐ	NSAppTransportSecurity		
ĐIỀU KIỆN	N/A		

Mã nguồn không an toàn:

```
<key>NSAppTransportSecurity</key>
  <dict>
    <key>NSAllowsArbitraryLoads</key>
    <true/>
    <key>NSExceptionDomains</key>
    <dict>
      <key>localhost</key>
      <dict>
        <key>NSExceptionAllowsInsecureHTTPLoads</key>
        <true/>
      </dict>
    </dict>
  </dict>
```

6. PHẦN MỞ RỘNG A: THÔNG TIN ĐÁNH GIÁ

6.1. DANH SÁCH CÔNG CỤ

No.	CATEGORY	TOOLS
1	Open-Source tools	Nmap, Firefox addons, Grabber, Zed, Sqlmap, WebScarab, Wireshark and other tool in Kali Linux (advanced penetration testing platform)
		Framework scanner: Microsoft ASP.NET
		Software: notepad++, sublime, python, RDP, putty
2	Commercial tools	Burpsuite – Proxy for application scanning, analyzing and modifying requests, responses
		Nessus - Vulnerability scanner for servers, databases, applications and network devices
3	Self-developed tools	Scanner & Tool - Analyzing application structure - Enumerate application components (functions, url, parameters, …) - Password dictionary and signs of critical vulnerabilities, for example XSS, SQL based on errors - Sensitive components detection (GHDB, Module, keyword, parameters, contents, email, notes, backup data, …) - Customized exploit for critical vulnerabilities: SQL Injection, XSS, Heartbleed, XPath, XXE, File Upload, File Inclusion, OS Command Injection, … and others vulnerabilities in and out of OWASP Top 10 - Customization support for web service

7. PHẦN MỞ RỘNG B: PHÂN LOẠI RỦI RO

Mỗi rủi ro tìm thấy trong quá trình kiểm thử được tham chiếu việc đánh giá theo OWASP Risk Rating Methodology.

Phương pháp tiếp cận theo OWASP được đề cập trong tài liệu được dùng làm chuẩn tham chiếu/phương pháp tiếp cận và tuỳ biến theo từng ứng dụng để đáp ứng/tinh chỉnh cho phù hợp các test-cases/kịch bản.

Mô hình đánh giá mức độ rủi ro:

Rủi ro = Khả Năng * Ảnh hưởng

MỨC ĐỘ NGHIÊM TRỌNG				
MỨC ĐỘ ẢNH HƯỞNG	**CAO**	TRUNG BÌNH	CAO	NGHIÊM TRỌNG
	TRUNG BÌNH	THẤP	TRUNG BÌNH	CAO
	THẤP	NOTE	THẤP	TRUNG BÌNH
		THẤP	**TRUNG BÌNH**	**CAO**
	KHẢ NĂNG XẢY RA			

Tài liệu tham khảo:

https://www.owasp.org/index.php/OWASP_Risk_Rating_Methodology