Software Requirements Specification

for

GDC-EDaSA-IDS

Version 1.0 approved

Prepared by Bernd Landgraf

Green Danube Cloud GmbH

2014-07

Table of Contents

1. Introduction 1

1.1 Purpose 1

1.2 Document Conventions 1

1.3 Intended Audience and Reading Suggestions 1

1.4 Product Scope 2

1.5 References 2

2. Overall Description 4

2.1 Product Perspective 4

2.2 Product Functions 6

2.3 User Classes and Characteristics 6

2.4 Operating Environment 7

2.5 Design and Implementation Constraints 7

2.6 User Documentation 8

2.7 Assumptions and Dependencies 8

3. External Interface Requirements 9

3.1 User Interfaces 9

3.2 Hardware Interfaces 9

3.3 Software Interfaces 9

3.4 Communications Interfaces 10

4. System Features 13

4.1 Setup client devices to use the product 13

4.2 Provide methods to register and authenticate a user 13

4.3 [REQ 3] Redirect-Proxy: Intercept communication between the client and a foreign server 13

4.4 [REQ 4] Suffix-Proxy: Handle communication between the client and a foreign server 14

4.5 15

4.6 [REQ 5] Suffix-Proxy: Analyze the response from the foreign server to find potential attack vectors 16

4.7 Provide feedback to the user, based on the analysis of the communication data 17

5. Other Nonfunctional Requirements 18

5.1 Performance Requirements 18

5.2 Safety Requirements 18

5.3 Security Requirements 18

5.4 Software Quality Attributes 19

5.5 Business Rules 19

6. Other Requirements 19

Revision History

|  |  |  |  |
| --- | --- | --- | --- |
| Name | Date | Reason For Changes | Version |
| Bernd Landgraf | 2014-07-17 | Initial Version | 1.0 |
|  |  |  |  |

# Introduction

## Purpose

This document describes the requirements for the first version of the software product “Green Danube Cloud-Self-Adaption-Intrusion-Detection-System” developed by the Green Danube Cloud GmbH. It describes the basic requirements and features for the first prototype of this product which should provide a proof of concept. This document does not claim to provide a fully developed, complete requirements-specification but will be extended and adapted in an iterative fashion.

## Document Conventions

<Describe any standards or typographical conventions that were followed when writing this SRS, such as fonts or highlighting that have special significance. For example, state whether priorities for higher-level requirements are assumed to be inherited by detailed requirements, or whether every requirement statement is to have its own priority.>

## Intended Audience and Reading Suggestions

This document is intended for developers working on this product as well as for senior management executives as a documentation and for supervising the ongoing development process.

## Product Scope

This product should increase endpoint security of mobile devices as well as personal computers when using web- and cloud-applications. Our approach is based on the assumption that attacks generated by infected web- or cloud-applications can be detected by analyzing the response of the application before it even reaches the endpoint (e.g. the browser on a user’s device).

To allow a high detection rate as well as the possibility to detect yet unknown attacks – so called 0-day-attacks – the product will use signature-based as well as anomaly-based detection algorithms.

Additionally, the product will intercept the line of attack (as outlined in [1]) in an very early phase of the attack cycle (see Illustration 1). Therefore it will help not only to detect malicious changes or infections but will have the ability to prevent infections on endpoints before they actually happen.



Illustration 1: Position of the product in the line of attack

## References

|  |  |
| --- | --- |
| [1] | Green Danube Cloud GmbH, FFG-Application, 2014. |
| [2] | "Datenschutzgesetz Deutschland," [Online]. Available: http://www.gesetze-im-internet.de/bundesrecht/bdsg\_1990/gesamt.pdf. [Accessed 21 07 2014]. |
| [3] | "Österreichisches Datenschutzgesetz," [Online]. Available: https://www.ris.bka.gv.at/GeltendeFassung.wxe?Abfrage=bundesnormen&Gesetzesnummer=10001597. [Accessed 21 07 2014]. |
| [4] | "Akka Webseite," [Online]. Available: http://akka.io/. [Accessed 17 07 2014]. |
| [5] | "HTTP/1.1," [Online]. Available: http://www.w3.org/Protocols/rfc2616/rfc2616.html. [Accessed 21 07 2014]. |
| [6] | "HTTP/2," [Online]. Available: http://http2.github.io/. [Accessed 21 07 2014]. |
| [7] | "WebSocket," [Online]. Available: http://tools.ietf.org/html/rfc6455. [Accessed 21 07 2014]. |
| [8] | "Server-Sent Events," [Online]. Available: http://www.w3.org/TR/eventsource/. [Accessed 21 07 2014]. |
| [9] | "WebRTC," [Online]. Available: http://www.webrtc.org/. [Accessed 21 07 2014]. |
| [10] | "RTP," [Online]. Available: http://www.ietf.org/rfc/rfc1889.txt. [Accessed 21 07 2014]. |
| [11] | "OWASP," [Online]. Available: https://www.owasp.org. [Accessed 23 07 2014]. |
| [12] | "SSL/TLS Interception Proxies and Transitive Trust," [Online]. Available: https://media.blackhat.com/bh-eu-12/Jarmoc/bh-eu-12-Jarmoc-SSL\_TLS\_Interception-WP.pdf. [Accessed 23 07 2014]. |
| [13] | H. Lampesberger, A Grammatical Inference Approach to Language-Based Anomaly Detection. |
| [14] | S. Axelsson, The base-rate-fallacy and the difficulty of intrusion detection. |
| [15] | "QUIC," [Online]. Available: http://blog.chromium.org/2013/06/experimenting-with-quic.html. [Accessed 21 07 2014]. |
| [16] | "Certificate Pinning," [Online]. Available: https://www.owasp.org/index.php/Certificate\_and\_Public\_Key\_Pinning. |

# Overall Description

## Product Perspective

This product will be a new, innovative, extensible product. The usage of encryption technologies (e.g. HTTP + TLS in the HTTP/1.1 standard and most likely SPDY in the upcoming HTTP/2.0 standard), as well as security methods used by applications (e.g. “Certificate Pinning”), limit the basic architecture of our system as displayed in the following illustration:



Illustration 2: Basic System Architecture

**CLIENT:** can be any browser or application on any device that allows the user to provide custom proxy settings.

**PROXY = REDICRECT PROXY:** A simple proxy server that redirects all requests to the suffix-proxy.

**SUFFIX-PROXY:** The suffix-proxy will contain the major part of our application:

* It will receive the request from the client
* Forward the request to a foreign server and:
  + has to be able to handle different protocols (http(s), spdy, quic, WebRTC, push-technologies (WebSocket, SSE),… -> see Paper „Technologies for Web and Cloud Service Interaction“ by Harald Lampesberger)
  + has to be able to handle sessions between the client and the foreign server,…
* Receive the response from the foreign server and:
  + Inspect the response for attack-vectors by
    - Detecting the content-type of the resource
    - Performing various detection-algorithms depending on the content-type
  + Perform URL-rewriting
  + Either forward the response to the client if it is considered to be safe or warn the client otherwise

A very simple communication between these components might look like shown in the following diagram:



The basic steps performed are listed in the following activity diagram:



## Product Functions

The product’s functions can be divided in 3 categories:

### User Management

* Register User
* Setup Client Devices

### Redirect-Proxy

* Authenticate Users
* Redirect the client to the suffix proxy

### Suffix-Proxy

* Handle communication between client and foreign server and inspect the communication for possible attack vectors using various attack-detection-algorithms

## User Classes and Characteristics

Basically we can distinguish between 2 user classes:

* **Single Users:**

Private users who want to use the product to enhance security of their private, personal devices.

* **Users using an appliance:**

Users or companies who want to use the product to enhance security of all devices connected in their company network.

|  |  |  |
| --- | --- | --- |
|  | Single Users | Users using an appliance |
| Technical Expertise | Low | High |
| Distribution Channel | Download/Appstore | S&T |
| Importance | Medium | High |
|  |  |  |

## Operating Environment

The product will have to communicate with the user’s applications as well as with foreign web-applications the user wants to access.

To allow maximum availability and scalability, the program will be deployed in a cloud environment. A specific cloud environment is [TBD].

## Design and Implementation Constraints

**Security**

As the product will try to limit an attackers possibilities to attack the users system, it will be a high-value target for attackers itself. DoS attacks could severely affect the availability and therefore the usability of our product. Additionally, if an attacker gets access to our product, he could manipulate data so that his attacks against clients using the product are not recognized anymore. Therefore, security will be of very high importance for our product and relevant development-guides and quality-assurance methods will have to be applied.

**Data protection and data security**

As the data that is transferred to our product might be used to identify the user of our product and because the data might contain sensible information, we will have to take laws governing data protection and data security into account (see e.g. [2] and [3]).

**Framework**

To allow maximum availability and scalability, the program will be based on an actor-framework, namely Akka (see [4])

**Legal Constraints**

Decrypting and analyzing encrypted, private data might have legal implications – Legal requirements and constraints will have to be clarified with a legal expert. [TBD]

## User Documentation

<List the user documentation components (such as user manuals, on-line help, and tutorials) that will be delivered along with the software. Identify any known user documentation delivery formats or standards.>

[TBD]

## Assumptions and Dependencies

<List any assumed factors (as opposed to known facts) that could affect the requirements stated in the SRS. These could include third-party or commercial components that you plan to use, issues around the development or operating environment, or constraints. The project could be affected if these assumptions are incorrect, are not shared, or change. Also identify any dependencies the project has on external factors, such as software components that you intend to reuse from another project, unless they are already documented elsewhere (for example, in the vision and scope document or the project plan).>

# External Interface Requirements

## User Interfaces

<Describe the logical characteristics of each interface between the software product and the users. This may include sample screen images, any GUI standards or product family style guides that are to be followed, screen layout constraints, standard buttons and functions (e.g., help) that will appear on every screen, keyboard shortcuts, error message display standards, and so on. Define the software components for which a user interface is needed. Details of the user interface design should be documented in a separate user interface specification.>

[TBD]

## Hardware Interfaces

<Describe the logical and physical characteristics of each interface between the software product and the hardware components of the system. This may include the supported device types, the nature of the data and control interactions between the software and the hardware, and communication protocols to be used.>

[TBD]

The product can be used on any device with any application that allows configuring a proxy server. The server-components (the Redirect-Proxy and the Suffix-Proxy, see Illustration 2) of the product can be deployed on any machine that supports running a JVM.

## Software Interfaces

<Describe the connections between this product and other specific software components (name and version), including databases, operating systems, tools, libraries, and integrated commercial components. Identify the data items or messages coming into the system and going out and describe the purpose of each. Describe the services needed and the nature of communications. Refer to documents that describe detailed application programming interface protocols. Identify data that will be shared across software components. If the data sharing mechanism must be implemented in a specific way (for example, use of a global data area in a multitasking operating system), specify this as an implementation constraint.>

[TBD]

## Communications Interfaces

Basically, the product will be distributed in 2 scenarios:

### Single-User-Scenario



This opens 2 communication channels:

1. **Client – GDC – Client**

Instead of issuing a request directly to an (potentially infected) web application, the client sends the request to our product and receives the reponse from the product.

1. **GDC – Foreign Server – GDC**

Our product will issue and process the request instead of the client, receive the response from the foreign Webserver and analyze the received response for attack vectors.

### Appliance-Scenario



The Appliance-Scenario opens a third communication channel:

1. **Client – Appliance – Client**

= Client – GDC -Client

1. **Appliance – Foreign Server – Appliance**

= GDC – Foreign Server – GDC

1. **Appliance – GDC – Appliance**

To inspect the Foreign Server’s response, additional data stored in the GDC will be necessary for some exploit-detection-algorithms. This data has to be transferred from the GDC to the appliance.

### Protocols

#### HTTP

As communication between all entities will happen over the internet (or over the intranet in the appliance-scenario), mostly HTTP will be used as a communication protocol. The current version of HTTP (1.1, see [5]) will soon be replaced by a major revision of the protocol HTTP 2.0 (see [6]).

The product should – depending on the capabilities of the client and the foreign webserver – use the most advanced possible protocol. Additionally, the data-exchange between Client and GDC should always be encrypted, even if the foreign server does not support encryption itself. Preferably, the communication between Client and GDC should happen using the SPDY protocol which will be standardized as the new communication protocol for the web in HTTP/2.0. If the client does not support the SPDY protocol, HTTP+TLS (https) should be used for communication.

Also, using the HTTP Upgrade request to switch the protocol e.g. to WebSocket has to be considered.

#### Other Protocols and Technologies

In addition to HTTP, at least the following protocols and technologies have to be considered - this list may be extended in the future:

* WebSocket [7]: A protocol that is initiated by a HTTP-Upgrade request that enables push-technologies.
* Server-Sent Events [8]: A technology where a browser gets automatic updates from a server via a HTTP connection, standardized as part of the HTML5 standard
* WebRTC [9]: A technology that allows cross-browser communication using the (S)RTP [10] protocol

### Security implications

The main goal of our product is to enhance the security of clients regarding malicious responses from infected web- or cloud-applications. However, we have to be constantly aware of the fact that our approach also increases the attack surface, especially when handling benign responses.

As our product itself will be a web-application, it can be targeted by attackers just as well as any other web-application. Therefore high security standards will have to be defined and followed during development (see e.g. [11]).

Our product will provides a single point of failure for a lot of users and therefore will be a very attractive target for attackers, also for DoS-attacks. Additionally, the need to inspect encrypted traffic, which will be necessary especially with the upcoming http/2 standard where all communication will be encrypted, requires that our product breaks the end-to-end encryption between the Client and the Foreign Server. The fact, that the transferred data will be decrypted to plain text by our product, makes it an even more attractive target. (see [12])

# System Features

## Use Case Overview Diagram



## Actors

* **User**: A user using a single license
* **Client**: Any device or application belonging to a registered User

## [REQ 1] User-Management: Register and manage Users

### Description and Priority

|  |  |
| --- | --- |
| Description | The Product has to provide methods to register single users and provide methods to setup client devices to use our product |
| Priority | Critical |

### Use-Cases

|  |  |  |  |
| --- | --- | --- | --- |
| Use Case ID: | UM-1 [TBD] | | |
| Use Case Name: | Register User | | |
| Created By: |  | Last Updated By: |  |
| Date Created: |  | Date Last Updated: |  |

|  |  |
| --- | --- |
| Actor: |  |
| Description: |  |
| Preconditions: |  |
| Postconditions: |  |
| Priority: |  |
| Frequency of Use: |  |
| Normal Course of Events: |  |
| Alternative Courses: |  |
| Exceptions: |  |
| Includes: |  |
| Special Requirements: |  |
| Assumptions: |  |
| Notes and Issues: |  |

### Functional Requirements

|  |  |
| --- | --- |
| **[REQ 1.1]** | **User Management: Register Users [TBD]** |
| Priority | Critical |
| Description |  |
| Error-Handling |  |
| Sub-Rqts |  |

|  |  |
| --- | --- |
| **[REQ 1.2]** | **User Management: Change User-Details [TBD]** |
| Priority | Important |
| Description |  |
| Error-Handling |  |

## [REQ 2] User-Management: Setup Client Devices

### Description and Priority

|  |  |
| --- | --- |
| Description | The product has to provide methods to setup devices and/or applications for the using our application |
| Priority | Critical |

### Use Cases

|  |  |  |  |
| --- | --- | --- | --- |
| Use Case ID: | UM-2 [TBD] | | |
| Use Case Name: | Setup Client Device | | |
| Created By: |  | Last Updated By: |  |
| Date Created: |  | Date Last Updated: |  |

|  |  |
| --- | --- |
| Actor: |  |
| Description: |  |
| Preconditions: |  |
| Postconditions: |  |
| Priority: |  |
| Frequency of Use: |  |
| Normal Course of Events: |  |
| Alternative Courses: |  |
| Exceptions: |  |
| Includes: |  |
| Special Requirements: |  |
| Assumptions: |  |
| Notes and Issues: |  |

### Functional Requirements

|  |  |
| --- | --- |
| **[REQ 2.1]** | **Setup Client Devices [TBD]** |
| Priority | Critical |
| Description |  |
| Error-Handling |  |

## [REQ 3] Redirect-Proxy: Authenticate Users and redirect the client to the suffix proxy

### Description and Priority

|  |  |
| --- | --- |
| Description | The product has to provide methods to authenticate already registered users when they try to use our product |
| Priority | Critical |

### Use Cases

|  |  |  |  |
| --- | --- | --- | --- |
| Use Case ID: | RP-1 [TBD] | | |
| Use Case Name: | Authenticate User | | |
| Created By: |  | Last Updated By: |  |
| Date Created: |  | Date Last Updated: |  |

|  |  |
| --- | --- |
| Actor: |  |
| Description: |  |
| Preconditions: |  |
| Postconditions: |  |
| Priority: |  |
| Frequency of Use: |  |
| Normal Course of Events: |  |
| Alternative Courses: |  |
| Exceptions: |  |
| Includes: |  |
| Special Requirements: |  |
| Assumptions: |  |
| Notes and Issues: |  |

|  |  |  |  |
| --- | --- | --- | --- |
| Use Case ID: | RP-2 | | |
| Use Case Name: | Redirect the client to the suffix-proxy | | |
| Created By: | Bernd Landgraf | Last Updated By: | Bernd Landgraf |
| Date Created: | 2014-07-22 | Date Last Updated: | 2014-07-22 |

|  |  |
| --- | --- |
| Actor: | Client |
| Description: | When the Redirect-Proxy receives a request from an authenticated client, it will redirect the client’s request to the suffix-proxy |
| Preconditions: | Client is authenticated |
| Postconditions: | The client receives a redirect to the suffix-proxy |
| Priority: | Critical |
| Frequency of Use: |  |
| Normal Course of Events: | 1. The client issues a request to the redirect-proxy 2. The redirect-proxy receives the request 3. The redirect-proxy sends a redirect to the client that points him to the suffix-proxy |
| Alternative Courses: | - |
| Exceptions: | - |
| Includes: | - |
| Special Requirements: | * **Security/Privacy:** All communication between the Client and the Redirect-Proxy has to be encrypted, either using SPDY or HTTP+TLS * **Performance:** The Redirect-Proxy should add as little latency as possible |
| Assumptions: |  |
| Notes and Issues: |  |

### Functional Requirements

|  |  |
| --- | --- |
| **[REQ 3.1]** | **Redirect-Proxy: Authenticate Single-Users [TBD]** |
| Priority | Critical |
| Description |  |
| Rationale |  |
| Error-Handling |  |

|  |  |
| --- | --- |
| **[REQ 3.1]** | **Redirect-Proxy: Authenticate Appliance-Users [TBD]** |
| Priority | Important |
| Description |  |
| Rationale |  |
| Error-Handling |  |

|  |  |
| --- | --- |
| **[REQ 3.2]** | **Redirect-Proxy: Determine/Negotiate protocol used for communication with client** |
| Priority | Critical |
| Description | The Redirect-Proxy has to determine which protocol should be used when communicating with the client. If possible, all communication between the client and the redirect-proxy has to be encrypted. |
| Rationale | Encrypting the transferred data is necessary to ensure data-protection and security. |
| Preconditions | - |
| Error-Handling | [TBD] |

|  |  |
| --- | --- |
| **[REQ 3.3]** | **Redirect-Proxy: Redirect the client to the suffix proxy** |
| Priority | Critical |
| Description | The Redirect-Proxy has to redirect the clients request to the suffix proxy, e.g. if the client requested the resource “https://www.example.com/faq”, the Redirect-Proxy has to redirect the client to “https://www.example.com.suffix.com/faq” |
| Rationale | Due to the increased usage of encryption and browser-techniques like certificate pinning, it would be extremely difficult – if not impossible – to implement a direct tls-interception-proxy. Therefore the redirect proxy needs to forward the client to the suffix proxy which will be able to handle encryption correctly |
| Preconditions | The user has to be authenticated |
| Error-Handling | [TBD]   * Unsupported Protocol |

|  |  |
| --- | --- |
| **[REQ 3.4]** | **Redirect-Proxy: Handle HTTP/1.1 requests** |
| Priority | Critical |
| Description | The redirect proxy has to handle requests transferred via the http/1.1 protocol |

|  |  |
| --- | --- |
| **[REQ 3.5]** | **Redirect-Proxy: Handle spdy requests** |
| Priority | Important |
| Description | The redirect proxy has to handle requests transferred via the spdy protocol – spdy will be standardized in HTTP/2.0 |

|  |  |
| --- | --- |
| **[REQ 3.6]** | **Redirect-Proxy: Handle Push-Technologies** |
| Priority | Critical |
| Description | The redirect-proxy has to handle push-technologies:   * Server-Sent-Events * Comet * ReverseHTTP * WebSockets |

|  |  |
| --- | --- |
| **[REQ 3.7]** | **Redirect-Proxy: Handle direct Client to Client Communication** |
| Priority | Desireable |
| Description | The redirect-proxy has to handle client-to-client communication via WebRTC |

## [REQ 4] Suffix-Proxy: Handle communication between the client and a foreign server

### Description and Priority

|  |  |
| --- | --- |
| Description | The suffix proxy has to receive requests from the client, send these requests to the server on the behalf of the client and receive the response from the foreign server. |
| Priority | Critical |

### Use Cases

|  |  |  |  |
| --- | --- | --- | --- |
| Use Case ID: | SP-1 | | |
| Use Case Name: | Handle communication between client and foreign server | | |
| Created By: | Bernd Landgraf | Last Updated By: | Bernd Landgraf |
| Date Created: | 2014-07-22 | Date Last Updated: | 2014-07-22 |

|  |  |
| --- | --- |
| Actor: | Client, Foreign Server |
| Description: | When the Suffix-Proxy receives a request from the Client, it forwards the request to the foreign server, receives the foreign server’s response, analyzes the response for attack vectors and forwards the response to the Client. |
| Preconditions: | Client is authenticated |
| Postconditions: | The Client receives a response |
| Priority: | Critical |
| Frequency of Use: |  |
| Normal Course of Events: | 1. The Client sends a request to the Suffix-Proxy 2. The Suffix-Proxy forwards the request to the Foreign Server 3. The Foreign Server send a response back to the Suffix-Proxy 4. The Suffix-Proxy analyzes the response for attack vectors 5. IF the response is considered safe, the Suffix-Proxy performs URL-Rewriting 6. IF the response is considered safe, the Suffix-Proxy forwards the rewritten response to the client |
| Alternative Courses: | SP-1.AC.1 5 IF the response is considered unsafe, the Suffix-Proxy sends a warning to the client  SP-1.AC.1 6 The client decides if he still wants to retrieve the response  SP-1.AC.1.7 IF the client decides to receive the response, the Suffix-Proxy forwards the response to the client  SP-1.AC.2.5 IF the response is considered to be dangerous, the Suffix-Proxy sends an information to the client |
| Exceptions: |  |
| Includes: |  |
| Special Requirements: | * **Security/Privacy:** All communication between the Client and the Redirect-Proxy has to be encrypted, either using SPDY or HTTP+TLS   If possible, all communication between the Suffix-Proxy and the Foreign Server has to be encrypted   * **Usability**: False-Alarm-Rate ~ < 10-5 * **Performance**: [TBD] |
| Assumptions: |  |
| Notes and Issues: |  |

### Functional Requirements

|  |  |
| --- | --- |
| **[REQ 4.1]** | **Suffix-Proxy: Determine/Negotiate protocol used for communication with client** |
| Priority | Critical |
| Description | The Suffix-Proxy has to determine which protocol should be used when communicating with the client. If possible, all communication between the client and the redirect-proxy has to be encrypted. |
| Rationale | Encrypting the transferred data is necessary to ensure data-protection and security. |
| Preconditions | - |
| Error-Handling | [TBD] |

|  |  |
| --- | --- |
| **[REQ 4.2]** | **Suffix-Proxy: Handle HTTP/1.1 requests** |
| Priority | Critical |
| Description | The suffix proxy has to handle requests transferred via the http/1.1 protocol |

|  |  |
| --- | --- |
| **[REQ 4.3]** | **Suffix-Proxy: Handle spdy requests** |
| Priority | Important |
| Description | The suffix proxy has to handle requests transferred via the spdy protocol – spdy will be standardized in HTTP/2.0 |

|  |  |
| --- | --- |
| **[REQ 4.4]** | **Suffix-Proxy: Handle Push-Technologies** |
| Priority | Critical |
| Description | The suffix-proxy has to handle push-technologies:   * Server-Sent-Events * Comet * ReverseHTTP * WebSockets |

|  |  |
| --- | --- |
| **[REQ 4.5]** | **Suffix-Proxy: Handle Push-Technologies** |
| Priority | Critical |
| Description | The suffix-proxy has to handle push-technologies:   * Server-Sent-Events * Comet * ReverseHTTP * WebSockets |

|  |  |
| --- | --- |
| **[REQ 4.6]** | **Suffix-Proxy: Perform URL-rewriting** |
| Priority | Critical |
| Description | The suffix-proxy has to rewrite all external references in the server’s response so that all following requests are also processed by the suffix-proxy. E.g. an external reference to “http://www.example.com/” should be rewritten to “http://www.example.suffix.com” |
| Rationale |  |
| Preconditions |  |

|  |  |
| --- | --- |
| **[REQ 4.7]** | **Suffix-Proxy: Determine the content-type of the foreign server’s response** |
| Priority | Critical |
| Description | The suffix server has to determine the content-type of the foreign server’s response and choose which attack-detection algorithms can be applied to the detected content-type.  Additionally, the content type has to be validated by the application, so that attacks like GIFAR (see <http://en.wikipedia.org/wiki/Gifar>) can be avoided. |
| Rationale | Detecting the content-type of the response is necessary because for different content types, different attack-detection-algorithms may be used. |

|  |  |
| --- | --- |
| **[REQ 4.8]** | **Suffix-Proxy: Analyze the foreign server’s response for attack vectors using language based anomaly detection** |
| Priority | Critical |
| Description | The foreign server’s response is analyzed using the language-based anomaly detection algorithm proposed by Harald Lampesberger [13] |
| Rationale |  |
| Preconditions | The foreign server’s response was determined to be xml/html-content |

|  |  |
| --- | --- |
| **[REQ 4.9]** | **Suffix-Proxy: Analyze the foreign server’s response for attack vectors using pattern matching** |
| Priority | Critical |
| Description | [TBD] |
| Rationale |  |
| Preconditions | The foreign server’s response was determined to be xml/html-content |

|  |  |
| --- | --- |
| **[REQ 4.10]** | **Suffix-Proxy: Send the foreign server’s response back to the client** |
| Priority | Critical |
| Description | The suffix-proxy has to decide, based on the analysis of the server’s response, whether the it is safe to send the response back to the client or if the access to the response has to be restricted and the client has to be warned about the security-violation |
| Rationale |  |
| Preconditions |  |

# Other Nonfunctional Requirements

## Performance Requirements

<If there are performance requirements for the product under various circumstances, state them here and explain their rationale, to help the developers understand the intent and make suitable design choices. Specify the timing relationships for real time systems. Make such requirements as specific as possible. You may need to state performance requirements for individual functional requirements or features.>

[TBD]

## Safety Requirements

<Specify those requirements that are concerned with possible loss, damage, or harm that could result from the use of the product. Define any safeguards or actions that must be taken, as well as actions that must be prevented. Refer to any external policies or regulations that state safety issues that affect the product’s design or use. Define any safety certifications that must be satisfied.>

By breaking up the encryption between client and foreign server, the communicated data will – temporarily – be available as plain data when inspected by our product. As this data might and most likely will often contain sensible and valuable information, this poses the constant danger of attackers getting access to confidential data. If this scenario emerges, the impact and damage cannot be foreseen, as they might range from almost no losses (insensitive information) to extreme losses (credit card numbers, company secrets ...), depending on the data that is leaked.

To minimize the likelihood of this scenario two different actions are taken:

1. Every communication between the client and the product, as well as between the product and a foreign webserver, is always encrypted if possible.
2. Additionally, tight security guidelines will be followed during development (see 5.3)

## Security Requirements

<Specify any requirements regarding security or privacy issues surrounding use of the product or protection of the data used or created by the product. Define any user identity authentication requirements. Refer to any external policies or regulations containing security issues that affect the product. Define any security or privacy certifications that must be satisfied.>

As stated in 2.5 and 3.4.4, although our product will enhance the security of clients regarding malicious responses, it also increases the attack surface and presents itself as a high value target for attackers as even encrypted communication will be available in plain form and as attacking our product will simplify attacking clients afterwards.

DoS attacks could severely affect the usability of our product or make it even unusable if they are not handled correctly

Therefore, the product has to be thoroughly secured against attacks and thoroughly tested itself.

We will have to develop or adapt existing secure-development-, review- and testing-guidelines and continuously enforce these activities during the software-development-cycle.

## Software Quality Attributes

<Specify any additional quality characteristics for the product that will be important to either the customers or the developers. Some to consider are: adaptability, availability, correctness, flexibility, interoperability, maintainability, portability, reliability, reusability, robustness, testability, and usability. Write these to be specific, quantitative, and verifiable when possible. At the least, clarify the relative preferences for various attributes, such as ease of use over ease of learning.>

### Modifiability

The internet and its protocols are a very changeable, constantly evolving environment. Therefore it should be easily possible to add support for new protocols.

The changeability is even greater concerning attacks on internet-users. This implies, that it should be easily possible to adapt or extend the existing attack-detection algorithms. Adding new attack-detection algorithms will be frequent changes in the future and should be easily possible.

### Testability

Due to the high security risks of the product, functional as well as penetration testing will need to be an integral part of our development cycle. Therefore, Testability has to be carefully considered in the software-architecture and design.

### Usability

The acceptance of our product will – apart from its ability to detect attacks- highly depend on its usability. There are 2 factors that have to be especially considered:

* **Performance**: If our product introduces too much latency, it might not get used anymore. Therefore Performance (5.1) has to be considered as a key aspect for the Usability of our Product.
* **False-Alarm-Rates**: To ensure the usability of our product, the false-alarm-rate has to be extremely low. A high number of false-alarms will also lead to users abandoning the use of our product. Axelsson [14] recommends a false-alarm-rate of < 10-5 to ensure the usability of an intrusion-detection-system.

## Business Rules

<List any operating principles about the product, such as which individuals or roles can perform which functions under specific circumstances. These are not functional requirements in themselves, but they may imply certain functional requirements to enforce the rules.>

# Other Requirements

<Define any other requirements not covered elsewhere in the SRS. This might include database requirements, internationalization requirements, legal requirements, reuse objectives for the project, and so on. Add any new sections that are pertinent to the project.>

Appendix A: Glossary

|  |  |
| --- | --- |
| GDC-EDaSA-IDS | Green Danube Cloud – Early Detection and Self Adaption – Intrusion Detection System |
| IDS | Intrusion Detection System |
|  |  |

Appendix B: Features Prioritization Scale

The features in chapter 4 are ranked according to the following prioritization scale:

|  |  |
| --- | --- |
| **Critical** | A critical requirement without which the product is not acceptable to the stakeholders. These are requirements that should be part of the first prototype of the application. |
| **Important** | A necessary but deferrable requirement which makes the product less usable but still functional. These requirements should |
| **Desirable** | A nice feature to have if there are resources but the product functions well without it. |

Appendix C: Analysis Models

<Optionally, include any pertinent analysis models, such as data flow diagrams, class diagrams, state-transition diagrams, or entity-relationship diagrams.>

Appendix D: To Be Determined List

<Collect a numbered list of the TBD (to be determined) references that remain in the SRS so they can be tracked to closure.>