

**PROJECT REPORT**

Project Semester January 2016 to June 2016

**INTERNSHIP REPORT**

Submitted by

**Shiwani Singh**

**13070121148**

Under the Guidance of

Himanshu Agrawal Vinay Potdar

**Associate Professor Dev Manger, CCS**

**SIT Symantec**

**Department of Computer Science Engineering**

**SYMBIOSIS INSTITUTE OF TECHNOLOGY, PUNE**

**January 2016- June 2016**

**DECLARATION**

I hereby declare that the project work entitled “File System monitoring for Windows Machines in Enterprise Network/ Conversion of Security Content Automation Protocol benchmark to proprietary format” is an authentic record of my own work carried out at Symantec Software India Pvt. Ltd. as requirements of six months project semester for the award of degree of B.Tech. in Computer Science Engineering, Symbiosis Institute of Technology, Pune, under the guidance of Vinay Potdar and Himanshu Agrawal, during January to June, 2016.

(Signature of student)

Shiwani Singh

13070121148

Date: 30/06/2016

Certified that the above statement made by the student is correct to the best of our knowledge and belief.

Himanshu Agrawal Vinay Potdar

**Associate Professor, SIT Manager, CCS, Symantec**

**Experience Certificate from Company / Certificate on Company Letter Head**

**(Student may attach a Scanned Copy / Xerox Copy of certificate, retain original for your Resume.)ACKNOWLEDGEMENT**

I would like to thank **Mr. Yogesh Patil**, Director of **Control Compliance Suite, Symantec** for providing me the opportunity to intern at their esteemed organization. A wonderful experience that indeed helped me cast my career.

I also feel obliged to **Prof. Shraddha Phansalkar**, Head of Department, Computer Science and **Dr. T.P.Singh**, Director, Symbiosis Institute of Technology, for lending their full support throughout my internship tenure.

I am also greatly thankful to **Mr. Himanshu Agrawal**, Associate Professor, Symbiosis Institute of Technology, Pune under whose able guidance I completed my internship program successfully.

I express sincere gratitude to my guide, **Mr. Vinay Potdar**, Manager, Development, CCS for his insightful guidance and support throughout the course of my internship and **Mrs. Vani Naik**, **Principal Software Engineer** for assigning me projects, giving me day to day tasks and providing mentorship to execute the same .

Lastly, I would like to thank my parents and friends for their unconditional support and constant encouragement which was an invaluable asset throughout the assignment.

Shiwani Singh

**TABLE OF CONTENTS :**

1. INTRODUCTION 6

2. Work Industry Review 8

3. Introduction to department- CCS (Control Compliance Suite) 9

3.1 Software and hardware requirements of Control Compliance Suite: 11

3.2 System Architecture Diagram: 14

4. DETAILS OF WORK 16

**For Project 1 -**

4.1 PURPOSE OF ASSIGNMENT 16

4.3 OVERALL DESCRIPTION 17

4.4 Project Design 23

4.5 Test Cases considered 26

4.6 Software tools used 26

4.7 DIAGRAMS 27

**For Project 2-**

5.1 Purpose of Assignment 30

5.2 Job Description 31

5.3 Overall description 32

5.4 Project Description : 45

5.5 Test cases considered : 49

5.6 Software tools used : 49

5.7 DIAGRAMS 49

6. Conclusion and future scope of work 52

7. References : 52

1. INTRODUCTION

I pursued my internship at Symantec Software India Pvt Ltd. under the tutelage of Mrs. Vani Naik who offered me two projects:

**Project 1 : File system monitoring for Windows machines in enterprise network** : It is an utility that makes it easy to monitor folders, subfolders and individual files for changes during a specific time in an enterprise network. This module targets all windows platforms and provides you with a comprehensible list of all the files that have been created, modified, deleted along with the type of changes that have occurred. It works with specifying a target along with a template where desired monitoring parameters are configured. Various checks include :

changed files,new files, removed files, changed ownership, changed permissions , changed signature and modification time . Hence, it can also be used to track any changes made by any malicious software in an enterprise network.

**Working of File watch module**: When the file watch module is first run, a snapshot is taken. The next time the module is run, it will again take a snapshot and report for all the files which have been modified by comparing the two snapshots.

Example: The /etc/system file has been specified in the file watch template for Windows. On the first run of the module, a message indicating that snapshot is taken will be in the results. If the file has not been changed on the subsequent run, it will report no problems found. If changes are made to the content of /etc/system file, it will report that the /etc/system file was modified.

I had worked in all stages of this project including coding as well as testing. The coding was done in C++ .

Concepts learned :

1. **C++ STL**
2. **Win32 API to create processes**
3. **Base64 encoding/decoding in C++**
4. **File handling to read CSV files in C++**

Project 2. Conversion of Security Content Automation Protocol benchmark to proprietary format**:** The utility is used for converting the Windows Security Content Automation Protocol (SCAP) standards to the proprietary Control Compliance Suite’s standards in order to automate the high-level monitoring of a system’s security in an enterprise network.

The **Security Content Automation Protocol** (**SCAP**) is a protocol that standardises the format by which software flaws and security configurations are embedded in XML . Essentially , this is a means of establishing some automated “on/off” switches when checking to see if a server or desktop is compliant with the standard in question. The National Institute of Standards and Technology (NIST) is the U.S. government content repository for SCAP .They measure systems to find vulnerabilities and offer methods to score those findings in order to evaluate the possible impact.

In the product I was working on, i.e., Control Compliance Suite, proprietary standards are used for determining the compliance of windows machines in the enterprise networks through a console. User can import these standard in Control Compliance Suite console and run a Data collection-evaluation-reporting job using them for the network machines.

These standards are primarily based on SCAP contents. However, the format is such that it can be understood by the Standards Manager. For example, in these standards we have checks. Each check has the **data collection section** which indicates what data needs to be collected for the check and an **evaluation section** which indicates how to evaluate on the collected data. Based on the evaluation, the check gives a result of pass/fail. In the end, a risk score is calculated for each check and an overall compliance score is calculated for the machine specifying its vulnerability level.

To automate the conversion of SCAP standards to the proprietary standard a tool was required and this was my project. Concepts learned :

**1.C# and .Net basics**

**2. XML parsing, using Collections and File handling in C#**

**3. Security Content Automation Protocol (SCAP) Concepts**

**4.Open Vulnerability Assessment Language(OVAL) Concepts**

# 2. Work Industry Review

Symantec Software Pvt. Ltd. is an American based company headquartered in Mountain View, California, United States. It is a computer and network security company. It is a product , service as well as solution based company providing numerous IT solutions at the global level. Founded in 1982, Symantec has evolved to become the global leader in cyber security, with more than 11,000 employees in more than 35 countries. Symantec is a fortune 500 company and a member of the S&P 500 stock-market Index.

Operating one of the world’s largest cyber intelligence networks they protect customers from the next generation of attacks. They help companies, governments and individuals secure their most important data wherever it lives.

To list few of the **products** , it provides :

* Threat Protection- Endpoint protection, Data Center Security, Advanced Threat Protection
* Information Protection- Data loss prevention, VIP Access Manager, Encryption
* Website Security-SSL/TSL certificates

To list few of the **services,** it provides :

* Cyber Security Services
* Consulting Services
* Customer Success Services
* Business Critical Services

I was assigned internship at the Pune based office located in EON IT Park, Kharadi. The department was called Control Compliance Suite (CCS) and I was under Mr. Vinay Potdar , Manager, Development.

# 3. Introduction to department- CCS (Control Compliance Suite)

My department consisted of a dedicated team delivering products for Symantec Enterprise Security. The department was mainly divided into : Content and Infrastructure. I was into content team working with Mrs. Vani Naik ,Principal Software Engineer. The content team works for the client side of CCS. My job role was primarily into development.

**What is CCS?**

It is an enterprise product used by various organisations like BMC software, barclays, HSBC, etc to keep track of their networks. The product is designed keeping in mind that it is very difficult to monitor the functioning of large network systems which are always vulnerable to external attacks.

Symantec Control Compliance Suite (CCS) automates key IT risk and compliance management tasks. It automates compliance and prioritise risks. Any amount of raw data coming from both inside or outside the network is almost always vulnerable to risks. What CCS does is to analyse this data on an enterprise level , normalise it and present it in a customisable dashboard. Hence, any risk of severe level can easily be identified among all others. This can be easily understood by business executives. It also drills down to asset level for the IT team which can further monitor these risks.

CCS supports automated assessment of the system security configuration, permissions, patches, and vulnerabilities. CCS includes system reporting capabilities.

CCS is an integrated solution comprising of different modules. You can use a combination of these modules to meet your objectives.

**Capabilities of Control Compliance Suite**

Managing IT risk and compliance in an enterprise is a complex task. Control

Compliance Suite helps you to build your IT governance, risk and compliance

program in such a way that you can carry out a comprehensive procedure to improve

the overall security and compliance posture. CCS provides a comprehensive framework that allows customers to do the following:

▪ Lower the cost of risk and compliance posture assessment.

▪ Use automated agent-less or agent-based capabilities to audit and scan technical controls.

▪ Provide an ability to attest procedural controls.

▪ Define, review, and disseminate written policies to end-users as mapped to specific, measurable controls.

▪ Produce evidence of due care in an IT audit process.

▪ Pull in third-party checks and controls data as evidence and for the integrated assessment of technical standards.

▪ Integrate the compliance process with existing asset management systems.

**How to achieve your business objective with CCS** You can use CCS to meet different business objectives such as:

▪ Plan for internal and external audits

▪ Assess technical controls

▪ Evaluate exposure to external threats

▪ Assess procedural controls

▪ Assess data controls

▪ Report on IT risk and compliance posture

# Software and hardware requirements of Control Compliance Suite:

### Software Requirements

#### Operating System

* Windows Server 2003 SP2 x64 Enterprise or Standard edition
* Windows Server 2003 R2 SP2 x64 Enterprise or Standard edition
* Windows Server 2008 SP2 x64 Enterprise or Standard edition
* Windows Server 2008 R2 x64 Enterprise or Standard edition

#### Database

* Microsoft SQL Server 2005 SP2 or later (32-bit and 64-bit computers)
* Microsoft SQL Server 2008 SP1, SP2 (32-bit and 64-bit computers)
* Microsoft SQL Server 2008 R2 (32-bit and 64-bit computers)

#### Other Software

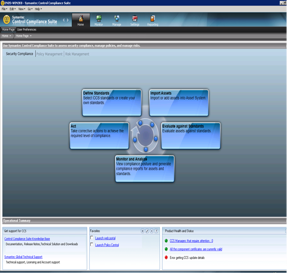
* Microsoft .Net Framework 3.5 SP1
* Oracle Instant Client 10.2.0.4
* Internet connection for CCS service
* Internet Explorer 8.0
* Internet Information Service (IIS)
* ASP.NET v4.0.30319
* ASP.NET v4.0.30319 Web Service Extensions

## Hardware Requirements

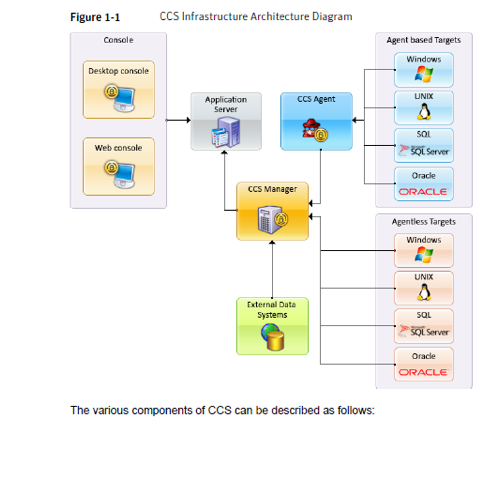
### For deployment on a single server

* Minimum memory: 4GB
* Minimum processor: Dual Proc 3GHz
* Minimum hard disk space: 140GB

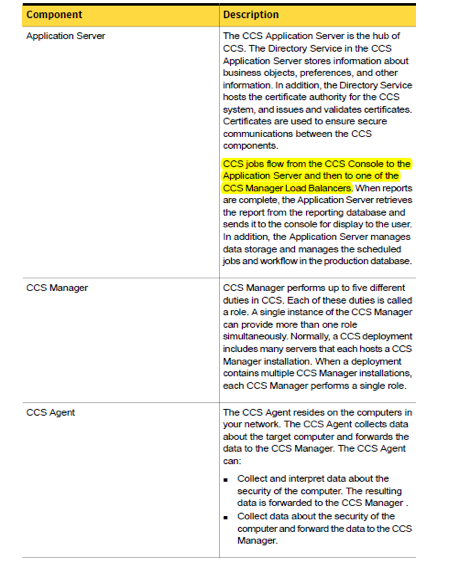
**This is how a typical CCS console looks like -**

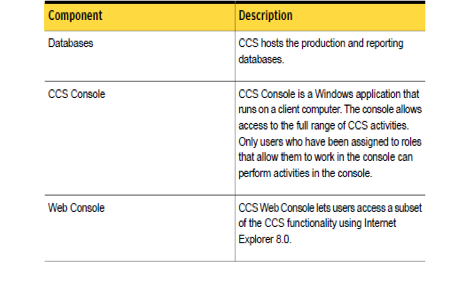


# System Architecture Diagram:



**Explanation of various components :**





# 4. DETAILS OF WORK

**PROJECT 1 : Development of File system monitoring for Windows machines in enterprise network** called “**File watch” .**

# 4.1 PURPOSE OF ASSIGNMENT

1.To analyse file watch module for Enterprise Security Manager (ESM) .

2. Code the input and output modules to migrate this feature on Control Compliance Suite.

**What is File watch module?**

It is an utility that makes it easy to monitor folders,subfolders and individual files for changes during a specific time in an enterprise network. This module targets all windows platforms and provides you with a comprehensible list of all the files that have been created, modified, deleted along with the type of changes that have occurred. It works with specifying a target along with a template where desired monitoring parameters are configured. Hence, it can also be used to track any changes made by any malicious software in an enterprise network.

It reports these changes in the format of a report along with the name of the file.It works with a template where desired directories or files are configured. When the file watch module is first run, a snapshot is taken. The next time the module is run, it will report for all the files which have been modified.

Example: The /etc/system file has been specified in the file watch template for UNIX. On the first run of the module, a message indicating that snapshot is taken will be in the results. If the file has not been changed on the subsequent run, it will report no problems found. If changes are made to the content of /etc/system file, it will report that the /etc/system file was modified and will also report the new state along with the old one.

4.2 JOB DESCRIPTION: My job was to first analyse the working of File Watch module in Symantec Enterprise Security Manager™ ,understand its working and significance, note the way its input and output works and then to code on similar grounds. This feature was to be added in Symantec Control Compliance Suite. As Control Compliance Suite follows a certain pattern of inputs and outputs it was important that the same pattern was followed while migrating this tool. I had worked on all phases of this tool development, i.e., from coding to unit testing to adding features.

# 4.3 OVERALL DESCRIPTION

**How was it done on ESM?**

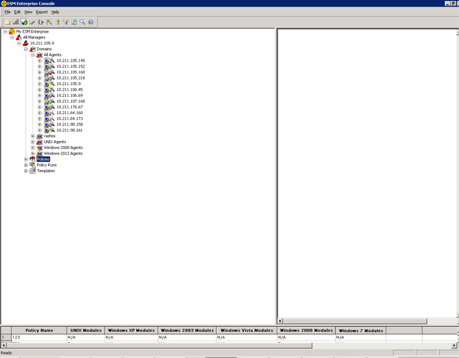
Symantec Enterprise Security Manager™ automates the discovery of security vulnerabilities and deviations from the security policy in mission critical applications and servers across the enterprise. The file watch module of ESM is a console based application which uses the policies, modules and templates to report file changes. It contains agents, managers, templates, modules and policies.

**About the policies, modules, and templates**

Symantec ESM uses policies, templates, and modules to identify and evaluate network vulnerabilities and security policy violations. Policies set the standard that Symantec ESM uses to measure the security state of agent computers. Policies contain modules. Modules contain the security checks that perform the assessments. The templates and the snapshots serve as baselines to determine what conditions should exist on agent computers.

This I how the console looks like. It contains managers(10.211.105.9 in this case), agents and policies.

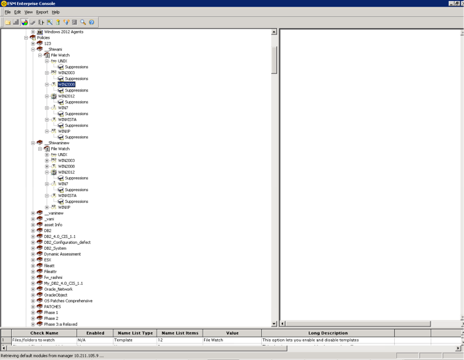
Pter



**About the policies :**

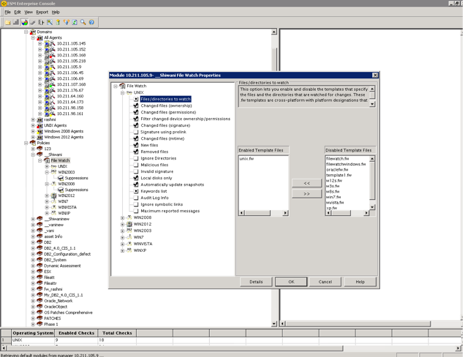
Policies specify the settings, the authorisations, or the permissions that network resources must have to comply with your company policy. Symantec ESM compares the current state of each assessed computer to standards that are defined in the policy, and reports each discrepancy with its severity score.

You can run policies on a single agent or on all agents in a manager domain. Here, I have added a new policy called “\_\_Shiwani”.

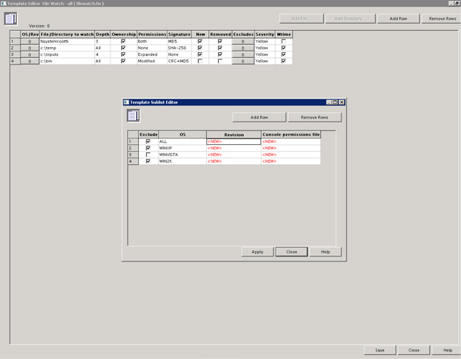


**About editing a module**

By editing the modules, you can match Symantec ESM policies to the relevant parts of your organization's security policy. Hence, policies run on top of module files. You can enable or disable the security checks in a module, or edit the templates to associate with a check. Here, I have edited the policy “\_\_Shiwani” in which checks for changed files ownership, permissions, signature has been added.

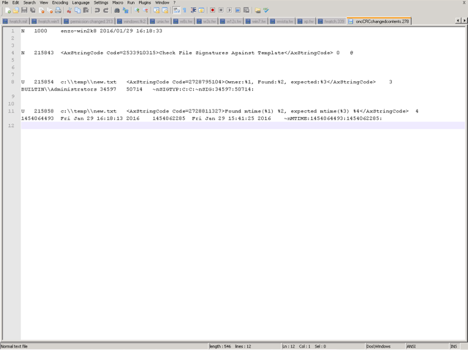


**About managing templates :**

A template is a file that contains module control directives and definitions of objects (files or directories) with their expected states. Symantec Enterprise Security Manager lets you create, edit, and delete templates. With each new security update, Symantec Enterprise Security Manager overwrites all of the templates that ship with the program. Consequently, you lose any changes that you have made to these templates. It contains the list of files/ directories to be reported on.

After adding a policy, we edit its module file in which we check/uncheck changes, add/remove template files which contains a list of parameters to be reported on and finally run the policy on the agent(or all agents). After taking a snapshot of the existing state of folders or subfolders it compares it with the last snapshot and reports for any changes. This report is generated in the ESM program directory. A sample report looks as shown below.

**Sample Report Generation :**



# 4.4 Project Design

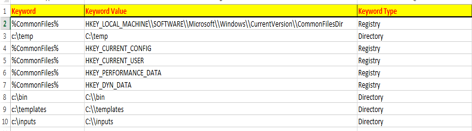
Now, my job was to assist in a making a similar application for Symantec Control Compliance Suite which does all this in C++.

The first job was to take inputs from the user. This was done easily in MS-Excel 2013 using data validation features . This was later on converted to CSV files for code handling.

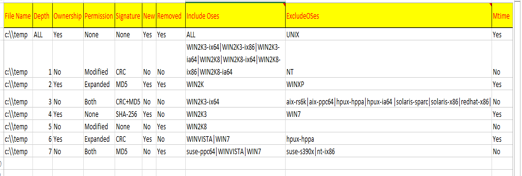
**Module Input file (lists of all checks to be performed)**



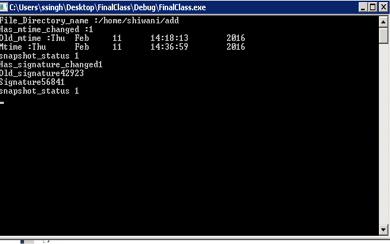
**Keyword input file (contains information about registry keys and values)**



**Template Input file (list all files and what has to be reported**)

The datasource will read the results from output and convert it to records to be written in the reply xml.This is done by comparing snapshots to the previous snapshot taken and reporting all changes. 5. For each individual file object in the program, the outputs are mapped to their corresponding changes and saved.

For demonstration purpose, I printed it on the console.



  6. All error conditions are reported to the user.

**List of all checks performed :**

File/Directory name (primary)

1. AddedSinceLastSnapshot  (Yes/No)
2. RemovedSinceLastSnapshot (Yes/No)
3. ModifiedSinceLastSnapshot (Yes/No)
4. Owner
5. Signature
6. mTime
7. Has signature changed (Yes/No)
8. Old signature
9. Has ownership changed (Yes/No)
10. Old owner
11. Has mtime changed (Yes/No)
12. Old time
13. EventLogInformation (Yes/No)

Using **Event Log in windows :**

In Windows /Linux OSs, there is an Auditing subsystem built-in, that is capable of logging data about file and folder deletion, as well as user name and executable name that was used to perform an action. The Auditing is not enabled by default because any monitoring you use consumes some part of system resources, so tracking down too much events may cause a considerable system slowdown. Even more, since not all user activity is of interest for logging.Auditing policies enable us capturing only event types that we consider being important. This is a very important feature. This was also made use of in File Watch . The event codes associated with file addition, modification, deletion were added as switch cases in the code. These are directly taken from the event log to generate reports.

# 4.5 Software tools used :

1. Microsoft Visual Studio 2010 Express edition
2. Microsoft Excel 2013
3. Putty
4. Event log viewer in SQL server 2010

**5. Project 2 : Conversion of Security Content Automation Protocol benchmark to proprietary format**

# 5.1 Purpose of Assignment

Symantec Control Compliance Suite Standards Manager is one of the five modules of CCS which delivers asset autodiscovery across network devices, servers, and databases and assesses the security configuration of these assets. Organizations employ Symantec Control Compliance Suite Standards Manager to discover and identify rogue and misconfigured assets, detect configuration drifts, and evaluate if systems are secured, configured, and patched according to the customer's security standards.



This CCS SM standard was made manually using the SCAP standards according to the format that is understood by CCS , which resulted in a lot of wastage of time. Hence, a tool was required to automate this conversion.

User can import the SM standard in Control Compliance Suite console and run a Data collection-evaluation-reporting job using them for the network machines.

This standard has checks. Each check has the data collection section which indicates what data to collect for the check from the asset and an evaluation section which indicates how to evaluate on the collected data. Based on the evaluation, the check gives a result of pass/fail.

# 5.2 Job Description

My job was to fully understand how the Security Content Automation Protocol (SCAP) works and then make changes in the existing automation tool in order to make it work. I had also added various objects according to the recent versions of SCAP. Also, heavy testing was done on my part by comparing the standard generated by the tool against that which existed and was made manually.

# 5.3 Overall description

**Automated auditing the system using SCAP (Understanding SCAP)**

We live in an electronic age. As computer systems are getting more capable and complex, it is ever more important to set and keep the underlying computer system secure against security threats. Besides the requirement the system to be designed with security in mind right from the scratch, the subsequent related actions often involve (but are not limited to) the following:

* Proper system configuration,
* Presence of means for users' privilege separation,
* Periodically updating the underlying system software with available security patches,
* Presence of system tools regularly performing security scans, integrity checks etc.

The **Security Content Automation Protocol** ([SCAP](http://scap.nist.gov/)) can be used for automated system monitoring and predefined security policy compliance checks. Keeping in mind the objective to comply with a predefined level of security, let's first look what we already have available:

* we have got a set (often many thousands) of computer systems we want to monitor / administer, and
* central security policy we want to be applied to each of these systems.

From observation, the security policies often come in a form of set of rules (a checklist), where the system has to be compliant to all the rules in order to comply with that security policy. Also, since there might be differences in the controlled systems, we would want to abstract from concrete system's specifics in order to fulfill the main goal.

Considering the above, let's define the protocol features we are searching for as follows:

* The mechanism should be capable of performing system state scans, regularly and in automated way (locally and / or remotely),
* It should be possible to specify what should be audited in a form of a checklist,
* That checklist would ideally not depend on the type of the systems, we are going to apply the security policy against,
* The obtained results should be reported preferably in some inter-operable form for their further (possibly again automated) reuse / processing,
* Also, once results have been analyzed, the mechanism should be capable of correcting the local system inconsistencies for compliance with the predefined centralized policy.

SCAP has been chosen as a representative of a community evolved protocol, which meets the above criteria. SCAP defines how the following standards (referred to as SCAP 'Components') are combined:

**SCAP Components**:

* [Common Vulnerabilities and Exposures](https://en.wikipedia.org/wiki/Common_Vulnerabilities_and_Exposures) [(CVE)](http://cve.mitre.org/)
* [Common Configuration Enumeration (CCE)](http://nvd.nist.gov/cce/)
* [Common Platform Enumeration (CPE)](http://scap.nist.gov/specifications/cpe/)
* [Common Weakness Enumeration (CWE)](https://nvd.nist.gov/cwe.cfm)
* [Common Vulnerability Scoring System (CVSS)](http://www.first.org/cvss/)
* [Extensible Configuration Checklist Description Format](https://en.wikipedia.org/wiki/Extensible_Configuration_Checklist_Description_Format) [(XCCDF)](http://scap.nist.gov/specifications/xccdf/)
* [Open Vulnerability and Assessment Language](https://en.wikipedia.org/wiki/Open_Vulnerability_and_Assessment_Language) [(OVAL)](http://oval.mitre.org/)

Out of these, to abstract from underlying computer system characteristics we will use **Open Vulnerability and Assessment Language** ([OVAL](https://oval.mitre.org/)) standard of SCAP. For representation of a security policy we will entertain the **Extensible Configuration Checklist Description Format** ([XCCDF](http://scap.nist.gov/specifications/xccdf/)) concept of SCAP .

**Extensible Configuration Checklist Description Format** ([XCCDF](http://scap.nist.gov/specifications/xccdf/)) :

XCCDF is a specification language for writing security checklists, benchmarks, and related kinds of documents. An XCCDF document represents a structured collection of security configuration rules for some set of target systems. The specification is designed to support information interchange, document generation, organizational and situational tailoring, automated compliance testing, and compliance scoring. The specification also defines a data model and format for storing results of benchmark compliance testing. The intent of XCCDF is to provide a uniform foundation for expression of security checklists, benchmarks, and other configuration guidance, and thereby foster more widespread application of good security practices.

In XCCDF terminology :

1. The security policy is constituted as **checklist.**

2. Checklist is represented by **benchmark**. Benchmark consists of items (groups, rules, values, profiles, etc.).

3. **Rule** is a named entity, that should act as system check holder.

4. **Group** merges particular rules into logically related sections.

5. **Value** is a named entity, which can be used in other items to hold particular state information and to be able to pass this information further.

6. Finally **profile** is a subset of rules available in the benchmark would be executed when performing the system scan.

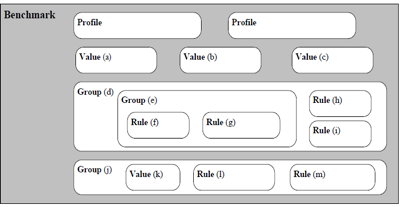
Hence,

We have a protocol to represent the automated checks (SCAP)

A way how to represent a computer system details (OVAL)

And a manner how to comprise a security policy (XCCDF).

This is a typical representation of XCCDF.

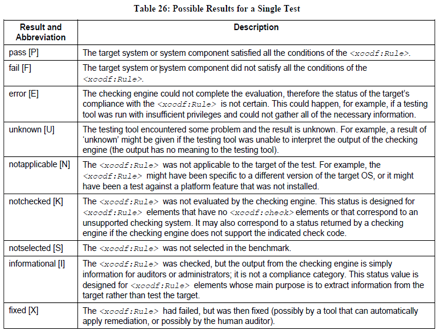


**Workflow of a system scan (consuming the benchmark)**

During the scan evaluation an interpreter interprets each rule of the XCCDF benchmark (one at a time), using OVAL definitions to compare actual system property values with the expected ones (defined in the particular OVAL check for each of the tests). After evaluating all rules, partial results of the scan are given as Pass/fail.

**Understanding the results of the scan**

There are various XCCDF rule's evaluation results possible:



**Structure of an OVAL system check**

Open Vulnerability Assessment Language is the mechanism allowing us to abstract from concrete computer system properties, and express them in unified way so it would be understandable for the OVAL interpreter on one hand, but also for producing final system XCCDF results on the other.Each OVAL definition can contain one or more tests, that should check if the system is compliant with the desired policy. Each test element consists of **object** and **state.**

Example- Let's suppose an example of implementing OVAL check validating if minimum length of user provided password (defined in */etc/login.defs* for passwords managed via *shadow-utils* package) file meets required length, the object here would be /etc/login.defs file, while the state would be the current value of PASS\_MIN\_LEN row within that file.

<definition id="accounts\_password\_minlen\_login\_defs" version="1">

<metadata>

<title>Set Password Expiration Parameters</title>

<affected family="unix">

<platform>Fedora 19</platform>

</affected>

<description>

The password minimum length should be set appropriately.

</description>

</metadata>

<criteria operator="AND">

<criterion test\_ref="test\_etc\_login\_defs" />

</criteria>

</definition>

<ind:textfilecontent54\_test check="all" comment="check PASS\_MIN\_LEN in /etc/login.defs"

id="test\_etc\_login\_defs" version="1">

<ind:object object\_ref="object\_etc\_login\_defs" />

<ind:state state\_ref="state\_accounts\_password\_minlen\_login\_defs" />

</ind:textfilecontent54\_test>

<ind:textfilecontent54\_object id="object\_etc\_login\_defs" version="1">

<ind:filepath>/etc/login.defs</ind:filepath>

<ind:pattern operation="pattern match">

^PASS\_MIN\_LEN\s+(\d+)\s\*$

</ind:pattern>

<ind:instance datatype="int">1</ind:instance>

</ind:textfilecontent54\_object>

<ind:textfilecontent54\_state id="state\_accounts\_password\_minlen\_login\_defs" version="1">

<ind:subexpression operation="greater than or equal"

var\_ref="var\_accounts\_password\_minlen\_login\_defs" datatype="int" />

</ind:textfilecontent54\_state>

<external\_variable comment="password minimum length" datatype="int"

id="var\_accounts\_password\_minlen\_login\_defs" version="1" />

**Explanation of OVAL structure:**

The definition itself is encapsulated into **<definition>** element. There are five possible types of a class: *compliance, inventory, patch, vulnerability,* and *miscellaneous.* We have used the *compliance* one as it best suits for our purpose. *id* should be unique definition identifier.

The *<definition>* element contains **<metadata*>***element to further clarify title, description (purpose) of the OVAL definition, and system environment the test is intended for (family and platform).

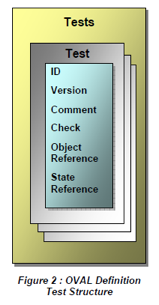
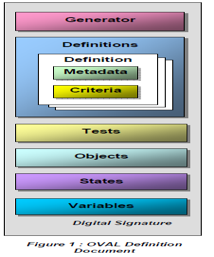
Once the metadata has been added to the definition, it's time to include **criteria** operator. Purpose of criteria element is to join the individual tests together and clearly specify the logical operation (possible *AND, OR, XOR* or *ONE* values of the *operator* attribute), which should be performed to obtain the final result value. c*riteria* element contains one or more *criterion* elements, with actual reference to the test (*test\_ref attribute),* possible comments (*comment* attribute), and if the result of the test should be negated (*negate* attribute “true”) prior applying the logical operation mentioned before, or not.

As can be seen, in our example the *criterion* element references “test\_etc\_login\_defs”, which is the actual *id* of our *textfilecontent54\_test.* The test contains reference to both, *textfilecontent54\_object* and *textfilecontent54\_test.* The required *check=”all”* attribute determines how many of the existing objects must satisfy the specified state requirements ('all' in our case).

In *textfilecontent54\_object* definition we specify subset of allowed child elements, namely that the file path we are interested in is /etc/login.defs, the operation that should be applied against the chunk of that text file is matching a pattern (we also specify the form of that pattern), and that in a case a pattern match is found we are interested in the first occurrence (value of *instance* element)1.

The *textfilecontent54\_state* element is an actual expression of the rule expectations. In our example we require the operation to be performed with value found to be “greater or equal” comparison against the value of ''var\_accounts\_password\_minlen\_login\_defs'' variable (whose value is defined outside of our OVAL definition), and that data type of value of that variable should be integer number.

**Typical structure of OVAL file and tests within it**



**<tests>**

The tests section contains all of the individual system level tests needed by the definitions . For each platform or application addressed by OVAL, there is a corresponding set of tests to express the various configuration states for that entity.The structure of each test, with respect to the information being collected, is meant to reflect the structure of the information for the corresponding entity on the end host. For example, for the Microsoft Windows platform, there are tests for Active Directory, the Registry, and file data. For Red Hat Linux, there are tests for RPM, uname, and file data.

**<objects>**

The objects section contains all of the system entities (e.g. files, Registry keys, packages,etc. ) that are being analyzed by a definition. For each object of interest on the platforms supported by the OVAL Language, there is a corresponding schema detailing the properties being examined for that object. For example, in the case of a Microsoft Windows Registry object, the properties being examined are the hive, key, and name, which represent the path components to a specific Registry key value. The reason for dedicating a specific section of the definition to objects is, in many cases, the same objects are being examined over and over again. By assigning unique ID’s to the objects and maintaining them in a separate section, it allows the same objects to be referenced from multiple tests, and eases the task of data collection. For some objects, it may be desirable to control the way in which it is gathered, or the amount of data gathered. For this reason, <behavior> and <filter> elements provide a means to perform such actions on an object.

**<states>**

The states section complements the objects section, in that it contains the values against which the actual system objects are being compared . Returning to the Microsoft Windows Registry example from the object section – the object defines the Registry path to traverse, while the corresponding state defines the specific value of

interest, and the operation to be performs when conducting a comparison. As explained in the object section, it is possible to reference a single object from multiple tests. It is also possible that the value comparison being conducted against a specific object can differ from one test to the next. Therefore, the states are defined in a distinct section, and each has an associated unique ID to allow for greater flexibility in the reuse and combination of objects and states within a test.

**<variables>**

It is not always the case that all of the values within a definition are known at the time a definition is written . Some values can vary based upon an organization’s policy or the specific configuration of a machine. As a result, the variables section contains placeholders for values that must be filled in at definition execution time – with

one exception. There are three types of variables defined in OVAL, external, local, and constant.

* External variables are values that have to be obtained from an outside source – e.g. user, configuration file… - and plugged into the definition at run time.
* Local variables are also introduced into the definition at run time, but they represent values that can be retrieved from objects on the system itself.
* Constant variables represent common static values used across a number of definitions. These variables have been broken out and hard coded into the individual definition document to ensure consistency across definitions.

**Producing sample XCCDF rule**

Now let's see how a corresponding XCCDF rule for this OVAL system check would look like:

<Rule id="accounts\_password\_minlen\_login\_defs" selected="false" severity="medium">

<title xml:lang="en-US">

Set Password Minimum Length in login.defs

</title>

<description xmlns:xhtml=<http://www.w3.org/1999/xhtml> xml:lang="en-US">

To specify password length requirements for new accounts, edit the file <xhtml:code>/etc/login.defs</xhtml:code> and add or correct the following lines:

<pre xmlns="http://www.w3.org/1999/xhtml">PASS\_MIN\_LEN 12</pre>

<br xmlns="http://www.w3.org/1999/xhtml"/>

<br xmlns="http://www.w3.org/1999/xhtml"/>

Nowadays recommended values, considered as secure by various organizations focused on topic of computer security, range from <xhtml:code>12 (FISMA)</xhtml:code> up to <xhtml:code>14 (DoD)</xhtml:code> characters for password length requirements. If a program consults <xhtml:code> /etc/login.defs</xhtml:code> and also another PAM module (such as <xhtml:code>pam\_cracklib</xhtml:code>) during a

password change operation, then the most restrictive

must be satisfied. See PAM section for more information about

enforcing password quality requirements.

</description>

<reference

href="http://csrc.nist.gov/publications/nistpubs/800-53-Rev3/sp800-53-rev3-final.pdf"> IA-5(f)

</reference>

<reference href="http://iase.disa.mil/cci/index.html"> 205

</reference>

<rationale xmlns:xhtml="http://www.w3.org/1999/xhtml" xml:lang="en-US">

Requiring a minimum password length makes password cracking

attacks more difficult by ensuring a larger search space.

However, any security benefit from an onerous requirement

must be carefully weighed against usability problems, support

costs, or counterproductive behavior that may result.

</rationale>

<check system="http://oval.mitre.org/XMLSchema/oval-definitions-5">

<check-export export-name="oval:ssg:var:153"

value-id="var\_accounts\_password\_minlen\_login\_defs"/>

<check-content-ref name="oval:ssg:def:127" href="ssg-fedora-oval.xml"/>

</check>

</Rule>

The rule definition starts with unique *id,* value of *selected* attribute says that this rule would not be selected to be run by default, *severity* level is used for metrics and tracking.Rule's definition then continues with title (short rule summary), description (longer description), possible references, and rationale (clarify purpose of the rule).The aforementioned OVAL system check is linked with this rule via the *<check>* element. The *system* attribute of the *check* element specifies lower-level system language, the OVAL rule should be expected to be written in.

The presence of *check-export* name suggests this OVAL test uses certain OVAL variable (*var\_accounts\_password\_minlen\_login\_defs* in our example), and that the check definition itself can be found in *ssg-fedora-oval.xml* file present on the local system, and *oval:ssg:def:127* is the OVAL internal name for the **accounts\_password\_minlen\_login\_defs** OVAL system check, we have defined above.

This definition to be complete we need to provide definition of the **var\_accounts\_password\_minlen\_login\_defs** variable yet:

<Value id="var\_accounts\_password\_minlen\_login\_defs" type="number">

<title xml:lang="en-US">minimum password length</title>

<description

xmlns:xhtml="http://www.w3.org/1999/xhtml" xml:lang="en-US">

Minimum number of characters in password

</description>

<warning xmlns:xhtml="http://www.w3.org/1999/xhtml"

xml:lang="en-US" override="false" category="general">

This will only check new passwords

</warning>

<value>12</value>

<value selector="6">6</value>

<value selector="8">8</value>

<value selector="10">10</value>

<value selector="12">12</value>

<value selector="14">14</value>

</Value>

In the definition we specify the variable to be numeric, and having actual value of '12'. We use the *selector* attribute to define the allowed values for this variable to be just one of '6', '8', '10', '12', or '14'. XCCDF profiles can be used to assign actual values to OVAL variables (various profiles can define various values for particular variable).

**Summary**

* SCAP gives us a transparent, interoperable, repeatable ,and ultimately automated way to assess security software flaws and misconfiguration in the enterprise
* Efficiencies gained through SCAP give our IT security teams additional cycles to address other important aspects of IT security
* By linking compliance to configuration, SCAP makes compliance reporting a byproduct of good security, allowing IT security teams to focus on securing the enterprise.

**Using SCAP in CCS standards manager :**

**About assets :**

These are basically the components whose data can be collected. Primitive technology assets include User accounts, Computers, Printers, Network,I nfrastructure, and Services. Control Compliance Suite collects data on these primitive assets*.* Control Compliance Suite supports the data collection, analysis, and reporting on the following platforms:

■ Enterprise Security Manager

■ Oracle

■ SQL

■ UNIX

■ Windows

■ Exchange

■ NDS

■ NetWare

■ Cisco

■ VMware

**About standards :**

Control Compliance Suite makes available a set of predefined standards that are

installed along with the product. These standards are mostly derived from some

published guidelines such as SCAP.

In Control Compliance Suite, the standards hierarchy is explained as follows:

■ A standard contains one or more sections.

■ Each section can further contain other sections or checks.

■ A check is always contained within a section in a standard.

**Typical Standards Manager’s check :**



**About sections**

You use a section to organize or to group related checks. A section can contain

another section. Hence, a section can be a collection of checks and other sections.

For example, consider that you have one set of checks that relate to account

passwords. Another set of checks concern the account lockout policy.

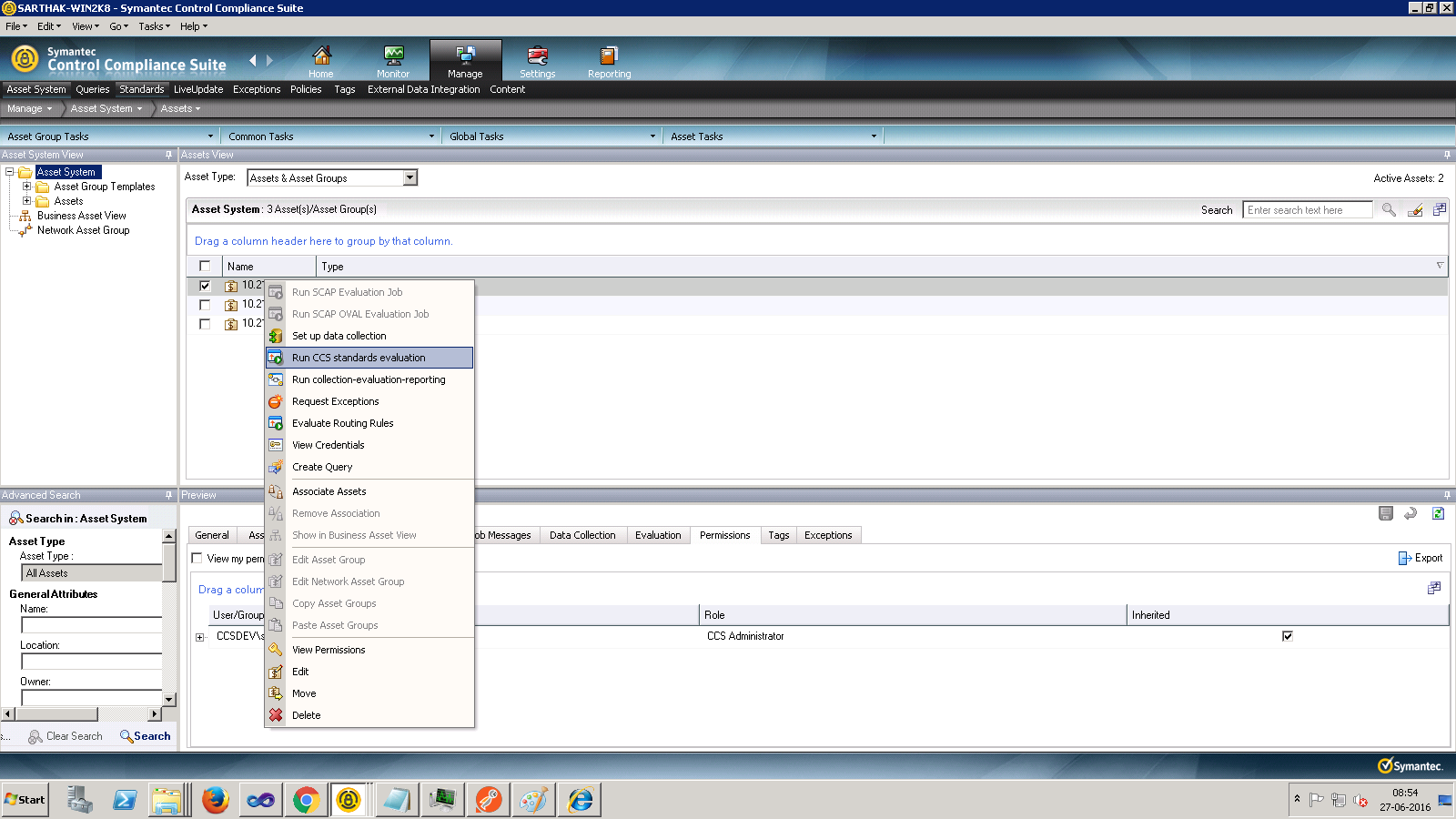
**About checks**

A check is a test that is performed against one or more assets to determine a pass

or a fail status.

A check is composed of one or more check expressions. Multiple check expressions

can be joined through operators to form a check formula.



**About data collection jobs**

You create a data collection job to collect data from the assets for specific standards.

The information that you specify during the data collection process is saved in the

data collection job. Hence you do not need to specify the collection criteria every

time you perform the collection process. Data collection jobs can be scheduled to

run at predefined intervals. The jobs can also be modified and deleted.

**About evaluation jobs :**

You create an evaluation job to evaluate the assets in your organization against

specific standards. The information that you specify during the evaluation process is saved in the evaluation job. Hence, an evaluation job lets you perform the evaluation process repeatedly without having to specify the evaluation criteria again. Evaluation jobs can be scheduled to run at predefined intervals. You can modify and delete the

evaluation jobs.

**About compliance score :**

The compliance score is a percentage value between 0 and 100 that represents

the level of adherence to a standard. This score is derived from the checks that are

present in a standard. The checks in the Not Applicable status are not considered when you calculate the compliance score. The compliance score is available when you evaluate an asset against one or more standard. The result of the evaluation process provides the compliance and the risk score.

**About compliance score calculation :**

The Control Compliance Suite uses the following formulae to calculate the

compliance score:

No. Of checks Passed / (Total number of checks – Not Applicable checks)

Compliance score of STD against an Asset = No. Of checks Passed / (Total number

of checks – Not Applicable checks)

**About risk score :**

In Control Compliance Suite, a risk score is used to quantify the risk that is

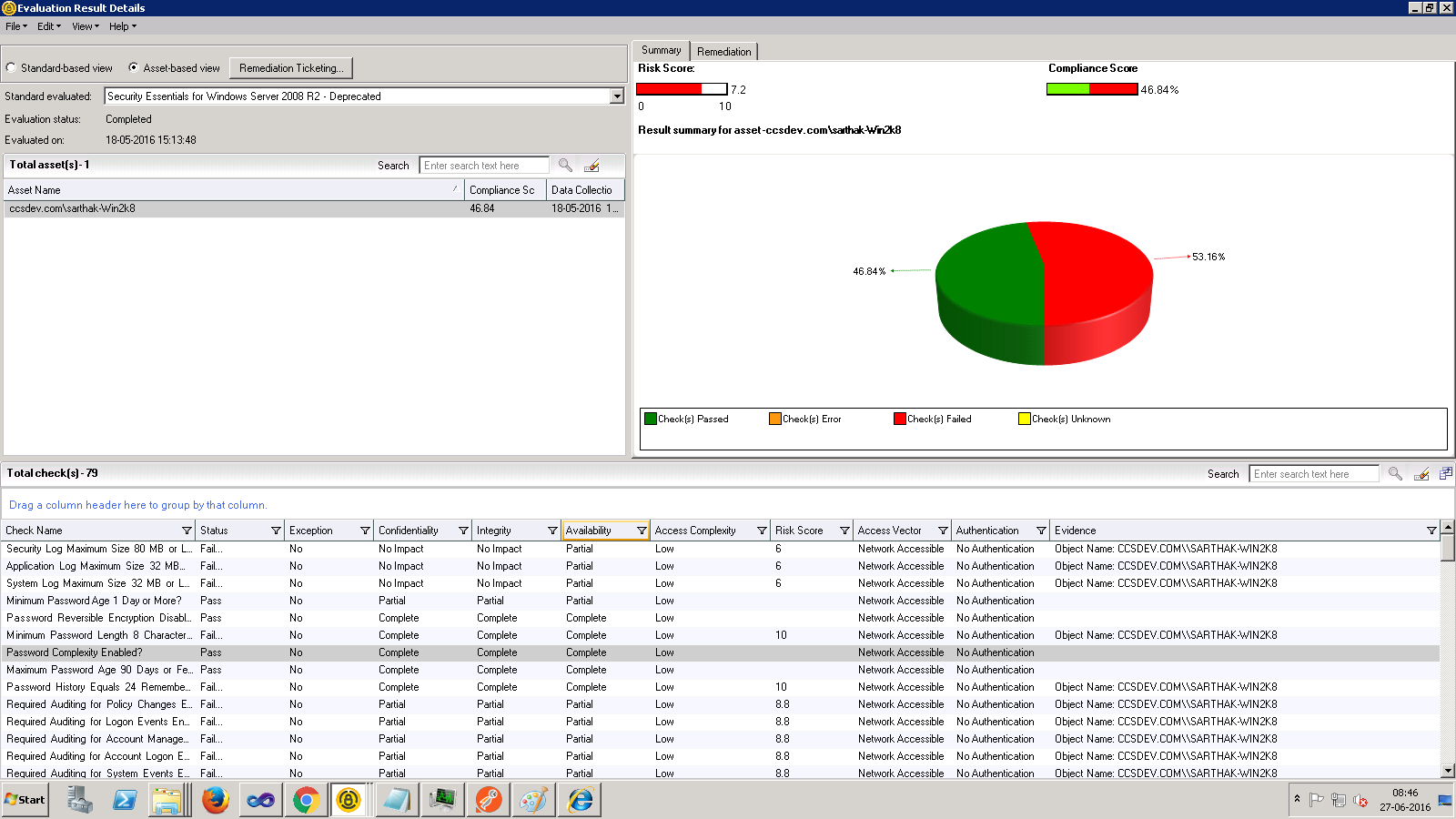
associated with an asset in your organization. The risk score is calculated on the basis of the CIA values for an asset and the risk attributes of a check. You should give due consideration before you specify these values in the product. You can specify the asset CIA values through the assets details pane or with the pre rules in the asset view.

You can specify the check risk attributes through the checks details pane or at the

time of check creation.

The risk calculations are based on the Common Vulnerabilities Scoring System (CVSS) version 2.

This is the end result of running a data collection-evaluation job on a particular asset



# 5.5 Test cases considered :

The generated standard was compared against the CCS standard which was manually developed before using SCAP standards issued by the National Institute of Standards and Technology (NIST) . Any discrepancy found at each stage was resolved by making code amendments by adding support for new objects and states.

# Software tools used : Microsoft Visual studio 2008

# 6. Conclusion and future scope of work

My project on file monitoring system for windows was successfully implemented after going through all stages of developments and amendments. It was a very enriching experience as I learnt about the various stages of software development

through it. I not only enhanced my coding skills through it, but also learnt about static and dynamic analysis of the code.

In the future, this module is scoped to be implemented on agentless models too in Control Compliance Suite. It will also be enhanced to work with different flavors of UNIX. In future, we had also planned to add an UI for module input and template input.

My second project on conversion of Security Content Automation protocol standards to proprietary standards was yet another enriching experience for me. To realize how intricate things could get to define even the basic vulnerability assessment models and how unknowingly we use it everyday, so knowingly was truly amazing.

We were able to successfully convert benchmarks for Windows Server 2012, Windows 8 and Windows 7 platforms. As this project was a Proof Of Concept(POC), we had planned to enhance the tools so that it will also serve the purpose for all Windows and UNIX platforms.

All in all, my internship semester transformed me from a coder to a Software Developer and the journey was truly exciting.

# 7. References :

1. <https://scap.nist.gov/>
2. https://en.wikipedia.org/wiki/**Security**\_**Content**\_**Automation**\_**Protocol**
3. https://access.redhat.com/blogs/766093/posts/1976103
4. <https://www.first.org/cvss/grance-tim-slides.pdf>
5. <https://confluence.ges.symantec.com/pages/viewpage.action?pageId=86408529>