Report for:

Mobile CRM Solution Security Assessment – Management Summary

Resco

August 2018

Version: 1.0

Prepared By:	Ian Cornish
Email:	ian.cornish@nccgroup.trust
Telephone:	+44(0)7949 332573

NCC Group PLC - Security Testing Audit and Compliance

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

Executive Summary

This report presents the findings of the Mobile CRM Solution Security Assessment on behalf of Resco. The assessment was conducted between 11/06/2018 and 22/06/2018 and was authorised by Resco. The solution was further assessed on 20/07/2018 and 30/07/2018 in order to determine the effectiveness of remedial activities performed by Resco.

Resco has developed a mobile CRM solution that allows businesses to manage their Microsoft Dynamics and Salesforce CRM systems. The solution is built upon a shared code base using the Xamarin C# framework and encompasses a variety of platforms (including Android, iOS and Windows Mobile). All platforms use a web service API in order to manage CRM data. As it is expected that the data stored by the solution will be sensitive in nature, it is important that the solution is secure to ensure that such data is appropriately protected.

NCC Group hereby gives Resco permission to disclose this report to third parties. NCC Group carried out the testing for Resco and accepts no liability to any other party that relies on this report. The results set out in this report are only applicable to the system as tested by NCC Group during the dates of testing as set out above.

Overview

Resco have worked with NCC Group who were contracted to assess the security of the Mobile CRM Solution. A white box approach was taken for the assessment which consisted of reviewing the source code of the mobile application and associated web service API. This approach was preferred over a black box assessment as it allows for a more thorough assessment to be performed, resulting in the identification of a wider range of vulnerabilities and instances thereof.

Further information that details the approach taken in testing the solution can be found in Tailored Methodologies, Section 2.1.

Following the initial assessment, Resco were swift to act upon the identified risks by implementing a programme of remedial actions in order to mitigate the most significant issues. This included addressing an issue that could result in an authenticated attacker obtaining data stored in the back-end database. Two rounds of retesting were performed by NCC Group in order to assess the effectiveness of the remediation. The first of these, performed on 20/07/2018 focussed on the significant issues found during the Web Service Assessment, while the second, performed on 30/07/2018 focussed on issues found during the Mobile Application Code Review. These retests concluded that the most significant risk had been successfully addressed, alongside a small number of other issues that were determined to be of importance to the security posture of the solution.

After verifying Resco’s remedial activities it can be considered that the Mobile CRM solution presents a good security posture that is appropriate to the data which requires protection. While some of the issues raised remain, it is not expected that they represent a significant risk to the security of the solution and the data that it provides access to. Subsequent discussion with Resco indicated that some risk would need to be accepted so as to maintain application functionality. Where this is the case, it is important that this is documented within the relevant Risk Register so that Resco maintain visibility of the risk to which the solution is exposed. It is recommended that, where possible, the remaining issues are addressed to ensure that the solution adheres to a defence in depth approach to security, in accordance with security best practice.

The following table breaks down the issues which were identified by phase and severity of risk (issues which are reported for information only are not included in the totals). This table reflects the status of the issues after the retest of 30/07/2018:

Phase	Description	Critical	High	Medium	Low	Total
1	Web Service Assessment	0	0	0	5	**5**
2	Mobile Application Code Review	0	0	0	7	**7**
	Total	**0**	**0**	**0**	**12**	**12**

Assessment Summary

A security assessment was performed of the Mobile CRM solution. The assessment was conducted from a white box perspective and included a review of the source code of the mobile application and web service components that combine to form the solution. Two further assessments were performed, following a programme of remedial activity by Resco, in order to determine the effectiveness of the remediation.

The initial assessment of the web service identified an issue that was assessed to pose a high risk. It was possible for an authenticated user to inject SQL statements via XML web service calls. The web service implemented a flexible procedure to translate XML requests into SQL statements in order to perform database queries. Due to this flexibility and a lack of input validation, it was possible to inject arbitrary SQL statements in order to retrieve and manipulate information stored in the database server. This issue was not limited to the user’s organisation’s database; but rather it was possible to access data belonging to other organisations that use the platform. Due to the expected sensitivity of the data stored within the backend Resco database (should customers choose to use this backend solution over the Salesforce or Microsoft Dynamics integrations) this issue was considered to pose a significant business risk for Resco’s customers.

Resco were quick to acknowledge this risk by implementing an effective mitigation strategy. This involved the creation of a new procedure within the application to provide comprehensive validation of the dynamic fetch query. The issue was retested by NCC Group on 20/07/2018 and was determined to be effective in mitigating the risk.

Other issues relating to the web service that were considered to be of significance were as a result of authentication controls that had been implemented, but had not been enabled. Resco have since enabled the account lockout and password policy mechanisms. This has the benefit of mitigating automated password guessing attacks while ensuring that users are required to choose strong passwords. This said, it was noted that the password policy did not require the use of mixed case characters. This may be an oversight, however requiring the use of lower and upper case characters would increase the complexity of passwords and so make them more resistant to guessing.

The review of the mobile application source code identified no issues considered to pose a high risk. A small number of medium risk issues were identified which included the ability to bypass a control designed to prevent application configuration files from being tampered with. The impact of this is that changes could be made to a configuration file in order to send potentially sensitive data to an unauthorised URL. The risk associated with this issue was lessened as exploitation would require an attacker to have already gained a position of high privilege, such as local access to the device. This issue was addressed by Resco during their remediation programme by updating the affected code to prevent the application falling back to the legacy behaviour. The updated code was reviewed by NCC Group on 30/07/2018 and was considered to appropriately mitigate the risk.

As is often the case, the mobile application made use of a number of third party libraries. The inclusion of third party libraries within the code base can represent a security risk in the event that they are not updated as new versions are released that address publically disclosed vulnerabilities. This was found to be the case for the mobile application, whereby a number of the libraries were outdated – some of which were affected by security issues. Efforts were made by Resco during their remediation programme to update the affected libraries. The retesting conducted by NCC Group on 30/07/2018 showed that the remediation had been effective, with the version of jQuery having been updated for all HTML pages that made use of the library. The retesting also highlighted that the outdated versions of the Moment, Knockout and JSON3 libraries remained within the code base; however, it was determined that these libraries were not linked by any application page and consequently do not represent a security risk.

Of the issues that have not been addressed through Resco’s remediation programme, all were assessed to pose a low risk and do not represent a direct threat to the security of the solution. Nevertheless, it is recommended that these outstanding issues are addressed, where possible, to bring the solution into line with security best practice. Feedback from Resco indicated that some of these issues cannot be addressed due to the expected impact on application functionality. In these instances, it is suggested that this is documented within the relevant Risk Register so that Resco maintain full visibility of the outstanding risks.

A brief description of each of the issues which were identified is included in Section 1.4 of this report. The issues that were retested during the assessments of 20/07/2018 and 30/07/2018 have been marked as CLOSED, PART CLOSED or OPEN, depending on their state of remediation. Issues that were not retested have been marked as NOT TESTED.

Table of Contents

Using This Report

To facilitate the dissemination of the information within this report throughout your organisation, this document has been divided into the following clearly marked and separable sections.

Document Breakdown		
	Executive Summary	Management level, strategic overview of the assessment and the risks posed to the business
1	Technical Summary	An overview of the assessment from a more technical perspective, including a defined scope and any caveats which may apply
2	Appendices	This section usually includes the security tools which were used, outlines the assessment methodologies and lists the assessment team members

Document Control

Client Confidentiality

This document contains 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 Resco.

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

Document Version Control	
Data Classification	Public
Client Name	Resco
Project Reference	63893
Proposal Reference	Resco-2018-001
Document Title	Mobile CRM Solution Security Assessment
Authors	Ian Cornish Luke Rogerson Paul Collett Peter Winter-Smith Ramon Salvador

Document History			
Issue No.	**Issue Date**	**Issued By**	**Change Description**
0.1	06/09/2018	Ian Cornish	Draft for NCC Group internal review only
0.2	13/09/2018	Luke Rogerson	Draft released to client
1.0	26/09/2018	Luke Rogerson	Released to client

Document Distribution List	
Miro Pomsar	Project Sponsor, Resco
Ian Cornish	Technical Author, NCC Group
Sherief Hammad	Account Manager, NCC Group

1 Technical Summary

NCC Group was contracted by Resco to conduct a security assessment of the Mobile CRM solution in order to identify security issues that could negatively affect Resco's business or that of its customers if they led to the compromise or abuse of the solution.

1.1 Scope

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

- Web Service Assessment including Code Review of the FetchXML API
- Mobile Application Code Review

Source code was provided in the form of zip files, the specific files provided are detailed below:

Mobile application source code

- File: AppSource.zip
- MD5 hash: 1ec00ad911262a3438ec5c37a0b8de49

Web service source code

- File: RescoCRM.Server.zip
- MD5 hash: 2a5d2cb15c076fff3ce8e13a9aac2174

Resco deployed a test environment to perform the assessment at the following URL:

- progres-dev.rescocrm.com

A threat modelling exercise was conducted during production of the statement of work (SOW). The result of this exercise identified the following key areas of risk which formed the focus of the assessment:

Mobile Application Code Review

- Assess the security of areas of the mobile application using encryption and hashing
- Assess the authentication and session implementation between the Mobile CRM application and Microsoft Dynamics, Salesforce and Resco's own backend component
- Confirm that code between trust boundaries is suitably robust
- Ensure that all sensitive data is handled safely and encrypted at rest
- Ensure that any project dependencies are up to date and do not contain known security vulnerabilities
- Take note of any other security issues that arise from the review of the code

Web Service Assessment including Code Review of the FetchXML API

- Ensure that any project dependencies are up to date and do not contain known security vulnerabilities
- Perform a security review of each API endpoint
- Review the implementation of the underlying SOAP-XML and OData4 service to ensure it is suitably robust
- Take note of any other security issues that arise from the review of the code

Following a programme of remediation performed by Resco – two further assessments were conducted by NCC Group, in order to assess the effectiveness of the remediation. Specifically, the following issues were retested:

Retest 20/07/2018

- RESO-001-1-1 - SQL Injection
- RESO-001-1-2 - No Account Lockout
- RESO-001-1-3 - Verbose Web Service Errors
- RESO-001-1-5 - Ineffective Input Validation
- RESO-001-1-7 - Password Policy Disabled

Retest 30/07/2018

- RESO-001-2-1 - Outdated Third Party Libraries
- RESO-001-2-2 - Config File HMAC Bypass
- RESO-001-2-8 - Use of SHA-1

Resco provided updated source code and a test environment with the applied fixes in order to perform the retest assessments. The source code files provided for review are detailed below.

The following source code file was provided for the retest of 20/07/2018:

Web service source code

- File: RescoCRM.Server.Update_July.zip
- MD5 hash: 1c73007922078fb9f47b1d24f0f0f9b7

The following source code file was provided for the retest of 30/07/2018:

Web service source code

- File: AppSource_Updated.zip

MD5 hash: 9fc204fbebdc4f83d418da9c81355e25

The test environment was located at the following URL:

- progres-dev.rescocrm.com

1.2 Caveats

The scope originally included an assessment of the OData4 service. This service provides functionality similar to that of the FastXML service (querying back-end data stored on Resco systems). As the OData4 service is not currently used by Resco’s customers, and is optional and will be set to off-by-default in the future, the decision was made that the time allocated to the assessment would be better spent focusing on the FastXML service. Therefore no assurance can be given as to the security of the OData4 service or how this feature could affect the security posture of the wider solution.

1.3 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.
ℹ	INFO	0 - 0.9	A discovery was made that is reported for information. This should be addressed in order to meet leading practice.
✓	N/A	N/A	Good security practices were being followed or an audit item was found to be present and correct.

1.4 Findings Overview

All the issues identified during the assessment are listed below with a brief description and risk rating for each issue. The risk ratings used in this report are defined in Section 1.3 Risk Ratings.

Phase 1 – Web Service Assessment

Ref	Finding	Retest	Risk
RESO-001-1-1	***SQL Injection*** The application implemented a procedure to convert fetchxml queries into SQL statements. While all parameters to be used by the SQL query were parameterised, the procedure also converted fetchxml variables to parts of the SQL statement, allowing custom SQL to be injected. It should be noted that in order to exploit this vulnerability, an attacker would need to be authenticated to the service.	CLOSED	**High** ⛔
RESO-001-1-2	***No Account Lockout*** The service implemented an account lockout mechanism, however this mechanism was disabled. Such a mechanism prevents any further authentication attempts after a certain number of consecutive failed login attempts within a specified time frame. Lockout mechanisms are important for the prevention of successful automated password attacks.	CLOSED	**Medium** ❗
RESO-001-1-3	***Verbose Web Service Error Messages*** The web services returned detailed error messages when the transmitted request was not properly formatted or caused an application error. Although this could be helpful to a legitimate developer, it could also aid an attacker in crafting malicious yet well-formed requests, and could leak information about the application environment.	CLOSED	**Low** ⚠
RESO-001-1-4	***Code Comments Suggest Incomplete or Missing Code*** A search for code comments with the terms “FIXME” or “TODO” and derivatives thereof identified a number of instances where developers have noted incomplete or missing code.	NOT RETESTED	**Low** ⚠
RESO-001-1-5	***Ineffective Input Validation*** Weaknesses were identified in the way the service handled user-supplied input. This allowed the injection of HTML tags through the alias attribute which allowed manipulating the resulting XML document.	CLOSED	**Low** ⚠
RESO-001-1-6	***Unsafe Use of SHA-1*** The application made use of the SHA-1 algorithm. SHA-1 is now considered to be cryptographically weak, in that it is vulnerable to collision attacks. This means that it is possible for an attacker to create two messages that have the same computed SHA-1 hash value.	NOT RETESTED	**Low** ⚠
RESO-001-1-7	***Password Policy Disabled*** The password policy implemented by the web service had not been enabled. Consequently it would be possible to set weak password values. Weak passwords can be easier to guess or to determine through a brute-force attack and could therefore lead to the compromise of user accounts.	PART CLOSED	**Low** ⚠

Ref	Finding	Retest	Risk
RESO-001-1-8	***Weak SSL Cipher Suites Supported*** A cipher suite supported by the web service was not sufficiently cryptographically secure and, as a result, cannot provide as much protection against brute-force decryption when compared to more modern cipher suites, should the traffic be captured.	NOT RETESTED	**Low** ⚠
RESO-001-1-9	***No TLS/SSL Downgrade Attack Prevention*** The affected SSL/TLS service did not implement any protections against SSL downgrade attacks, such as the TLS_FALLBACK_SCSV mitigation. This meant that an attacker could potentially use a man-in-the-middle attack against an active connection between a client and the web server and downgrade connections to a more vulnerable protocol such as TLSv1.	NOT RETESTED	**Low** ⚠
RESO-001-1-10	***BEAST SSL/TLS Weaknesses*** A vulnerability that could allow information disclosure exists in SSL and version 1.0 of TLS. The weakness is caused by an improper choice of initialisation vector (IV) used by block ciphers operating in cipher block chaining (CBC) mode. The exploit that takes advantage of the vulnerability is known as “browser exploit against SSL/TLS” (BEAST). BEAST allows attackers to compromise the confidentiality of connections to reveal short sections of plaintext, with session cookies being the most likely target. The potential for the BEAST vulnerability has been reported here because the affected SSL/TLS service supported block ciphers in CBC mode operating under vulnerable SSL/TLS protocol versions.	NOT RETESTED	**Info** ℹ
RESO-001-1-11	***HTTP “Basic” Authentication in Use*** The web service did not implement a session management mechanism. Instead, the service required the username and password to be provided for each request. As a result, should an attacker gain access to one single request, it would be possible to compromise the user’s account. This is contrary to a session managed using cryptographically secure tokens, where compromising a single token would only allow an attacker to compromise the user’s active session.	NOT RETESTED	**Info** ℹ

Phase 2 – Mobile Application Code Review

Ref	Finding	Retest	Risk
RESO-001-2-1	***Outdated Third Party Libraries*** A number of third party libraries included within the code base were found to be outdated. Third party libraries should be kept up to date to ensure they contain the latest security patches and enhancements.	CLOSED	**Medium** ❗
RESO-001-2-2	***Config File HMAC Bypass*** The software did not correctly handle the case when the HMAC is not present in the configuration file. As a result, it might be possible to forge the contents of a configuration file, although this would require local access to the device where the configuration file was stored.	CLOSED	**Medium** ❗
RESO-001-2-3	***Exposed API OAuth Application Keys and Secrets*** Various third party API keys were discovered in the code base, including keys for Microsoft, Google, Salesforce, DropBox, Docusign and Universign services. An attacker would be able to extract these credentials from the compiled application and potentially use them to impersonate the application and trick users into connecting to services used by the application.	NOT RETESTED	**Low** ⚠
RESO-001-2-4	***Hardcoded HMAC Credentials*** The codebase contained hardcoded credentials used as HMAC secrets. This does not conform to security best practice guidelines as the same passwords would be used on all device's installations.	NOT RETESTED	**Low** ⚠
RESO-001-2-5	***Encrypted File Password Stored in Plaintext*** The application settings were stored twice, once with secure storage and a second time in an encrypted file. The key used to encrypt the second file was stored in plaintext within the config.xml file.	NOT RETESTED	**Low** ⚠
RESO-001-2-6	***No Certificate Pinning*** It was possible to intercept the encrypted traffic being passed between the application and the various servers it communicated with as certificate pinning was not enforced. This means that a suitably-placed or equipped attacker could monitor the data in transit and possibly acquire sensitive information.	NOT RETESTED	**Low** ⚠
RESO-001-2-7	***Application Sends Device identifiers to Resco*** The application sent detailed device-specific information to Resco servers in the form of the device name (UDID for iOS devices) and operating system. Identifiers such as these should not be sent to third parties.	NOT RETESTED	**Low** ⚠
RESO-001-2-8	***Use of SHA-1*** The application made use of the SHA-1 algorithm. SHA-1 is now considered to be cryptographically weak, in that it is vulnerable to collision attacks. This means that it is possible for an attacker to create two messages that have the same computed SHA-1 hash value.	CLOSED	**Low** ⚠

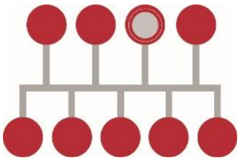
Ref	Finding	Retest	Risk
RESO-001-2-9	***Code Comments Suggest Incomplete or Missing code*** A search for code comments with the terms “FIXME” or “TODO” and derivatives thereof identified a number of instances where developers have noted incomplete or missing code.	NOT RETESTED	**Low** ⚠
RESO-001-2-10	***No Root Detection*** The mobile application did not implement security controls designed to detect when it was running on a ‘rooted’ device or an Android emulator. Devices that have been rooted essentially have a degraded security model. This can cause sensitive data to be exposed to a malicious user (e.g. somebody who has stolen the device), or a malicious application installed on the device. Furthermore, an attacker can use various tools such as debuggers, hooking frameworks and profilers to study the application while it is running on a rooted device or emulator.	NOT RETESTED	**Low** ⚠
RESO-001-2-11	***Stack Trace Written to Log File*** The application wrote stack trace information to a number of different log files, potentially revealing information on the inner workings of the application. This information was also easily visible to a user, because an option was available for users to email crash information to the developers.	NOT RETESTED	**Info** ℹ

2 Appendices

2.1 Tailored Methodologies

2.1.1 Code Review – Mobile Apps

Key Information



Detailed white-box analysis of source code for mobile applications, which can uncover a wide range of vulnerabilities.

NCC Group has extensive experience of analysing applications for iOS, BlackBerry, Android, Windows Phone, and many other embedded mobile platforms at source code level.

Analysis of the data being stored locally on the device, and any methods being used to encrypt it.

Assessment of the techniques used for secure communication.

Can be combined effectively with dynamic black-box testing – these often prove to be useful complementary approaches.

Test Highlights

The initial phase of the source code review will involve threat modelling to aid in the identification of security-critical and other high-priority areas. In many cases there is not sufficient time available to perform a full manual review of all the available source code. Using the threat-modelling approach ensures that NCC Group reviews code in order of criticality.

Detailed manual review of the source code then begins, seeking to identify implementation-level bugs (caused by coding errors or insecure development practices) as well as design-level issues (where the application does not successfully address its threat model). Mobile platforms typically provide large parts of the security functionality required by applications, and much of the source code review work involves checking that the platform-provided APIs are being used correctly.

The focus of the assessment typically includes:

- Identification of application data which is being stored locally on the device (either in databases or on the file system), ensuring that all sensitive data is encrypted. Ideally this should make use of platform-provided APIs such as Keychain on iOS or content protection and media card encryption on BlackBerry.

- Checking for situations where sensitive data from the application is unintentionally stored on the device, perhaps due to web caching or stored screenshots, and ensuring that these are handled safely.

- Privacy leaks are often a major concern for mobile applications, with numerous high-profile cases involving location information and contact data – code review will ensure that this data is being handled securely and not leaked over the network or onto the file system.

- Most mobile applications use SSL/TLS for traffic encryption – the assessment will ensure that this is configured correctly, with a particular focus on the validation of certificates. Many applications are weak in this area, which can potentially enable man-in-the-middle attacks.

- Code which implements key trust boundary functions such as login, authentication, key generation, or input validation and filtering is examined in detail.

- Logging and error-handling code will be reviewed. It is common to find mobile applications accidentally logging sensitive information which can end up on the device's file system.

- Source code review is the ideal means of finding hardcoded credentials, test data, or debug functionality which should not be present in the application but frequently remains due to developer oversight. An attacker may be able to find these by reverse engineering the application

- The documentation, code comments, and coding conventions will be assessed. Extensive documentation and comments and consistent coding conventions can help to minimise the chance of security-related problems being introduced during maintenance of the code.

NCC Group can also review the build configuration for the application to ensure that full advantage is taken of any anti-exploitation features offered by the compiler and mobile platform, such as ASLR, and analyse the effectiveness of any anti-reverse-engineering or jailbreak-detection technologies which are being used.

The final report produced by NCC Group will include detailed descriptions of any vulnerabilities found, along with an overall assessment of the level of security exhibited by the code.

2.1.2 Web Service Assessment

Key Information

The primary areas of concern in web service security are code execution, authentication bypass, injection, privilege escalation, and data extraction.

NCC Group's web service assessment will find common vulnerabilities such as message replay attacks, XML complexity attacks, and transport security weaknesses.

Web service assessments can be performed either remotely or on site, depending on the exposure of the service. The purpose of the assessment is to identify any vulnerabilities which can be exploited in order to attack the system or other users, bypass controls, escalate privileges, or extract sensitive data.

During the assessment the consultants will use proven non-invasive testing techniques to quickly identify any weaknesses. The service is assessed from several perspectives, including with no credentials, user credentials, and privileged user credentials.

More Details

Unvalidated Input

Where information from web requests is not validated before being used by a web service, an attacker could use this flaw to access and attack the supporting back-end components or other users. Examples of this type of attack include SQL injection, OS command injection, and SOAP injection.

Broken Access Control

Access control restrictions determine what authenticated users are allowed to do in a web service. When they are not properly enforced an attacker can exploit these flaws to access other users' accounts, view sensitive files, or use unauthorised functions.

Buffer Overflows

Some web service components may be vulnerable to buffer overflow attacks. A remote attacker may be able to provide specially-crafted malicious input which causes the components to crash and, in some cases, can lead to remote code execution.

Injection Flaws

Web services pass data between the user and server using a protocol called SOAP (the Simple Object Access Protocol), the basis of which is an XML structure defined in a WSDL (Web Services Description Language) document. If an attacker can embed malicious commands in the SOAP parameters, the external system may execute those commands on behalf of the web service.

Improper Error Handling

There are instances where error conditions occur during normal operations and are not handled properly. If an attacker can identify the errors that the web service fails to handle correctly, they can systematically force those errors, revealing system information.

Insecure Storage

Storing information such as credentials usually involves cryptography. Integrating cryptography into a web application can be complex, and as a result there are often deficiencies in its execution. When the cryptographic function is not coded properly, or is not integrated appropriately, information is not protected.

Denial of Service

An attacker can survey a service to determine what processes use the most resources. With this knowledge it is possible to consume web service resources to a point where legitimate users can no longer access or use the service. In extreme cases the service can be knocked over and cease functioning completely.

Detailed Methodology

We will perform an in-depth and thorough assessment of in-scope web services to ensure that correct configuration and recommended practices have been followed to minimise client exposure. The following is a sample list of common tests that are performed when carrying out a web service test. It will vary depending on the technology and protocols that have been implemented.

Web Server Specific

- Identify known vulnerabilities related to the web server version.
- Assess configuration issues.
- Search for default web server content.
- Identify information leakage.

Authentication

- Find valid login credentials with password grinding.
- Ensure a lockout policy for failed attempts is implemented.
- Assess if a lockout timeout is in place.
- Assess use of generic authentication error messages, preventing username enumeration.
- Bypass authentication with spoofed tokens.
- Bypass authentication with replay of authentication information.
- If SSL is implemented, ensure the certificate is correctly configured.

Input Manipulation

- Find limitations of defined variables and protocol payload, data length and type, construct format.
- Use exceptionally long character strings to find buffer overflow vulnerabilities.
- Inject malicious commands in the SOAP messages.
- Examine unauthorised directory or file access with path and directory traversal.
- Execute remote commands through server-side includes.
- Check validation, ensuring strong type, length, and data-format input.
- Determine the protocol specification of the server or client application.

Session Management

- Determine session management information – number of concurrent sessions, IP-based authentication, role-based authentication, and identity-based authentication
- Estimate session ID sequence and format.
- Determine if the session ID is maintained with IP address information; check if the same information can be retrieved on another machine.
- Replay gathered information to fool services.
- Ensure session variables are kept server side.
- Check if a session timeout is enforced.
- Check that simultaneous logins are not permitted.
- Ensure that the user session is deleted on logout.
- Ensure the client-server communication channel is adequately secured for its intended use.

Service Vulnerabilities

- Check for vulnerability to XML complexity, serialization, and external reference attacks.
- Examine SOAP messages for WSDL/WS-Inspection information disclosure vulnerabilities.
- Check for incorrect use of WS-Security standards.

- Check for transport security weaknesses, including insufficient certification chain validation and weak cipher suite configuration.

2.2 Assessment Team

The following members of staff were assigned to this assessment:

Name	Job Title	Comments
Paul Collett	Principal Security Consultant	Mobile Application Code Review
Ramon Salvador	Managing Security Consultant	Web Services Assessment and Code Review/Mobile Application Retesting
Peter Winter-Smith	Principal Security Consultant	Web Service Retesting
Luke Rogerson	Managing Security Consultant	Project Management
Ian Cornish	Technical Author	Document Creation