Report for:

Web Application, Internal Infrastructure and Mobile Application Security Assessment Sample Report

ClientName

March 23, 2021

Version: 3.0

Prepared by: NCC Consultant

Email: n.consultant@nccgroup.com

Telephone: +44 (0)161 209 5200

NCC Group PLC - Security Testing Audit and Compliance

XYZ Building,
2 Hardman Boulevard,
Spinningfields,
Manchester,
M3 3AQ
https://www.nccgroup.com

1 Executive Summary

This report presents the findings of the Web Application, Internal Infrastructure and Mobile Application Security Assessment conducted on behalf of Client Company Ltd (ClientName). The assessment was conducted between 15/03/2021 and 19/03/2021 and was authorised by ClientName.

The system being assessed was a website together with its associated mobile application and supporting infrastructure. These platforms allowed Client to provide lifestyle advice to their customer base. The iOS application collected data on the customer's physical activity during the day. It also allowed the customer to enter details of the food they consumed by scanning bar codes and the weight eaten. Data could also be uploaded to Client's website for analysis by experts employed by Client.

1.1 Overview

The assessment determined that the security posture of all three components which were examined would benefit from further work to improve the security which was afforded. A number of medium risk issues were identified during the assessment of the mobile application. The issues which were identified could threaten the confidentiality of application data held on a device. Only two issues were identified during the assessment of the internal infrastructure; however, one of these was assessed to pose a high risk. The two medium risk issues identified in the web application could be exploited by an attacker in order to capture the credentials of other users or to attack the server.

The following table breaks down the issues which were identified by component and severity of risk (issues which are reported for information only are not included in the totals):

Component	Critical	High	Medium	Low	Total
Phase 1 –Web Application Assessment	0	0	2	3	**5**
Phase 2 –Internal Infrastructure Assessment	0	1	0	1	**2**
Phase 3 –Mobile Application and Web Service Assessment	0	0	4	3	**7**
Total	**0**	**1**	**6**	**7**	**14**

1.2 Assessment Summary

The most significant issue which was identified was a high risk issue found during the internal infrastructure assessment. One item of software had reached the end of its life and was no longer supported by the manufacturer. As such it contained publicly documented security vulnerabilities that would never be addressed by software updates. It is worth noting that all other software was found to be up-to-date and that no security patches were still to be applied. As such, it is possible that this version was in use to provide compatibility with a legacy system. Nevertheless, the use of unsupported software still exposes Client to long term risk.

The assessment of the mobile application identified five medium risk issues. It was found that the application lacked mechanisms to detect whether it was being run on a jailbroken device. This increases the risk that a malicious user could override operating system restrictions. This could allow them to access data which would otherwise be restricted and also to download software from unofficial sources. In addition, it was found that the mobile application used hard-coded credentials to communicate with the web service. An attacker who had jailbroken their device could therefore potentially bypass the normal authentication mechanisms and gain access to server-side systems. The two remaining medium risk issues were associated with the configuration of the encryption used by the web service. One of the encryption protocols is now regarded as cryptographically insecure and has been superseded by a more secure version. The other issue could allow an attacker to launch a denial of service attack.

The assessment of the web application identified two medium risk issues. The application was vulnerable to reflected cross-site scripting. Attacks of this nature can be used to target other legitimate users of an application and are typically used to recover sensitive information such as login credentials or other personal information. A weakness in the file upload mechanism was also assessed to pose a medium risk. The types of file which could be uploaded were not suitably restricted and, as a consequence, it was possible to upload a file containing malicious code. It is acknowledged that no practical means of exploiting this activity was identified during the time available for this security assessment. Nevertheless, a determined attacker with more time available might be able to identify a viable technique. This would then allow them to execute code on the server from a remote location.

More detailed information on each of the issues which were identified is included in the Technical Details section of this report.

1.3 Strategic Recommendations

It is recommended that the reason for the use of the outdated software identified in the infrastructure assessment should be investigated. Steps should be taken to prepare for the decommissioning and replacement of the unsupported software and any associated or legacy systems, as relevant. It is acknowledged that projects of this nature can require considerable resource; nevertheless, this is the only effective way to mitigate the risk these systems expose given that the operating system is no longer supported by the vendor. Any gaps in the otherwise robust patching process should also be addressed.

It is recommended that the credential handling and authentication methods used by the application should be reviewed. In particular, the use of hard-coded credentials should be replaced with a more secure mechanism. It is recognised that this will require some redevelopment of the application and so this may also require significant resource.

Several instances of cross-site scripting were identified in the web application. It is seldom possible to identify all the instances of this issue within a black box assessment of this type and so it is recommended that a code review of the application should be performed. It is important that any remedial activity which is taken should be applied to the entire application rather than just those instances documented here.

It is acknowledged that no practical method of exploiting the file upload vulnerability was identified during the assessment. Nevertheless, given the potential impact if a practical attack were identified, it is recommended that this issue should be addressed as part of a robust, defence in depth approach to security.

2 Table of Contents

3 Technical Summary

NCC Group was contracted by Client to conduct a security assessment of the systems within scope in order to identify security issues that could negatively affect Client's business or reputation if they led to the compromise or abuse of systems.

3.1 Scope

The security assessment was carried out in the live environment and included:

- Web application assessment
- Internal infrastructure assessment
- Mobile application assessment

The IP address and URL within the scope of this test are listed below:

- **Phase 1 Assessment: Web Application Assessment**

https://www.nccgroup-client.co.uk (192.1.1.111)

- **Phase 2 Assessment: Internal Infrastructure Testing**

internaladdress.nccgroup-client.co.uk (10.1.1.123)

- **Phase 3 Assessment: Mobile Application Assessment and back-end Web Service Assessment**

Sample MobileApp –version 1.023 publicly released to the store.

Web Service located on: mobilesample.nccgroup-client.co.uk (192.1.1.121)

3.2 Caveats

During the initial day of testing, test credentials were only available for a standard user for the web application. Credentials for a user with administrator's privileges were provided on the second day of testing. This did not significantly affect the coverage that was achieved.

Due to the nature of the environment, checks that would have a high probability of causing disruption to the named hosts were excluded. Denial of service attempts were excluded for the same reason.

3.3 Post Assessment Cleanup

Any test accounts which were created for the purpose of this assessment should be disabled or removed, as appropriate, together with any associated content.

Revert any WAF/IDS/IPS/firewall changes which were made for the purposes of the assessment.

3.4 Risk Ratings

The table below gives a key to the icons and symbols used throughout this report to provide a clear and concise risk scoring system. It should be stressed that quantifying the overall business risk posed by any of the issues found in any test is outside our remit. This means that some risks may be reported as high from a technical perspective but may, as a result of other controls unknown to us, be considered acceptable.

Symbol	Risk Rating	CVSSv2 Score	Explanation
	CRITICAL	9.0 - 10	A vulnerability was discovered that has been rated as critical. This requires resolution as quickly as possible.
	HIGH	7.0 - 8.9	A vulnerability was discovered that has been rated as high. This requires resolution in the short term.
	MEDIUM	4.0 - 6.9	A vulnerability was discovered that has been rated as medium. This should be resolved as part of the ongoing security maintenance of the system.
	LOW	1.0 - 3.9	A vulnerability was discovered that has been rated as low. This should be addressed as part of routine maintenance tasks.
	INFO	0 - 0.9	A discovery was made that is reported for information. This should be addressed in order to meet leading practice.
	GOOD	N/A	Good security practices were being followed or an audit item was found to be present and correct.

3.5 Findings Overview

All the issues identified during the assessment are listed below with a risk rating for each issue.

Phase 1 –Web Application Assessment

Ref	Finding	Risk
SR-102-001	Reflected Cross-Site Scripting (XSS)	Medium
SR-102-002	Uploaded File Types Were Not Restricted	Medium
SR-102-003	HTTP Header Discloses Internal IP Address	Low
SR-102-004	Username Enumeration via Login Function	Low
SR-102-005	Verbose Error Messages	Low

Phase 2 –Internal Infrastructure Assessment

Ref	Finding	Risk
SR-102-006	Unsupported Operating System	High
SR-102-007	Certificate Signed With MD5 Hash	Low
SR-102-008	Closed UDP Ports Discovered	Info

Phase 3 –Mobile Application and Web Service Assessment

Ref	Finding	Risk
SR-102-009	Hard-Coded Credentials	Medium

Ref	Finding	Risk
SR-102-010	**SSL Certificate Checking Disabled**	Medium 
SR-102-011	**No Jailbreak Detection**	Medium 
SR-102-012	**SSL/TLS Renegotiation DoS**	Medium 
SR-102-014	**Backgrounding Screenshots Enabled**	Low 
SR-102-015	**Manual Screenshots Not Disabled**	Low 
SR-102-016	**No Support for TLS Versions Above 1.0**	Low 

4 Technical Details

4.1 Phase 1 –Web Application Assessment

SR-102-001	Reflected Cross-Site Scripting (XSS)		
Risk Rating	Medium	**Status**	Open

Description:
The website was vulnerable to reflected, or non-persistent, cross-site scripting (XSS) attacks. This type of vulnerability occurs when data provided by a web client is used immediately by server-side scripts to generate a page of results for the user. If unvalidated user-supplied data is included in the resulting page without full and proper HTML escaping, client-side executable code may be injected into the dynamic page.

In the case of a GET request, this means that a URL which appears to be associated with the site (and therefore trustworthy to regular users) could contain malicious code that would be executed by the user's browser within the context of the website when the link is visited. In the case of a POST request, a victim user would have to first be coerced to an otherwise unrelated site which then launches attack using a form.

In this case it was established that the `email_address` parameter in the following two URLs were vulnerable.

```
Evidence of URLs requests, payloads and responses redacted
```

Other vulnerable locations are noted in Supplemental Data - Reflected Cross-Site Scripting on page 28.

Reflected cross-site scripting vulnerabilities are extremely common in web applications but can have a serious impact. They are typically used to launch site impersonation or phishing attacks, in which unsuspecting users are lured to malicious sites via links that appear legitimate. The attacker is then free to present the user with what appears to be genuine content, in an attempt, for example, to capture authentication credentials. Another common method of exploitation is to capture the session token of the victim user, allowing their session to be hijacked by the attacker.

Recommendation:
Reliable avoidance of cross-site scripting vulnerabilities should consist of two stages - input validation and output encoding.

Input validation involves the application rejecting any characters which are invalid for the field in question, preferably by whitelisting a limited set of characters (in a telephone number field, for example, the whitelisted characters could be 0-9, parentheses, and hyphens). This strategy can also help in mitigating other flaws which stem from a failure to sanitise input, such as SQL or HTTP header injection attacks.

Output encoding requires the encoding of all special characters (such as those used in HTML and JavaScript) in potentially malicious data. This is generally done directly before display by web applications (or client-side script), and many programming languages have built-in functions or libraries which provide this encoding (also called quoting or escaping in this context). Note that the correct encoding of the output depends on the location that the data is to be used within the response. In the case of it being within the main body of the document, HTML entities must be encoded. If the input is to be used within a script inside of a string, the quotes used for that string must be escaped. In general, it is important to ensure that it is not possible for the data to include whatever sequence is used to demark the end of that data and the beginning of something else.

The application should be reviewed and, if necessary, modified, to handle malicious data properly. The specific instances identified in this finding should be addressed, and the application code base should also be examined for any similar issues which may exist.

Affects:

IP Address	DNS Name	Page	Parameter(s)
10.x.x.x	www.nccgroup-client.co.uk	vulnerablepage.html vulnerablepage2.html	email_address email_address

SR-102-002	Uploaded File Types Were Not Restricted		
Risk Rating	Medium	**Status**	Open

Description:

The web application allowed any type of file to be uploaded, although anti-virus scanning was in place and would detect any attempt to upload malware. The lack of restrictions could allow an attacker to upload an otherwise legitimate file containing malicious code. For example, an attacker might be able target the server itself, by uploading a web shell that would provide the attacker with the capability to run commands on the server and access the file system with the full privileges of the web server process.

As a proof of concept, a .php file was uploaded containing malicious code. If triggered, this would have given the attacker command line execution.

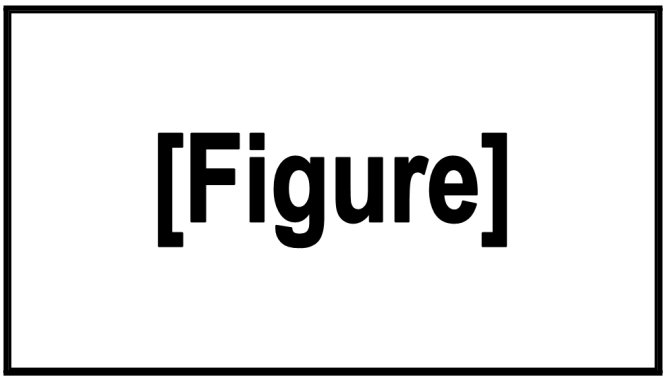

Figure 1: Uploaded PHP file

Although it was possible to upload a file containing malicious code, access to the directory used to store the uploaded files required an Active Directory account. Accordingly, the risk associated with this issue has been lowered to Medium.

Recommendation:

The file types allowed to be uploaded should be restricted to only those that are necessary for business functionality.

- The application should perform filtering and content checking on any files which are uploaded to the server. Files should be thoroughly scanned and validated before being made available to other users. If in doubt, the file should be discarded.

Note that the effectiveness of content-checking controls should not be wholly relied upon, and that they will only provide benefit if the anti-virus and malware signatures are regularly updated.

It is also recommended that some consideration should be given to storing uploaded files in a database, rather than on the file system. This would significantly reduce the risk associated with the file upload facility, but it is recognised that this might require extensive changes to the current application design

Affects:

IP Address	DNS Name
192.1.1.111	www.nccgroup-client.co.uk

SR-102-003	HTTP Header Discloses Internal IP Address		
Risk Rating	Low	**Status**	Open

Description:

It was possible to determine an internal IP address by sending a crafted request to the application. An attacker could use this information to gain a greater understanding of the internal network and tailor further attacks.

The HTTP request and response shown below illustrate this issue.

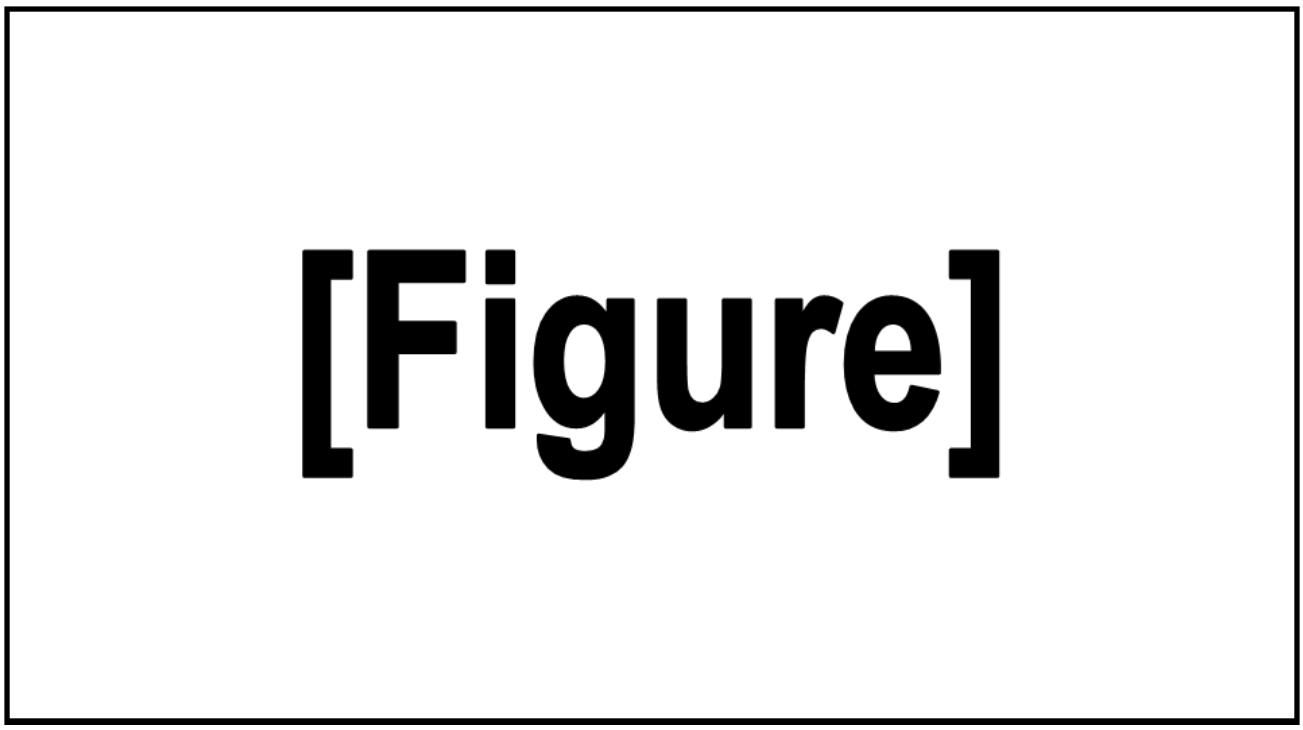

Figure 2: Figure 2 –HTTP response

Recommendation:

On IIS version 7 and above, this issue can be addressed by setting the IIS alternateHostname property (as described at the www.iis.net link below). This allows the specified hostname to be used in place of the internal IP address in redirection responses. Testing should be performed to ensure that this does not adversely affect any legitimate redirection functionality in the application.[1,2]

Affects:

IP Address	DNS Name
192.1.1.111	www.nccgroup-client.co.uk

[1] CWE-200: Information Exposure: https://cwe.mitre.org/data/definitions/200.html
[2] Microsoft Support - FIX: http://support.microsoft.com/kb/834141

SR-102-004	Username Enumeration via Login Function		
Risk Rating	Low	**Status**	Open

Description:

It was possible to enumerate users of the website through the login mechanism. This provided different responses to failed login attempts depending on whether or not the supplied username was valid. An attacker could potentially use this difference in behaviour to compile a list of valid usernames which could then be used as the basis for further attacks against the application.

When an authentication attempt was made with a valid username but an incorrect password, the application returned '*Invalid password –please try again*'. When the username was not in use, the application returned '*Your user name has not been recognised –please re-enter it*'.

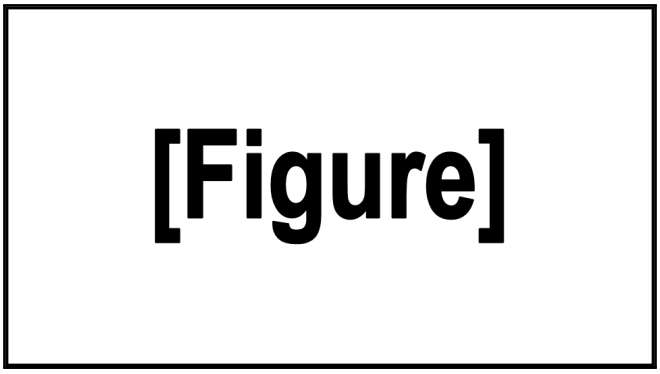

Figure 3: Different responses returned by login functionality

Recommendation:

The login mechanism should be modified to return the same error message for a failed login attempt regardless of whether or not the supplied username is valid.[3,4]

Affects:

IP Address	DNS Name
192.1.1.111	www.nccgroup-client.co.uk

[3] **Username Enumeration Vulnerabilities** https://www.gnucitizen.org/blog/username-enumeration-vulnerabilities/
[4] **Preventing username enumeration** https://blog.portswigger.net/2007/04/preventing-username-enumeration.html

SR-102-005	Verbose Error Messages		
Risk Rating	Low	**Status**	Open

Description:
The web application was configured to display detailed error messages when an application exception was generated. Printing debugging information in the form of a stack trace associated with an error can leak information about the internal structure of the application which could potentially lead to the discovery or classification of additional vulnerabilities.

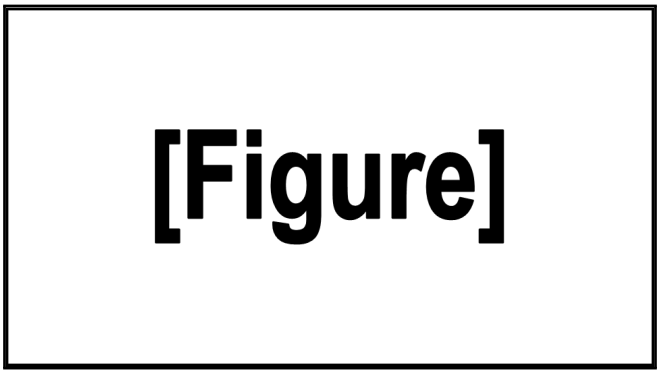

Figure 4: Example of verbose error message

Recommendation:
Application errors should be handled gracefully, and unnecessary technical information should not be presented to users. Applications should return suitably generic but user friendly error messages that do not disclose sensitive information.

Affects:

IP Address	DNS Name
192.1.1.111	www.nccgroup-client.co.uk

4.2 Phase 2 –Internal Infrastructure Assessment

SR-102-006	Unsupported Operating System		
Risk Rating	High	**Status**	Open

Description:
Windows Server 2008 was found to be installed on the affected host. This software reached its end of life date in January 2020 and is no longer supported by Microsoft. It may contain security vulnerabilities that will never be addressed by Microsoft.

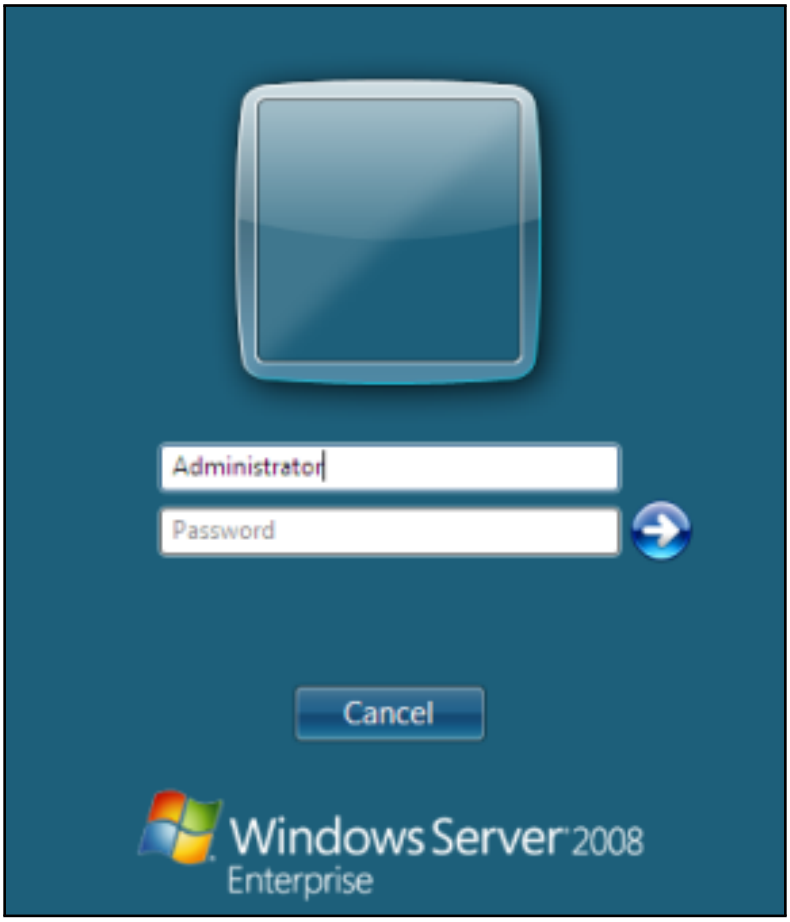

Figure 5: Windows Server 2008 login page

In addition, the length of time this issue has remained unaddressed may indicate a gap in the software patching process.

Recommendation:
This software should be upgraded to the latest stable, secure and supported version of the software. [5]

Investigate the software patching policy and procedures and ensure that updates are applied to all software installations, including third party applications, on a regular basis.

[5] **Microsoft Product Support Status** https://docs.microsoft.com/en-us/lifecycle/products/windows-server-2003-

In particular, it should be ensured that the patching procedures are sufficient to identify software which is about to reach end of life sufficiently early for extended support to be purchased or alternative software purchased *before* end of life is reached. These procedures should acknowledge that software upgrades sometime also require hardware upgrades and factor this into the relevant timescales.

Affects:

IP Address	DNS Name
192.1.1.111	www.nccgroup-client.co.uk

SR-102-007	Certificate Signed With MD5 Hash		
Risk Rating	Low	**Status**	Open

Description:

The SSL/TLS service on the Affected Hosts listed below presented an SSL certificate whose digital signature relied on the MD5 hashing algorithm. MD5 is cryptographically weak and is vulnerable to collision attacks (in which input can be crafted that will result in a hash identical to some other input). This principle can be used to generate spoofed SSL certificates which will be accepted as valid by browsers.

This would allow an attacker to create their own certificate with the same signature as the genuine one. They could then use this to masquerade as the affected service as the spoofed certificate would be indistinguishable from the genuine one.

Whilst this attack has been shown to be possible, it requires a significant amount of computing effort to produce the false certificate. Nevertheless, major browser vendors have already implemented updates which treat MD5 based certificates as insecure and issue warnings to users (see References below).

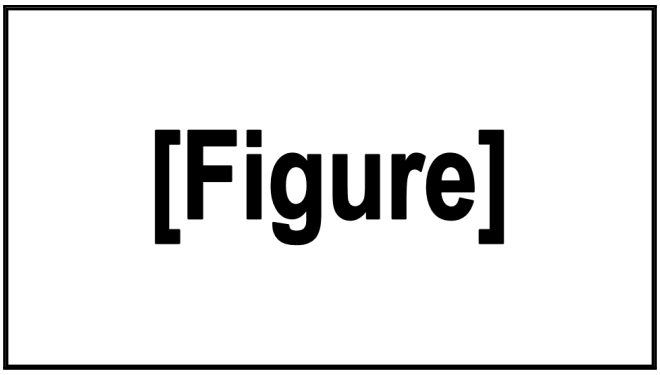

Figure 6: Screenshot with signed with MD5 hash

Recommendation:

Request that the certificate authority reissue the certificate using a stronger hashing algorithm such as SHA-256.[6,7,8,9]

Affects:

IP Address	DNS Name
192.1.1.111	www.nccgroup-client.co.uk

[6] **SSL Certificate Attack Using MD5 Collision** https://www.win.tue.nl/hashclash/rogue-ca/
[7] **Apple** https://support.apple.com/kb/HT4999
[8] **Microsoft** https://support.microsoft.com/kb/2862973
[9] **Chromium** https://code.google.com/p/chromium/issues/detail?id=101123

SR-102-008	Closed UDP Ports Discovered		
Risk Rating	Informational	**Status**	Open

Description:

Port scans revealed the presence of closed UDP ports, which can be useful in fingerprinting systems and greatly speed up an attacker's network scans.

Open UDP ports do not have to send an acknowledgement in response to a request, and closed ports are not even required to send an error packet. However, some devices do send an ICMP port unreachable packet when a packet is sent to a closed UDP port. Thus it may be possible to determine if a port is not open.

Closed UDP ports can be useful for fingerprinting remote systems, and increase the accuracy of scan results. Unused ports allowed through the firewall also provide an attacker with the ability to bind new services to them if a compromise has occurred. This can enable an attacker to gain extended access or to bounce packets to deeper layers within the architecture.

Furthermore, closed ports can reveal the administrative habits and history of the device, potentially enabling identification of services that have been active in the past and that could be accessible in the future.

Recommendation:

All attempts to access unused UDP ports should be silently dropped by filtering devices. This can be achieved through blocking the relevant port on the firewall device.

Affects:

IP Address	DNS Name
192.1.1.111	www.nccgroup-client.co.uk

4.3 Phase 3 –Mobile Application and Web Service Assessment

SR-102-009	Hard-Coded Credentials		
Risk Rating	Medium	**Status**	Open

Description:
During testing, it was discovered that the application communicated with the web service using HTTP Basic authentication. When HTTP Basic authentication is used, the credentials are sent to the server in a base64 encoded format. No cookie or session is upheld, as the authentication header (the base64 string) is sent with each request to authenticate the application's request to the server.

The authentication token was the same for every request sent to the web service, suggesting that authentication was on an application level, and users were only authenticated locally. The base64 encoded authentication token was decoded, as per below:

```
Authentication Token: abcdefghiJKlmnopQrstuvwFtJTRATVhxWV5aSDRV
Decoded from Base64: SVC-SITESBCert:R!m123456qY^ZH4U
```

The consultant then inspected the application binary, and discovered that these credentials were hard-coded into the application.

If an attacker is able to gain access to the application binary, they would be able to recover the clear-text credentials used to communicate with the web service. Additionally, any attacker performing a successful man in the middle attack could sniff the credentials from the HTTP session. Once the attacker has successfully obtained the credentials, they would then be able to communicate directly with the web service, and carry out transactions with the same permissions as the application.

Recommendation:
Credentials should not be hard-coded into the application as these can easily be recovered by an attacker with access to the binary. If HTTP Basic authentication is used, it should be used in conjunction with properly configured TLS/SSL. In addition to authentication, each user of the application should be identified to the web service using an authorisation token, which should be checked to ensure the user is authorised to carry out the requested action. This ensures that even if an attacker gains access to the web service authentication credentials, they would not be authorised to carry out any transactions without an authorisation token.

Affects:
"ABC 123.ipa"

SR-102-010	SSL Certificate Checking Disabled		
Risk Rating	Medium	**Status**	Open

Description:

The application was found to ignore the in-built iOS SSL certificate checking functionality. This meant that it was possible for an attacker to perform a man-in-the-middle attack by presenting their own SSL certificate, and thus nullifies the use of SSL.

In iOS, when connecting to an HTTPS website, if the Common Name field in the SSL certificate presented by the server does not match the host and domain component of the URL, the warning message "Cannot Verify Server Identity" is automatically displayed. The application should handle this case gracefully with NSURLConnection's "`connection: didFailWithError`" method.

It was discovered that the application disabled the default certificate checking, by using the NSURLConnection `continueWithoutCredentialForAuthenticationChallenge` method. This may have been set by the developer to aid in debugging, or for use when the SSL certificate is self-signed or unknown. Certificate checking should never be disabled in a production environment, and the server should always present a valid certificate from a trusted CA.

Recommendation:

In a production environment, SSL certificate checking should be enforced to ensure the confidentiality and integrity of users' information. The application should handle certificate mismatch errors, using the NSURLConnection "`connection:didFailWithError`" method, and deny connections where there is a certificate mismatch. For a defence in depth strategy, consider the use of certificate pinning to allow only connections using a specific certificate or signed by a specific CA.

Affects:

"ABC 123.ipa"

SR-102-011	No Jailbreak Detection		
Risk Rating	Medium	**Status**	Open

Description:
Allowing the application to be installed or run on a jailbroken device exposes the application to trivial and sophisticated attacks, and to reverse engineering.

iOS jailbreaking refers to the process of removing the limitations imposed by Apple on devices (such as the iPhone, iPod touch and iPad) through the use of exploit tools. Jailbreaking enables users to gain complete control of the iOS operating system, allowing them to download additional applications which are not available via the Apple store, and to use the device freely with any carrier. Jailbreaking an iOS device can be a trivial process which takes around five or ten minutes; this process can be undone by reinstalling iOS (in around thirty minutes).

However, jailbreaking an iOS device in this way has security implications, because untrusted code can be installed, security protections can be removed, and installed applications and data can be tampered with (as well as an increased potential for malware on the device). In short, all iOS protection mechanisms can be bypassed if a device has been jailbroken, and as such there can be no inherent guarantee of a secure platform, and any data at rest can potentially be decrypted.

Additionally, jailbreaking can be used on a stolen device in order to help retrieve data from it and access installed applications and services. If the device has no passcode, or a weak passcode set, then the device could be jailbroken to break the security protections included by default.

Recommendation:
It is considered a good security practice to: [10]

- Detect the presence of a jailbreak
- Refuse to install or run on a system which has been jailbroken

Below is a very simple example of a test for jailbroken devices (obviously to be successful, more sophisticated checks are required):

```
BOOL isJailbroken()
{
#if TARGET_IPHONE_SIMULATOR
return NO;
#else
FILE *f = fopen("/bin/bash", "r");
if (errno == ENOENT)
{
// device is NOT jailbroken
fclose(f);
return NO;
}
else {
// device IS jailbroken
fclose(f);
return YES;
}
#endif
}
```

Affects:
"ABC 123.ipa"

[10] **OWASP Jailbreak Cheatsheet** https://www.owasp.org/index.php/Mobile_Jailbreaking_Cheat_Sheet

SR-102-012	SSL/TLS Renegotiation DoS		
Risk Rating	Medium	**Status**	Open

Description:
The remote service encrypts traffic using TLS/SSL but allows a client to renegotiate the connection after the initial handshake.

An attacker could use this vulnerability to trigger renegotiation of existing SSL connections to cause a denial of service due to resource exhaustion on the server. Publicly available DoS tools could take advantage of the renegotiation to automate an attack against the server.

Recommendation:
Disable renegotiation on the server-side completely.

Affects:

IP Address	DNS Name	Port
192.1.1.121	nccgroup-client.co.uk	7083

SR-102-014	Backgrounding Screenshots Enabled		
Risk Rating	Low	**Status**	Open

Description:

By default, when an iOS application is sent to the background (e.g. by pressing the Home button), the operating system will take a screenshot of the current UI and store it for future use. The mobile application did not disable this feature, and hence screenshots containing information on an on-going credit card transaction could be written to the device file system.

A user can put an application into a background or suspended state by either single-clicking or double-clicking the home button on the iOS device. Double-clicking the home button will show the current list of backgrounded applications available for switching to the foreground. To help users identify each application, iOS by default will capture a screenshot of the application as it is closed and display that to the user. Two images will be saved in the following locations:

```
<App>/Library/Caches/Snapshots/<app identifier>/Main/<image.png>
<App>/Library/Caches/Snapshots/<app identifier>/Main-downscaled/<image.png>
```

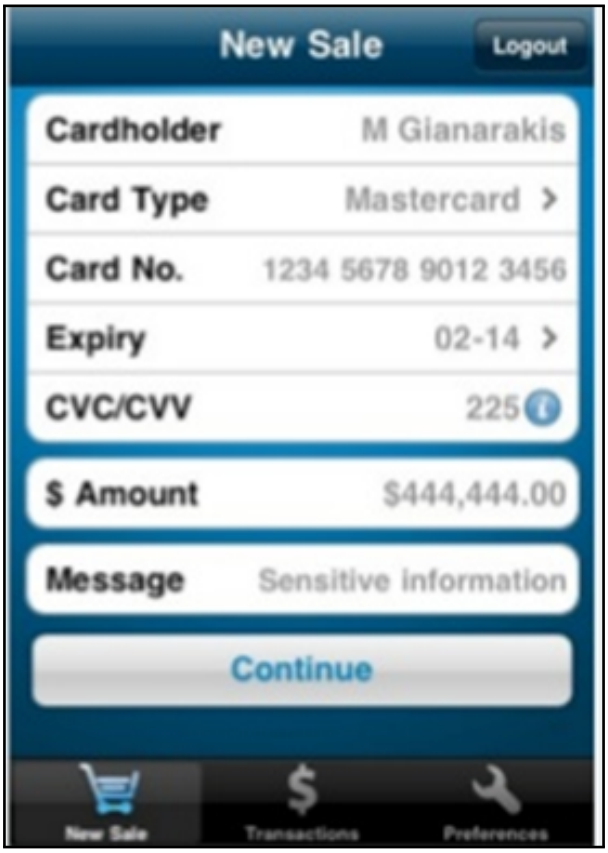

Figure 7: Backgrounding screenshot [example]

In order to exploit this issue, an attacker would have to gain access to the device file system (e.g. by stealing a device, obtaining the user's PIN, and jailbreaking it).

Recommendation:
The application should prevent iOS from taking a snapshot of sensitive data when the background or suspended state is entered.[11,12,13]

The application can programmatically hide or eliminate sensitive data from the screen before the snapshot is taken. This can be accomplished by setting window.hidden to YES in the applicationDidEnterBackground delegate and window.hidden to NO in the applicationWillEnterForeground delegate. This will have the effect of blanking out the UI before the screenshot is taken and redraws is when the application is relaunched.

The exact method used to hide the sensitive data will depend on the application, but some common techniques include blanking of sensitive data fields, covering the entire UI with a default image, or distorting the UI using the UIImageEffects class which is available from iOS 7 onwards.

Affects:
"ABC 123.ipa"

[11] iOS Developer Documentation for UIImageEffects: https://developer.apple.com/library/prerelease/ios/samplecode/UIImageEffects/Introduction/Intro.html#//apple_ref/doc/uid/DTS40013396-Intro-DontLinkElementID_2
[12] iOS Background Screen Caching: https://www.virtuesecurity.com/blog/ios-background-screen-caching/
[13] iOS Developer Documentation IgnoreSnapshot: https://developer.apple.com/library/ios/documentation/UIKit/Reference/UIApplication_Class/#//apple_ref/occ/instm/UIApplication/ignoreSnapshotOnNextApplicationLaunch

SR-102-015	Manual Screenshots Not Disabled		
Risk Rating	Low	**Status**	Open

Description:
It was possible for the user to take screen captures of the mobile application, using iOS's screenshot key combination. This could lead to images containing sensitive information (such as recently accessed client information) being stored in unencrypted form on the device file system. Although it is perhaps unlikely that the user would deliberately take screenshots of their online banking data, it is relatively easy to press the relevant key combination by accident, and this could lead to the inadvertent leakage of sensitive data.

The image below is a manual screenshot taken on an iOS device (using the *Power+Home* key combination):

Figure 8: Mobile application screenshot [example]

Recommendation:
There is no documented method to prevent manual screenshots being taken on iOS at the time of writing, and hence this risk has to be accepted by ClientName.[14]

Affects:
"ABC 123.ipa"

[14] **How to take a screenshot on your iPhone, iPad, and iPod touch** https://support.apple.com/en-gb/HT200289

SR-102-016	No Support for TLS Versions Above 1.0		
Risk Rating	Low	**Status**	Open

Description:

The affected server did not support TLS above version 1.0. Since 2008, TLS has been at version 1.2, with version 1.3 being defined in August 2018. Versions 1.2 and 1.3 of TLS are more resistant to known attacks, and TLSv1.2 supports more modern cipher suites that are widely acknowledged to offer the best cryptography available for securing Internet connections. TLSv1.3 further improves on security by removing unsafe or unused features, eliminating unnecessary handshake steps and forcing the use of newer encryption methods. It should be noted that use of TLS 1.0 and 1.1 is now flagged as insecure (such as by browser warnings) by the major browsers, and in January 2021 the NSA issued guidance urging system administrators to only provide support for TLS versions 1.2 or 1.3.[15]

TLS 1.0 was the highest protocol version supported, as shown in Supplemental Data - SSLScan Output on page 30.

Recent versions of all web browsers support 1.2 and 1.3, and therefore supporting these protocols server-side offers better security for those clients using modern browsers. TLSv1.2 also supports a class of cipher suites that offer Authenticated Encryption with Associated Data (AEAD). These cipher suites include an authenticated integrity check within the encryption operation, and are resistant to more attacks than their older counterparts. Google's Chrome browser states that the security of connections to websites that do not use AEAD ciphers is "obsolete". Currently, this is not presented as an error to the user, merely as information for anyone viewing the page's security report (in the browser's developer console). However, as Google continues to push for the rapid adoption of more robust Internet cryptography, this message may become more prominent.

TLSv1.3 was approved for use in August 2018 and goes to further enhance security. The security benefits of TLSv1.3 include the removal of unsafe or unused features, elimination of unnecessary handshake steps and the forced use of newer encryption methods. As well as preventing encryption downgrade attacks, the streamlined approach to session initiation will offer performance gains over previous TLS implementations.

It should be noted that major browser vendors have stated that support for TLSv1.0 and 1.1 will be dropped in the near future.

Supporting TLS 1.0 as a security control is non-compliant with the Payment Card Industry (PCI) Data Security Standard (DSS). While these regulations may not be directly relevant, it should be expected that this directive will become security best practice and may also be reflected in more obvious ways, such as web browser warnings.

Recommendation:

Add support for TLS v1.2 and v1.3.[16,17,18,19,20,21,22]

Support for TLS version 1.0 should be disabled unless there is a specific business requirement to allow users of older, less secure browsers to continue to connect. Consider also disabling support for version 1.1, which will become unsupported by major browsers in the near future.

All affected hosts should be configured to prefer the latest cipher suites that they offer, such as AES-GCM (TLS 1.2 and 1.3 only).

The reference from Mozilla below provides recommendations on cipher suite ordering based on the profile of con-

[15] Eliminating Obsolete Transport Layer Security (TLS) Protocol Configurations - https://media.defense.gov/2021/Jan/05/2002560140/-1/-1/0/ELIMINATING_OBSOLETE_TLS_UOO197443-20.PDF
[16] Mozilla Server Side TLS: https://wiki.mozilla.org/Security/Server_Side_TLS#Recommended_configurations
[17] Obsolete Cipher Suites: https://www.chromium.org/Home/chromium-security/education/tls#TOC-Obsolete-Cipher-Suites
[18] OWASP Transport Layer Protection Cheat Sheet: https://cheatsheetseries.owasp.org/cheatsheets/Transport_Layer_Protection_Cheat_Sheet.html
[19] NCC Group Whitepaper on the Configuration of SSL/TLS Services: www.nccgroup.com
[20] PCI Security Standards Council: https://www.pcisecuritystandards.org/documents/Migrating-from-SSL-Early-TLS-Info-Supp-v1_1.pdf
[21] Saying Goodbye to SSL/Early TLS: blog.pcisecuritystandards.org
[22] Overview of TLS v1.3: https://owasp.org/www-pdf-archive/OWASPLondon20180125_TLSv1.3_Andy_Brodie.pdf

necting clients.

Affects:

IP Address	DNS Name	Port
192.1.1.121	nccgroup-client.co.uk	7083

5 Supplemental Data - Reflected Cross-Site Scripting

The vulnerable locations included:

```
<http://www.nccgroup-
➜  client.co.uk/ vulnerablepage.html?sortColumn=&ascending=&nocache=1234567890123456&abCde=
➜  <script>alert(document.cookie)</script>;//&kLmno=&isFromHomePage=

http://www.nccgroup-
➜  client.co.uk/ vulnerablepage.html?sortColumn=&ascending=&nocache=1234567890123456&abCde =&kLmno=
➜  <script>alert(document.cookie)</script>;//&isFromHomePage=>

http://www.nccgroup-client.co.uk/ vulnerablepage2.html?ajYard=';alert(document.cookie)//kLmno =A

http://www.nccgroup-client.co.uk/ vulnerablepage2.html?kLmno=A';alert(document.cookie)//

http://www.nccgroup-client.co.uk/vulnerablepage2.html?ajType=';alert(document.cookie)//

http://www.nccgroup-client.co.uk/vulnerablepage2.html?isFromHomePage=';alert(document.cookie)//
```

An example response is included below:

Request:

```
http://www.nccgroup-
➜  client.co.uk/c2/vulnerablepage.html?sortColumn=&ascending=&nocache=1234567890123456&abCde =
➜  </script><script>alert(document.cookie);//&ajLane=&isFromHomePage=
```

Response:

```
HTTP/1.1 200 OK
Server: Resin/3.1.8
Cache-Control: private
Content-Language: en-GB
Vary: Accept-Encoding
Set-Cookie: PSID=12AbcdefghijKLMNnpqr==; path=/; expires=Tue, 09-MAR-2021 23:13:36 GMT
Content-Type: text/html; charset=UTF-8
Date: Mon, 08 Mar 2021 23:13:36 GMT
Content-Length: 18349

[…]
<script type="text/javascript"> //<![CDATA[ autoJoin('</script><script>alert(document.cookie);//','',aj
➜  Type); //></script>
```

6 Supplemental Data - Network Scan Results

This section has been redacted.

7 Supplemental Data - SSLScan Output

```
Redacted.
```

8 Document Control

Client Confidentiality

This document contains Client Confidential information and may not be copied without written permission.

Proprietary Information

The content of this document should be considered proprietary information and should not be disclosed outside of Client Company Ltd (ClientName).

NCC Group gives permission to copy this report for the purposes of disseminating information within your organisation or any regulatory agency.

8.1 Document Data

Data Classification	Client Confidential
Client Name	ClientName
Project Reference	99999
Proposal Reference	P99999-Sample
Document Title	Web Application, Internal Infrastructure and Mobile Application Security Assessment Sample Report
Author	NCC Consultant

8.2 Document History

Issue No.	Issue Date	Issued by	Change Description
0.1	2015-01-15	NCC Consultant	Draft for NCC Group internal review only
0.2	2015-01-16	A Reviewer	Revised QA
1.0	2018-01-15	NCC Consultant	Released to client
2.0	2018-02-14	NCC Consultant	Cover page address updated
3.0	2021-03-23	NCC Consultant	Minor content and format changes

8.3 Document Distribution List

Name	Role
A. Sponsor	Project Sponsor, ClientName
NCC Consultant	Senior Security Consultant, NCC Group
A. Manager	Account Manager, NCC Group

9 Assessment Team

The following members of staff were assigned to this assessment:

Name	Job Title	Comments
NCC Consultant	Senior Security Consultant	CREST Registered Tester (CRT), CHECK Team Member (CTM)