UTELISYS THREAT
MANAGEMENT

Allow students to authenticate with federation through the Utelisys Threat Management before gaining Internet access

Student thesis for
System and Network Engineering
Hogeschool van Amsterdam
Chris Cha
514919
May 22, 2012
Preface

This thesis is related to the research activities for further expanding the Utelisys Threat Management of Utelisys Communications B.V.

The goal of this thesis is to introduce the readers with federation and Utelisys Threat Management. This will further lead to the concept and overview of the implementation. The target audiences are System and Network Engineers and the graduation committee. To comprehend this thesis, basic knowledge of C, HTML, PERL and PHP is required.

Chris Cha
Amsterdam, May, 2012
Summary

Utelisys is a service provider of data and telecommunication services within the Benelux region. The focus of Utelisys is to provide Internet connections in schools, businesses, student accommodations, hotels, institutions and to other service providers. Besides Internet connections, they are also providing TV streaming at the accommodations.

Utelisys is a company that is trying to expand their current software by including authentication to its current software. Utelisys has unique naming software called the Utelisys Threat Management which is based on the software PacketFence for managing the Internet connections that they offer to the users at the student accommodations.

Utelisys is working with SURFnet to provide Internet connections at the accommodations. SURFnet requires Utelisys to have better management over the users who are making use of their network connection. SURFnet is very strict with users who access the Internet, because they are a non-profit organization that ensures researchers, instructors, and students can simply and effectively work together with the aid of ICT [2].

Utelisys wants the students to authenticate themselves with the credentials they have obtained from their institution before they can access the Internet at the student accommodations. To achieve this, they want to expand the Utelisys Threat Management by including federation to its arsenal. The project will lead to achieving the main goal of this thesis “allow students to authenticate with federation through the Utelisys Threat Management before gaining Internet access”. This is achieved by fulfilling the following project objectives:

- Allow authentication process while limiting Internet access through the Utelisys Threat Management before authentication
- Integrate federation software with the Utelisys Threat Management
Acknowledgement

This project would not be possible if not for Mr. Maikel van der Roest. He was my advisor and guider during this project. He was a very helpful and patient person. Personally, I would like to thank him for his kindness, patience and support throughout my internship. I would also like to thank the co-workers of Utelisys for providing me the resources needed to make this research possible, especially Mr. Jeremy Martijn and Mr. Michiel Timmers. This project would not be possible if not for the help of Douwe van der Meer, he was my coordinator.
# Table of Contents

Preface ................................................................................................................................. II
Summary .................................................................................................................................. III
Acknowledgement ................................................................................................................ IV
List of Abbreviations ............................................................................................................. VII
1 Introduction .......................................................................................................................... 8
2 Utelisys ................................................................................................................................. 9
3 General description of project ............................................................................................ 10
   3.1 Current Situation ........................................................................................................... 10
   3.2 Problem ......................................................................................................................... 10
   3.3 Desired Situation ........................................................................................................... 11
   3.4 Project Objectives ........................................................................................................ 11
4 Plan of Approach ................................................................................................................ 12
   4.1 Waterfall Method .......................................................................................................... 12
      4.1.1 Analysis ................................................................................................................. 12
      4.1.2 Design .................................................................................................................. 12
      4.1.3 Implementation ................................................................................................... 12
      4.1.4 Test ..................................................................................................................... 12
      4.1.5 Maintenance ....................................................................................................... 12
   4.2 Project Organization ..................................................................................................... 13
   4.3 Project Risks ................................................................................................................ 13
   4.4 Project Costs ................................................................................................................ 14
   4.5 Global Project Planning ............................................................................................... 14
5 Preliminary Research .......................................................................................................... 15
   5.1 Traditional Authentication Method .............................................................................. 15
   5.2 What is Federation? .................................................................................................... 15
      5.2.1 Service Provider .................................................................................................. 16
      5.2.2 Identity Provider ................................................................................................. 16
   5.3 SURFfederatie Gateway ............................................................................................. 16
   5.4 SimpleSAMLphp ........................................................................................................... 17
      5.4.1 SimpleSAMLphp federation ............................................................................... 17
      5.4.2 Problem .............................................................................................................. 19
      5.4.3 Solution ............................................................................................................... 19
   5.5 Utelisys Threat Management ....................................................................................... 20
      5.5.1 Access Control .................................................................................................... 20
      5.5.2 Authentication Methods ..................................................................................... 20
      5.5.3 How PacketFence Works .................................................................................. 21
   5.6 Results/Findings .......................................................................................................... 22
      5.6.1 ZXID .................................................................................................................. 22
      5.6.2 Decision ............................................................................................................... 22
6 Design .................................................................................................................................. 23
   6.1 Utelisys Threat Management and ZXID ...................................................................... 23
   6.2 Registration page ......................................................................................................... 23
7 Implement/Test ................................................................................................................... 25
   7.1 Private network ............................................................................................................. 25
      7.1.1 Solution ............................................................................................................... 26
<table>
<thead>
<tr>
<th>Section</th>
<th>Title</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>7.1.1.1</td>
<td>Proxies</td>
<td>26</td>
</tr>
<tr>
<td>7.1.1.2</td>
<td>Apache ProxyPass</td>
<td>26</td>
</tr>
<tr>
<td>7.2</td>
<td>Automatic Recognition</td>
<td>27</td>
</tr>
<tr>
<td>7.2.1</td>
<td>Desired process</td>
<td>27</td>
</tr>
<tr>
<td>7.2.2</td>
<td>Problem</td>
<td>27</td>
</tr>
<tr>
<td>7.2.3</td>
<td>Solution</td>
<td>28</td>
</tr>
<tr>
<td>8</td>
<td>Reflection</td>
<td>29</td>
</tr>
<tr>
<td>9</td>
<td>Conclusion</td>
<td>30</td>
</tr>
<tr>
<td></td>
<td>Definition of Terms</td>
<td>31</td>
</tr>
<tr>
<td></td>
<td>References</td>
<td>33</td>
</tr>
<tr>
<td></td>
<td>Appendices Maintenance</td>
<td>34</td>
</tr>
<tr>
<td></td>
<td>Appendix SimpleSAMLphp Federation</td>
<td>34</td>
</tr>
<tr>
<td></td>
<td>Appendix ZXID</td>
<td>35</td>
</tr>
<tr>
<td></td>
<td>ZXID Federation</td>
<td>35</td>
</tr>
<tr>
<td></td>
<td>Appendix PacketFence Apache</td>
<td>36</td>
</tr>
<tr>
<td></td>
<td>Appendix ProxyPass</td>
<td>36</td>
</tr>
<tr>
<td></td>
<td>Appendix Selection Page</td>
<td>37</td>
</tr>
<tr>
<td></td>
<td>Appendix Automatic Recognition</td>
<td>37</td>
</tr>
</tbody>
</table>
### List of Abbreviations

<table>
<thead>
<tr>
<th>Abbreviation</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>CENTOS</td>
<td>Community ENTerprise Operating System</td>
</tr>
<tr>
<td>CSS</td>
<td>Cascading Style Sheets</td>
</tr>
<tr>
<td>DHCP</td>
<td>Dynamic Host Control Protocol</td>
</tr>
<tr>
<td>DNS</td>
<td>Domain Name Server</td>
</tr>
<tr>
<td>HTML</td>
<td>HyperText Markup Language</td>
</tr>
<tr>
<td>HTTPD</td>
<td>HyperText Transfer Protocol Daemon</td>
</tr>
<tr>
<td>HTTPS</td>
<td>HyperText Transfer Protocol Secure</td>
</tr>
<tr>
<td>IP</td>
<td>Internet Protocol</td>
</tr>
<tr>
<td>LDAP</td>
<td>Lightweight Directory Access Protocol</td>
</tr>
<tr>
<td>MAC-Address</td>
<td>Media Access Control Address</td>
</tr>
<tr>
<td>MINI-HTTPD</td>
<td>Mini HyperText Transfer Protocol Daemon</td>
</tr>
<tr>
<td>NAT</td>
<td>Network Address Translation</td>
</tr>
<tr>
<td>P2P</td>
<td>Peer-to-Peer</td>
</tr>
<tr>
<td>PC</td>
<td>Personal Computer</td>
</tr>
<tr>
<td>PERL</td>
<td>Practical Extraction and Report Language</td>
</tr>
<tr>
<td>PHP</td>
<td>P Hypertext Preprocessor</td>
</tr>
<tr>
<td>RADIUS</td>
<td>Remote Authentication Dial-In User Service</td>
</tr>
<tr>
<td>RFC</td>
<td>Request for Comments</td>
</tr>
<tr>
<td>SNMP</td>
<td>Simple Network Management Protocol</td>
</tr>
<tr>
<td>SSH</td>
<td>Secure Shell</td>
</tr>
<tr>
<td>SSL</td>
<td>Secure Socket Layer</td>
</tr>
<tr>
<td>SSO</td>
<td>Single Sign On</td>
</tr>
<tr>
<td>SVN</td>
<td>Subversion</td>
</tr>
<tr>
<td>SWIG</td>
<td>Simplified Wrapper and Interface Generator</td>
</tr>
<tr>
<td>URL</td>
<td>Uniform Resource Locator</td>
</tr>
<tr>
<td>VLAN</td>
<td>Virtual Local Area Network</td>
</tr>
<tr>
<td>XML</td>
<td>eXtensible Markup Language</td>
</tr>
</tbody>
</table>
1 Introduction

After almost five years of studying at the Hogeschool van Amsterdam, it is time to put the theoretical learning’s into practice. Therefore, during the final year of the education, a compulsory project needs to be conducted that will be fully approved by the Hogeschool van Amsterdam. At the “Opdrachtensite” of the institute of computer science, the project that Utelisys described has immediately caught my interest. This project will provide me greater knowledge in the System, and Network engineering field. Also, along the way, I will develop my skills in the programming field.

This thesis is written for my graduation at the Hogeschool van Amsterdam. This project will be conducted at Utelisys communications B.V. at Pietersbergweg 15, Amsterdam Zuid-Oost.

This thesis is separated into three parts. The first part, chapters two to four of this thesis, will explain who Utelisys is, give a general description of the project and the plan of approach.

The second part, chapters five to seven, describes the preliminary research, design and implementation of the project.

The final part, chapters eight and nine, gives the reflection and concludes with “allow students to authenticate with federation through the Utelisys Threat Management before gaining Internet access”.
2 Utelisys

Utelisys is a service provider of data and telecommunication services within the Benelux region. The focus of Utelisys is to provide Internet connections in schools, businesses, student accommodations, hotels, institutions and to other service providers [1]. Besides Internet connections, they are also providing TV streaming at the accommodations.

The organization of Utelisys consists of three directors (see figure 1). The directors and Maikel van der Roest manages all the workers below them in the figure. The other developers (software designers) that are working with Utelisys are not shown in this figure.

![Organizational structure](image-url)
3 General description of project

Authentication is an important aspect of the term security. Authentication is the process where an individual identifies themselves by means of biometrics, passwords, username, etc. The person claiming to have access to a specific resource is a legitimate user\(^1\).

Utelisys is a company that is trying to expand their current software by including authentication to its current software.

Utelisys has unique naming software called the Utelisys Threat Management which is based on the software PacketFence for managing the Internet connections that they offer to the users at the student accommodations.

3.1 Current Situation

In the current situation, the student can access the Internet at the accommodation by only accepting the policy of Utelisys Threat Management at the registration page. The student will get an E-mail with a link where they need to click and obtain permanent Internet access until the date has expired.

3.2 Problem

Utelisys is working with SURFnet to provide Internet connections at the accommodations. SURFnet requires Utelisys to have better management over the users who are making use of their network connection. SURFnet is very strict with users who access the Internet, because they are a non-profit organization that ensures researchers, instructors, and students can simply and effectively work together with the aid of ICT \(^2\).

Officially, when a student resides at a student accommodation, they need to sign a contract with the administration of the accommodation. Basically, this should prove that the users are student and attending school. This should provide a form of authentication to the administration of the accommodation. Therefore, the students are allowed to access the Internet\(^2\) or make use of the resources from the accommodation.

However, the students might not have received their registration papers on time, the students have graduated or the students have resigned from the institution. This can be determined by the administration of the accommodation, but this will require an extra effort from the administration. The administration of the accommodation will need to ask every student either via mail or post to prove that they are attending school.

---

\(^1\) Entity theft or fraud, stolen identifications, etc are possible to alter this term.
\(^2\) Not only the student accommodations get Internet connections, also business partners, but using separate fiber optic connection.
3.3 Desired Situation

The desired situation is to further expand the Utelisys Threat Management by including federation to its arsenal. With federation, they can have better management over the users who are accessing the network and reduce the amount of work for the administration of the accommodation. The students will need to authenticate with the credentials that they have obtained from school before they can access the Internet.

The benefits with the new improvement are:

- More control over the users who accesses the Internet (The user is a legitimate student)
- When the user is not a student, Internet access would not be given, because their credentials aren’t valid anymore
- Keeping the bandwidth low (Limiting the users who are accessing the network and allow other users to experience faster Internet)

The drawbacks with the new improvement are:

- Students can use other student’s legitimate credentials to gain Internet access
- It will be very hectic for the helpdesks of Utelisys (There will be a lot of complaints from the students i.e. students got their registration information late or decided to change studies, etc.).

3.4 Project Objectives

The main goal of this project is to “allow students to authenticate with federation through the Utelisys Threat Management before gaining Internet access”.

The project objectives are:

- Allow authentication process while limiting Internet access through the Utelisys Threat Management before authentication
- Allow federation software to operate with the Utelisys Threat Management
4 Plan of Approach

4.1 Waterfall Method

To fulfill this project, the waterfall model was used. This model is a sequential of process that starts from the top to the bottom and will go down to the following step with every progress [3]. The reason for choosing this model is, because with every completion of the objectives, it’s possible to proceed to the next step. The following part shows the parts of the model used:

- Analysis
- Design
- Implementation
- Test
- Maintenance

4.1.1 Analysis

As for every project, before we can create or design or implement an infrastructure, an analysis needs to be done on the current software and the desired software to be used. An example can be seen on chapter 5 preliminary research of this thesis. During all the phases, we will be working on the UNIX environment based on CENTOS. The servers that we will be using are based on an x86_64 architecture which is a 64 bit operation system.

4.1.2 Design

During the design phase, we will create a design based on the analysis that was done. This can be reflected in chapter 6 Design.

4.1.3 Implementation

During the implementation phase, we need to achieve the goals that were set for this project described in chapter 3.4 Project Objectives. During this phase, we will also start by implementing the design described previously.

4.1.4 Test

For this project, a test phase needs to occur. During the test phase we will be simultaneously testing and implementing the design. We will be testing to see if the implementation is working according to plan or if we need to make some changes according to the design. However, changes may occur during the project which is normal considering the complexity of the subject.

4.1.5 Maintenance

The last part of the model is to provide Utelisys with the code needed to achieve the results. This will be found in chapter Appendix Maintenance of this thesis and would be the technical part of this project.
4.2 Project Organization

The team of this project consists of only one person and that would be me. I will take full responsibility by working on the Utelisys Threat Management. The project was carried out from the 21st of September 2011 until the 20th of February 2012.

The following engagements and acceptances should be present during this project:

- The co-workers at Utelisys comply by helping me with the necessary resources needed for this project.
- The co-workers at Utelisys comply by assisting me with the proper feedback on a topic that was vague.

For efficiency, all information and documents of the project should be stored in a SVN server that is going to be installed in Utelisys. This server allows the documents to be retrieved through TortoiseSVN and can be accessed anywhere through the Internet with authentication. Also, all the other servers\(^3\) for this project can be accessible anywhere around the world. This is handy, because the research of the project can be conducted after working hours. For security reasons, the firewall will only allowed secure access to the servers (SSH). All other connections besides web access and secure web access will not be allowed.

4.3 Project Risks

Project risks are risks that might affect the chances for completing the objectives of this project, even being sick can impede with the process.

The following risks are taken into consideration.

- Hardware failure
- Not enough resources (computers, switches, cables, etc)
- Accidental deletion of a software which causes the server to dysfunction and leads to the reinstalling of a new server
- Specification which are not conform the end result
  - Failed agreements
  - Environmental disasters

\(^3\) SimpleSAMLphp, ZXID and PacketFence server
4.4 Project Costs

When a company is seeking ways to solve a problem, it is likely that they will hire interns for their projects. A beneficial reason to hire an external person would be to bring new talents to the team and for the company to give the students an opportunity to learn and gain experience by working in the field. This will require some cost on the company for the interns’ efforts, but this will only be done in a short term which is a win-win situation for both parties. Other costs such as hardware or software are already available or free (i.e. CENTOS, SimpleSAMLphp, ZXID, Packetfence, Servers).

4.5 Global Project Planning

This is the global project planning (see figure 3) of this project. It wasn’t necessary to put task such as the completion by installing SimpleSAMLphp, ZXID or PacketFence. The global project planning only shows the beginning and end of every goal. The fourth task of this project was not fully completed, hence the short date.

<table>
<thead>
<tr>
<th>ID</th>
<th>Task Name</th>
<th>Start</th>
<th>Finish</th>
<th>2011</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Oct</td>
</tr>
<tr>
<td>1</td>
<td>SimpleSAMLphp</td>
<td>9/21/2011</td>
<td>10/21/2011</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>PacketFence</td>
<td>10/24/2011</td>
<td>11/14/2011</td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>NAT and Reverse Proxy</td>
<td>12/19/2011</td>
<td>12/23/2011</td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>ProxyPass</td>
<td>1/20/2012</td>
<td>1/31/2012</td>
<td></td>
</tr>
<tr>
<td>6</td>
<td>Automatic recognition</td>
<td>2/1/2012</td>
<td>2/20/2012</td>
<td></td>
</tr>
<tr>
<td>7</td>
<td>Thesis</td>
<td>9/21/2011</td>
<td>1/9/2012</td>
<td></td>
</tr>
<tr>
<td>8</td>
<td>Thesis v 1.0</td>
<td>1/20/2012</td>
<td>3/21/2012</td>
<td></td>
</tr>
</tbody>
</table>

Figure 3 Global project planning
5 Preliminary Research

5.1 Traditional Authentication Method

In traditional authentication method, if a student wants to access the Internet at an institution, they need to authenticate themselves with the credentials (see figure 4) they have obtained from their institution. For this to happen, the students need to prove that they are part of the institution by paying the tuition fee.

![Figure 4 Traditional authentication method]

However, with the traditional authentication method, Internet access can only occur inside of an institution. A possible method that Utelisys could have make use of was an authentication server (for example a RADUIS server). This server needs to be synchronized with the authentication server of all the institution. This will allow Utelisys to know all the credentials from the students. It will require too much work and it is not certain that the credentials would be provided to Utelisys from the institution. An enormous database would also be required and should not be the best solution for managing (database needs to be frequently updated). Therefore, the solution was to introduce federation. With federation it is possible to allow the students to authenticate themselves with the credentials that they have obtained from their institution for Internet access from another location i.e. student accommodations without requiring Utelisys knowing the credentials of the students.

5.2 What is Federation?

Federation server is a key component for securely\footnote{http://www.w3.org/TR/xmldsig-core/#sec-X509Data} transporting SSO tokens (an assertion) across applications that is trusted by the federation \cite{4}. Federation uses the Security Assertion Markup Language (SAML), developed by the Security Services Technical Committee of OASIS. It is an XML-based framework for communicating user authentication, entitlement, and attribute information. As its name
suggests, SAML allows business entities to make assertions regarding the identity, attributes, and entitlements of a subject (an entity that is often a human user) to other entities, such as a partner company or another enterprise application. [5]

For federation servers to work, a service provider (entity) and identity provider (entity) are needed which the federation trusts.

### 5.2.1 Service Provider

A service provider is a server running a web application in a trusted federation zone that relies on an identity provider to offer attributes to the users on behalf of the service provider.

### 5.2.2 Identity Provider

An identity provider is an organization that is running their own authenticating server users (institutions or any businesses that contains their own authentication method e.g. LDAP server or Active Directory to authenticate users) in a trusted federation zone that is responsible for authenticating local or external users. It also provides optional attributes back to the service provider [5].

### 5.3 SURFfederatie Gateway

SURFnet, uses a federation server called the SURFfederatie Gateway. SURFfederatie Gateway resides between multiple service providers and identity providers (see figure 5) [6].

This server acts like a bridge between the service provider and identity provider, because information requested from the service provider to the identity provider is conveyed through the SURFfederatie Gateway. Information returned to the service provider (attributes) is conveyed through the SURFfederatie Gateway. The SURFfederatie Gateway has all the SAML (metadata)
information of the service provider and the identity provider that they trust. This way it reduces the amount of work to link all the metadata of the identity provider to the service providers. Utelisys has a program called SimpleSAMLphp (service provider) which participates in federation, also known as Starbase. The SimpleSAMLphp software is connected to the SURFfederatie Gateway (SURFnet).

5.4 SimpleSAMLphp

SimpleSAMLphp is an award-winning application written in native PHP that deals with authentication [6]. This software allows an existing authentication infrastructure to be extended with single sign on capabilities using the SAML (2.0) protocol. It can be used as an identity provider for existing authentication sources and as a service provider using identity providers to authenticate users. [7]

5.4.1 SimpleSAMLphp federation

A SimpleSAMLphp federation process is a process where a user can select which identity provider they want to authenticate at. In this section we will show an example of the SimpleSAMLphp federation. The service provider will be the SimpleSAMLphp software. The idea with SimpleSAMLphp is to connect SimpleSAMLphp (service provider) to another SimpleSAMLphp so it can act like the SURFfederatie Gateway (which we will further refer to as the bridge). Also, we need to create an account at https://openidp.feide.no/ (Feide test server). This account is an identity provider to provide authentication and attributes back to the service provider. This account will only contain test information. We are using SimpleSAMLphp, because this was the simplest way to understand how federation works.

Step 1

When a user wants to connect to the Feide test server in a SimpleSAMLphp federation process, the user needs to login in the page of the service provider (see figure 6). From there, the user will get a page where he/she can select which identity provider he/she wants to connect to. In this case the user can only select the bridge, because the service provider only has the metadata of the bridge. Once the user has selected the bridge, the user will be redirected to the bridge page.
Step 2

At the page of the bridge (see figure 7), the user will get a list of identity providers that the bridge has. In this case the bridge only has the Feide test server\(^5\). When the user selects the Feide test server, the user will be redirected to the login page of the Feide test server.

![Figure 7 Bridge page](image7.png)

Step 3

At the webpage of the Feide test server, the user needs to authenticate himself/herself (see figure 8). If the login is successful, the user will be redirected invisibly back to the service provider via the bridge.

![Figure 8 Identity provider page](image8.png)

\(^5\) In the standard installation, there is more than one identity providers included.
Step 4

The attributes of the user are seen on the website of the service provider (see figure 9).

<table>
<thead>
<tr>
<th>Your attributes</th>
</tr>
</thead>
<tbody>
<tr>
<td>User ID</td>
</tr>
<tr>
<td>Mail</td>
</tr>
<tr>
<td>Given name</td>
</tr>
<tr>
<td>Surname</td>
</tr>
<tr>
<td>Common name</td>
</tr>
<tr>
<td>Person’s principal name at home organization</td>
</tr>
<tr>
<td>Persistent pseudonymous ID</td>
</tr>
<tr>
<td>urn:oid:0.9.6.3.4.1.11001101</td>
</tr>
<tr>
<td>urn:oid:0.9.6.3.4.1.11001101</td>
</tr>
<tr>
<td>urn:oid:2.5.4.42</td>
</tr>
<tr>
<td>urn:oid:2.5.4.4</td>
</tr>
<tr>
<td>urn:oid:2.5.4.3</td>
</tr>
<tr>
<td>urn:oid:1.3.6.1.4.1.5923.1.1.1.6</td>
</tr>
<tr>
<td>urn:oid:1.3.6.1.4.1.5923.1.1.1.10</td>
</tr>
<tr>
<td>groups</td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
</tbody>
</table>

Figure 9 Attribute page at service provider

5.4.2 Problem

Step 1 and 2 of this request is not efficient, because this requires the user to select twice before being authenticated. This will not allow the bridge to act like the SURFfederatie Gateway.

5.4.3 Solution

With SimpleSAMLphp, a solution was to make use of scoping, because scoping will allow a service provider to specify a list of identity providers which allows the bridge to handle the list. The user can select an identity provider he/she wants to authenticate at, but this will only occur at the page of the bridge. The link location of the bridge was added to allow the user to see the page of the bridge instead of the service provider page. However, this is not the recommended solution, because the service provider cannot create their own design page. The selection page should occur at the service provider page. This is possible by making some modifications in the program of the service provider. (See Appendix SimpleSAMLphp Federation for the code). As simple as the name stated in SimpleSAMLphp, it was not quite that simple to modify the software, because of the huge programming codes needed to be understood.

---

6 Adding 'idp' => 'http://xx.xx.xx.xx/simplesaml/saml2/idp/metadata.php' to the authsources.php file
5.5 Utelisys Threat Management

Utelisys Threat Management is based on the software PacketFence. PacketFence is a fully supported, trusted, Free and Open Source (General Public License) system. PacketFence controls how a device can access a network. This software is written in the PERL programming language which is a highly capable, feature-rich programming language with over 23 years of development. [8]

5.5.1 Access Control

Access control is a method that controls how the user can access the network. PacketFence offers the following in VLAN mode:

- MAC-Detection (Detects the device)
- Registration VLAN (Allows a device to be registered)
- Isolation VLAN (A problematic device that can cause network problems i.e. a device that is infected by a virus would be put on this VLAN)
- Normal VLAN (Allows Internet access)

5.5.2 Authentication Methods

Authentication method is the method that requires the user to authenticate before they can access the Internet. This software offers different authentication method:

- LDAP
- Microsoft Active Directory
- RADIUS
- Local user file
- Guest Registration (This approach is the current situation for allowing the students to access the Internet, however it is a form of authentication)

All these authentication methods are potential for allowing the students to authenticate before they can get access to the Internet, but would require more work for Utelisys. Every student needs obtain their own credentials from Utelisys. It is not the best solution for solving the problem for authenticating the students before they can access the Internet and does not solve the problem defined in chapter 3.2 Problem.

---

7 More features are available, but the once that were used during this project are mentioned.
5.5.3 How PacketFence Works

When a user connects a device to the switch, an SNMP trap will be sent from the switch to the PacketFence server. The PacketFence server will control if the device’s MAC address exists in its database. It checks if the device has to be registered, isolated or has been registered. If the device needs to be registered, the device will be sent to the registration VLAN 2 and the PacketFence server will assign a private IP address\(^8\) to the device using its local DHCP server (see figure 10).

![Semi-Utelisys infrastructure diagram](image)

The PacketFence server will then save the MAC address, web browser information, operating system type, username of PC, etc of the device as unregistered in the database. When the user inserts their personal information at the guest registration page, the user will be able to connect to the Internet (normal VLAN xxx) with a public IP address. The PacketFence server will then put the device status as registered.

On the other hand, if the device needs to be isolated, it will be sent to the isolation VLAN 3. A violation will be triggered to warn the administrator that a device has been affected by a virus. If the device’s status is registered, the user can gain Internet access (normal VLAN xxx) without requiring them to re-authenticate (see Appendix PacketFence Configuration for complete configuration).

---

\(^8\) IPv4 address RFC 1918
5.6 Results/Findings

As a result, the SimpleSAMLphp software needs to operate with the Utelisys Threat Management. The Utelisys Threat Management is also written in the PERL language. Therefore, searching in Google, we have found a software which is also SAML based and it written in the PERL programming language. This software is called ZXID.

5.6.1 ZXID

ZXID is light weight, has a small foot print, and is implemented in C. Scripting languages are supported using SWIG, including Perl, PHP and Java. The "full stack" nature of ZXID means it’s self contained and has minimal external library dependencies (see downloads).

Targeted Federated Identity Standards

• SAML 2.0 (fully done, Service Provider and Identity Provider roles) [9]

On the other hand, with ZXID, some modifications in the program also needed to be done to allow it to act in federation (see section 5.4.1 SimpleSAMLphp federation). (See appendix ZXID Federation for the code).

The ZXID can either run with HTTPD or MINI_HTTPD. Using HTTPD⁹ would require more configurations than MINI_HTTPD. But the MINI_HTTPD works with a different port number (8443) when using SSL. Therefore, we used the HTTPD, because it runs on the standard port number with SSL (443).

5.6.2 Decision

The table below shows a comparison between the ZXID and SimpleSAMLphp software. There isn’t much difference between the two by comparing its functionality. The main difference lies on the module support, because the ZXID module is already written in PERL and this is what Utelisys wanted to use with the Utelisys Threat Management. This should provide better maintenance when the Utelisys Threat Management is upgraded to a newer version, where the code doesn’t need to be changed and requires less man hour to maintain the software.

<table>
<thead>
<tr>
<th></th>
<th>SimpleSAMLphp</th>
<th>ZXID</th>
</tr>
</thead>
<tbody>
<tr>
<td>Identity provider</td>
<td>x</td>
<td>x</td>
</tr>
<tr>
<td>Service provider</td>
<td>x</td>
<td>x</td>
</tr>
<tr>
<td>Universally support (PHP, PERL, JAVA)</td>
<td>Only PHP</td>
<td>All</td>
</tr>
<tr>
<td>SAML support</td>
<td>x</td>
<td>x</td>
</tr>
<tr>
<td>PERL module</td>
<td></td>
<td>x</td>
</tr>
<tr>
<td>PHP module</td>
<td>x</td>
<td></td>
</tr>
</tbody>
</table>

6 Design

6.1 Utelisys Threat Management and ZXID

Based on the preliminary research, during this phase we will be creating the design for this research using the Utelisys Threat Management and ZXID software. The ZXID software will be connected to the Starbase of Utelisys, because the ZXID software is not connected directly to the SURFfederatie Gateway. In figure 11 shows how the design should look like. The ZXID software should be working on the same server as the Utelisys Threat Management.

![Figure 11 Design](image)

6.2 Registration page

At the guest registration page of the Utelisys Threat Management, users can register without requiring them to use password. Below shows (figure 12) an example of a guest registration page.

![Figure 12 Guest registration page](image)

The idea is to allow an extra field to be added at the registration page of the Utelisys Threat Management containing a list of the institution where students can choose and allow the ZXID
software to handle the process. And when the authentication succeeds, the user will get Internet access. In figure 13 shows how the end result should look like in a case.

![Diagram](image.png)

**Figure 13 End-result case**
7 Implement/Test

7.1 Private network

The users need to authenticate with the ZXID software, but the software requires Internet to work and the users on the network are on a private network. Therefore, their needs to be a way for students to authenticate at the institution while being on the private network.

There were two choices to work with; one was allowing the users to access the Internet for a limited amount of time using NAT and the other method was to use the Reverse Proxy Passthrough method of PacketFence.

The first method was not the best choice, because it doesn’t provide sufficient restriction of Internet traffic (P2P, Instant Messaging, E-mail, etc). This will give the users the ability to access the Internet, but it is possible to block the traffic using IPtables which is a firewall software for UNIX. But we will need to work with a squid server which is a proxy server that we can use to block website traffic, but this will require too much configuration and might be very complex to configure within the amount of time for this project.

The other solution provides better Internet access restriction than the first method and requires fewer configurations and it already restricts the Internet traffic. The Reverse Proxy Passthrough of PacketFence allows an infected device to visit another webpage or it allows a user to visit a specific website when they are on an unregistered network or residing on a private network.

The Reverse Proxy Passthrough method will use the reverse proxy of the PacketFence Apache to only allow the users to visit the web pages that are specified in a list containing different URLs. It is also possible to allow a range of IP addresses to be visited, but whenever this is done, all of the contents of a web site are retrievable, but this would not be a concern (website of institution will only be visible). This feature needs to be enabled for this approach to work.

The Reverse Proxy Passthrough method was not functioning when the user was allowed to visit Google at the registration page. When searching for Google, the user gets redirected back to the registration page. As a result, we have chosen the wrong method, because the correct method was to use the ProxyPass of PacketFence. It isn’t certain what the Reverse Proxy Passthrough main usage was. It seems that this method is used to redirect the user to a proxy server. Therefore, we are further using the Proxy Pass of PacketFence and the following section will further explain the concept.

10 /usr/local/pf/addons/captive-portal/proxy-bypass/squid.redirector.pl
7.1.1 Solution

7.1.1.1 Proxies

PacketFence has a method called Proxies [10] which redirects an infected device to another webpage or it allows a user to visit a specific website when they are on private network. The Proxies method will use the ProxyPass of the PacketFence Apache\textsuperscript{11} to only allow the users to visit the web pages that are specified in a list containing different URLs and disallowing other traffics (P2P, Instant Messaging, E-mail, etc). The code below shows an example of the ProxyPass.

\begin{verbatim}
Suppose the local server has address http://wibble.org/; then
   ProxyPass /mirror/foo/ http://foo.com/
will cause a local request for the <http://wibble.org/mirror/foo/bar> to be internally converted into a proxy request to <http://foo.com/bar>.[11]
\end{verbatim}

The federation software accesses different resources (different files) and PacketFence uses a method called rewriterule. This method changes the URL or redirects the user back to the registration page of PacketFence if the resources that are being accessed are not correctly specified in the ProxyPass. This method seemed to work by configuring three servers (Feide Test Server), but whenever we have configured the resources to be access at the fourth server (Hogeschool van Amsterdam), it seems that the user does not get redirected to the authentication page. The user gets redirected back to the registration page. Theoretically speaking, this method should work, but it seems to be a bizarre behavior of either the on the ProxyPass or it doesn’t seem to pass on authentication sessions.

7.1.1.2 Apache ProxyPass

We have tried to use a separate ProxyPass of Apache instead, while running the PacketFence software. For every website we wanted to access we have created a separate Virtual Hosts with the ProxyPass containing different domain name. Virtual Hosts gives a server the possibility to maintain more than one web server on the same machine using the same IP address (see Appendix ProxyPass for an example).

We have tried to configure a website that runs with HTTP connection. The first Virtual Host that was specified in Apache was the first webpage that the user sees. As a test purpose, we have tried to setup the first page to show the home page of a test website. Then we created a link to allow the users to access the other website that were specified, this proven to work.

\footnote{\texttt{/usr/local/pf/addons/captive-portal/proxy-bypass/squid.redirector.pl}}
Since all the connections should be with SSL, the next step was to test the connectivity with HTTPS. When we have tried to visit another webpage other than the first webpage defined in the first Virtual Host, the webpage seems to retrieve the information from the first Virtual Host defined in Apache. The problem was caused by the missing configuration in Apache with the Server Name Indication (SNI).

With SNI, you can have many virtual hosts sharing the same IP address and port, and each one can have its own unique certificate (and the rest of the configuration)\[12\]

The final step was to apply the configuration to the Apache of PacketFence instead of using a separate Apache configuration. When we have tried to restart PacketFence, it seems when Apache has been started first, it doesn’t seem to load the Apache configuration of PacketFence. This will not allow us to see the registration page. Therefore, all the configuration of Apache needs to be put into the Apache configuration of PacketFence and we would need to stop the normal Apache from running. Also, we need to apply the configuration before the rewriterule in the Apache otherwise the Virtual Host defined will not work.

Figure 14 shows that the PacketFence, ZXID and ProxyPass should be working on the same server (For test purposes, we have used the ZXID as a separate server to test if the ProxyPass also worked with ZXID).

7.2 Automatic Recognition

Utelsys Threat Management and ZXID are two different programs; therefore some modification needs to be done to make sure that both of the software is interoperable. When the authentication succeeds with ZXID, a session id is being returned as an attribute. There might be different ways to allow the Utelsys Threat Management to know that the user has been authenticated with this session id.

7.2.1 Desired process

As explained in section 6.1.2 registration page, the idea was to create a list of institution at the guest registration page. The ZXID code needs to be implemented in PacketFence to allow the software to operate with each other.

7.2.2 Problem

This cannot be possible, because the domain name that PacketFence uses at the registration page is a local or fake domain name. Federation requires real domain name for operation, therefore implementing the ZXID code in PacketFence was not possible. Implementing PacketFence code
into the ZXID software would require too much time and might need a lot of changes on both of the software. The solution that we will be working on will be explained in the following section.

7.2.3 Solution

The solution is to allow the user to select the school that they are attending at the registration page (this will be a custom page created containing a selection box). When the user has chosen his/her institution, the user will be invisibly redirected to the authentication page of the institution that they are attending. (See appendix Selection Page for the code)

When the student has been authenticated, the user will then see the guest registration page instead of the normal ZXID page where a session id is being returned. This page was generated with the use of the PacketFence module (See Appendix Automatic Recognition). When generating the registration page with the module in a PERL script, it seems that the files that are not in a PERL format (HTML and CSS) were not executed as a normal file. The user will only see a plain page containing only text. We needed to make a configuration in Apache of PacketFence to allow files that were not written in PERL format, to be executed as a normal file (See Appendix PacketFence Apache). This took quite some time to achieve.

After this configuration, this will allow the process to work with two different domain names, however one of them is real one (guest registration) and the other one is the fake domain (school selection page).

The idea was to use the same domain name for the fake and real one, but when accessing the selection page of the institution, it seems that Apache tries to access the resource from the ZXID directory instead of the PacketFence directory. This caused the process to fail and not work according to the idea. Cookies could have been the solution, but were not discussed in this research (privacy and security concerns).

At the ZXID page the user needs to enter their personal information. There will be Javascript used for error checking on the fields entered. The reason we are using this method is because the module of PacketFence was used to generate the guest registration page. All the error checking does not allow the code to generate the correct URL of the newly created registration page. The URL could not be used, because when the authentication succeeded, users cannot copy paste the URL to see the registration page. The user will be redirected back to the selection page of the institution instead.

When they click login, the information will be processed by PacketFence. The PacketFence code will automatically control\textsuperscript{12} the user information and redirect the user to the connectivity page for allowing them to get Internet access.

As a result, the process to continue normally like the current situation, but with an extra step which requires users to authenticate before Internet access.

\textsuperscript{12} This is to avoid users by trying to bypass the registration page if no information are being entered at the guest registration page.
8 Reflection

This project has proving to be a value for me during my time at Utelisys. This project helped me integrate the different skills that I have developed during my time at the Hogeschool van Amsterdam and applying them in a practical environment. Most of these the software and OS that were used during my project, I have not worked with before. Therefore, providing me with greater accomplishment and understanding on how to work in a new environment and quickly adapting to the system.

Also what was also important was the teamwork at Utelisys. The most important part was communication. Without any communication, this project would lead to confusion and could have taken longer to achieve than within the period of time required for a student to accomplish his goal for a project.

In short, a lot was learned at Utelisys. The opportunity was given to me by Utelisys to allow me to develop my educational, individual and team skill.

These are the competencies that were realized during this project.

Analyzing

During this project a lot of researches needed to be done on PacketFence and federations. This took quite some time to research, because I have never heard of both of the software. Therefore, it took quite longer than the implementation itself.

Programming

A lot of reading programming code needed to be understood during this project, especially C and PHP. The programming codes were incredibility complicated. I have learned how to read and understand how the programming codes works.

Troubleshooting

One of the toughest assignments was troubleshooting, because not every program ran without errors. Debugging was the most helpful skill learned during this project, because a lot of problems were solved using log files or error logs and it also saved a lot of time. Some problems took longer to solve than expected.
9 Conclusion

After spending quite some time on reading programming codes, applying changes to the software and creating my own test environment, it is time to come up with a solution.

The objectives that needed to be achieved are:

- Allow authentication process while limiting Internet access through the Utelisys Threat Management
- Allow federation software to operate with the Utelisys Threat Management

Objective 1

According to the research, not all of the process worked according to the plan. We could not use the Proxies method of PacketFence to allow the students to authenticate while restricting Internet access, because the process didn’t seem to work with four servers. We needed to work with a separate ProxyPass to achieve this goal and applying this into the Apache configuration of PacketFence.

Objective 2

It seems that the student needs to select their institution before they can authenticate. After they authenticate, they need to type in their personal information. This requires extra step for the students, but it would be nicer if the student could have inserted their information at the first page, but we were not working with cookies. Cookies were not discussed in this research, but would not be the recommended solution, because of privacy and security concerns.

To improve the Utelisys Threat Management we can do the following recommendations:

Recommendations 1

Trigger an SNMP alert to PacketFence server whenever the switch port has been down and allow PacketFence to unregister the device. This will allow the user to register every time whenever they turn off their computer or when they disconnect their PC from the port.

Recommendations 2

Remove the registration page and only allow the user to authenticate at the institution, this will remove the burden for the student to always typing their information at the registration page. This will also allow the Utelisys Threat Management to work with one domain name and remove the problem for allowing the student to get permanent Internet access until their policy expires. This is only possible if the first recommendation is achieved.
Definition of Terms

Assertions - Statements about a user that contains a package made by SAML. SAML defines three different kinds of assertion: authentication, attribute and authorization.

CENTOS (Community ENTerprise Operating System) - Enterprise-class Linux Distribution derived from sources freely provided to the public.

Cascading Style Sheet (CSS) – A format for giving a website or HTML a look and feel.

Domain Name Server (DNS) - It is an internet service that translates domain names into IP addresses.

Dynamic Host Control Protocol (DHCP) - A protocol that assigns dynamic IP addresses to devices on a network.

Feide test server - The Feide OpenIdp is a SAML 2.0 Identity Provider for users that do not have an account in Feide.

HyperText Markup Language (HTML) – A language for creating a website.

HyperText Transfer Protocol Daemon (HTTPD) - Software program that runs in the background of a web server and waits for incoming server requests.

HyperText Transfer Protocol Secure (HTTPS) – Provides encrypted communication and identifies that you are communicating with a legitimate web server.

JAVA - A high level, object-oriented, cross-platform that is developed by Sun Microsystems.

LDAP (Lightweight Directory Access Protocol) - An Internet protocol that programs uses to look up information from a server.

Mini Hypertext Transfer Protocol Daemon (MINI-HTTPD) – A Small web server that should be adequate for low or medium traffic performance.

Media Access Control Address (MAC) - Physical hardware address of the network card.

Network Address Translation (NAT) - Translates private addresses into public addresses

P2P (Peer-to-Peer) - A network protocol that is used for sharing files.

PHP (P Hypertext Preprocessor) - A scripting language that is mostly used for web development.

RADIUS (Remote Authentication Dial-In User Service) – A network protocol that provides authentication, authorization and accounting for a computer.
RFC (Request for Comments) - Defines a standard number for the Internet.

Simple Network Management Protocol (SNMP) - Protocol that manages devices on a network.

Secure Socket Layer (SSL) - Allows an encrypted data communication to be established between a web server and a web browser.

Single Sign On (SSO) - Allows a user to login once with his credentials and access to their resources from multiple applications without requiring them to re-authenticate.

SSH (Secure Shell) - A network protocol that connects two unsecure networks to provide secure data communication.

SVN (Subversion) - A software where computer files and documents can be stored.

SWIG (Simplified Wrapper and Interface Generator) - A software development tool that connects programs written in C and C++ with a variety of high-level programming languages (PHP, JAVA, PERL).

TortoiseSVN - An application that allows files to be retrieved from an SVN server.

URL (Uniform Resource Locator) - A unique address for a webpage.

XML (eXtensible Markup Language) - A markup language that defines a standard set of rules for encoding documents in a format (human-readable and machine-readable)

Virtual Local Area Network (VLAN) - A group of hosts that is connected to the same broadcast domain.
References

[1] Website: Utelisys
   http://www.utelisys.com/over  23 - 10 - 11
   http://community.jboss.org/wiki/FederationServer 17 - 10 - 11
[7] Website: SimpleSAMLphp
   http://simplesamlphp.org/  15 – 10 - 11
[8] Website: The Perl Programming Language
   http://www.perl.org/  25 – 11 - 11
[9] ZXID Home - Open Source IdM for the Masses - SAML SSO
   http://www.zxid.org/index-ol.html  28 - 11 -11
[10] Article: PacketFence version 3.1.0 Administration guide
    http://httpd.Apache.org/docs/1.3/mod/mod_proxy.html#proxypass  24 – 1 -12
[12] Website: SSL with Virtual Hosts Using SNI
    http://wiki.Apache.org/httpd/NameBasedSSLVHostsWithSNI  26 - 1 - 12
Appendices Maintenance

Appendix SimpleSAMLphp Federation

A list of identity provider needs to be created at the service provider page (see example below).

```
$IDPList = array("https://openidp.feide.no" => "Feide OpenIdP - guestusers",
"https://betawayf.wayf.dk" => "DK-WAYF Quality Assurance");

foreach ($name1 as $key=>$value){
    echo '<option value ='.htmlspecialchars($key).'></option>
    echo htmlspecialchars($value) . '</option>;
}
```

The code below shows swapping of the identity provider list value.

```
public function startSSO($idp, array $state) {
    $IDPselected = $idp;
    $idpMetadata = $this->getIdPMetadata($idp);

    $type = $idpMetadata->getString('metadata-set');
    switch ($type) {
        ...
        $this->startSSO2($idpMetadata, $state, $IDPselected);
    }
}
```

The code below allows the identity provider value being passed with one value for scoping ($IDPselected).

```
$state['IDPList'] = $IDPselected;

if (isset($state['IDPList'])) {
    $IDPList = $state['IDPList'];
} else {
    $IDPList = array();
}
```
Appendix ZXID

ZXID Federation

This software didn't have a build in scoping. The scoping value wasn't available in the code. Therefore, a scoping attribute needs to be created for allowing scoping to be possible. Also, an identity provider list at the service provider page needs to be created.

The identity provider list can be put in the zxidhlo.pl file or in the zxidsimple.c. The handiest way is to put it in the zxidhlo.pl file (see code below for the list), because it was easier and requires less understanding of the complicated coding parts of ZXID.

```perl
@list = ("urn:federation:HvA", "https://openidp.feide.no", "https://betawayf.wayf.no");
$size = @list;
print <<<HTML;
</br></br>
</select name=d>
for ($i=0; $i<$size; $i++)
{
    <option value="$list[$i]">  ($list[$i])</option>
}

A 'for' loop needs to be created to test if there is a match between the selected identity provider. If that exists, we need to swap the identity provider to the scope value (cgi->IDPList) and another value (cgi->eid) will contain the link location of the bridge (see example below).

```c
static char scope_list[100] = {"https://openidp.feide.no", "https://betawayf.wayf.no");
int i;
int res;
for( i =0;  i < (sizeof(scope_list)/100); i++)
{
    res = strcmp(scope_list[i],cgi->eid);
    if(!res){
        cgi->IDPList= cgi->eid;
        cgi->pxy_count = "1";
    }
}
```

The code below shows the creating of the scoping attribute (IDPList) and allowing it to be processed at the bridge passed.

```c
struct zx_sp_IDPEntry_s* idpList;
...
ar->Scoping->IDPList = zx_NEW_sp_IDPList(cf->ctx, &ar->Scoping->gg);
```
ar->Scoping->IDPList->IDPEntry = idpList = zx_NEW_sp_IDPEntry(cf->ctx, &ar->Scoping->IDPList->gg);
        idpList->ProviderID = zx_ref_attr(cf->ctx, &idpList->gg, zx_ProviderID_ATTR, cgi-IDPList);

Appendix PacketFence Apache

The code below shows the configuration of the directory for ZXID. The ZXID is running in an HTTPS (443 which is the standard port number for SSL connection) connection. It also contains a self assigned certificate for test purposes. In the <FilesMatch></FilesMatch> the code will avoid executing a non PERL script as a normal file (HTML, CSS) instead of assuming that the code is a PERL extension.

<VirtualHost *.443>
   SSLStrictSNIVHostCheck off
   SSLEngine on
   SSLCertificateFile /etc/pki/tls/certs/localhost.crt
   SSLCertificateKeyFile /etc/pki/tls/private/localhost.key
   Alias /var/zxid-1.02/ ServerName pf.utelisys.net
   <Directory /var/zxid-1.02>
      <FilesMatch "^\.+\.pl">
         SetHandler perl-script
         AddHandler perl-script .pl
         PerlResponseHandler ModPerl::Registry
         PerlOptions +ParseHeaders
         Options +ExecCGI
         Order allow,deny
         Allow from all
      </FilesMatch>
   </Directory>
</VirtualHost>

Appendix ProxyPass

The code below shows an example of how the ZXID software can be accessed while residing on the private network. For every domain that the federation accesses, we create a VirtualHost for every domain name that is being accessed with the ProxyPass. There are more domain names being accessed, however we are only showing one for an example.

<VirtualHost *.443>
   SSLEngine on
   SSLCertificateFile /etc/pki/CA/certs/megan2.crt
   SSLCertificateKeyFile /etc/pki/CA/certs/megan2.key
   ServerName megan.utelisys.net
   SSLProxyEngine On
   ProxyPass / https://megan.utelisys.net/
Appendix Selection Page

The code below shows if the parameter is defined in the selection page. Once it is defined, it will redirect the information to the ZXID. This code is shown with the redirect function.

```perl
if (defined($params{'d'}) && $params{'d'} ne '') {
    $PFurl="https://pf.utelisys.net";
    $PFfile = "zxidhlo.pl";
    $d= uri_escape($params{'d'});
    print $cgi->redirect(-uri=>'https://pf.utelisys.net/zxidhlo.pl?d='. $d . '&l1=+Login+);
    exit(0);
}
```

The code below checks if the parameter being passed is correct. It will return an option (\$op), if the option is correct (option ‘L’ or ‘C’), the user will be redirected to the authentication page of the institution, otherwise the user will be redirected back to the selection page (with the else statement).

```perl
$res = Net::SAML::simple_cf($cf, -1, $qs, undef, 0x1828);
$op = substr($res, 0, 1);
if ($op eq 'L' || $op eq 'C') {
    warn "res($res) len=".length($res); print $res; exit;
} else {
    my $cgi = new CGI; print $cgi->redirect("https://localhost.utelisys.test/captive-portal");
    exit(0);
}
```

Appendix Automatic Recognition

Once the user has been authenticated, the code will check if the session id (\$sid) being returned is true. If that is the case, the code will generate the guest registration page and the user will be able to enter its personal information. Otherwise the user will be redirected back to the selection page and repeat the process of selection (code is shown below).

```perl
if ($sid){
    pf::web::guest::generate_selfregistration_page( $cgi, $session,
    "/signup?mode=$GUEST_REGISTRATION", $destination_url, $mac);
    exit(0);
} else {
    print $cgi->redirect("https://localhost.utelisys.test/captive-portal"); exit(0);
}