Global Information Assurance Certification Paper

Auditing the authentication process of an e-banking application

An auditor perspective

GSNA Practical Assignment Version 3.2, Option 1

by Laurent Kempenaar
2/9/2005

1 ABSTRACT

This report documents an independent audit of an E-banking B-to-B application. The application runs in a J2EE environment. It uses strong authentication, session management and transactional operations on corporate bank accounts.

The focus of the audit is put on the Identification-Authentication-Authorization (IAA) and session management processes.

The audit process is organized in four logical parts, each part serving as input for the following one. Those parts are:

Phase I – Research in audit measurement practices and control :
Three objectives are present in this first phase of the audit :

- Defining precisely the application, its role in business terms and subsequent systems and processes that are part of the audit [WHAT];
- Determining the most significant risks to the application and systems, this will give a precise direction to the auditor on the security implication of the application [WHY];
- Analyzing the current best practices in security audit of the identification, authentication, authorization and session management processes of web application [HOW].

Phase II – Create the audit checklist :
The checklist is created according to the outputs of the phase one (what-why-how).

Phase III – Conducting the audit :
The processes on which the audit focuses have to be tested and this part of the report shows the results and findings of the test. The checklist created during Phase II is used.

Phase IV – Audit report :
Findings, recommendations and conclusion of the audit are summarized in this part.

2 TABLES

TABLE OF CONTENT

TABLE OF TABLES

TABLE OF FIGURES

3 PART I - RESEARCH IN AUDIT, MEASUREMENT PRACTICES AND CONTROLS

3.1 APPLICATION AND SYSTEMS TO BE AUDITED

This paragraph gives an overview of the application and systems to be audited. It also covers the scope and objectives of the audit.

3.1.1 FUNCTIONAL OVERVIEW OF THE APPLICATION

Presentation
The application that is the subject of the audit is known internally to the bank as Octave. This name is a "code name" chosen according to the bank policy in terms of application development. We will refer to Octave as the web application in the rest of this document.

Needs and functionalities
Octave is intended to answer to a growing customer need. More and more companies have expressed the will of using a web-banking application specifically designed for businesses instead of the traditional home-user application.

Companies have different and more precise needs than individual users in terms of banking transaction (at both volume and frequency levels) and more specifically in terms of security.

In a first phase of the project, the business objectives have been determined by the bank using a survey sent to all their business customers. The results of this need analysis are summarized in the table hereunder :

Need	Description
Functional needs	
Payments	This is the traditional payment functionality where users fill-in a web form with all the requested information.
Mass payments	Back-end transaction systems on company often generate an output file containing a lot of transaction. The mass payment functionality would allow the customers to send this file (specifically formatted to a standard banking format) to the bank on a regular basis (for example once a day). This would reduce the time employees spend to enter individual payments.
Account reporting	This would allow the customer to have a global view off all its accounts at the bank and also to group them (by activity for example), to hide some of them.
Financial information	Financial information should be provided on a profile base. Those information should include stock positions, interests rates, market trends and news…

Security needs	
Strong authentication	The authentication mechanism has to be robust and give the customer the certainty that the authentication mechanism cannot be abused.
User profiles	Different profiles should be attributed to the users according to their roles in their company. For example an Executive Assistant should be able to consult an account status but not to enter a transaction. This multi-profile scheme should be transparent to the users.
Non repudiation	Customer should be able to legally prove that a particular transaction has been (or has not been) passed, by who, when…

Table 1 – Needs table

3.1.2 LOGICAL OVERVIEW OF THE APPLICATION

Development and environment
Octave runs in a J2EE environment. This technology has been chosen for its efficiency, portability and also according to other developments and notably the home-user banking application. All the experience gained by the development, security and infrastructure teams in this environment have thus been capitalized and reused in the Octave project.

The J2EE application server hosting Octave is a Websphere Application Server (WAS). Once again, the reason of this choice is to capitalize on existing infrastructure and also to rely on a close and long-term existing relationship with IBM at both technical and human resource levels.

The application itself has been developed using the portlet technology. Portlets have been used as Enterprise Java Beans in the context of Octave.

Components
The following components are present on the infrastructure :

- *Two-level firewalls* :
  Function : the first level firewall is composed of two redundant Nokia-Checkpoint firewalls. Their role is to protect the trusted perimeter from the untrusted internet, from which customer requests will arrive. The first level of firewalls gives access to the DMZ zone, hosting relaying assets (proxies, public webserver)
  The second level of firewalls is composed of two redundant Cisco PIX firewalls. Their role is to protect the private zone from access coming from the DMZ. It protects assets such as authentication servers, LDAP, UserDB and Application server.
  Details : Frontend firewall : Nokia IP530 with Checkpoint Firewall-1 NG Feature Pack 3. Backend firewall : Cisco PIX 535 with Cisco PIX Firewall Software 6.0

- *Load-balancer router* :

Function : this load-balancing router has the responsibility to dispatch the incoming traffic directed on the two Webseal servers. Since a session-cookie is used between the authenticating Webseal and the client, *Stinking labeling* is used on the load-balancer in order to avoid an authenticated user having to re-authenticate in case the Webseal on which he logged-on goes down for any reason.
Details : Cisco Catalyst 6509 with Content Switching Module (CSM) Software Release 4.1.1

- *Webseal reverse proxy* :
  Function : Webseal is a reverse proxy acting also as a basic web server. It also covers the role of intercepting access request, forwarding them to the authentication system and also requests initial pages (logon pages) to the public web server. This is the entry point of the infrastructure.
  The CMAN connector is a piece of software developed on the model of the CDAS of Webseal.
  Details : IBM Tivoli Access Manager Webseal 4.1, SecurIT C-MAN 2.1

- *Reverse proxy with content validation* :
  According to the bank security policy, these proxies are mandatory for accessing any porting of a web server, even a public web servers. Those proxies act as validation of content, but are limited to checking the consistency of the requested URLs.
  Details : Bluecoat Proxy SG 8000

- *Authentication server* (AS) :
  Function: those two redundant servers are the heart of the authentication mechanism. They receive access requests from the Webseal servers and have to validate or invalidate this access based on the users profiles at both global Bank level (LDAP) and application level (UserDB).
  Details : IBM Websphere Application Server 4.0.7, IBM HTTP Server 2.0

- *LDAP slave server* (LDAP_SL):
  Function: the LDAP slave servers contain a duplicated copy of the Bank directory containing all the registered users. The user base is duplicated from the LDAP Hub (see below). LDAP validation is the part of the authentication process that checks that the user is a valid customer of the Bank.
  Details : Sun Iplanet 5.1 Directory Server

- *LDAP Hub* :
  Function: the LDAP Hub is the server contacting the global LDAP server of the bank in order to duplicate its entire directory.
  Details : Sun Iplanet 5.1 Directory Server

- *User database* (UDB):
  Function: this database contains all the registered Bank customer having access to Octave. A valid customer of the Bank that forgot to pay its fees to have access to Octave will be put in this database as “no-access”. The database validation is

the part of the authentication process that checks if the valid customer of the Bank (proven by LDAP check) is allowed to access Octave.
Details : Oracle 8i

- *Network dispatcher* :
  Function: these servers act as load-balancers and has the responsibility to dispatch the traffic directed to the Application Servers, the Authentication servers and the LDAP servers.
  Details : IBM Websphere Edge Server 2.0 with Network Dispatcher

- *Public webserver* (PWS) :
  Function: the public web server hosts only the logon pages application.
  Details : Websphere Application Server 4.0.7 and IBM HTTP Server 2.0

- *Application Server* (WAS) :
  Function: this server hosts the application itself and is the link to the back-end systems of the bank.
  Details : Websphere Portal Server 4.2.2, Websphere Application Server 4.0.7, IBM HTTP Server 2.0

The scheme on the following page presents a logical view of the application and its components:

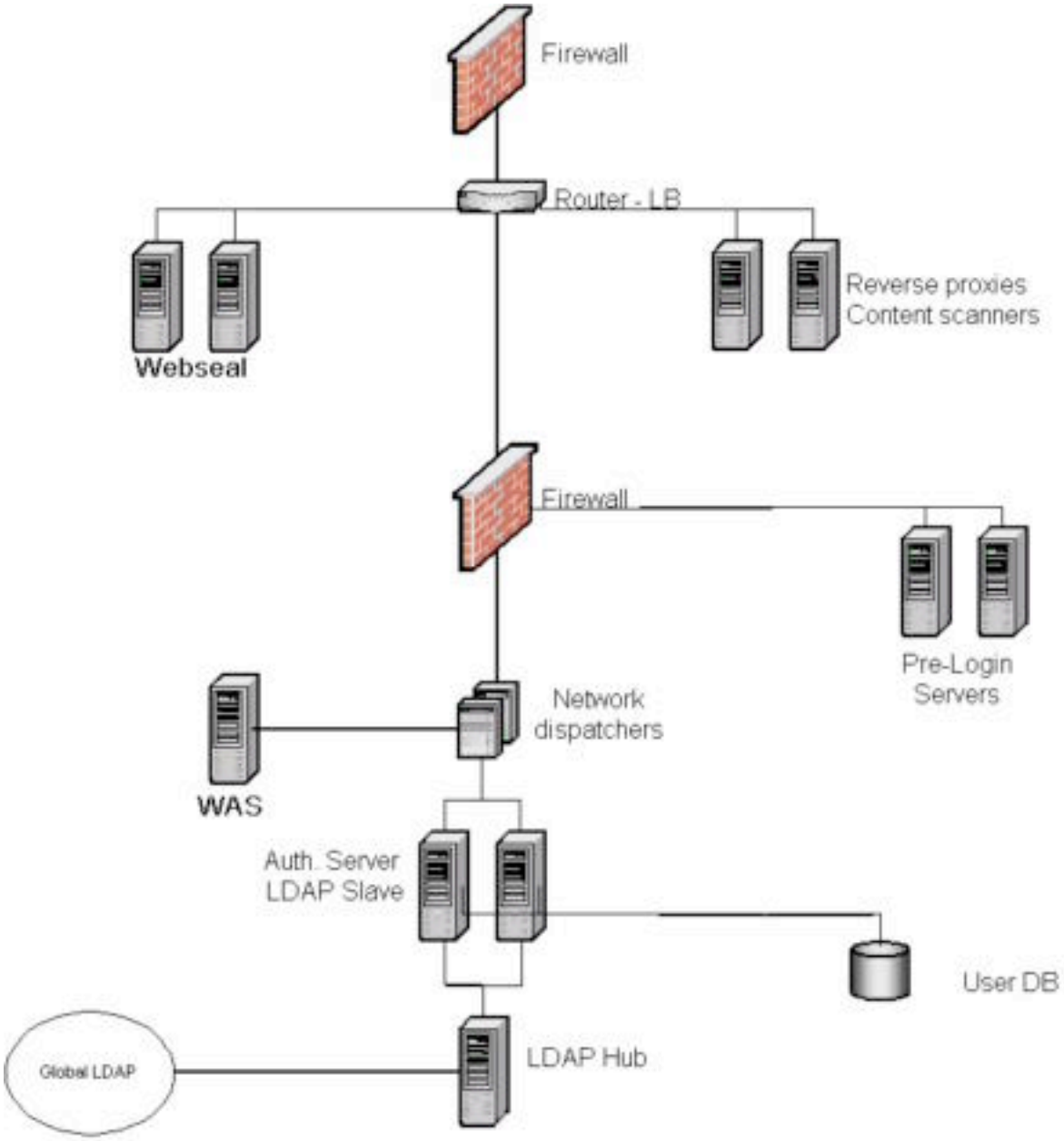


Figure 1 – Logical scheme

3.1.3 SCOPE AND OBJECTIVE OF THE AUDIT

Scope
The audit will focus on the Identification-Authentication-Authorization process of the application. The audit will not address applicative issues.

It must be noted that the audit will not cover the code reviewing part. This issue will be addressed in a future step of the security testing integrated process (see the table below for details on the security testing process at the Bank).

An Integrated Security Testing Framework (ISTF) has been determined together with the customer and integration teams. This framework follows the guidelines of a generic Software Development Life Cycle (SDLC) in place at the Bank. At each level of the deployment, different security tests will have to be done.

Level	Security tests
alpha-a	Stress test of the infrastructure supporting the application
alpha-b	*Test the security of the IAA process*
beta-a	Test the security of the application itself and review of the code
beta-b	Test of the complete infrastructure and operating system security

Table 2 – Security testing by level

The ISTF ensures that all aspects of the security on and around the application are tested during the development life cyle (SDLC).

This document describes the security tests of the alpha-b level. Other functional tests take place concurrently but are not mentioned in this report.

From the integration point of view, the Bank is now ready to implement the infrastructure on which Octave relies on. The application development is almost terminated but some application modules have to be validated from a functional point of view and stress-tested before passing to beta-a level.

Operating systems security audit are also out of the scope of the project.

This means that the precise scope of the audit to be made at this level of Octave development is clearly limited to the IAA process.

3.1.4 METHODOLOGY

The objective being to have a precise view on the security level of the IAA process, various methods will be used to reach it. This paragraph describes the methodology applied in order to reach the objective of the audit defined above in the scope paragraph.

Groups of control objectives
First of all, groups of control objective can be determined. These groups permit us to have a clear and high-level perspective on what has to be done.

The following table presents those groups of control objectives :

Group	Description
Design	The control objectives in this group have as objective to validate the architecture of the IAA process designed for Octave.
Webseal configuration	Being the main element of the IAA process, the Webseal configuration has to be analyzed in details. This group of control objectives is dedicated to validate the configuration Webseal since it is the main technical element of the IAA process.
Behavior testing	Those control objectives analyze the IAA process from a behavior perspective (white box tests).

Table 3 – Groups of control objectives

Auditing techniques
Different auditing techniques will have to be used.

Auditing Technique	Description
Document review	This auditing technique consists in analyzing documentation coming from Octave design and configuration : - Strategical documents : objective of application, business needs… - Tactical documents : technical concepts, description of IAA model, conceptual models… - Operational documents : technical configuration, cookbooks, procedures…
Configuration analysis	This is the traditional auditing of the technical elements constituting the IAA infrastructure.
White box testing	This technique consists in acting as a user of the application trying to connect and analyzing if no abuses are possible.
Interviews	It must be noted that interviewing is a particular auditing method that requires all the auditors "feeling". Its particularity comes from the fact that by nature, **interviews are subjective**. Moreover when interviewing technical and security people, the auditor will often realize that the answers are almost every time in concordance with the **ideal situation** and not the **real situation**. It is up to the auditor knowledge and experience to know how he has to ask questions, in which sequential order and even sometimes with which voice intonation in order to have answers that describes the reality. In other words, the auditor must be "smart" enough and have the necessary technical skills to avoid answers like "what we wanted was…" and go a step further and obtain "actually we had to do it this way…"

Table 4 – Auditing techniques used

These 4 auditing techniques will allow to have a transversal view on the IAA process of Octave, from concepts to implementation and real-world behavior.

Groups control objectives and auditing techniques
A particular auditing technique does not always fits with a group of control objectives. Some techniques will overlap more than one control objective. The following tab summarizes this :

Group of Control objectives	Auditing techniques
Design	Document review Configuration analysis Interviews
Webseal configuration	Document review Configuration analysis White box testing
Behavior testing	White box testing

Table 5 – Control objectives and auditing techniques mapping

3.1.5 SYSTEMS IMPLIED

As stated before, the audit will focus on the IAA process of Octave. In this perspective, it is useful to determine the specific components of the infrastructures directly involved in IAA, even though the Integrated Security Testing Framework (ISTF) of the Bank does not require to test all of them (neither at application, nor at operating system levels).

Name	Details	
	OS	Application
Webseal	MS Windows 2000	**IBM TAM Webseal 4.1** SecurIT C-MAN 2.1
PWS	SUN Solaris 8	Websphere Application Server 4.0.7 IBM HTTP Server 2.0
AS	SUN Solaris 8	IBM Websphere Application Server 4.0.7 IBM HTTP Server 2.0
UDB	SUN Solaris 8	Oracle 8i
LDAP Slave	SUN Solaris 8	Sun Iplanet 5.1 Directory Server
WAS	SUN Solaris 8	Websphere Portal Server 4.2.2 Websphere Application Server 4.0.7 IBM HTTP Server 2.0

Table 6 – Systems involved

As presented in the tab above, only the IBM TAM Webseal application has to be audited according to the Integrated Security Testing Framework (ISTF) of the Bank.

3.1.6 SUMMARY

Object of the audit
The object of the audit is the Octave application. It is an application specifically designed for the business customers of the Bank, answering to their particular needs.

Octave has been designed using Java and runs in a Java 2 Entreprise Environment (J2EE).

Scope
The scope of the audit is clearly focused on the Identification-Authentication-Authorization (IAA) process.
Operating systems and the application itself are out of the scope.

Control objectives and auditing techniques
Three main classes of control objectives have been defined :

- Design
- Webseal configuration
- Behavior testing

From the auditing point of view, four main auditing techniques will be used :

- Document review
- Interviews
- Configuration analysis
- White box testing

3.2 MOST SIGNIFICANT RISKS TO THE APPLICATION AND SYSTEMS

A PRIMER ON RISK ANALYSIS [1]

This primer defines some terms relative to risk analysis in order to fix the concepts besides used words. Often, terms like threat, threat agent, vulnerability and risk are used in such a particular context that they tend to be unclear. Risk analysis is such a precise work that it cannot suffer any misunderstanding on terms.

Threat[1]: threats are any action or inaction that could cause damage, destruction, alteration, loss, or disclosure of assets or that could block access to or prevent maintenance of assets.

- Threat agent : intentional exploitation of vulnerabilities.
- Threat event : accidental exploitations of vulnerabilities.

Vulnerability[1]: the absence of or the weakness of a safeguard or countermeasure.

Risk[1]: the possibility that any specific threat will exploit a specific vulnerability to cause harm to an asset. It is an assessment of probability, possibility, or chance. The more likely it is that a threat event will occur, the greater the risk. Every instance of exposure is a risk.

When written as a formula, risk can be defined as risk = threat + vulnerability. Thus, reducing either the threat agent/event or the vulnerability directly results in a reduction of risk.

Exposure[1]: the fact of being susceptible to asset loss due to a threat. There is the possibility that a vulnerability can or will be exploited by a threat agent or event.

The following figure synthesizes the risk model :

Figure 2 – Risk model

[1] For all definitions in this Primer : Tittel, Ed. Chappel, Mike. Steward, James Michael. CISSP Study Guide. Alameda: Sybex, 2003. 180.

3.2.1 METHODOLOGY OF RISK ANALYSIS

First of all we have to chose the right approach for analyzing the risk involved in the IAA process of Octave. We have the choice between quantitative and qualitative approach.

The quantitative approach results in concrete probability percentage and hard money loss expectancy for each individual risk that would happen[1]. On the opposite, the qualitative approach focuses on considering the happening of a scenario and ranking threats on a scale to evaluate their risk, costs and effects[2].

We do not have financial values of assets at our disposition. Moreover, analyzing the potential financial losses is out of the scope of the mission assigned by the Bank. The risk analysis objective in the context of this audit focuses on determining the technical threats and vulnerabilities that could be harmful to the IAA process of Octave. The right risk analysis method is thus the quantitative one.

In the following pages, we will analyze both threats, assets affected by those threats and the main vulnerabilities of the application. This three-steps methodology allows us to better understand how the application could represent a risk and what elements or vulnerabilities could lead to this situation. In other words after this three-steps risk analysis, we will have an accurate view on the global risk exposure of Octave.

Each of the three steps has its own objective :

Step	Objective
Main threats	Defining the main threats will give us a view on the actions (or inactions) that could represent a risk for Octave. According to the definition of risks (see above), threats are not the only element to evaluate the risk exposure, vulnerabilities are the other elements needed for this.
Assets affected	Before being able to determine vulnerabilities, affected assets must be defined. Once the assets known, it will be easier to define the vulnerabilities on them. Assets are not the systems themselves but must be understood from a more general point of view as “information assets”.
Vulnerabilities	Vulnerabilities are the other key components to define the risk exposure. Defining vulnerabilities will allow us to focus our control objectives on the real vulnerabilities of Octave and consequently on the real risks.

Table 7 – Three-steps risk analysis

[1] Tittel, Ed. Chappel, Mike. Steward, James Michael. CISSP Study Guide. Alameda: Sybex, 2003. 184.

[2] Tittel, Ed. Chappel, Mike. Steward, James Michael. CISSP Study Guide. Alameda: Sybex, 2003. 186.

3.2.2 MAIN THREATS AND CAPACITY TO DAMAGE

Given the definition of the threat (see above), we will now determine the main threats that our system is submitted to.

Threats are presented in the table below using the Confidentiality – Integrity – Availability criteria.

Each threat is described and a likelihood estimation is given. This estimation follows the scheme described hereunder :

Value	Description
Almost certain	The threat is common and its occurrence must be foreseen in the application as a normal situation.
Likely	The threat may not occur with certainty but it must be taken into account.
Possible	The threat may occur in exceptional circumstances.
Unlikely	The threat occurrence is rare and is not taken into account.

Table 8 – Threat likelihood values

This assessment has been carried keeping in mind that the environment will be used by external users (customers having access to Octave and its underlying applications) and internal users (Bank employees having access to the portal, its underlying applications and the administration systems). When answering the questions we looked at the nature of data (confidentiality, sensitivity…), the direction of the information flow (from bank to customer, from customer to bank) and the communication channel used (HTTP, e-mail, FTP).

(see next pages for the threats table)

ID	Threat	Capacity to damage	Likelihood
Confidentiality			
T-C-1	Unauthorized access to data by employees	• Disclose confidential technical information (functioning of Octave, security measures…) • Disclose confidential customer information (authentication data, financial intelligence…) • Theft confidential information	Possible
T-C-2	Unauthorized access to data by external personnel	Same than for T-C-1 except that they may use an external trusted partner-access to gain access to the application. Some external personnel may also be inside the Bank and act just like a normal employee.	Possible
T-C-3	Confidentiality problem with connected systems	• Octave may send confidential information to fed connected systems inside the Bank (mainly authentication data) • Connected systems may have a network path to one of the application component and are thus potentially harmful • Connected systems may not have the same security level than Octave, consequently they can represent an entry point for hackers	Unlikely
T-C-4	Interception of communication links	• Authentication information could be sniffed and reused immediately or at a later time (replay attack) • Authentication information could be intercepted and reused directly by an attacker impersonating a regular user (man-in-the-middle attack) • Ciphered data during the session could be sniffed and used later to launch a cryptanalysis attack	Possible
Integrity			
T-I-1	Input errors	• Customers/inside users may badly enter data when using the application (customers) or its administrative side (inside users)	Almost certain
T-I-2	Operator errors	• An operator could do a mistake and put the application at risk. This threat mainly applies to administering the infrastructure supporting Octave, e.g. misconfiguration of an applicative server a router, deletion of non-repudiation data… • A business operator could badly enter the data of a new customer, e.g. allowing	Likely

		too much rights than necessary	
T-I-3	Manipulation or suppression of input documents	• An internal employee could destroy/corrupt accidentally/intentionally customer transaction data	Possible
Availability			
T-A-1	Day to day system outage	• The day-to-day upgrades and administration of the infrastructure supporting Octave could be inefficient and put the infrastructure at risk. This is the typical case where an information system continuously lose its security level. • Specific vulnerability or incidents may not be handled correctly and render the application less effective.	Likely
T-A-2	Degraded system performance	• In addition to T-A-1, some specific events can degrade the performance of the application, like e.g. a peak in the usage of the application (end of fiscal year for example) or a heavy load on access infrastructure (either due to an external or internal event)	Possible

Table 9 – Main threats and capacity to damage

3.2.3 INFORMATION ASSETS AFFECTED

Octave role is to give companies the possibility to use a web-banking application that perfectly fits their needs. As for every web-banking application, a number of assets are impacted by Octave. Those assets can thus be put at risk by any threat.

The enumeration of the assets impacted hereunder is the result of interviews with business managers and technical managers in the Octave project. Business managers are the ones that drive the functional development of Octave since there are in direct contact with the customers. Technical managers are the ones that are responsible to traduce the customer's needs in technical solutions. Both view are different but necessary in order to have a complete view on the assets impacted by Octave.

ID	Asset	Description
A-1	Banking transaction data	This represents all the transactions done by the customers. Those information are the heart of Octave, any impact on both their confidentiality, integrity and availability would represent a major loss. Those information also comprise non-transactional data (in a strict sense of the term) like e.g. account status, stock values…
A-2	User confidential data	Those data are the identification, authentication and profile information like e.g. a smartcard, a password…
A-3	Non-repudiation data	Non-repudiation data are log traces that could be used in forensics case involving a customer of the Bank or the Bank itself. These are not only technical logs, but functional logs (who did what, when, with which authorization level ?) archived by legal obligation.
A-4	Bank reputation	The Bank reputation is also an asset that could be impacted by Octave in case of problem. This is not a technical asset but is completely part of the "information asset" category.
A-5	Legal liability	This is the other main non-technical asset. The Bank is legally responsible for any misuse of Octave. Legal liability could thus be impacted if it can be proved that the Bank did not reach the minimum security requirements for such an application.

Table 10 – Major information assets impacted

3.2.4 MAIN VULNERABILITIES

This third and last part of the risk analysis consists in describing the main vulnerabilities to the application.

The following table lists the vulnerabilities, their exposure and impact. It also establishes links with the threats and assets analyzed in the first two-steps. The table can be thus seen as a synthesis view of the risk analysis for Octave.

ID	Vulnerability	Exposure	Impact	Linked threat(s)	Affected asset(s)
V-1	Bad logical access controls to data and application	High	High	T-C-1,2	A-1,2,3,4,5
V-2	Bad user and password policy	Medium	High	T-C-1	A-1,2,3,4,5
V-3	Bad access rights revocation policy	Medium	Medium	T-C-1	A-1,2,3,4,5
V-4	Too permissive external access policy	Low	Medium	T-C-2	A-1,2,3,4,5
V-5	Inadequate classification policy for business information	Low	High	T-C-3	A1,4,5
V-6	No protection on transferred data	Low	High	T-C-4	A1,2,4,5
V-7	Too much data transferred	Medium	Low	T-C-4	A1,4,5
V-8	Lack of user training	Low	Medium	T-I-1	A1,4,5
V-9	Lack of customer awareness	Medium	Medium	T-I-1	A1,2,4,5
V-10	No input validation policy	Medium	High	T-I-1,2,3	A-1,2,4,5
V-11	Bad operating procedure	Low	Medium	T-I-2	A-1,2,3,4,5
V-12	Assignment of duties	Low	Low	T-I-3	A-1,2,3,4,5
V-13	Unapplied or bad upgrade policy	Low	Medium	T-A-1	A-1,2,3,4,5
V-14	Bad incident management policy	Low	High	T-A-1	A-1,2,3,4,5
V-15	Unacceptable degradation of system performances	Medium	High	T-A-2	A4,5
V-16	Bad capacity and performance planning procedure	Medium	High	T-A-2	A4,5

Table 11 – Vulnerabilities

Important note : in the table we often refer to the term “policy”, which does refer to organizational tasks that could be considered out of scope for this technical audit project. Although policy can also comprise a number of not-only-organizational tasks like in our case the technical configuration of the Octave infrastructure.
For example, when we talk about “bad user and password policy” (V2), we are not only talking about the existence or the content of a written document, but also the application of this policy on the assets themselves. The term “policy” addresses thus both “paper” existence and accuracy AND technical conformance.

This table can now be used to determine the checklists and to focus on the real risks that Octave is facing.

3.3 CURRENT STATE OF PRACTICE

This part list the resources used to conduct the audit. They have been used in the entire process, beginning with understanding the application and the technology used, building the checklists, conducting the audit and concluding it.

3.3.1 APPLICATION AUDIT SOURCES :

The OWASP project
OWASP stands for Open Web Application Security Project. This project exist since September 2001 and its objective is to be an open-source reference point for every aspects of the security of web applications.

- Curphey, Mark. Van der Stock, Andrew. Collective work. "The OWASP Testing Project". Draft Version 1.0. July 2004. http://www.owasp.org/documentation/tesing/application.html

- Mark Curphey, David Endler, William Hau, Steve Taylor, Tim Smith, Alex Russell, Gene McKenna, Richard Parke, Kevin McLaughlin, Nigel Tranter, Amit Klien, Dennis Groves, Izhar By-Gad, Sverre Huseby, Martin Eizner, Martin Eizner, Roy McNamara. "Owasp Guide to Building Secure Web Applications". Version 1.1.1. September 2002. http://www.owasp.org/documentation/guide/guide_downloads.html

Tutorials on J2EE application
The first tutorial is an online example of the development process of a web application using J2EE technology.

- http://www.tusc.com.au/tutorial/html

The second tutorial comes from the Sun website.

- http://java.sun.com/j2ee/tutorial/1_3-fcs/index.html

Auditnet web site
Auditnet is a website that freely proposes a comprehensive number of auditing checklists. It must be noted that no "editorial control" of any sort is done by the administrator of the site. The checklists found there can be seen as starting points for technical auditors. The document "audit web application checklist" has been used.

- http://www.auditnet.org/docs/Web Based Applications.doc

IBM Redbook
IBM Redbook series is a collection of technical papers that focus on specific IBM technologies and put them in practice through real-world cases study.

- John Ganci, Hinrich Boog, Melanie Fletcher, Brett Gordon, Ashwin Manekar, Normunds Saumanis, Kai Schwidder, Jonas Tingeborn. "Develop and Deploy

a Secure Portal Solution Using Websphere Portal V5 and Tivoli Access Manager V5.1". Version 1. August 2004.
http://www.redbooks.ibm.com/redbooks/SG246325

3.3.2 WEBSEAL AUDIT SOURCES :

Webseal Administrator guide
Webseal administrator guide is the official IBM guide on deploying and managing Webseal and is part of the suite Tivoli Access Manager.

- IBM Corp. "Webseal Administrators Guide". Version 1. November 2003. Search on http://publib.boulder.ibm.com

IBM Redbook
IBM Redbook series is a collection of technical papers that focus on specific IBM technologies and put them in practice through real-world cases study.

- Axel Bücker, Chris Eric Friell, Armando Lemos, Rick McCarty, Jani Perttilä, Dieter Riexinger, Andreas Schmengler. "Enterprise Business Portals with Tivoli Access Manager". Version 1. August 2002.
http://www.redbooks.ibm.com/redbooks/SG246556

3.3.3 BEHAVIOR/WHITE BOX AUDIT SOURCES :

OSSTMM
OSSTM stands for Open Source Security Testing Manual. This project can be compared with OWASP but takes a higher view on the various tests to be done in the general security field.

- http://isecom.securenetltd.com/osstmm.en.2.1.pdf

MISC newspaper
MISC is a French newspaper that stands for Multi-System and Internet Security Cookbook. Its publication rate is once every two months. It presents (in French) various security subjects and deep technical analyzes.

- MISC 11 (January-February, 2004) Les tests d'intrusions (information on http://www.miscmag.com)

4 PART II – AUDIT CHECKLIST

In order for the testers to have a clear and concise view and to provide them a sequential checklist that they can easily follow, the control objectives listed hereunder will follow the objective-methods framework explained before and will be based on the three main area of the audit.

4.1 DESIGN CONTROL OBJECTIVES

Category	Design
Item number	**CO-D-1**
Item title	The complete IAA scheme is in line with industry best practices
References	Personal experience
Risks	The IAA scheme could be poorly designed and lead to an insecure IAA process that would allow to bypass or abuse the system.
Testing procedure	**Step 1 :** Review the documents that describes the IAA process (conceptual and tactical documents). **Step 2 :** Interview the developers and administrators to validate that the information in the documents have been correctly implemented.
Test nature	Objective for Step 1 Subjective for Step 2
Evidence	
Findings	

Category	Design
Item number	**CO-D-2**
Item title	The authorization scheme is clearly defined and in line with the security needs of the customer
References	Personal experience
Risks	The authorization scheme could be inadequate comparing to the expressed business needs (and specifically the “profile” approach) and wouldn’t provide the required granularity.
Testing procedure	**Step 1 :** Review the documents that describes the authorization process (conceptual and tactical documents). **Step 2 :** Interview the developers and administrators to validate that the information in the documents have been correctly implemented.
Test nature	Objective for Step 1 Subjective for Step 2
Evidence	
Findings	

Category	Design

Item number	CO-D-3
Item title	Specific security zones exist in the architecture
References	Personal experience
Risks	If no security zones exist, the security of the data on the infrastructure could be threatened. Security must implement the following concepts : - Security zones are delimited by a mechanism of network filtering (firewall, filtering router); - Data of different levels have to be hosted on different security zones; - No flow can directly go from an untrusted zone to a trusted zone without a relaying mechanism as an intermediary step.
Testing procedure	**Step 1 :** Review the documents describing the architecture at both functional, network and physical levels. **Step 2 :** Check "on the field" if the physical design in the documents is correctly implemented by going onsite and have a look on the physical rooms, switches…
Test nature	Objective
Evidence	
Findings	

Category	Design
Item number	CO-D-4
Item title	Security zones are based on a classification policy
References	Personal experience
Risks	The classification must clearly indicate the security level of the asset or data and consequently, the security zone on which they are hosted. Some assets and data could be hosted on a different security level zone than the one required by their classification.
Testing procedure	**Step 1 :** Review the documents describing the security classification of the assets. **Step 2 :** Review the documents describing the architecture at both functional, network and physical levels. **Step 3 :** Check that the implemented infrastructure is in line with the technical and classification documents.
Test nature	Objective
Evidence	

Findings	

Category	Design
Item number	**CO-D-5**
Item title	Communication between elements of the architecture are correctly filtered
References	Personal experience
Risks	Filtering the logical access from the internet is not enough to guarantee the security of the infrastructure. A poorly implemented filtering policy between the assets inside the infrastructure could also lead to serious security issues. If one component is corrupted, other ones may be corrupted too if the corruption mechanism uses network communication to propagate (e.g. a network DDOS on locally connect segments, a hacker compromising a machine that can join all the others)
Testing procedure	Use a network scanner (e.g. NMAP) and check that the communication allowed (open ports) are only restricted to the ones that need to be open. This check has to be performed from every zones (DMZ) to every other zones (DMZ).

	`NMAP use :`
	`nmap -sS –P0 $DMZ_NET_ADDRESS$/$NETMASK$`
	`(check every TCP port on destination DMZ)`
	
	`nmap -sU –P0 $DMZ_NET_ADDRESS$/$NETMASK$`
	`(check every UDP port on destination DMZ)`

Test nature	Objective
Evidence	
Findings	

Category	Design
Item number	**CO-D-6**
Item title	The modelisation of the zones are based on the “content”, “security registry”, “data storage” and “controller” model
References	OWASP – A guide to building secure web applications – Chapter 5 : architecture – Chapter 5 : architecture
Risks	Web application are multi-tiered and must follow a particular framework. Each zone (tier) is intended to render a particular service. In a typical web application, those zones are the following ones : - Presenter : first part of the presentation layer. This tier is in contact with the client and the security registry zone; - Controller : second part of the presentation layer. This tier is in contact with the data storage; - Security registry : contains the IAA data. Responsible for authenticating the users; - Data storage : all the data related to the application are stored in this zone;

	If this model is not followed, this could put the entire application design at risk regarding the functions performed and data stored.
Testing procedure	**Step 1 :** Review the documents describing the application and its infrastructure. **Step 2 :** Check that the implemented infrastructure is in line with the documents (physical verification in server rooms, localization of switches…)
Test nature	Objective
Evidence	
Findings	

Category	**Design**
Item number	**CO-D-7**
Item title	Data flows between elements of separate zones are ciphered
References	OWASP – A guide to building secure web applications – Chapter 5 : architecture
Risks	If communication are not filtered inside the application, this can cause a confidentiality risk.
Testing procedure	**Step 1 :** Review the documents describing the application and its infrastructure. **Step 2 :** Use a sniffer (e.g. Tcpdump) to check that every communication is ciphered.

	`TCPDUMP use :`
	`tcpdump host $host_IP$`
	`(check every connection from/to a particular host)`
	
	`tcpdump net $dest_net$`
	`(check every connection from/to local host to destination network)`

Test nature	Objective
Evidence	
Findings	

Category	**Design**
Item number	**CO-D-8**
Item title	Flows between “content” and “security registry” are logged and contain information needed to specifically identify a particular user (or source) attempting to connect
References	OWASP – A guide to building secure web applications – Chapter 5 : architecture
Risks	If no traces are kept of connection attempts, it could be impossible to prove any action of any user on the application. It would also be

	difficult to trace brute force or abuse attempts. This is a legal constraint from the local authorities to trace such actions.
Testing procedure	**Step 1 :** Review the documents describing the application and its infrastructure and specifically the non-repudiation process. **Step 2 :** Check in the non-repudiation database that the requested logs are correctly stored.
Test nature	Objective
Evidence	
Findings	

Category	**Design**
Item number	**CO-D-9**
Item title	Traces of authentication attempts flows (see CO-D-6) are signed, time stamped, ciphered and securely archived on read-only storage
References	OWASP – A guide to building secure web applications – Chapter 5 : architecture
Risks	The risk involved is the same than for CO-D-8 (see above). While CO-D-8 concerns only the "functional" non-repudiation, this particular control (and the measures to be implemented) are needed for any legal action that could use the traces kept by the application. This is a legal constraint from the local authorities that the traces kept respond to such conditions.
Testing procedure	**Step 1 :** Review the documents describing the application and its infrastructure and specifically the non-repudiation process (see CO-D-8). **Step 2 :** Check the existence of a read-only storage for the traces. **Step 3 :** Check in the non-repudiation database that the requested logs are stored with the appropriate legal measures.
Test nature	Objective
Evidence	
Findings	

Category	**Design**
Item number	**CO-D-10**
Item title	Only one path is possible between a client and any elements of the architecture
References	Enterprise Business Portals with IBM Tivoli Access Manager

Risks	If multiple path are allowed, it may be possible for a malicious user to bypass security measures or in a more general way, to try to gather information on the application infrastructure.
Testing procedure	**Step 1 :** Review the documents describing the application and its infrastructure. **Step 2 :** Use network discovery tools (NMAP, Traceroute, Firewalk) to ensure that only one network-level entry-point to the application is allowed.

	`NMAP use :`
	`nmap -sS -PO $DMZ_NET_ADDRESS$/$NETMASK$`
	`(same command than for CO-D-5 launched from the internet on all DMZ)`

Test nature	Objective
Evidence	
Findings	

Category	**Design**
Item number	**CO-D-11**
Item title	Data entry validation (at web application level) is conform to the "accept only known valid data" policy
References	OWASP – A guide to building secure web applications – Chapter 10 : data validation
Risks	From a general point of view, if no input data validation policy is applied, the application could be vulnerable to the following attacks (amongst others) : - Cross-site scripting, - SQL injection , - Direct OS commands input, - … Different mitigation techniques exist for each of those risks. Above the technical mitigation techniques, a clear mitigation policy should exist. Different approach are possible : - Accept only known valid data; - Reject known bad data; - Sanitize bad data. The most efficient and less resource-consuming policy is the "accept only known valid data". Note : this data input validation policy applies for the entire application, not only the IAA process. Since our study focuses on IAA, we will only consider input validation at logon time.
Testing	**Step 1 :**

procedure	Review the documents describing the application and its infrastructure and specifically the input validation part. **Step 2 :** See the behavior testing control objective part.
Test nature	Objective
Evidence	
Findings	

4.2 WEBSEAL CONFIGURATION CONTROL OBJECTIVES

Category	Webseal configuration
Item number	**CO-WSC-1**
Item title	Webseal is chrooted
References	Webseal Administrator's Guide Personal work
Risks	If Webseal does not run chrooted and is compromised, the aggressor would have access to the entire filesystem.
Testing procedure / Compliance criteria	In the `webseald-internet.conf` file, check that the `server-root` parameter is added under the `[server]` stanza.
Test nature	Objective
Evidence	
Findings	

Category	Webseal configuration
Item number	**CO-WSC-2**
Item title	Webseal does not run with administrative rights
References	Webseal Administrator's Guide Personal work
Risks	If Webseal runs with administrative rights and is compromised, the aggressor would have administrative access on the system.
Testing procedure / Compliance criteria	Check that the Webseal process does not run with administrative rights. **Step 1 :** In the `webseald-internet.conf` file, identify the names of the `unix-user` and `unix-group` parameter in the `[server]` stanza. **Step 2 :** Connect to the machine with Administrator rights. In the `user and group manager` of the Windows 2000 Server, check that neither the user, nor the group are member of the `Administrator` group.
Test nature	Objective
Evidence	
Findings	

Category	Webseal configuration
Item number	**CO-WSC-3**
Item title	The root directory for web server is only accessible by the Webseal user and an administrator
References	Webseal Administrator's Guide Personal work
Risks	The directory could be corrupted by a user or process that has

	nothing to do in this area.
Testing procedure / Compliance criteria	Collect the NTFS rights applied on the root-directory of Webseal. **Step 1 :** Locate the `server-root` parameter in the `[server]` stanza (cfr. OC-WSC-1) **Step 2 :** Connect to the machine with Administrator rights. In the Windows Explorer, right click on the root-directory, collects its rights and make sure only required users and groups are allowed to access it.
Test nature	Objective
Evidence	
Findings	

Category	**Webseal configuration**
Item number	**CO-WSC-4**
Item title	Webseal limits the size of the post request to read
References	Webseal Administrator's Guide Personal work
Risks	Webseal could cache more data than it is able to read and create a denial of service (DOS) situation.
Testing procedure	In the `webseald-internet.conf` file, check that `request-max-cache` and `request-body-max-read` parameters in the `[server]` stanza are short enough.
Test nature	Objective
Evidence	
Findings	

Category	**Webseal configuration**
Item number	**CO-WSC-5**
Item title	The `dynurl` mapping file is protected and only accessible to Webseal user and an administrator
References	Webseal Administrator's Guide Personal work
Risks	Corruption of the mapping file by an attacker to allow any URL request.
Testing procedure	Collect the NTFS rights applied on the `dynurl` file. **Step 1 :** Locate the `dynurl-map` parameter in the `[server]` stanza. **Step 2 :** Connect to the machine with Administrator rights. In the Windows Explorer, right click on the `dynurl` file, collects its

	rights and make sure only required users and groups are allowed to access it.
Test nature	Objective
Evidence	
Findings	

Category	**Webseal configuration**
Item number	**CO-WSC-6**
Item title	The server identity is suppressed in answers
References	Webseal Administrator's Guide Personal work
Risks	A malicious user could identify precisely the Webseal server (information gathering about the infrastructure).
Testing procedure	In the `webseald-internet.conf` file, check that the `suppress-server-identity` parameter is activated with the `yes` parameter under the `[server]` stanza.
Test nature	Objective
Evidence	
Findings	

Category	**Webseal configuration**
Item number	**CO-WSC-7**
Item title	The Compare mode is not used by LDAP for authenticating users
References	Webseal Administrator's Guide Personal work
Risks	Using the Compare mode is faster but insecure comparing to a real Bind operation.
Testing procedure	In the `webseald-internet.conf` file, check that the `auth-using-compare` parameter is deactivated with the `no` value under the `[ldap]` stanza.
Test nature	Objective
Evidence	
Findings	

Category	**Webseal configuration**
Item number	**CO-WSC-8**
Item title	The connection to the LDAP server is SSL-enabled
References	Webseal Administrator's Guide Personal work
Risks	All the communication between LDAP and Webseal are in clear text by default.
Testing procedure	In the `webseald-internet.conf` file, check that the `ssl-enabled` parameter is activated with the `yes` value under the `[ldap]` stanza.
Test nature	Objective

Evidence	
Findings	

Category	**Webseal configuration**
Item number	**CO-WSC-9**
Item title	Check KEY DB file (and password storage)
References	Webseal Administrator's Guide Personal work
Risks	The key database file could be compromised and consequently all the mechanisms of client and server SSL authentication could be compromised.
Testing procedure	In the `webseald-internet.conf` file, check the following parameters in the `[ssl]` stanza : **1) Key database default filename :** Check that the name of the key database file is not the default name. Localize the `webseal-cert-keyfile` parameter and check that its value is not `pdsrv.kdb`. **2) Key database file access rights :** Connect to Webseal with administrative rights, use the Windows explorer, locate the key database file (using findings of the previous test), collects its rights and make sure only required users and groups are allowed to access it. **3) Key database password directly in the conf file :** Check that the `webseal-cert-keyfile-pwd` parameter is deactivated. **4) Key database password in a stash file :** Check that the `webseal-cert-keyfile-stash` parameter exists and if yes check its value (path).
Test nature	Objective
Evidence	
Findings	

Category	**Webseal configuration**
Item number	**CO-WSC-10**
Item title	Check SSL KEY file (and password storage)
References	Webseal Administrator's Guide Personal work
Risks	The SSL key file could could be compromised and consequently all the mechanisms of client and server SSL authentication could be compromised.
Testing procedure	In the `webseald-internet.conf` file, check the following parameters in the `[ssl]` stanza :

	1) SSL key file access rights are restrictive : Locate the ssl-keyfile parameter and note its value. Then connect to Webseal with administrative rights, use the Windows explorer, locate the SSL key file, collects its rights and make sure only required users and groups are allowed to access it. **2) SSL key file password is not directly in the conf file :** Check that the `ssl-keyfile-pwd` parameter is deactivated. **3) SSL key file password is in a stash file :** Check that the `ssl-keyfile-stash` parameter exists and if yes check its value (path).
Test nature	Objective
Evidence	
Findings	

Category	**Webseal configuration**
Item number	**CO-WSC-11**
Item title	Check SSL type and timeout
References	Webseal Administrator's Guide Personal work
Risks	SSL type could be weak and would not provide the safest possible encryption mechanism.
Testing procedure	In the `webseald-internet.conf` file, check the following parameters in the `[ssl]` stanza : **1) SSL type accepted :** Check that the `disable-ssl-vX=` parameter is activated and only allows SSLv3 (in other words, the parameter `disable-ssl-v3` should be the only one to be deactivated with the `no` parameter) **2) SSL timeout :** Check that the `ssl-vX-timeout` parameter is configured with reasonable parameters.
Test nature	Objective
Evidence	
Findings	

Category	**Webseal configuration**
Item number	**CO-WSC-12**
Item title	Check the CRL mechanism
References	Webseal Administrator's Guide Personal work
Risks	The CRL mechanism could be compromised and render the

	authentication mechanism less trustworthy.
Testing procedure	In the `webseald-internet.conf` file, check the following parameters in the `[ssl]` stanza : **1) Activation of the CRL mechanism :** Check that the `gsk-crl-cache-size` and `gsk-crl-cache-entry-lifetime` are not null. **2) Definition of a LDAP server :** Check the name of the LDAP server in the `crl-ldap-server` parameter. **3) LDAP user used to connect is not null :** Check that the `crl-ldap-user` parameter is not null.
Test nature	Objective
Evidence	
Findings	

Category	**Webseal configuration**
Item number	**CO-WSC-13**
Item title	Check junction list and parameters
References	Webseal Administrator's Guide Personal work
Risks	The junction list contains all the definition of the connection between Webseal and the other servers that are part of the Octave application. If the file is compromised, all the communication inside the application are compromised too.
Testing procedure	**1) Access rights on the junction DB file :** Collect the NTFS rights applied on the junction database of Webseal. **Step 1 :** Locate the `junction-db` parameter in the `[junction]` stanza of the `webseald-internet.conf` file. **Step 2 :** Connect to the machine with Administrator rights. In the Windows Explorer, right click on the junction DB file, collects its rights and make sure only required users and groups are allowed to access it **2) Junction type :** In the junction file (located above) check that HTTPS is used instead of HTTP. **3) Worker threat limit is not too heavily configured :** In the webseald-internet.conf file, check that the `worker-thread-hard-limit` and `worker-thread-hard-limit` parameters are not too high (above 80 %).

Test nature	Objective
Evidence	
Findings	

Category	**Webseal configuration**
Item number	**CO-WSC-14**
Item title	Authentication mechanism
References	Webseal Administrator's Guide Personal work
Risks	Failure in the authentication mechanism could allow an attacker to abuse the authentication mechanism. Even a well-intentioned user could be weakly authentified without knowing it.
Testing procedure	In the `webseald-internet.conf` file, check the following parameters in the `*authentication*` chapter : **1) Locate the authentication method(s) definition :** Locate the following stanza : `[ba]/[forms] /[token]/[certificate] /[http-headers]/[ipaddr]` **2) Check activation of authentication methods :** For each stanza (see above), check whether the authentication method is used or not (check if value is a valid parameter enumerated in the explanation of the configuration file or if the value is `none`). **3) Check active method(s) validity :** Check that the authentication method(s) activated are valid method(s) for the application. **4) Check authentication mechanism and libraries :** In the [authentication-method] check the authentication methods used and the associated shared libraries.
Test nature	Objective
Evidence	
Findings	

Category	**Webseal configuration**
Item number	**CO-WSC-15**
Item title	Check Session management
References	Webseal Administrator's Guide Personal work
Risks	A non-secure session management could allow an attacker to reuse the user credentials and is thus at risk concerning the authentication mechanism.
Testing procedure	In the `webseald-internet.conf` file, check the following parameters :

	1) Check that SSL-ID is not used as a state conservation mechanism : Locate the `ssl-id-session` parameter in the `[ssl-client-session]` stanza and check that the parameter is not activated. **2) The session cookie is resent with every client response :** Locate the `resend-webseal-cookies` parameter in the `[sending-session-cookie]` stanza and check that the parameter is activated (value is `yes`). **3) A failover mechanism is used if the authenticating Webseal is unavailable :** Locate the failover-auth parameter in the [failover] stanza and checks that the parameter is activated and only HTTPS is allowed. **4) The failover key is securely stored :** Collect the NTFS rights applied on the key used to encrypt the failover cookie. **Step 1 :** Locate the `failover-cookies-keyfile` parameter in the `[failover]` stanza. **Step 2 :** Connect to the machine with Administrator rights. In the Windows Explorer, right click on the file identified at Step 1, collects its rights and make sure only required users and groups are allowed to access it.
Test nature	Objective
Evidence	
Findings	

4.3 BEHAVIOR TESTING CONTROL OBJECTIVES

Category	Behavior testing
Item number	**CO-BT-1**
Item title	Identifier, card code and data provided to client are anonymous
References	Personal work
Risks	If too much information are provided, an attacker could use them to identify some layer of the business logic.
Testing procedure	Review the information provided by the bank to log in the application and check that no direct link can be established between one of this information and the client name. Tests have to be made for application without samrtcard (username and password) and application with smartcard.
Test nature	Objective
Evidence	
Findings	

Category	Behavior testing
Item number	**CO-BT-2**
Item title	Username format prevents guessing existing usernames
References	OWASP guide Testing Web Application Exposed Personal work
Risks	If too much information are given on the username construction, an attacker could try to lock a high number of account (by flooding the application with existing usernames without valid credentials, this would be a DOS situation) or even to try to connect to the application.
Testing procedure	Review the information provided (especially any kind of user identification) and check that the way of constructing usernames cannot be guessed too easily. Tests have to be made for application without smartcard (username and password) and application with smartcard.
Test nature	Objective
Evidence	
Findings	

Category	Behavior testing
Item number	**CO-BT-3**
Item title	Authentication strength (secure transport and account lockout policy) is aligned with current industry good practices
References	OWASP guide Testing Web Application Exposed Personal work
Risks	An attacker could sniff a user credentials if the transport is not secured or try to brute force an account if no account lockout policy is

	active.
Testing procedure	**Step 1 :** Try to connect to the application using standard web browser with the given credentials and check the transport method. **Step 2 :** Try to connect repetitively to the application using standard web browser with invalid credentials until the account is locked-out. Tests have to be made for application without smartcard (username and password) and application with smartcard.
Test nature	Objective
Evidence	
Findings	

Category	**Behavior testing**
Item number	**CO-BT-4**
Item title	Simultaneous connection to the application are not possible
References	OWASP guide Testing Web Application Exposed Personal work
Risks	Simultaneous connection can allow an attacker to replay immediately intercepted credentials and to log in concurrently to the application. Also the fact of allowing multiple connections increases the risk of poorly terminated session that could be replayed later.
Testing procedure	Try to connect the same account simultaneously using standard web browser. Tests have to be made for application without smartcard (username and password) and application with smartcard.
Test nature	Objective
Evidence	
Findings	

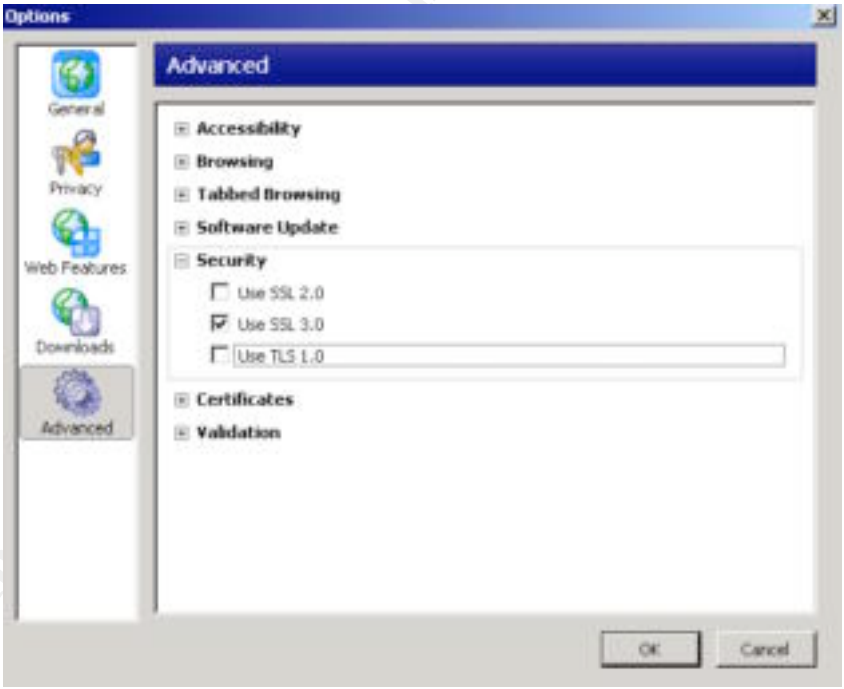
Category	**Behavior testing**
Item number	**CO-BT-5**
Item title	Try to abuse the web server and the local application
References	OWASP guide Testing Web Application Exposed Personal work
Risks	Replaying the authentication flow could allow an aggressor to compromise the entire authentication process.
Testing procedure	Try different techniques to abuse both local and remote ends : - XSS - Connection on the local application Use a local relaying proxy to keep a detailed trace of every request and reply (e.g. Odysseus Proxy).

	Tests have to be made for application without smartcard (username and password) and application with smartcard.
Test nature	Objective
Evidence	
Findings	

Category	**Behavior testing**
Item number	**CO-BT-6**
Item title	A customer can not access information of other customers
References	OWASP guide Testing Web Application Exposed Personal work
Risks	An attacker could abuse the system and have unallowed access to confidential data.
Testing procedure	Try different techniques to abuse both local and remote ends : - XSS - Webserver abuse - … Use a local relaying proxy to keep a detailed trace of every request and reply (e.g. Odysseus Proxy). Tests have to be made for application without smartcard (username and password) and application with smartcard.
Test nature	Objective
Evidence	
Findings	

Category	**Behavior testing**
Item number	**CO-BT-7**
Item title	Cookies are not persistent
References	OWASP guide Testing Web Application Exposed Personal work
Risks	Persistent cookies can reveal information on the application to an attacker that would have an access on the client workstation. Moreover, cookie replay could be attempted to have access to the application.
Testing procedure	Analyze the locally stored cookie on the client workstation. Use a local relaying proxy to keep a detailed trace of every request and reply (e.g. Odysseus Proxy). Use a browser that allows to manage cookies easily and consult the cookie history too (e.g. Netscape, Mozilla). Tests have to be made for application without smartcard (username and password) and application with smartcard.
Test nature	Objective

Evidence	
Findings	

Category	**Behavior testing**
Item number	**CO-BT-8**
Item title	The application only accepts strong SSL connections (i.e. 128 bits)
References	OWASP guide Testing Web Application Exposed Personal work
Risks	Weak SSL connection could be cracked by an attacker.
Testing procedure	Try to connect to the application forcing the client browser to use all types of SSL encryption (especially weak ones). Use a standard web browser and change its configuration so that it only accept a particular SSL type. Example on Firefox :  Tests have to be made for application without smartcard (username and password) and application with smartcard.
Test nature	Objective
Evidence	
Findings	

Category	**Behavior testing**
Item number	**CO-BT-9**
Item title	Non-persistent cookie in memory is secured
References	OWASP guide Testing Web Application Exposed

	Personal work
Risks	Cookies stored in volatile memory are said to be non-persistent cookies. Those non-persistent cookies, although they are more secured than the ones written on non-volatile memory (hard disk) due to their timely nature. Even if the cookie is not written, it can be read and a common mistake is that the cookies stored in memory contains clear-text information that can be gathered and used to compromise the security of the application/session.
Testing procedure	Use a memory dump analyzer to gather RAM memory information and analyze the output. Check that the following information are not be present in the memory dump : - clear text information on user credentials - various identification information on the client/application…
Test nature	Objective
Evidence	
Findings	

Category	**Behavior testing**
Item number	**CO-BT-10**
Item title	Cookie manipulation is prevented
References	OWASP guide Testing Web Application Exposed Personal work
Risks	Cookie manipulation is often used to abuse or to generate a non-conform behavior of a web application.
Testing procedure	Use a local proxy (e.g. Odysseus Proxy) on the test workstation to intercept all the cookies sent to the application with the HTTPS requests. Once the information (cookies) are intercepted, try to manipulate some parameters (id level, username…) and send them to the application. Monitor the behavior of the application.
Test nature	Objective
Evidence	
Findings	

5 PART III – CONDUCTING THE AUDIT

5.1 DESIGN CONTROL OBJECTIVES

Category	Design
Item number	**CO-D-1**
Item title	The complete IAA scheme is in line with industry best practices
References	Personal experience
Risks	The IAA scheme could be poorly designed and lead to an insecure IAA process that would allow to bypass or abuse the system.
Testing procedure	**Step 1 :** Review the documents that describes the IAA process (conceptual and tactical documents). **Step 2 :** Interview the developers and administrators to validate that the information in the documents have been correctly implemented. In order to analyze completely the IAA scheme as stated in the Control Objective, we have to constitute what is called a login sequence. No document were found at the Bank to completely describe this process. Only parcels of information have been found. Our work consists thus in putting them all together by reading and interviewing people working on them.
Test nature	Objective for Step 1 Subjective for Step 2
Evidence	Since a lot has to be said for this Control Objective, the layout of the table has been modified to allow a clear presentation.

A PRIMER ON IAA

Identification is the fact of claiming to be a particular user.
For example in a traditional userid and password authentication scheme, writing down the userid is the identification.

Authentication is the fact of proving that we are with certainty this user.
While Identification is a "client-side" process (the user has to write its userid), authentication is a mixed process between the client and the server. In our traditional userid/password example, the client has to write its password and the server will validate it. Both writing the password and checking its validity compose the process of authentication.

Authorization happens once authentication is successful. It consists in determining the level of trust the user has to be granted inside the application. Authorization determines the action the user is allowed to perform in a specified area (application, information system…)

First of all, two identification authentication method are used by Octave :

- UserID/Password combination;
- Smartcard access.

The authentication method is chosen by the user at logon time. It must be noted that according to customer specific needs, some users may be forced to log-on with a specific method.
For example, a company could chose to force its employees to authenticate with a Smartcard. The normal case is that each application is available with both authentication methods but the chosen authentication method gives different authorization level.

The first steps of both UserID/Password and Smartcard authentication are an initial request to an application. It must be admitted here that either the user typed in the correct URL to directly access the application, or that he followed a link on the Bank website that conducted him to the application request URL.

The following description of the two login sequences (UserID/Password and Smartcard) is very important in the understanding of the IAA mechanism. It is in fact the complete IAA mechanism explained. It is of course specific to Octave but can be transposed (and often simplified) for a lot of web applications.

Scenario 1 : UID/PWD authentication
ACCESS REQUEST TO AN APPLICATION
1. Initial request of the client (https://obfuscatedurl.com/octave+$application$)
IDENTIFICATION
2. Webseal asks PLS for authentication methods for the requested application (contained in URL) [Webseal keeps the initial request (application) in cache for future use] [Webseal actually hosts only a frameset and submit this page to PLS that has to feed the page]
3. PLS sends the constructed page with the authentication methods to the browser [PLS has two mapping files : URL->Application + Application->AuthMethod]
4. The client choose an authentication method and returns it to PLS
AUTHENTICATION
5. PLS sends the authentication page (password specific) to browser
6. Client submits the page with filled fields (pkmslogin script)
7. Webseal intercepts the credentials and sends them to CMAN
8. CMAN transforms the received data in XML and submit them to AS over HTTPS
9. AS asks WSA_DB for UID validity (enforced login policy)
10. AS asks LDAP_SL for credentials validity (validate authentication credentials)
11. Build actor context
12. Build profile list (WSA_DB)
CASE 1 : ONLY ONE PROFILE FOR THE USER
13. authentication timestamp written in DB
14. AS sends data to CMAN
15. CMAN checks Actor Type (internal/external) (internally coded)
16. CMAN passes UID to CDAS/Webseal
AUTHORIZATION
17. CDAS/Websal checks ACL for this actor (LDAP connection)
CASE 2 : MORE THAT ONE PROFILE FOR THE USER
18. AS sends data to CMAN
19. CMAN checks Actor Type (internal/external) (internally coded)
AUTHORIZATION

20. CDAS/Websal checks ACL for this actor (LDAP connection) 21. The profile selection page is constructed and sent to the client [Webseal constructs a HTTP header containing all the profiles for this actor] [Webseal sends the HTTP header and the frameset of the page to the client] [The client connects PLS to construct the profiles list to show in the page] 22. The client sends the profile selection (pkmslogin script) to CDAS/Webseal 23. Webseal sends the profile to CMAN and CMAN sends the profile to AS 24. AS checks that user is allowed to use this profile with the credentials on the LDAP
ACCESS TO THE APPLICATION
25. The initial request of the client (cached at step 2) is taken back by webseal and the connection to the portal (and application) is done

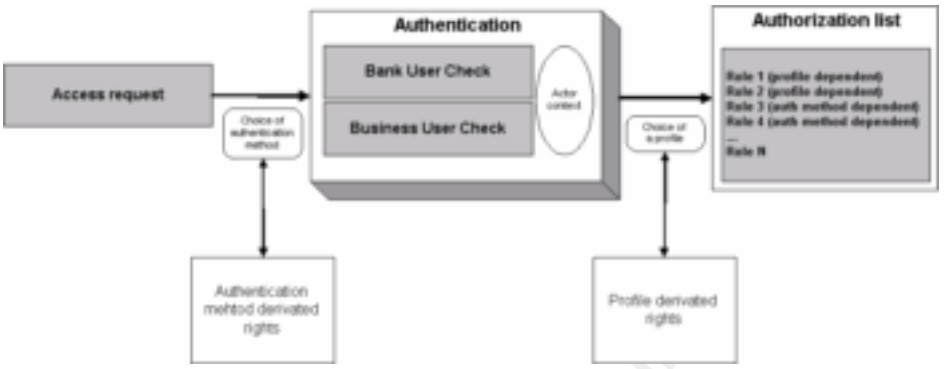
The Smartcard login sequence is almost the same than the UID/PWD one. The only changes take place at steps 9 and 10 of the authentication since the validation process is not the same.

Scenario 2 : Smartcard authentication
ACCESS REQUEST TO AN APPLICATION
Same steps than for Scenario 1
IDENTIFICATION
Same steps than for Scenario 1
AUTHENTICATION
5. PLS sends the authentication page (password specific) to browser 6. Client submits the page with filled fields (pkmslogin script) 7. Webseal intercepts the credentials and sends them to CMAN 8. CMAN transforms the received data in XML and submit them to AS over HTTPS **9. AS asks WSA_DB for UID validity (enforced login policy)** **10. AS asks LDAP_SL for credentials validity (validate authentication credentials)** 11. Build actor context 12. Build profile list (WSA_DB)
CASE 1 : ONLY ONE PROFILE FOR THE USER
Same steps than for Scenario 1
AUTHORIZATION
Same step than for Scenario 1
CASE 2 : MORE THAT ONE PROFILE FOR THE USER
Same steps than for Scenario 1
AUTHORIZATION
Same steps than for Scenario 1
ACCESS TO THE APPLICATION
Same step than for Scenario 1

Findings	No weakness was found in the login sequence for both UID/PWD and SMARTCARD authentications. The login sequences described above are present for a better comprehension of the IAA process of Octave application. It must be noted that the authentication mechanism goes even further than the commonly admitted best practices by implementing a double-check on the customers : - Bank User Check - “Is the user a valid customer of the Bank ?” - Business User Check - “Is the known bank customer allowed to access the Octave application ?”

	The second question (Business User Check) can be seen as a first step of the authorization process, although for coherence reason, we prefer to consider it as the second step of the authentication process. **Control objective PASSED**

Category	Design
Item number	**CO-D-2**
Item title	The authorization scheme is clearly defined and in line with the security needs of the customer
References	Personal experience
Risks	The authorization scheme could be inadequate comparing to the expressed business needs (and specifically the "profile" approach) and wouldn't provide the required granularity.
Testing procedure	**Step 1 :** Review the documents that describes the authorization process (conceptual and tactical documents). **Step 2 :** Interview the developers and administrators to validate that the information in the documents have been correctly implemented.
Test nature	Objective for Step 1 Subjective for Step 2
Evidence	**Authorization scheme** Understanding the login sequences is halfway to success when auditing an authentication mechanism. The other key success factor in such a project is understanding the authentication scheme. Together, login sequence and authorization scheme cover the entire scope of IAA process. This model has been developed to correctly answer to the user profile need expressed by the business customers. The main idea sustaining this model is that the rights according to a user depends on both the authentication method and the profile. Both of them are chosen by the user itself. When a user requests access to an application, he is asked to chose an authentication method. Authentication then checks that the user is known to the bank (Bank User Check) and to the application itself (Business User Check). Once the two checks are performed, the system looks in what is called the Actor's context. This context contains the list of the profiles associated with the user. Authentication process checks that the profile chosen is valid and then constructs the authorization list. The authorization list is constructed by associating various roles to the authenticated user. Those roles come from two distinct sources : - The authentication method ; - The profile

	The scheme hereunder illustrates the authorization model : 
Findings	The authorization profile responds to the customers needs and is conform to the best practices. **Control objective PASSED**

5.2 WEBSEAL CONFIGURATION CONTROL OBJECTIVES

Category	Webseal configuration
Item number	**CO-WSC-1**
Item title	Webseal is chrooted
References	Webseal Administrator's Guide Personal work
Risks	If Webseal does not run chrooted and is compromised, the aggressor would have access to the entire filesystem.
Testing procedure / Compliance criteria	In the `webseald-internet.conf` file, check that the `server-root` parameter is added under the `[server]` stanza.
Test nature	Objective

Evidence	Extract of the `webseald-internet.conf` file :
	`[server]`
	`#Root directory for the webserver`
	`server-root = D:/Program Files/Tivoli/PDWeb/www-internet`

Findings	This parameter should be present to indicate a path that would be defined as the root path of Webseal. Consequentially, all the relatives paths expressed in the configuration file are relative to this root. In other words, Webseal has only access to a limited part of the filesystem located above the root directory and not to the entire filesystem of the system. The relative path parameter is correctly defined in the `[server]` stanza. It appears thus that the server is chrooted and does not share the same filesystem than the operating system. **Control objective PASSED**

Category	Webseal configuration
Item number	**CO-WSC-3**
Item title	The root directory for web server is only accessible by the Webseal user and an administrator
References	Webseal Administrator's Guide Personal work
Risks	The directory could be corrupted by a user or process that has nothing to do in this area.
Testing procedure / Compliance criteria	Collect the NTFS rights applied on the root-directory of Webseal. **Step 1 :**

	Locate the `server-root` parameter in the `[server]` stanza (cfr. OC-WSC-1) **Step 2 :** Connect to the machine with Administrator rights. In the Windows Explorer, right click on the root-directory, collects its rights and make sure only required users and groups are allowed to access it.
Test nature	Objective
Evidence	**Step 1 :** Extract of the `webseald-internet.conf` file :

	`[server]`
	`#Root directory for the webserver`
	`server-root = D:/Program Files/Tivoli/PDWeb/www-internet`

	Step 2 : The `Administrators` group is listed in the access list with `full control` rights together with the `ivmgr` group.
Findings	The complete `Administrators` group should not have access to this directory. It is necessary to have at least one member of the `Administrators` group, not the entire group. **Control objective FAILED**

Category	**Webseal configuration**
Item number	**CO-WSC-4**
Item title	Webseal limits the size of the post request to read
References	Webseal Administrator's Guide Personal work
Risks	Webseal could cache more data than it is able to read and create a denial of service (DOS) situation.
Testing procedure	In the `webseald-internet.conf` file, check that `request-max-cache` and `request-body-max-read` parameters in the `[server]` stanza are short enough.
Test nature	Objective
Evidence	Extract of the `webseald-internet.conf` file :
	`request-body-max-read = 15000` `request-max-cache = 20000`
Findings	Those two parameters are used by Webseal when it intercepts the first request of a non-authenticated client. Webseal is designed to cache the initial resource request, and launch the authentication process. Once authentication is done, Webseal reads its cache and redirects the client to its original (cached) request.

	`Request-max-cache` is the maximum size of the entire HTTP request (body + header + footer + potential cookie…) `Request-body-max-read` determines the amount of data that Webseal can read for a POST request.. Two parameters are important : 1) `Request-max-cache` must be larger than `request-body-max read` (in order to cache the entire request in a POST method) This is the case in the findings. 2) The values of both parameters must be reasonable. This is not the case since values would allow a request of 20K and a body of 15K (nearly 20000 and 15000 characters) **Control objective FAILED**

Category	Webseal configuration
Item number	**CO-WSC-5**
Item title	The `dynurl` mapping file is protected and only accessible to Webseal user and an administrator
References	Webseal Administrator's Guide Personal work
Risks	Corruption of the mapping file by an attacker to allow any URL request.
Testing procedure	Collect the NTFS rights applied on the `dynurl` file. **Step 1 :** Locate the `dynurl-map` parameter in the `[server]` stanza. **Step 2 :** Connect to the machine with Administrator rights. In the Windows Explorer, right click on the `dynurl` file, collects its rights and make sure only required users and groups are allowed to access it.
Test nature	Objective
Evidence	**Step 1 :** Extract of the `webseald-internet.conf` file :
	`dynurl-map = lib/dynurl.conf`
	Step 2 : The `Administrators` group is listed in the access list with `full control` rights together with the `ivmgr` group.
Findings	The `dynurl` mapping file is useful for the authorization function of dynamic URLs. Dynamic URLs are generated mainly for database request. Those requests are **dynamic** (because they

	change according to each request) and **temporary** (because they exists only once). Each URL has to be authorized by Webseal before the request is sent. Webseal uses the dynurl.conf file to group dynamic URLs by type. This file is thus very important and must not be compromised. The complete `Administrators` group should not have access to this directory. It is necessary to have at least one member of the `Administrators` group, not the entire group. **Control objective PASSED**

Category	Webseal configuration
Item number	**CO-WSC-6**
Item title	The server identity is suppressed in answers
References	Webseal Administrator's Guide Personal work
Risks	A malicious user could identify precisely the Webseal server (information gathering about the infrastructure).
Testing procedure	In the `webseald-internet.conf` file, check that the `suppress-server-identity` parameter is activated with the `yes` parameter under the `[server]` stanza.
Test nature	Objective
Evidence	Extract of the `webseald-internet.conf` file :
	`suppress-server-identity = yes`
Findings	The parameter is correctly activated. Thus Webseal answers without giving its identification (application name and version). **Control objective PASSED**

Category	Webseal configuration
Item number	**CO-WSC-7**
Item title	The Compare mode is not used by LDAP for authenticating users
References	Webseal Administrator's Guide Personal work
Risks	Using the Compare mode is faster but insecure comparing to a real Bind operation.
Testing procedure	In the `webseald-internet.conf` file, check that the `auth-using-compare` parameter is deactivated with the `no` value under the `[ldap]` stanza.
Test nature	Objective
Evidence	Extract of the `webseald-internet.conf` file :

	`auth-using-compare = no`
Findings	With this configuration setting to `no`, any authentication request to the LDAP server is done by Webseal using Bind and Unbind operations instead of a Compare operation (if parameter set to yes). COMPARE operation is faster when authenticating a lot of clients but transfers data insecurely. **Control objective PASSED**

Category	**Webseal configuration**
Item number	**CO-WSC-8**
Item title	The connection to the LDAP server is SSL-enabled
References	Webseal Administrator's Guide Personal work
Risks	All the communication between LDAP and Webseal are in clear text by default.
Testing procedure	In the `webseald-internet.conf` file, check that the `ssl-enabled` parameter is activated with the `yes` value under the `[ldap]` stanza.
Test nature	Objective
Evidence	Extract of the `webseald-internet.conf` file :
	`ssl-enabled = no`
Findings	The connection with the LDAP server is done by Webseal using clear HTTP. **Control objective FAILED**

Category	**Webseal configuration**
Item number	**CO-WSC-11**
Item title	Check SSL type and timeout
References	Webseal Administrator's Guide Personal work
Risks	SSL type could be weak and would not provide the safest possible encryption mechanism.
Testing procedure	In the `webseald-internet.conf` file, check the following parameters in the `[ssl]` stanza : **<u>1) SSL type accepted :</u>** Check that the `disable-ssl-vX=` parameter is activated and only allows SSLv3 (in other words, the parameter `disable-ssl-v3` should be the only one to be deactivated with the `no` parameter) **<u>2) SSL timeout :</u>**

	Check that the `ssl-vX-timeout` parameter is configured with reasonable parameters.
Test nature	Objective
Evidence	Extracts of the `webseald-internet.conf` file : **1) SSL type accepted :**

	`# Selectively disable SSL version support`
	`disable-ssl-v2 = no`
	`disable-ssl-v3 = no`
	`disable-tls-v1 = no`

	2) SSL timeout :
	`ssl-v2-timeout = 100` `ssl-v3-timeout = 7200`
Findings	1) The SSL versions used are too large. Both V1, V2 and V3 are used. 2) The timeout used are too different and show a lack of coherence in the timeouts settings. It seems that these are the default parameter used by Webseal. **Control objective FAILED**

Category	**Webseal configuration**
Item number	**CO-WSC-14**
Item title	Authentication mechanism
References	Webseal Administrator's Guide Personal work
Risks	Failure in the authentication mechanism could allow an attacker to abuse the authentication mechanism. Even a well-intentioned user could be weakly authentified without knowing it.
Testing procedure	In the `webseald-internet.conf` file, check the following parameters in the `*authentication*` chapter : **1) Locate the authentication method(s) definition :** Locate the following stanza : `[ba]/[forms] /[token]/[certificate] /[http-headers]/[ipaddr]` **2) Check activation of authentication methods :** For each stanza (see above), check whether the authentication method is used or not (check if value is a valid parameter enumerated in the explanation of the configuration file or if the value is `none`). **3) Check active method(s) validity :**

	Check that the authentication method(s) activated are valid method(s) for the application. **4) Check authentication mechanism and libraries :** In the [authentication-method] check the authentication methods used and the associated shared libraries.
Test nature	Objective
Evidence	Extracts of the `webseald-internet.conf` file : **2) Check activation of authentication methods :**

```
[ba]
ba-auth = none
[forms]
forms-auth = https
[token]
token-auth = none
[certificate]
accept-client-certs = never
[http-headers]
http-headers-auth = none
[auth-headers]
#header = header-name
#header = entrust-client
[ipaddr]
ipaddr-auth = none
```

4) Check authentication mechanism and libraries :

```
[authentication-mechanisms]
# USERNAME/PASSWORD   such as Basic Authentication or
Forms
#passwd-cdas       = <passwd-cdas-library>
#passwd-ldap       = <passwd-ldap-library>
#passwd-uraf       = <uraf-authn-library>
# TOKEN
#token-cdas        = <token-cdas-library>
# CERTIFICATES
#cert-ssl          = <cert-ssl-library>
(…)
passwd-ldap                                              =
D:\PROGRA~1\Tivoli\POLICY~1\\bin\ldapauthn.dll
cert-ldap                                                =
D:\PROGRA~1\Tivoli\POLICY~1\\bin\certauthn.dll
passwd-cdas=D:\Program
Files\SecurIT\cman\bin\cdas_cman.dll&[D:\Program
Files\SecurIT\cman\etc\cman.conf]
failover-password=D:\Program
Files\SecurIT\cman\bin\cdas_cman.dll&[D:\Program
Files\SecurIT\cman\etc\cman.conf]
```

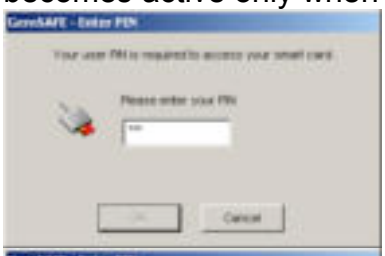
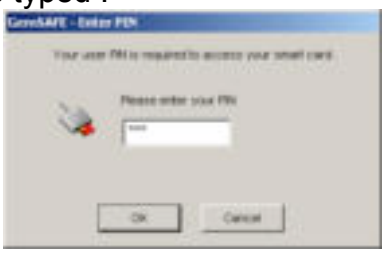
Findings	1) All the stanza were located in the configuration file. 2) Only the forms-auth authentication method is used. All the other ones are not configured. 3) In the context of the application, only the forms authentication

	method should be used. Moreover, forms should be sent to the client (and received from the client) using HTTPS in order to guarantee the confidentiality of the parameters transmitted. 4) This section of the configuration file defines the libraries used for each authentication method. Since we found that only the form-auth is used, we have to verify that no useless library is specified in the file. We can see that all the libraries links are inactivated. Only specific CMAN libraries are activated in the configuration file, which is the only ones that are needed. **Control objective PASSED**

5.3 BEHAVIOR TESTING CONTROL OBJECTIVES

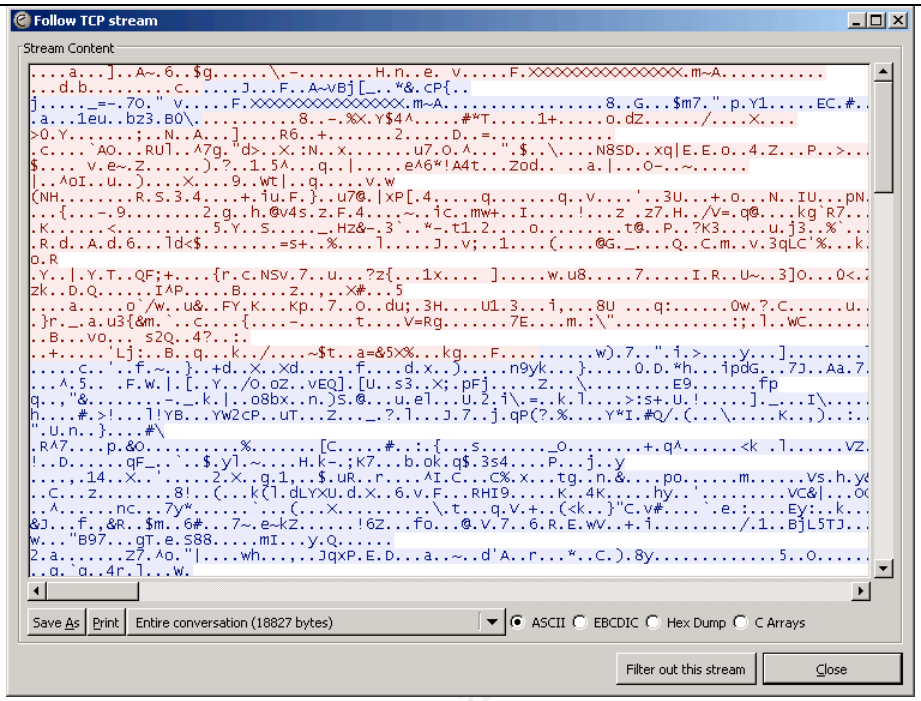
Category	Behavior testing
Item number	**CO-BT-1**
Item title	Identifier, card code and data provided to client are anonymous
References	Personal work
Risks	If too much information are provided, an attacker could use them to identify some layer of the business logic.
Testing procedure	Review the information provided by the bank to log in the application and check that no direct link can be established between one of this information and the client name. Tests have to be made for application without samrtcard (username and password) and application with smartcard.
Test nature	Objective
Evidence	For confidentiality reason, the identifier, card code and data provided by the Bank are not mentioned in this report. The following information/material were provided by the Bank : - Several user names and passwords with different level of authorization - A smartcard associated with one of the user name
Findings	**1) Application without smartcard :** According to the provided login/passwords and relating to the application without smartcard's use, a correlation between the customer and its username/password cannot exist, as accounts are at the test state and have a similar password. **2) Application with smartcard :** Relating to the smartCard concerning the application's access, the PIN code is the only posted data by the user; it should not be equal to user's personal data (typical birth's date). That means that, in the current state of advance of the application's development, this control objective cannot be correctly qualified. Therefore, it is important to not use any correlation between identifier/card code/data sent to client in the application in the following production state. REMARK : Relating to the session cookie that personalizes pages, no account information were identified into. **Control objective PASSED**

Category	Behavior testing
Item number	**CO-BT-2**
Item title	Username format prevents guessing existing usernames
References	OWASP guide Testing Web Application Exposed Personal work
Risks	If too much information are given on the username construction, an attacker could try to lock a high number of account (by flooding the application with existing usernames without valid credentials, this would be a DOS situation) or even to try to connect to the application.
Testing procedure	Review the information provided (especially any kind of user identification) and check that the way of constructing usernames cannot be guessed too easily. Tests have to be made for application without smartcard (username and password) and application with smartcard.
Test nature	Objective
Evidence	**1) Application without smartcard :** According to the authentication page's source and/or error messages, the User-ID and password format is alphanumeric. (messages "Username must be alphanumeric" "Password must be alphanumeric").  Provided accounts are build with the same structure (2 minus letters and 4 numbers) and their password is similar; few attempts to authenticate using this username structure (trivial values and close to the provided login to verify that username's creation does not use a single increment and the common password) did not permit us to identify a valid login/password. **2) Application with smartcard :** The typical aggressor could be a person who found the SmartCard and try to use on its computer or a person that accesses physically to the user's computer. Previously he would surely know that this "user PIN" parameter is only composed of numbers. To access the client SmartCard, the required "user PIN" structure is

	made of 4 to 8 numbers; this is indicated by the “OK” button that becomes active only when 4 numbers are typed :   
Findings	**1) Application without smartcard :** The disadvantage in those structures is the login/password quality. Therefore a minimum of 10 alphanumeric characters should be required with at least one minuscule, one majuscule and one number for the password. Moreover, this login/password’s policy should be limited to the testing period and all these accounts should be deleted when the application will be in use. If the username are always built similarly, it indicates that there is a limited value for the account’s number (which would be 6 760 000). It is either recommended to expand username that the hit rate is lower than 1%. Depending on the number of accounts, the Bank should decide here if a mix of alphanumerical lower-uppercase values should be used. Using also majuscules characters could be a good solution to decrease the potential aggressor’s chances. **2) Application with smartcard :** This indicates that the limited value for the account’s number is 1 billion. Combining with the account lockout policy implemented (see below) this gives a few chances to an attacker to guess an existing pin code. **Control objective FAILED (application without smartcard)** **Control objective PASSED (application with smartcard)**

Category	Behavior testing
Item number	**CO-BT-3**
Item title	Authentication strength (secure transport and account lockout policy) is aligned with current industry good practices
References	OWASP guide Testing Web Application Exposed Personal work
Risks	An attacker could sniff a user credentials if the transport is not secured or try to brute force an account if no account lockout policy is

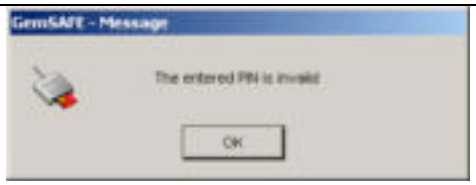
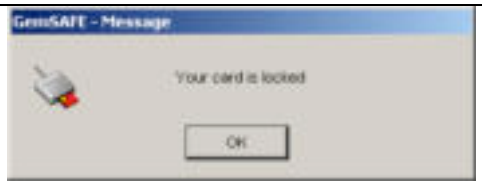
	active.
Testing procedure	**Step 1 :** Try to connect to the application using standard web browser with the given credentials and check the transport method. **Step 2 :** Try to connect repetitively to the application using standard web browser with invalid credentials until the account is locked-out. Tests have to be made for application without smartcard (username and password) and application with smartcard.
Test nature	Objective
Evidence	**1) Application without smartcard :** Two different types of user have been provided for the tests. It is also important to notice that theses users are in the state of test; that means that developers, etc. are using those accounts to verify or to qualify different aspects of the application. These types were: • external user : in this case, the application can only be acceded with the HTTPS ciphered protocol. • internal user : in this case, the application can only be acceded with the HTTP ciphered protocol. We clearly recommend to forbid the http protocol as data flows between server and client are not ciphered and, therefore, they could be intercept in “clear text” in a local area. The authentication process is to send to the server the 4 following fields :
	username=myusername password=<authenticationRequest><type>1</type><uid> myusername</uid> <pwd>mypassword</pwd><applid>webportal-internet</applid><pkcs7></pkcs7></authenticationRequest> x=25 y=1
	2) Application with smartcard : Server proposes SSL encryption and data transactions are correctly ciphered:



Once the SmartCard authentication type is chosen, the following window appears :

(only a portion of the web page is shown and the name of the Bank has been anonymized with the black square)

The protection is the account locking if three consecutive PIN are entered :

	 
Findings	**1) Application without smartcard :** No locking account policy has been seen ; if none exists, it should be implemented in order to avoid brute force attacks. **2) Application with smartcard :** This fact largely decreases the aggressor's possibilities to develop a brute force tool in order to discover the user's PIN and therefore access its pages. Indeed, it would require disconnecting the SmartCard USB interface. **Control objective FAILED (application without smartcard)** **Control objective PASSED (application with smartcard)**

Category	**Behavior testing**
Item number	**CO-BT-4**
Item title	Simultaneous connection to the application are not possible
References	OWASP guide Testing Web Application Exposed Personal work
Risks	Simultaneous connection can allow an attacker to replay immediately intercepted credentials and to log in concurrently to the application. Also the fact of allowing multiple connections increases the risk of poorly terminated session that could be replayed later.
Testing procedure	Try to connect the same account simultaneously using standard web browser. Tests have to be made for application without smartcard (username and password) and application with smartcard.
Test nature	Objective
Evidence	**Application with and without smartcard :** Multiple connections are allowed. No limitation have been noticed.
Findings	**Application with and without smartcard :** After discussing with people inside the bank, it appears that simultaneous sessions on the same account are possible. This policy seems to be particularly important for the business part of the Bank. Also developers seem to need this feature in order to log into the application while only a small number of accounts exist. **Control objective FAILED**

Category	Behavior testing
Item number	**CO-BT-5**
Item title	Try to abuse the web server and the local application
References	OWASP guide Testing Web Application Exposed Personal work
Risks	Replaying the authentication flow could allow an aggressor to compromise the entire authentication process.
Testing procedure	Try different techniques to abuse both local and remote ends : - XSS - Connection on the local application Use a local relaying proxy to keep a detailed trace of every request and reply (e.g. Odysseus Proxy). Tests have to be made for application without smartcard (username and password) and application with smartcard.
Test nature	Objective
Evidence	**<u>Application with and without smartcard :</u>** Generally, the stealing of the cookie that personalizes pages can be executed by two different means : - client workstation's bugs (a remote aggressor has admin or system rights) - servers bug (as XSS attacks possibilities…)  The screenshot above shows that no special character is allowed in the authentication fields. **<u>Local server - Application with smartcard :</u>** This test is specific to the server that runs locally on the client workstation when working with the smartcard. We tried to connect to the workstation local server to check out if it

	could permit malicious acts. It was not possible to use it locally even with the Bank certificate without connecting on the WebSeal server :

	`>curl https://127.0.0.1:5683/pls/prelogin/ -vigL -A "Java Plug-in" --cacert bank.pem --capath "C:\Documents and Settings\tester" -k`
	`* About to connect() to 127.0.0.1 port 5683`
	`* Connected to 127.0.0.1 (127.0.0.1) port 5683`
	`* successfully set certificate verify locations:`
	`*  CAfile: bank.pem`
	`  CApath: C:\Documents and Settings\tester`
	`* SSL: error:140770FC:SSL routines:SSL23_GET_SERVER_HELLO:unknown protocol`

Findings	**<u>Application with and without smartcard :</u>** Special characters are not allowed in username and password fields ; that avoid SQL injections and XSS attacks possibilities: **<u>Local server - Application with smartcard :</u>** It seems that no remote compromise of the local server is possible. **Control objective PASSED**

Category	**Behavior testing**
Item number	**CO-BT-6**
Item title	A customer can not access information of other customers
References	OWASP guide Testing Web Application Exposed Personal work
Risks	An attacker could abuse the system and have unallowed access to confidential data.
Testing procedure	Try different techniques to abuse both local and remote ends : - XSS - Webserver abuse - … Use a local relaying proxy to keep a detailed trace of every request and reply (e.g. Odysseus Proxy). Tests have to be made for application without smartcard (username and password) and application with smartcard.
Test nature	Objective

Evidence	**1) Application without smartcard :**

Step 1 : web server error messages on authentication
A way to access other customers' information or account is to deal with the web server answers to the authentication.

A very good point is that it is not possible to identify valid accounts with the
server's messages after authentication. Indeed, this message is generic; whether you post a valid or a invalid login, it is equal :

Moreover, special characters are not allowed in username and password fields. This avoid SQL injections and XSS (Cross Site Scripting) attacks possibilities.

Step 2 : cookie identifying and replay
Requesting a URL specific to the personalized pages of the application with different cookies we obtained after the authentication with a provided account, we noticed that only one cookie is used to identify the session.

Here, we replay the client's URL request to an application page with all cookies :

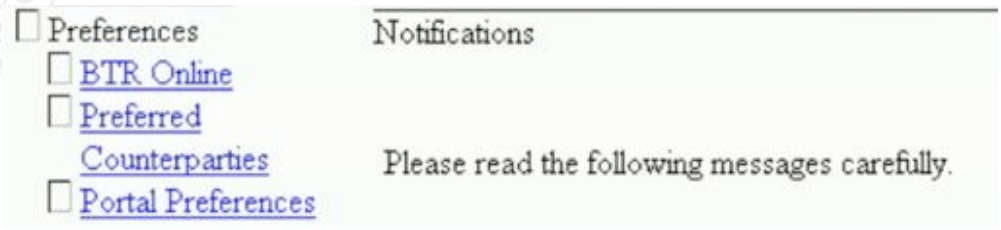
```
>curl -k -vigL
"https://10.66.33.226/octave/wps/myportal/.cmd/ad/.
ar/sa.GetNotification/.c/1013/.ce/1507/.p/1107?PC_1107_Notific
ationStatus=Open&PC_1107_NotificationSummary=Dear%20users,WeB%
20Release%202%20is%20now%20available%20in%20Alpha%202%20enviro
nment.%20Enjoy%20testbank%20!If%20you%20find%20any%20bug,%20pl
ease%20call%20the%20Man%20in%20Black.&PC_1107_NotificationTitl
e=WeB%20Release%202%20is%20now%20available%20in%20Alpha%202&PC
_1107_NotificationDate=1093008240000#1507"  -b  "PD-S-SESSION-
ID=2_zIyD72PxRAvPreYS38IL170k+HKG0Iaoli9HFexPuckjAAAA;
AMWEBJCT!/pls!JSESSIONID=0000ATDSWG4L2SKFC4TYD2OFS0Y:vulfaf99;
AMWEBJCT!/pls!lang=en;        IV_JCT=/application;        PD-
ID=vdXEJ4y3gLlKWcGhmTU-32wxhMK6cAlHb
YA7mEgFp4pIRehv7FhEnzv1bZgT1ZbIcBppDStVjY-
```

```
WCuSR4rPzwCLKt9jxxEyqMVukaG8Mv
zkZsz+MvtMwMkT2Fp0+ITQHdVDz5RVyIZKFL6u9SC9LejFqrYpMnxBJjV1gA4R
UmNtle2kuQwN4svq254GiNLZu7lHwoXd7GYEyeySyCzS6dhQkKEyFvpEd0b1IJ
LEehmpRmp0q3lNPLpJEULcR4xaDMRhAyuQmxOaju8FrkkfxhSu-
ssXXD0AtWat9gnrGwdl7iqsjNE4w-ukhKxcfd72YiW-
ee29Oz6tGR+LW+8amgQCsglQw8ibxh0fvxRQjoYKC5arJj86+J3nnZFdUxAq18
jfVQCkVOcU0h-WQF-nQyf42AO+AhXiFdeq8Hfp0oSTyG1+o1zqGieI2A8QX3-
OoqqOe06SHRQJKIUKnIJ1yBV6CJ5Fj4KsN7D-
+SwpJxP2wweHwsM+7DY2JA56x9ikwSivrNHhvD8BGt
bE4aI6LPYqSFBPTm5qtVFPsWwMo1LlEXg2z2SZPOWPc5Q1oi6ZWQlBM6prdFSH
5RGN6ri3FPVBgLwdaf0oSy3vTM-
tLKFVKJz+hknmdR91+OgLMrWLQhh2kawvzw4AhMlbcMipABNZyK
TOxmdULHJ2sVVWy9zNquDQavED9B+evgaz5KTeZQHmmKRKtE1gZzL8Hp1urlob
fGCMgHxuvC+LXm21dIgBbMshSycam39IVAVL6lXiPSQsGFtRofvkQ7lygktsvp
I1tkXppmkyOhJAFFohjCDMJURb-
jrTQcQaE65GX4DMdPEjtJ9+199OhTmm4yuLO4LdvTba8A6ilykIwDx+3g5D
egpKqcaodiHG39D7KFE6hbDp1sGj+MhK0c0k4w6QIcpR6qwv1kZCVZXcXRd6ag
X4j6zsTIT1k6+v-sm-
ADTDYL5DpjwL1nsIF8h3A+Fz+NeVlDRElZo0DJvxdWDH71Buanzqpd0+nOcH
YBX0UHJDsmHii8+keYw=;AMWEBJCT!/application!JSESSIONID=00003WZ0
3JFCSIR0ZUBNWEBULUQ:vqdnhtfg;AMWEBJCT!/application!nd=sbedskat
"
```

Our attempts revealed that the same page can be acceded with the only use of the “PD-S-SESSION-ID” cookie :

```
>curl -k -vigL
"https://10.66.33.226/application/wps/myportal/.cmd/ad/.ar/sa.
GetNotification/.c/1013/.ce/1507/.p/1107?PC_1107_NotificationS
tatus=Open&PC_1107_NotificationSummary=Dear%20users,WeB%20Rele
ase%202%20is%20now%20available%20in%20Alpha%202%20environment.
%20Enjoy%20testing%20!If%20you%20find%20any%20bug,%20please%20
call%20the%20Man%20in%20Black.&PC_1107_NotificationTitle=WeB%2
0Release%202%20is%20now%20available%20in%20Alpha%202&PC_1107_N
otificationDate=1093008240000#1507" -b "PD-S-SESSION-
ID=2_zIyD72PxRAvPreYS38IL170k+HKG0Iaoli9HFexPuckjAAAA">>a.html
&a.html
```

We noticed that the request of a simpler URL (as https://10.66.33.226/application/wps/myportal) has the same result.



(only a small portion of the page is shown for anonymity reason)

Therefore, we study the generation of the “PD-S-SESSION-ID” cookie.

Step 3 : cookie generation and prediction

Using the authentication of a provided account, 1000 cookie values PD-S-SESSION-ID cookies were retrieved:

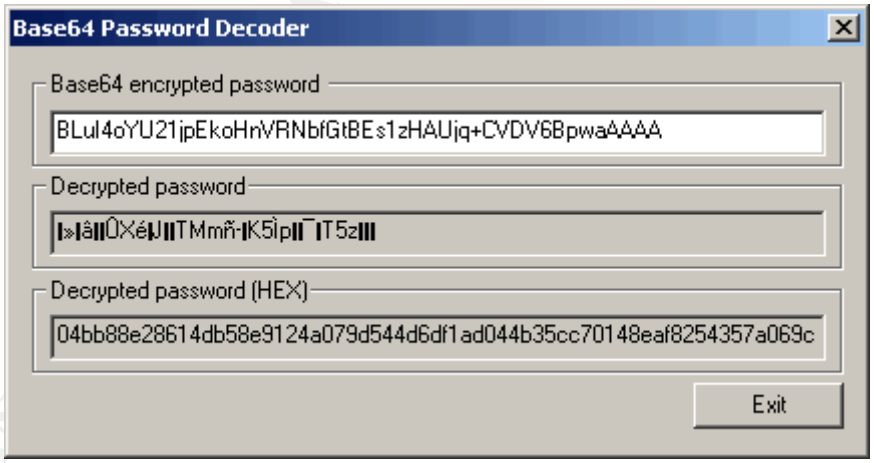
Table of consecutively received cookies:

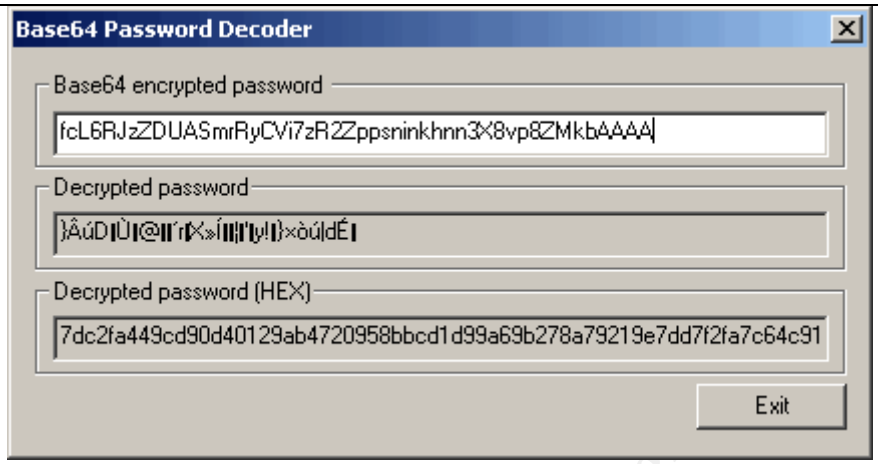
cookie number	"PD-S-SESSION-ID" cookie value
1	2_BLuI4oYU21jpEkoHnVRNbfGtBEs1zHAUjq+CVDV6BpwaAAAA
2	2_fcL6RJzZDUASmrRyCVi7zR2Zppsninkhnn3X8vp8ZMkbAAAA
3	2_EIqc2vn5J6xRK7fSL9RN4aFY6T1+QZmRwX2vZ831LIkcAAAA
(…)	(…)
998	2_piIKCAYTa3mKOyb6JBqTlAkynwaXAiVx5jDUsZQ1lDb-AwAA
999	2_ASby4Lwn3gxpPQcmxmETfg1bSQIp6auVqUj-paQOHRQABAAA
1000	2_nN0G0kcukcrQ+1biIrxwQO7MkdlpRDF7BljwXBU77TsBBAAA

Sorted table of received cookies:

"PD-S-SESSION-ID" cookie value
2_+2l8NpwMcZL3vhTLNiZ2Hrj0Rwn7HZRa3N5IR-TKLm5pAAAA
2_+3wQpW0wrQhDLui-wYV9-1ni1P-FOMoIEFp6qQHPYl1ZAAAA
2_+9DHC0ToeyXapGdPngNW42y2YKv1+LrF50VaJEdGm21eAQAA
(…)
2_ZYbueRhrXKXC80UW7PbvDpQ9h5TFbv4w2JYQvzvJum8fAAAA
2_zzRl40Nyn8ycSB5WpX3omDg1R2c+HdmHwO82IqS-bCNZAQAA
2_+2l8NpwMcZL3vhTLNiZ2Hrj0Rwn7HZRa3N5IR-TKLm5pAAAA

We tried to check out if the decryption of successive cookies shows that they are easily predictable; this is not the case:





The cookie generation seems to be random.

2) Application with smartcard :

The « Bank Key Utility » installs a local server on a non-random TCP port (5683) and the "User Agent" is a Java Plug-in.

When the PIN code is entered and accepted by the server, the application is similar to the application without smartcard. The only difference is that it is deployed on a different host. Therefore, Step 1 and Step 3 are the same than for the application without smartcard, only Step 2 evidence are shown below.

Step 1 : web server error messages on authentication

See above (application without smartcard).

Step 3 : cookie generation

See above (application without smartcard).

Step 2 : cookie identifying and replay

```
>curl -k -vigL -b
"PD-S-SESSION-ID=2_k5JuNTLjZRYkMEYEnVUNUD6y+Nbga2mlZ7tVQux
VnQsbAAAA"
"https://obfuscatedurl/.cmd/cs/.ce/155/.s/809/.r/8">>x.htm&x.h
tm
* About to connect() to obfuscatedintranet.url port 443
*  Connected  to  another.obfuscatedintranet.url  (10.66.39.62)
port 443
* successfully set certificate verify locations:
* CAfile: /usr/share/curl/curl-ca-bundle.crt
  CApath: none
* SSL connection using RC4-MD5
* Server certificate:
* subject: /C=US/O=IBM/OU=Tivoli Systems/CN=Test-Only
* start date: 2001-08-28 17:42:30 GMT
* expire date: 2011-08-27 17:42:30 GMT
*     common     name:     Test-Only     (does     not     match
'wdcsmdnob3.alpha.bank. intranet')
* issuer: /C=US/O=IBM/OU=Tivoli Systems/CN=Test-Only
* SSL certificate verify result: 18, continuing anyway.

> GET /obfuscated.url/.cmd/cs/.ce/155/.s/809/.r/8 HTTP/1.1
 User-Agent: curl/7.11.1 (i686-pc-cygwin) libcurl/7.11.1
 OpenSSL/0.9.7d zlib/1.2.2
```

```
Cookie: PD-S-SESSION-ID=2_k5JuNTLjZRYkMEYEnVUNUD6y+
Nbga2mlZ7tVQuxVnQsbAAAA
Host: obfuscatedurl.intranet
Pragma: no-cache
Accept: */*

% Total    % Received % Xferd  Average Speed
Time             Curr.
                                  Dload  Upload Total
Current  Left    Speed
  0     0    0     0    0     0      0      0 --:--:--
0:00:04 --:--:--     0<
HTTP/1.1 200 OK
< content-type: text/html; charset=UTF-8
< date: Tue, 26 Oct 2004 16:12:37 GMT
< connection: close
< content-language: en
< pragma: No-cache
< cache-control: no-cache
< expires: Thu, 01 Jan 1970 00:00:00 GMT
<                                                    Set-Cookie:
AMWEBJCT!/application!JSESSIONID=0000SDU2AUQL0RMDI4OY4Z2ZB
BI:vp4k1nh
d; Path=/
< Set-Cookie: AMWEBJCT!/application!nd=sbedskan; Path=/
100 16509    0 16509    0     0   2787      0 --:--:-
-  0:00:05 --:--:--  3923
* Closing connection #0
```

The result cannot be shown since it would give too much information on the Bank name and country of origin. The page obtained is a valid applicative page.

Findings

Application with and without smartcard :

Step 1 : web server error messages on authentication

The error messages concerning the use of a wrong username and/or password is not specific. This means that it is impossible to guess whether the username or the password is false. This is a good point since it avoid an attacker to guess too easily what is wrong in its connection attempt.

No special character is allowed in the username and password fields. This is another good point since it avoid attacks type like SQL injection or XSS (Cross Site Scripting).

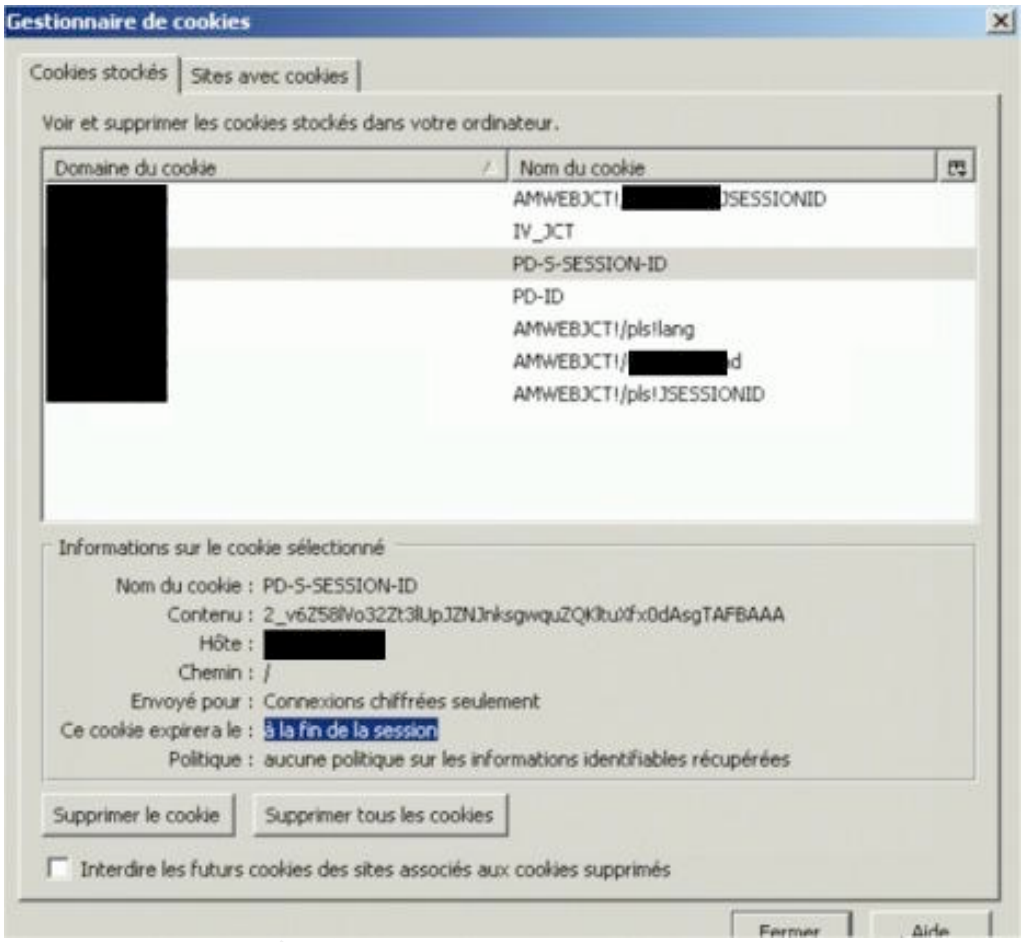
Step 2 : cookie identifying and replay

Only one cookie is used to personalize the page. In other words, only one cookie is used to maintain the session between the Webseal and the client. This is in conformance with the industry best-practices in order to simplify the session management scheme.

However only one cookie is really useful for the session management, other page-specific cookies were detected. The replay attempts were thus done with all the cookies AND only with the identified session-cookie.

	It appears that the result is the same. An access to the page is possible even if the session has been closed before. Moreover, a simple URL request also gives an access to the application page. Indeed, the request of a valid cookie associated to an application's URL brings back a user's personalized page. **Step 3 : cookie generation** The cookie generation is purely random. It was impossible to predict their value and use them to obtain information of other customers. **Control objective FAILED**

Category	**Behavior testing**
Item number	**CO-BT-7**
Item title	Cookies are not persistent
References	OWASP guide Testing Web Application Exposed Personal work
Risks	Persistent cookies can reveal information on the application to an attacker that would have an access on the client workstation. Moreover, cookie replay could be attempted to have access to the application.
Testing procedure	Analyze the locally stored cookie on the client workstation. Use a local relaying proxy to keep a detailed trace of every request and reply (e.g. Odysseus Proxy). Use a browser that allows to manage cookies easily and consult the cookie history too (e.g. Netscape, Mozilla). Tests have to be made for application without smartcard (username and password) and application with smartcard.
Test nature	Objective
Evidence	**1) Application without smartcard :** The cookies are not stored in the navigator. By analyzing each cookie parameter, it seems that there are killed at the end of a session :



Note : “à la fin de la session” means “at the end of the session”.

We verified that the “PD-S-SESSION-ID” cookie was correctly killed at the end of a session.

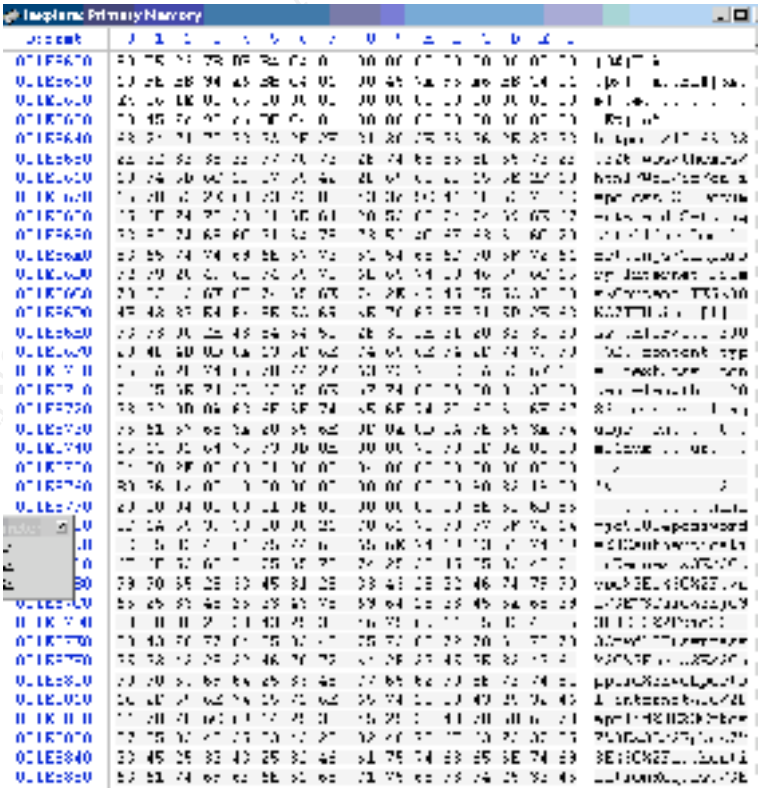
2) Application with smartcard :

We observed the same for the application with SmartCard.

Others actions were executed related to the « Bank Key Utility ». This last is written in Java language; this language’s advantage is that it used with many platforms; therefore, its decompilation does not require strong efforts. Consequently an aggressor is more able to discover bugs. However, special sensitive strings (as password, etc.) were searched in the decompiled files unsuccessfully.

The screen shot of this control objective is not shown for confidentiality reason.

Findings	**Application with and without smartcard :** All the stored cookies are killed at the end of the session. **Control objective PASSED**

Category	Behavior testing
Item number	**CO-BT-9**
Item title	Non-persistent cookie in memory is secured
References	OWASP guide Testing Web Application Exposed Personal work
Risks	Cookies stored in volatile memory are said to be non-persistent cookies. Those non-persistent cookies, although they are more secured than the ones written on non-volatile memory (hard disk) due to their timely nature. Even if the cookie is not written, it can be read and a common mistake is that the cookies stored in memory contains clear-text information that can be gathered and used to compromise the security of the application/session.
Testing procedure	Use a memory dump analyzer to gather RAM memory information and analyze the output. Check that the following information are not be present in the memory dump : - clear text information on user credentials - various identification information on the client/application…
Test nature	Objective
Evidence	The following screenshot shows a dump of the memory. The dump has been stripped to show only the session cookie information. 
Findings	The browser memory dump showed us the login and password

	value in a URL. Moreover, it is easy to associate to the Bank server with an URL or IP address. **Control objective FAILED**

Category	**Behavior testing**
Item number	**CO-BT-10**
Item title	Cookie manipulation is prevented
References	OWASP guide Testing Web Application Exposed Personal work
Risks	Cookie manipulation is often used to abuse or to generate a non-conform behavior of a web application.
Testing procedure	Use a local proxy (e.g. Odysseus Proxy) on the test workstation to intercept all the cookies sent to the application with the HTTPS requests. Once the information (cookies) are intercepted, try to manipulate some parameters (id level, username…) and send them to the application. Monitor the behavior of the application.
Test nature	Objective
Evidence	Some command injections in the cookie values were achieved. Some cookie values were tried to be forced.
Findings	Command injection was not successful. Forced cookie values were regenerated correctly by the server. **Control objective PASSED**

5.4 SYNTHESIS OF RESULTS

The following table presents a synthetic view of the most significant control objectives and their respective status (passed/failed).

CO ID	Title	Status
Design		
CO-D-1	The complete IAA scheme is in line with industry best practices	☑ Passed
CO-D-2	The authorization scheme is clearly defined and in line with the security needs of the customer	☑ Passed
Webseal configuration		
CO-WSC-1	Webseal is chrooted	☑ Passed
CO-WSC-3	The root directory for web server is only accessible by the Webseal user and an administrator	☒ Failed
CO-WSC-4	Webseal limits the size of the post request to read	☒ Failed
CO-WSC-5	The dynurl mapping file is protected and only accessible to Webseal user and an administrator	☑ Passed
CO-WSC-6	The server identity is suppressed in answers	☑ Passed
CO-WSC-7	The Compare mode is not used by LDAP for authenticating users	☑ Passed
CO-WSC-8	The connection to the LDAP server is SSL-enabled	☒ Failed
CO-WSC-11	Check SSL type and timeout	☒ Failed
CO-WSC-14	Authentication mechanism	☑ Passed
Behavior testing		
CO-BT-1	Identifier, card code and data provided to client are anonymous	☑ Passed
CO-BT-2	Username format prevents guessing existing usernames	☒ Failed-uid ☑ Passed-sc
CO-BT-3	Authentication strength (secure transport and account lockout policy) is aligned with current industry good practices	☒ Failed-uid ☑ Passed-sc
CO-BT-4	Simultaneous connection to the application are not possible	☒ Failed
CO-BT-5	Try to abuse the web server and the local application	☑ Passed
CO-BT-6	A customer can not access information of other customers	☒ Failed
CO-BT-7	Cookies are not persistent	☑ Passed
CO-BT-9	Non-persistent cookie in memory is secured	☒ Failed
CO-BT-10	Cookie manipulation is prevented	☑ Passed

Table 12 – Synthesis of control objectives

The status can be interpreted using the following scheme :

Status	Description
☑ Passed	Control objective is reached.
☒ Failed	Control objective is not achieved.

Table 13 – Status interpretation

6 PART IV – AUDIT REPORT

6.1 EXECUTIVE SUMMARY

As a global conclusion of the Octave assessment it can be said that **the level of security of the IAA process is currently low**. This level of security can even be considered as too low for such an application. Octave is meant to be a business tool and at this title must reach the highest possible security level, especially in the IAA process which is the first objective to reach when building security around a web application (and especially around a web-banking application).

Some **major flaws are present in the IAA process**. From an external point of view, we can say that the development work until now has been focused on the functionalities of the application and not on basic security measures. Whether this focus is coming from a lack of support from the management or a development use at the bank is out of the scope of this technical audit.

The **audit objectives are thus not reached** since some critical flaws have been found in the IAA functionality.

The security lifecycle framework in place at the Bank may be re-considered but once again, this is out of the scope of this audit. The next steps in the development should involve more security controls and most of all, more involvement in security since the flaws detected are part of the best practices commonly admitted in the web development field.

Although the result of the audit are technically low, it must be kept in mind that Octave is still in a development phase. All the corrections in both technical and non-technical fields still can be done and included in the next steps of the development.

6.2 AUDIT FINDINGS

Technical audit put in focus the fact that serious flaws are still present in the Octave application. The main flaws that make Octave audit control objectives are not reached are explained below. Pay attention to the fact that their almost all come from the Behavior Testing part (except one), which means that these are not only "configuration" issues. Those flaws are based on the real behavior of the application.

SSL types used are weak [CO-WSC-11]
The audit has shown that SSL types used to establish connection between the clients and the application are not secured enough. Actually, all versions of SSL are allowed. The following extract of the webseald.conf file shows the proof :

```
# Selectively disable SSL version support
disable-ssl-v2 = no
disable-ssl-v3 = no
disable-tls-v1 = no
```

SSL (Secure Socket Layer) is a protocol that allows an encryption of data between a client and a server and authentication of the server only (with SSLv2) and the server and the client (with SSLv3).

SSL has been succeeded by TLS (Transport Layer Security) which allows the same functions that SSLv3 (encryption and authentication of both end of the communication). It is different in the way it functions. Two layers are used, one for the encryption and another (Record Protocol) and another one for the authentication and negotiation of the encryption algorithm (Handshake Protocol).

In our case, SSLv2 should be disabled and SSLv3 could be too. This means that only TLSv1 would be used. It is possible to use only this protocol since recent browsers (IE, Netscape, Mozilla-Firefox) supports TLS without problems.

The fact of using TLS implies that the Bank customers should use a recent browser, this could be a prerequisite for accessing the application. If the Bank does not want to impose the use of a recent browser, SSLv3 could be kept together with TLSv1.

Beside of the SSL versions allowed, SSL timeouts are also wrongly configured. Another extract of the webseald.conf shows this :

```
ssl-v2-timeout = 100
ssl-v3-timeout = 7200
```

Actually the timeout (in seconds) for SSLv2 is too short and the timeout for SSLv3 is too long. This shows an incoherence in the way the timeouts are managed at the Webseal level.

It seems that no special attention has been brought to those particular parameters since they are the defaults value of an out-of-the-box Webseal installation.

Simultaneous connections are possible [CO-BT-4]
Once authenticated, it has been demonstrated that it is possible for a user to have multiple connections at the same time.

When trying to access the application a second time, the user is asked to re-authenticate. The authentication process follows its normal flows and the user can very easily create a second concurrent connection.

This behavior of the application represents a serious flaws since it means that if an attacker can steal authentication data (by sniffing and crypto-analyze them or by an over-the-shoulder-look attack), it means that the attacker can directly connect himself. The situation in this case would be the following : one legitimate connection and one illegitimate connection. What if afterwards the customer discovers illegitimates transaction and asks the Bank for an explanation ? In this situation, the Bank could even not use its non-repudiation data, since the traces of legitimate and illegitimate connections would be mixed.

The fact that the authentication is required when trying to establish a concurrent connection is a good point since it shows that the session is not based on the IP address of the client. It thus avoids more simple attack combining immediate replay of grabbed (sniffed) authentication data with the only condition of changing the IP address to the one of the legitimate client.

Customer may access other customers data [CO-BT-6]
The tests have shown that it was possible to replay a cookie and to obtain customer data after the session has been closed. In this case, the only challenge is to obtain a valid session cookie. As the test have shown, session cookies are generated randomly and are thus not predictable, which renders the task of a potential hacker more complex but not impossible.

Actually, if a malicious user has a physical access to a client machine (which could be the case by using Octave in a multi-user business environment), he can install tools that would grab the non-persistent session cookie and replay this data at later time to obtain a connection to the application.

The following is the request sent to the application using the Curl utility :

```
>curl -k -vigL
"https://10.66.33.226/application/wps/myportal/.cmd/ad/.ar/sa.
GetNotification/.c/1013/.ce/1507/.p/1107?PC_1107_NotificationS
tatus=Open&PC_1107_NotificationSummary=Dear%20users,WeB%20Rele
ase%202%20is%20now%20available%20in%20Alpha%202%20environment.
%20Enjoy%20testing%20!If%20you%20find%20any%20bug,%20please%20
call%20the%20Man%20in%20Black.&PC_1107_NotificationTitle=WeB%2
0Release%202%20is%20now%20available%20in%20Alpha%202&PC_1107_N
otificationDate=1093008240000#1507"      -b      "PD-S-SESSION-
ID=2_zIyD72PxRAvPreYS38IL170k+HKG0Iaoli9HFexPuckjAAAA">>a.html
&a.html
```

The following screenshot is the access obtained to the application (note that the ugly presentation of the page is due to the fact that only the session cookie was sent, and not every page-specific cookies. This does not change anything to the fact that the access is obtained :

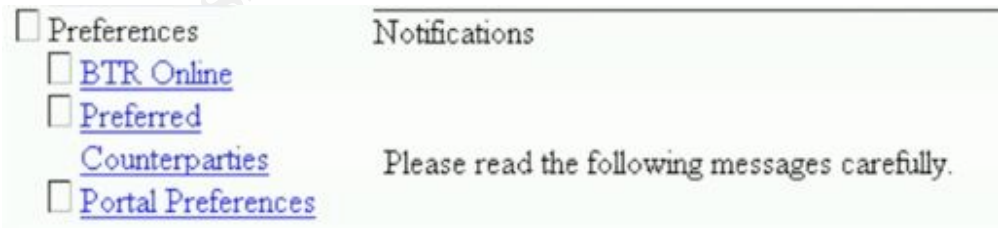


Figure 3 – Page obtained with reused cookie
(only a small portion of the page is shown for anonymity reason)

This is the most critical vulnerability of the application.

Local cookie containing authentication token is not secured [CO-BT-9]
The tests have shown that it was impossible to manipulate cookie content and that they are deleted once the session is closed.

Although, it is possible to read the cookies while they are stored in memory. This is a normal fact by nature impossible to solve, cookies have to be kept somewhere and memory is a better solution than the filesystem. The problem detected in the case of Octave is that the stored cookies are not ciphered in memory. The following screenshot shows this :

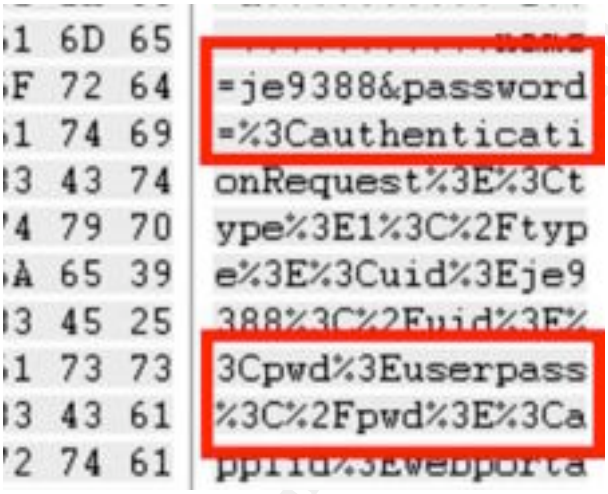


Figure 4 – Non-ciphered cookie

From this memory dump, we can easily see the password zones and thus reuse them. In the test environment, the password used was “pwd”.

The cookie should be stored ciphered in memory.

6.3 AUDIT RECOMMENDATIONS

The following paragraph explains the recommendations that we can do, based on the audit findings.

To achieve this objective, we propose a two-steps approach. This approach has the advantages to ensure that :

- ✓ The flaws found during the audit are corrected and consequently, the security level of Octave is higher than before;
- ✓ Future flaws of the same type will be either avoided or sooner detected by identifying their global cause;
- ✓ The security level of Octave is permanently kept at an acceptable level for the Bank.

Step 1 - Correcting existing flaws
The first step consists in correcting the detected flaws. This is a one shot work.

For each recommendation, the cost has been estimated considering various factors :

- Human time needed to achieve the recommendation;
- Technical constraints;
- Side impact/constraint of the changes.

The Cost scale used is the following :

Cost	Human time	technical constraints	Side impact
Expensive	Some days	Complicated	To be validated before
Moderate	Some hours	Simple	None
Cheap	Some minutes	Basic	None

Table 14 – Cost scale explanation

ID	Description	Cost
REC-1	Change ACLs on the root directory of Webseal in order to only allow access to the Webseal user (daemon user) and one member of the Administrators group	**Cheap**
REC-2	Limit the values of the size of the request that Webseal can read	Moderate
REC-3	Enable SSL on communications between Webseal and LDAP	Moderate
REC-4	Only allow SSLv3 or TLSv1 between clients and Webseal	Moderate
REC-5	Use a more random username policy determination	Moderate
REC-6	Force internal users to use HTTPS to connect to the application	**Cheap**
REC-7	Do not allow multiple connection for the same user account	**Expensive**
REC-8	Kill every information on the server side after the session is closed	**Expensive**
REC-9	Encrypt the session cookie stored in memory	**Expensive**

Table 15 – Correcting existing flaws

REC-7, 8 and 9 have a cost of **Expensive** since they both have :

- Complicated technical constraints : they all require programming or re-programming existing servlets or modules;
- Programming require some man/days for each recommendation;
- When programming, each decision has to be analyzed and has always at least one side-effect (for example, programming an encryption module for ciphering the session cookie could have an impact on the requirements of the client browser…)

For each of them, if the correction can not be made, compensating controls can be put in place to mitigate the risk :

ID	Compensating control
REC-7	✓ Try to keep traces of every action of a user with it's IP address associated
REC-8	✓ Force every client request to be authenticated before (and thus the replay of former session data require an authentication)
REC-9	✓ No compensating control is possible

Table 16 – Compensating controls

Step 2 - Considering the causes of the flaws to maintain the security level
Correcting the flaws is a first step but is not enough. In order to maintain the security level, the causes of those flaws should be determined and corrected.

The table hereunder presents the failed control objectives and determines the causes of each one of them.

REC reference	Cause(s)	Details of the cause
REC-1	Standard configuration	The root directory is configured with the default permission of an out-of-the box Windows 2000 installation
REC-2	Standard configuration	The size limitation of the post request is large by default
REC-3	Lack of security awareness	Not enabling SSL for internal connection can reveal too much trust in insiders and an underestimation of internal threats
REC-4	Standard configuration	Webseal is configured by default with both SSLv2, v3 and TLSv1 encryption
REC-5	Lack of security awareness	Username structure is a security basic for a web application based on username access
REC-6	Lack of security awareness	Allowing HTTP inside connection can reveal an underestimation of internal threats
REC-7	Business requirement	The owner of the application (the business service) has asked to allow multiple connection as a business requirement
REC-8	Lack of security awareness	The programmers seem to not be sensible to every security aspects of a web application
REC-9	Lack of security awareness	

Table 17 – Causes of failed Control Objectives

This analysis shows clearly that the main causes of the detected security flaws is a leak of security awareness, mainly in the development tasks. This is often the case in such development, developers tend to focus only on the functional aspect of the project and forget the security issues.

A second cause (that is in a way a consequence of the first one) is the standard configuration of technical assets. Webseal parameters are often left as they are in an out-of-the-box installation.

Those first two causes of security flaws could be addressed by security awareness sessions for technical teams. Although awareness sessions are useful, it is an entire "security culture" that has to be inserted in the technical team's work.

The last detected cause is the fact that business responsible put as a requirement the fact that simultaneous connections should be possible. In this case, the security risk is too high, business should be informed that this feature is at risk for the entire application.

7 APPENDICES

7.1 APPENDIX 1 – REFERENCES

7.1.1 PRINTED WORKS - BOOKS

- Tittel, Ed, Chappel, Mike and Steward, James Michael. CISSP Study Guide. Alameda: Sybex, 2003. 179 – 190.

- Hunter, Jason, Crawford, William. Java Servlet Programming - Second Edition.Sebastopol, CA. O'Reily, 2001.

- Sambray, Joel, Shema, Mike. Hacking Web Application Exposed. Berkeley: McGraw-Hill/Osborne, 2002. 131 – 200.

7.1.2 PRINTED WORKS - MAGAZINE/ARTICLES

- X, Manu. "Authentification : clé de voûte de la confiance." MISC. Sept. 2004: 18-54.

7.1.3 INTERNET SOURCES/URLS

- Curphey, Mark, Van der Stock, Andrew, et al. The OWASP Testing Project. Draft Version 1.0. July 2004. <http://www.owasp.org/documentation/tesing/application.html>

- Mark, Curphey, David, Endler, et al. Owasp Guide to Building Secure Web Applications. Version 1.1.1. September 2002/August 2004. <http://www.owasp.org/documentation/guide/guide_downloads.html>

- Tusc Corporation. Tutorial for building J2EE Applications using JBOSS and ECLIPSE. First version. July 2003/November 2004. <http://www.tusc.com.au/tutorial/html>

- Sun Corporation. J2EE Tutorial. Version 1.3. April 2002/August 2004. <http://java.sun.com/j2ee/tutorial/1_3-fcs/index.html>

- OWASP Fundation. OWASP Web Application Penetration Checklist. Version 1.1. July 2004/November 2004. <http://www.owasp.org/documentation/guide/guide_downloads.html>

- OWASP FUNDATION. Open Source Security Testing Methodology Manual. Version 2.1. August 2003/August 2004. <http://www.owasp.org/documentation/guide/guide_downloads.html>

- SecurIT Corporation. CMAN authentication for Tivoli Access Manager Whithepaper. Version 1.1. September 2003/November 2004.

<http://www.securit.biz/objects/C-ManWP.pdf>

- IBM Corporation. Webseal Administrators Guide. Version 1. November 2003/August 2004.
  <http://publib.boulder.ibm.com>

- IBM Corporation. Enterprise Business Portals with Tivoli Access Manager. Version 1. August 2002/August 2004.
  <http://www.redbooks.ibm.com/redbooks/SG246556>

- OSSTMM Fundation. Open Source Security Testing Manual. Version 2.1. August 2003/August 2004.
  <http://isecom.securenetltd.com/osstmm.en.2.1.pdf>

- IBM Corporation. Develop and Deploy a Secure Portal Solution Using Websphere Portal V5 and Tivoli Access Manager V5.1. Version 1. August 2004/August 2004.
  <http://www.redbooks.ibm.com/redbooks/SG246325 >