

TALLINN UNIVERSITY OF TECHNOLOGY
School of Information Technologies

Vjatšeslav Rukavišnikov 192919IVSB

Improving Reporting Process in CYBERS Security Operations Center

Bachelor Thesis

Supervisor: Jürgen Erm
BSc

Tallinn 2022

TALLINNA TEHNIKAÜLIKOOL
Infotehnoloogia teaduskond

Vjatseslav Rukavišnikov 192919IVSB

Aruandlusprotsessi täiustamine CYBERS küberkaitse operatsioonide keskuses

Bakalaureusetöö

Juhendaja: Jürgen Erm
BSc

Tallinn 2022

Author's declaration of originality

I hereby certify that I am the sole author of this thesis. All the used materials, references to the literature and the work of others have been referred to. This thesis has not been presented for examination anywhere else.

Author: Vjatšeslav Rukavišnikov

Date: 16.05.22

Abstract

Security Operations Center support IT goals of organizations and use reporting as a primary tool of communicating with their audience. Therefore, reporting is an integral process in Security Operation Centers.

Existing reporting solutions do not easily integrate with existing processes and infrastructure in SOCs. The objective of this work is to design a system that will complement current reporting system in CYBERS Security Operations Center. Author analyzes already implemented solutions and discusses some other solutions and their suitability. Taking into consideration drawbacks of implemented solutions author designs and implements a new reporting solution for CYBERS SOC.

The result of this work is a web application that interacts with a security monitoring solution and generates reports based on the criteria supplied by user of the application. System retrieves data from SIEM, processes it and presents a graphical report to the user. Web application was implemented ASP.NET Core, Entity Framework Core handles the database interactions and Chart.js plots graphs for the report.

The thesis is in English and contains 29 pages of text, 7 chapters, 27 figures, 0 tables.

Annotatsioon

Aruandlusprotsessi täiustamine CYBERS küberkaitse operatsioonide keskuses

Küberkaitse operatsioonide keskus toetab organisatsioonide IT-eesmärke ja kasutab aruandlust oma vaatajaskonnaga suhtlemise peamise tööriistana. Seetõttu on aruandlus küberkaitse keskustes üks peamistest protsessidest.

Olemasolevad aruandluslahendused ei integreeru kergesti küberkaitse keskustes olemasolevate protsessidega ja infrastruktuuriga. Selle töö eesmärk on välja töötada süsteem, mis täiendaks praegust CYBERSi küberkaitse operatsioonide keskuse aruandlussüsteemi. Autor analüüsib juba rakendatud lahendusi ja arutab mõningaid muid lahendusi ja nende sobivust. Võttes arvesse rakendatud lahenduste puudusi, kujundab autor ja viib ellu CYBERS SOC-i jaoks uut aruandluslahendust.

Selle töö tulemuseks on veebirakendus, mis suhtleb turvaseire lahendusega ja genereerib aruandeid kasutaja poolt antud parameetrite alusel. Süsteem teeb päringuid SIEM-ist, töötleb vastuseid ja esitab kasutajale graafilist aruannet. Veebirakenduse loomiseks kasutati ASP.NET Core, Entity Framework Core, mis vastutab andmebaasi interaktsioonide eest, ning Chart.js, mis koostab aruande graafikud.

Lõputöö on kirjutatud inglise keeles ning sisaldab teksti 29 leheküljel, 7 peatükki, 27 joonist, 0 tabelit.

List of abbreviations and terms

API	Application Programming Interface
AQL	Ariel Query Language
CSV	Comma Separated Values
HTTP	Hyper Text Transfer Protocol
JSON	JavaScript Object Notation
SIEM	Security Information and Event System
SOC	Security Operations Center

Table of Contents

List of Figures	9
1 Introduction	10
1.1 Problem statement	10
1.2 Goals of the Thesis	11
1.3 Structure of the Thesis	11
2 Background	12
2.1 Security Operations Center	12
2.1.1 Metrics and their reporting in SOC	13
2.1.2 Security Information and Event Management	14
2.2 About CYBERS	15
2.2.1 CYBERS SOC	15
2.2.2 Reporting and Metrics Management in CYBERS SOC	16
2.2.3 Authentication Failure Reporting in CYBERS SOC	16
2.2.4 Existing reporting solutions	17
3 CYBERS SOC Failed Authentication Reporting Process	18
3.1 Creating reports using QRadar SIEM platform	18
3.1.1 QRadar reporting system drawbacks	20
3.2 Creating reports using QReport	20
3.2.1 QReport advantages over QRadar reporting system	22
3.2.2 QReport shortcomings	22
4 Methodology	23
4.1 Approach	23
4.2 Microsoft .NET	23
4.3 ASP.NET Core	23
4.4 Entity Framework Core	24
4.5 Data visualization with Chart.js	25
4.6 QRadar API	25
5 Implementation	26
5.1 Application architecture	26
5.2 QRadar API client	26
5.3 Database	28
5.4 Failed authentication report generation process	31
6 Solution overview	34
6.1 Report creation process	34
6.2 Solution analysis	36

6.3 Future work	36
7 Summary	38
Bibliography	39
Appendices	41

List of Figures

1	A simple SIEM architecture	14
2	Report schedule setup dialogue	18
3	Report layout setup dialogue	19
4	Report chart configuration window	19
5	QReport structure	21
6	QReport report example	21
7	MVC pattern	24
8	Application architecture	26
9	QRadar API project structure	27
10	QRadar HTTP client initialization within application	28
11	Report model in application code	29
12	Search model in application code	29
13	Schedule model in application code	30
14	Application database schema	30
15	Code retrieving data from database	31
16	Offenses search sample	31
17	AQL search for failed authentication events	32
18	Code grouping events by username	32
19	Counting events by user	33
20	Grouping events by time and event type	33
21	Report creation initial dialogue	34
22	Search definition dialogue	34
23	Schedule setting dialogue	35
24	On-demand generation option	35
25	Event details per user	35
26	QRadar report example (obfuscated)	42
27	New solution report example (obfuscated)	43

1. Introduction

As use of informational technologies became widespread, cybersecurity became a concern of every business. Defending against targeted cyberattacks and threats has become an every day task of IT personnel. Over time, the spectrum and amount of tasks needed to be performed in order to maintain an adequate security posture became so large, that organizations had to form separate teams to fulfill these tasks. These teams are known today as Security Operations Center (SOC). SOC aims to address systems monitoring, incident detection and incident response. SOC service daily processes are carried out by analysts and engineers of different degrees, who process a large amount of information and are responsible for security of IT operations in the organization. Building an in-house SOC could prove both difficult and expensive, as organization would need to find people who could perform complex setup, analyze business needs and choose appropriate safeguards, implement tactics and procedures, and, finally, react to alerts. Not every organization can afford building their own SOC, but many want to bolster their security posture [1].

To satisfy the needs of such organizations, Security Operations Center was offered as a service. Using such service enables organizations to skip stages of building a SOC and hiring specialists for it, while getting help from experienced cybersecurity specialists. SOC processes a lot of information on a daily basis which should be analyzed. Reporting is necessary to gain an overview of SOC's performance and of quality of provided service [2].

1.1 Problem statement

Reporting of SOC activities is crucial to gain an overview of enterprise's security posture and assess the effectiveness of the currently implemented security measures. Monitoring solutions employed in Security Operations Centers may not always offer comprehensive reporting capabilities. They may be incapable of visualizing certain metrics, convey it in a hard-to-understand way, or reports produced by them may look out-of-date, which may undermine the trustworthiness and decrease the value of the SOC service in customers or management eyes. Lack of flexibility makes it challenging to integrate these solutions into the existing procedures established in SOC. In this work, author will focus on improving reporting capabilities of CYBERS SOC which uses IBM QRadar SIEM as its security

monitoring solution. The resulting application will be specifically tailored to work with SIEM employed by CYBERS SOC.

1.2 Goals of the Thesis

The author's goal is to design a new reporting solution which will integrate with existing security monitoring systems and provide more modular, more modern-looking and automatically generated reports, which will be fully compliant with CYBERS SOC procedures. The author will analyze implemented reporting solutions, find their shortcomings and attempt to fix them in the new solution while adding new functionality. New system will be designed based based on the research of the current reporting system and requirements of CYBERS SOC. Solution implemented by author will work specifically with QRadar SIEM.

Author will discuss only one of the reporting processes and new solution aims to improve it.

1.3 Structure of the Thesis

Chapter 1 described the problem and goals of this work. Chapter 2 provides necessary background and explains certain concepts so that reader could understand the rest of this thesis. SOC, metrics and their reporting are discussed in the background part. Chapter 3 is focused on CYBERS SOC failed authentication events' reporting process. Existing solutions are explained and analyzed. In chapter 4 author explains their approach to designing new system and lists used technologies. Chapter 5 is dedicated to implementation of the new system with explanation of the architecture and overall working principle. Chapter 6 discusses new solution's suitability to CYBERS SOC and describes its workflow, suggests future improvements. Finally, chapter 7 gives conclusions of results of this work.

2. Background

This chapter gives an overview of the primary concepts and processes which are subjects to imp.

2.1 Security Operations Center

The Security Operations Center, or SOC, is a dedicated team of people who ensure security of IT operations within organization. Their duties include monitoring information systems for vulnerabilities, unauthorized activity, acceptable use/policy/procedure violations, intrusions into and out of the network, and engaging with cyber incident response and remediation [3].

SOC is tightly integrated with organization's objectives. SOC develops technical processes that support achieving business goals in a secure manner. To successfully protect a business, SOC must fully understand business problems and weak spots to build a correct strategy of securing the IT operations. Security team that precisely chooses the main operations of a business and their critical aspects will bring the most value to the said business and will prove to be the most efficient for it. Business objectives are high level, more abstract goals. These goals are supported by SOC processes at a lower level [1][3].

SOC can support a business through various activities. These activities can be divided into several categories. Reactive activities include monitoring for alerts, detection of anomalies, investigation of cases, reporting of incidents, all in real time. Proactive activities are comprised of hunting for potential threats within the environment, gathering cyberthreat intelligence and integrating it into the monitoring systems, ensuring health of devices and services across the network. Other activities include continuous analysis of SOC's performance and its improvement, training the staff, and automating the procedures in order to make work more efficient [3].

Building a Security Operations Center requires considerable amount of effort and funds, as there are multiple challenges on the way to building one. SOC needs a work environment: work equipment for team and resources to accommodate technology, and people with right set of skills, who also understand the goals of SOC. Lastly, Security Operations Center

relies on technologies that security monitoring and control mechanisms [1].

2.1.1 Metrics and their reporting in SOC

Metrics are used for assessing performance and identifying problems. They can be divided into internal and external metrics. Internal metrics show the performance of systems and procedures employed in SOC. They are used to identify problematic areas, allowing SOC team to improve their system implementations and procedures, as well as measure themselves for continuous improvement. External metrics must communicate the value SOC brings to the organization and threats it faces to upper management. With this information management can make decisions concerning risk management and further funding of the SOC [4].

By measuring inputs and outputs of processes SOC ensures that they operate as intended. Various SOC properties like time to react, amount of processed alerts, true-positive/false-positive incident ratio are quantified to get a control whether key goals are achieved and if some action needs to be taken. Ideally these metrics are collected automatically and necessary actions to produce some of the metrics, like manual incident categorization, do not impede the work of the analysts [4].

SOC produces large quantities of data which should be analysed and subsequently reported. Overviews of log sources, events, threats, vulnerabilities, attacks, incident reports, investigations, incidents should be given to their respective audience. Reports should take target audience into account and deliver the message clearly. Failure to communicate could lead SOC to losing support from management or making wrong strategical decisions. Therefore, careful analysis and proper reporting are crucial activities within SOC [5].

Native reporting capabilities of various SOC systems (SIEMs, incident management systems, etc.) may not meet all the requirements and therefore, will prevent SOC from communicating effectively. On the other hand, dedicated security reporting solutions can be better at producing desired results, but those can come at a significant price which may render their usage infeasible depending on the SOC's budget. Finally, there are free or open-source alternatives for composing reports like JSReport, may include some features as a paid add-on and will still require effort to integrate them with rest of SOC's systems and configure for desired results [5].

2.1.2 Security Information and Event Management

Security information and event management (SIEM) centrally collects, parses and categorizes logs from various sources as well as traffic captures from network points. This tool provides enhanced visibility into the network by tracking activity of users, devices, services. This allows for quick detection and response to anomalies in the systems that can pose a threat to business security [6][7]. A simple SIEM architecture is illustrated on Figure 1.

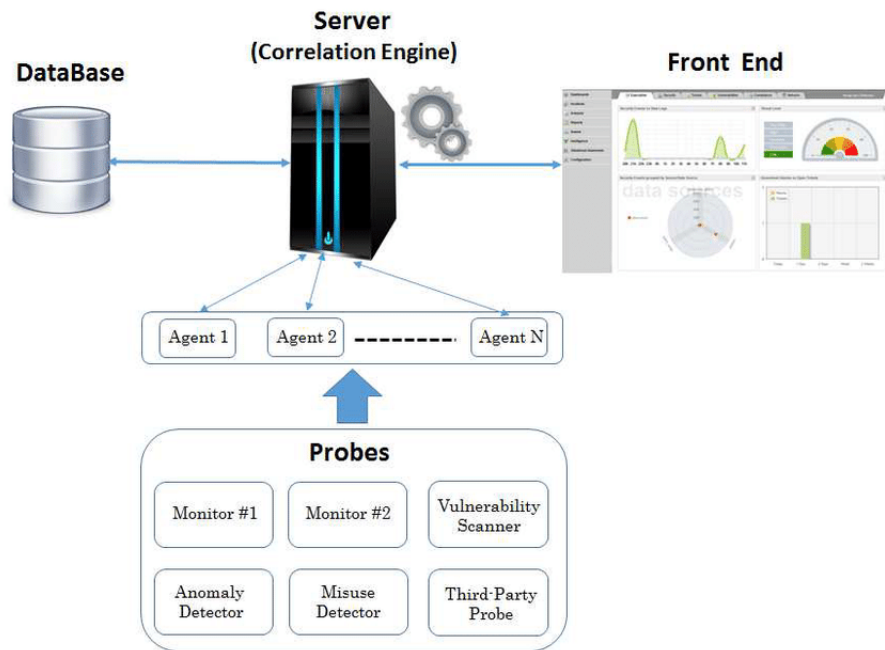


Figure 1. A simple SIEM architecture

Data that is collected from various log sources is being parsed with the purpose of extracting attributes that describe the event. Attributes may include usernames, event types, IP addresses, memory, processes and more. SIEM must be able to process logs of different formats. SIEM solution includes a log parser that converts imported data into a structured data format that the tool can understand. Many SIEM solutions provide parsing for common data sources; however, sometimes parsing has to be customized in order for the data to be converted to the proper structure that the SIEM solution can work with. Extraction of attributes allows for further analysis withing rule logic as well as querying logs based on specific attributes [6][1].

The categorization process involves adding context to the events. This can include system events, authentication data, application logs, and more, depending on what is collected. For example, a SIEM solution can group data about hosts, data about network devices, and

details from security tools into different buckets based on how the data will be analyzed. This sorting process is critical for converting data into actionable intelligence [1].

2.2 About CYBERS

CYBERS is a company founded in 2010 that offers a variety of cybersecurity services. Services provided by the company include, for example: vulnerability assessment, security architecture building, Microsoft 365 cloud-services security monitoring, compliance auditing, SOC as a Service (SOCaaS). In the context of this work, reporting process of CYBERS Security Operations Center will be studied [8][9].

2.2.1 CYBERS SOC

CYBERS provides SOC service to other organizations. New clients' environments are on-boarded for security monitoring. All logs are collected and analyzed by QRadar SIEM platforms. CYBERS specialists analyze incoming data and generated alerts, reacting where necessary. Periodically CYBERS reports its performance and customer environment's status to communicate value and make decisions.

CYBERS SOC is a so-called tiered SOC where analysts are divided into 3 categories: Level 1, Level 2 and Level 3. The day-to-day work of SOC Level 1 specialists includes, most importantly, real-time monitoring. The Level 1 specialists are first, who see the incoming alert, and it is also their task to verify if the incident described in the alert is a true-positive or a false-positive. After the verification has taken place, it is then necessary to carry out initial analysis and prioritization. If necessary, incident is further escalated, and related parties are notified. Verified cases are then viewed by a Level 2 analyst. Other duties of Level 1 analyst include review of different reports and sending them to their corresponding consumers.

The task of a SOC Level 2 specialist is reading analysis and verifying work of the previous analysts. Level 2 analyst conducts a more thorough investigation, if deemed necessary. It is also their task to do further response and remediation in cases where Level 1 employee does not have the access necessary tools, required training or knowledge, or time to dive deeper into the investigation. Incidents of highest priority are passed to the Level 3 specialists.

Level 3 analysts use their expertise to help investigating and remediating the incidents. Outside of incident response, they are occupied with threat hunting, as well as developing, testing and fine-tuning anomaly detection systems.

2.2.2 Reporting and Metrics Management in CYBERS SOC

CYBERS uses reports to give overview of Security Operation Center's performance as well as to assess implemented security measures' effectiveness. There is a set of common report types which are distributed to customers. New reports are created as per customer request or from internal initiative. When creating a new report, first step is always to review metrics and data to collect that are required to compile the requested report. After all requirements are reviewed, SOC team analyses the feasibility of establishing a new reporting procedure. The most common reason for rejecting the new procedure implementation is lack of necessary data. If new reporting procedure is approved, team starts implementing, testing and, finally, deploying the new reporting procedure. Reports' appearance and format are tailored for their respective audience.

CYBERS has several different reporting procedures in use for various purposes. Monthly for giving overview on security posture, incident report which documents the incident, operational reports that are used to ensure health of the infrastructure. Main focus of this work is on one of the weekly reports which is described in more detail later in this chapter.

2.2.3 Authentication Failure Reporting in CYBERS SOC

Some types of alerts that come to CYBERS SOC are not viewed on arrival due to their constant high count or individual insignificance. This can be referred to as 'ticket queue backlog'. But at the same time these alerts cannot be fully ignored. One of the possible ways to deal with 'noisy' alerts is periodically reviewing them in bulk. Authentication failures to various resources such as Windows or Linux machines, VPN interfaces, cloud platforms etc., are happening at a constant rate in CYBERS SOC service environment. These alerts are not considered important if they are not followed by a successful login to the same location by the same user, which could indicate a successful breach which is categorized differently. In most of the cases these alerts are triggered by users mistyping their passwords, sessions with expired passwords, or bruteforce attacks [5].

Each week these alerts are gathered into one report and report is sent to a customer on weekly basis. Authentication failure report shows the customer all alerts as well as users which triggered those alerts. Customer reviews the report to troubleshoot sign-in failures for users in their organization. Reviewing authentication related events may uncover some unexpected events. Each alert is supplied with events which contributed to it, which customer can read to understand the root cause of the failure. It is duty of the Level 1 SOC analysts to review the generated reports and look for suspicious occurrences [10].

2.2.4 Existing reporting solutions

This section mentions reporting solutions that are not used in the CYBERS SOC but could theoretically be implemented. Author also discusses why he chose against those solutions.

Kibana is a free and open-source user interface that lets you visualize your data. It is a very powerful visualization platform that ingests data from Elasticsearch and enables users to plot various graphs. Flexible dashboards allow creating user-defined layouts and importing additional design elements, resulting in an exceptionally powerful data exploration tool [11].

Since Kibana only works in conjunction with Elasticsearch, it requires data to be fed into the latter in order to perform its operations on the data. While Logstash, project created for ingesting various sorts of data, supports retrieving information from a remote API endpoint, it would still require implementation of some specifically programmed feeder due to the fact that QRadar SIEM's (SIEM solution used in CYBERS SOC) API is asynchronous: request is made against one endpoint, but results are retrieved from the other endpoint. Furthermore, to retrieve events associated with alerts triggered by SIEM's rule logic it is necessary to know the retrieve the alerts first. In other words, one search is dependent on the other and can only be constructed having the results of the first search. There is also another aspect of Kibana that prevents it from being used for reporting purpose in CYBERS SOC. Exporting reports as PDF or PNG file format is a feature only available with a paid subscription [11].

"jsreport" is an open-source reporting tool. It can dynamically produce reports in various formats like pdf, excel, docx and also many other text based-formats. It doesn't provide any graphical interface to design reports. Instead, it relies on users defining report structure and appearance by using code, HTML and JavaScript templating engines in particular. This approach gives great power and flexibility to the software developers and lets them use the knowledge they already have [12].

Similar to Kibana, data used in the report templates has to be submitted either manually or by some program via API. Free instance is limited to only 5 report templates, which limits the usefulness of jsreport since its report designer requires knowledge of HTML to create templates and other features like scheduling and template management can be achieved via other means [12].

3. CYBERS SOC Failed Authentication Reporting Process

This chapter gives an overview of reporting solutions that are already implemented in CYBERS SOC, their workflow, capabilities and limitations.

3.1 Creating reports using QRadar SIEM platform

QRadar SIEM has built-in reporting capabilities that enables users to create, edit, distribute, and manage reports generated based on information gathered by the system. Users can choose from default reports and self-defined reports, which can be further customized and branded [13].

Report creation begins with dialogue where report generation schedule is chosen. Both periodic and manual options are available (Figure 2) [13].



The image shows a 'Report Wizard' dialog box. At the top left is a logo with a red, blue, and orange circle. The title 'Report Wizard' is in white on a dark background. Below the title, the text 'This report should be scheduled to generate:' is followed by four radio button options: 'Manually', 'Hourly', 'Daily', and 'Monthly'. The 'Hourly' option is selected. Under 'Hourly', there is a label 'Every hour' and two dropdown menus: 'From:' and 'To:', both set to '2:00 AM'. Below these options is the question 'Allow this report to generate manually?' with two radio button options: 'Yes - Manually generate report.' (selected) and 'No - Schedule report only.'

Figure 2. Report schedule setup dialogue

Next dialogue window configures general layout of the report and number of charts or tables. There are 9 variants with portrait orientation and 6 variants with landscape orientation as portrayed on Figure 3 [13].

Report Wizard

Choose a Layout

Each divided section holds one chart. Click the layout that represents the size and number of charts required.

Orientation:

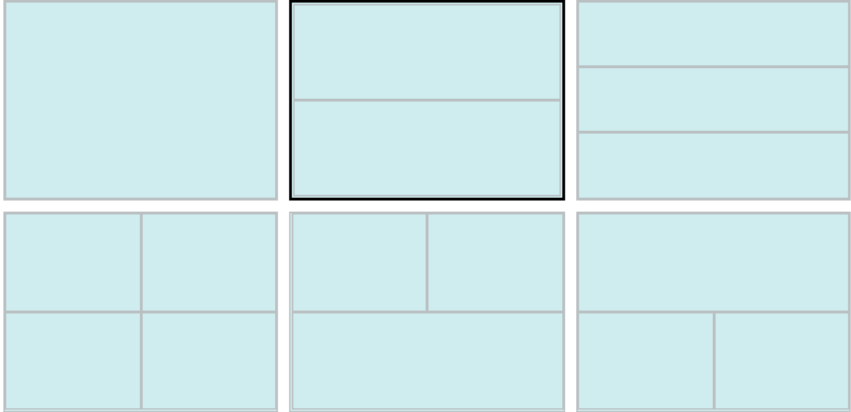


Figure 3. Report layout setup dialogue

After the layout has been configured, different charts can be chosen and assigned to their containers. First, a chart type which defines how data appears in the report has to be chosen. QRadar offers numerous chart types, although some of them are only usable if system has a respective extension installed. For querying event, network flow, and alert information "Events/Logs", "Flows" and "Offenses Over Time" chart types are used. "Events/Logs" and "Flows" are the most configurable as they can utilize "Ariel Query Language" - language with syntax similar to SQL used for extracting and manipulating data from QRadar databases, allowing to perform advanced searches not available from user interface. Figure 4 depicts main settings for the events chart. User can choose what data they want to get by choosing an appropriate AQL search as well as the graph type to represent the data [13][14].

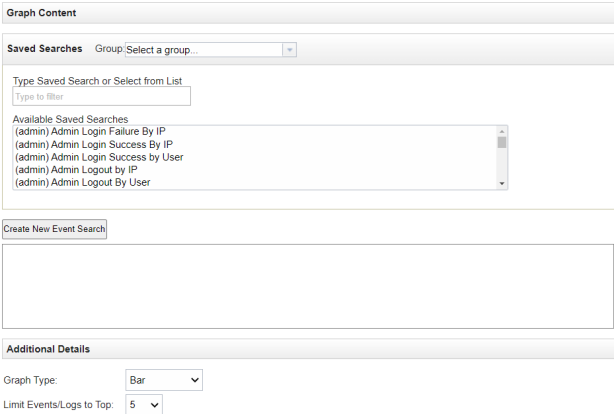


Figure 4. Report chart configuration window

After contents of the charts were defined, user can choose file format report will be compiled into. Next report configuration wizard's question is report distribution method: it is either done by sending it to the selected QRadar users or by sending it via email. Final configuration dialogue asks for brief report's description. With description submitted, report is considered created. Appendix 2 illustrates an example report which contains 2 charts: top authentication offences and top authentication failure events by user [13].

3.1.1 QRadar reporting system drawbacks

System works as intended and is conveniently built into the SIEM, however, there are a few drawbacks that made CYBERS consider another solution. These are the following in no particular order:

- There is no option to place alerts and events that contributed to them into one chart to show correlation. This makes investigation of the alert more difficult because report viewers will have to manually search events from a separate chart which is inconvenient.
- QRadar alert search functionality is limited to only search filters. There is no group offences by users or certain types.
- Generated reports are not interactive. There is no functionality to hide certain datasets from charts or collapse certain entries in order to hide information that is not needed at the moment.
- Very little control over final result's appearance. Only major appearance choice that user can make is report layout and logotype in the corner.

3.2 Creating reports using QReport

QReport is a custom solution made in CYBERS to generate custom reports using information retrieved from QRadar. Its purpose is to overcome QRadar reporting functionality limitations and provide a more flexible and more configurable reporting system. Project is a collection of Python scripts that interact with QRadar API, process retrieved information and insert results into templates. Figure 5 shows project's structure.

Reports are generated separately for each CYBERS SOC customer. To generate reports QReport retrieves a list of offenses which occurred within set time period for each customer and then, individually for each offense, it retrieves events that contributed to the offense. After the information has been retrieved QReport groups offences into a separate list where offences are grouped by the user that failed to authenticate. Finally, results are written

into 2 separate templates: first contains the offences in their order of occurrence, second lists the offences grouped by user. Figure 6 shows report where alerts listed in order of occurrence.

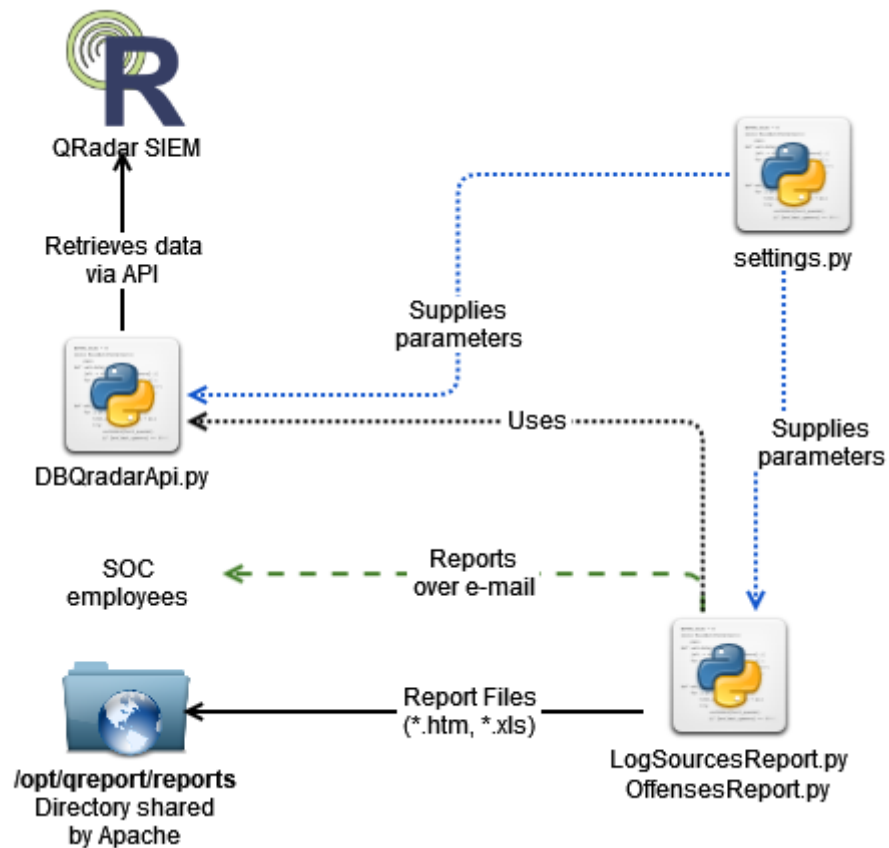


Figure 5. QReport structure

Failed Auth Offenses Report for Cybers - Mon, 25 Apr 2022, 04:59:45

Report Details	
Report Generation Date	Mon, 25 Apr 2022, 04:59:45
Report Period	Mon, 18 Apr 2022, 00:00:00 - Sun, 24 Apr 2022, 23:59:59
Reported Offenses	3 Offenses: <ul style="list-style-type: none"> Details Offense ID-s: 44415, 44416, 44412

Offense ID 44415																									
Description	Detected Potential Log4Shell Activity containing Bad Username																								
Status	OPEN																								
Start Time	23 Apr 2022, 12:35																								
Source Network	Cybers A																								
Categories	SSH Login Failed, Potential Web Exploit, Potential Web Exploit																								
Log Sources	2 Log Source(s): <ul style="list-style-type: none"> Details Custom Rule Engine-229 - qradar Linux OS @ machine B 																								
Offense Events	2 event(s) in 2 group(s), some recent event samples: <ul style="list-style-type: none"> Details <table border="1"> <thead> <tr> <th>Number of Events</th> <th>Log Source</th> <th>Event Type</th> <th>Username</th> <th>Source IP</th> <th>Destination IP</th> <th>Event Time</th> <th>Payload</th> </tr> </thead> <tbody> <tr> <td>1</td> <td>Linux OS @ machine B</td> <td>Bad Username</td> <td>test_user</td> <td>192.168.1.1</td> <td>192.168.1.1</td> <td>2022-04-23 12:35:49</td> <td><38-Apr 23 12:35:49 machine-b ssh@27624> input_userauth_request invalid user test_user [preauth]</td> </tr> <tr> <td>1</td> <td>Custom Rule Engine-229 - qradar</td> <td>Detected Potential Log4Shell Activity</td> <td>\$jndi</td> <td>192.168.1.2</td> <td>192.168.1.2</td> <td>2022-04-23 12:35:41</td> <td>Detected Potential Log4Shell Activity An event was found that could be associated with Log4Shell CVE-2021-44228</td> </tr> </tbody> </table>	Number of Events	Log Source	Event Type	Username	Source IP	Destination IP	Event Time	Payload	1	Linux OS @ machine B	Bad Username	test_user	192.168.1.1	192.168.1.1	2022-04-23 12:35:49	<38-Apr 23 12:35:49 machine-b ssh@27624> input_userauth_request invalid user test_user [preauth]	1	Custom Rule Engine-229 - qradar	Detected Potential Log4Shell Activity	\$jndi	192.168.1.2	192.168.1.2	2022-04-23 12:35:41	Detected Potential Log4Shell Activity An event was found that could be associated with Log4Shell CVE-2021-44228
Number of Events	Log Source	Event Type	Username	Source IP	Destination IP	Event Time	Payload																		
1	Linux OS @ machine B	Bad Username	test_user	192.168.1.1	192.168.1.1	2022-04-23 12:35:49	<38-Apr 23 12:35:49 machine-b ssh@27624> input_userauth_request invalid user test_user [preauth]																		
1	Custom Rule Engine-229 - qradar	Detected Potential Log4Shell Activity	\$jndi	192.168.1.2	192.168.1.2	2022-04-23 12:35:41	Detected Potential Log4Shell Activity An event was found that could be associated with Log4Shell CVE-2021-44228																		

Figure 6. QReport report example

3.2.1 QReport advantages over QRadar reporting system

QReport improves on QRadar built-in reporting capabilities in the following key areas:

- QReport allows additional post-processing of retrieved information: offences can be grouped by certain parameters and operations can be performed with retrieved data to calculate certain metrics like offence count per user, for instance.
- QReport allows usage of custom templates which gives greater control over report's appearance and viewing experience.

3.2.2 QReport shortcomings

QReport has a set of disadvantages which affect its usability and future development:

- QReport does not have a graphical user interface as it is essentially a console application. Application configuration is only possible by modifying the source code which requires direct access to the server. This limits CYBERS SOC employees' ability to interact with the system.
- QReport does not utilize a database for storing entities like customers. Adding a new customer to reporting process requires addition of a variable to the source code and adding filtering for certain offences requires inserting additional conditional statements into the report generation code.
- QReport does not have any graph plotting capabilities implemented. That leaves viewer with only textual representation of information which can be tedious to go through just to get an overview.

4. Methodology

In this chapter author describes tools and frameworks used in the new solution's design.

4.1 Approach

Based on the study of existing reporting systems employed in CYBERS SOC author will design a new system that is supposed to overcome their shortcomings. Resulting system will be comprised of freely available technologies with some of them being replaceable. Criteria for choosing technologies were ease of use, clear documentation, and being free and open-source.

4.2 Microsoft .NET

.NET(pronounced as 'dotnet') is a free, cross platform, and open-source(using MIT and Apache 2 licenses) development platform for building large variety of applications: web APIs and apps, mobile applications, desktop applications, machine learning. It is regularly updated for security and quality [15].

4.3 ASP.NET Core

ASP.NET Core is a framework used for building modern, cloud-enabled, Internet-connected apps. It runs on .NET runtime and has following features [16]:

- Ability to be hosted in variety of environments: IIS, Nginx, Apache, Docker
- Integration of modern client-side frameworks like Bootstrap
- Tooling that simplifies modern web development

ASP.NET Core MVC allows building web application and APIs using Model-View-Controller design pattern. MVC (Model-View-Controller) is a pattern in software design commonly used to implement user interfaces, data managing and application control logic. Its focal point is separation between the application's business logic and views displayed to the user. This "separation of concerns" provides for a better division of labor and improved maintenance. Figure 7 illustrates MVC pattern [16][17].

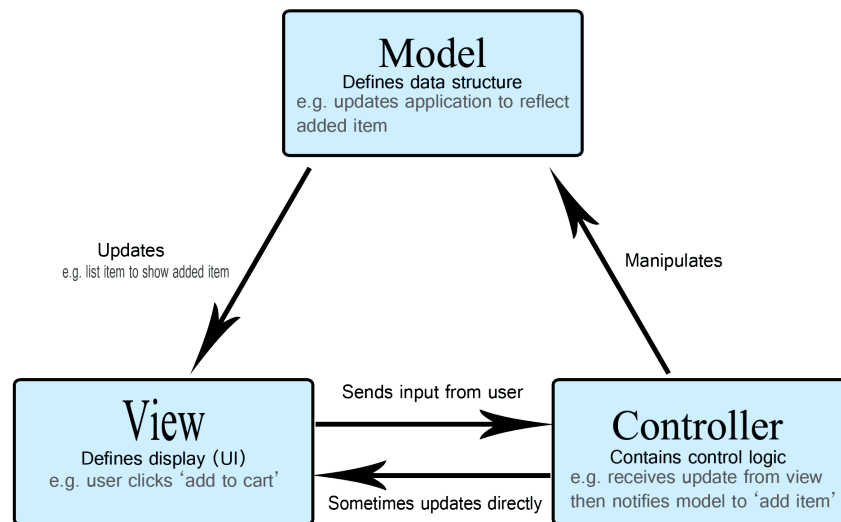


Figure 7. MVC pattern

ASP.NET Core MVC utilizes views to handle the application’s data presentation and user interactions. Views can be broken down into reusable components which allows to design modular pages. This functionality is beneficial to CYBERS reporting process, as different components can be utilized in multiple reports [18].

Implementing reporting solution as a web application solves the problem of not having a way to interact with the system. Modern web frameworks like Bootstrap simplify interface and report building process with the help of pre-built components [19].

4.4 Entity Framework Core

Entity Framework Core is a data access technology which supports many database engines. It enables developers to work with a database using .NET native objects and simplifies code by eliminating need for most of the code that handles connections to the database. Entity Framework Core also implements 'Code First Approach' where a database can be created based on the models defined within application logic [20].

Storing data in database and providing users with data manipulation capabilities via web interface will allow dynamic configuration of application as well as eliminating need for low-level access to solution.

4.5 Data visualization with Chart.js

It is much easier to discover and confirm the presence of patterns, relationships, and physical characteristics (such as outliers) through a visual display. Therefore, some data needs to be visualized to communicate in a clearer way [21].

Chart.js is a feature-rich JavaScript library used for client-side rendering of charts and graphs. Its wide variety of configuration options allow plotting complex graphs. Resulting graphs are interactive and dynamic: there is a possibility to filter out certain data and scales will be adjusted to ensure best viewing experience. Chart.js was chosen for its relative simplicity and object-oriented approach which matches with the rest of the technology stack. There are other feature-rich alternatives like "D3", but those can have a steeper learning curve. Data visualization component can be easily replaced with other variants, if need be [22][23].

4.6 QRadar API

QRadar SIEM has an API that allows retrieving information by sending specially crafted HTTP requests to specific endpoints. Each endpoint is responsible for specific functions and will respond differently depending on whether you send a GET, POST, or DELETE request. API allows integrating QRadar with other systems and taking advantage of its log attribute parsing, analytics applied to events, and optimized data retrieval [24].

5. Implementation

This chapter describes overall structure of the proposed solution and its components as well as general working principle.

5.1 Application architecture

Solution consists of a server-side application which is responsible for handling interactions with users, making calls to QRadar API and processing the data, as well as retrieving information from a database. Figure 8 illustrates application architecture.

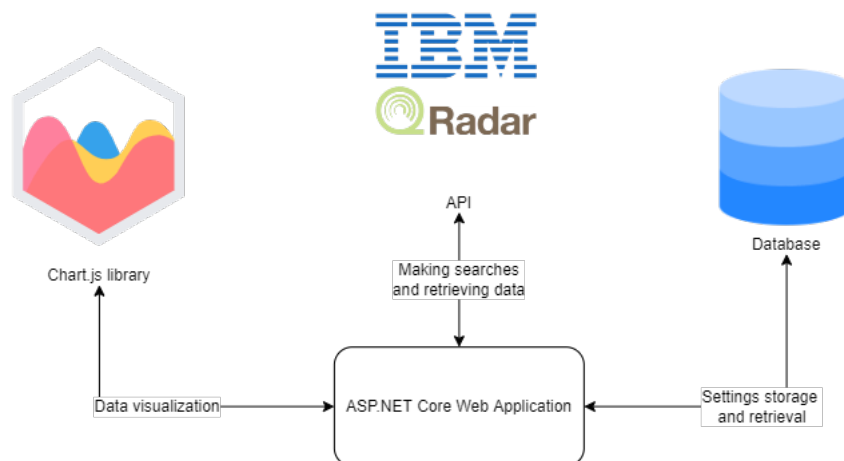


Figure 8. Application architecture

5.2 QRadar API client

QRadar API has many endpoints used for different purposes but only 2 are important in the context of this work [24]. These are as follows:

- **ariel/searches** - used for making queries with AQL and retrieving results.
- **siem/offenses** - used for retrieving alerts and information related to them.

QRadar API supports several response formats: JSON, CSV and XML etc, but not all endpoints support all formats [24]. Both aforementioned endpoints support JSON format

and this is what the application uses as standard. While JSON is not as space efficient as CSV, sticking to it as a default format allows for more uniform code base and easier parsing logic, which does not depend on the order of parameters.

Communication between application and QRadar API is done via separate HTTP client instance which encapsulates all logic related to constructing valid requests and parsing the responses into .NET objects. QRadar API client implementation is organized into "Client" and "Entities" namespaces (Figure 9). First contains methods for client configuration and its interaction with QRadar API, second contains logic for mapping of entities returned by API to .NET objects.

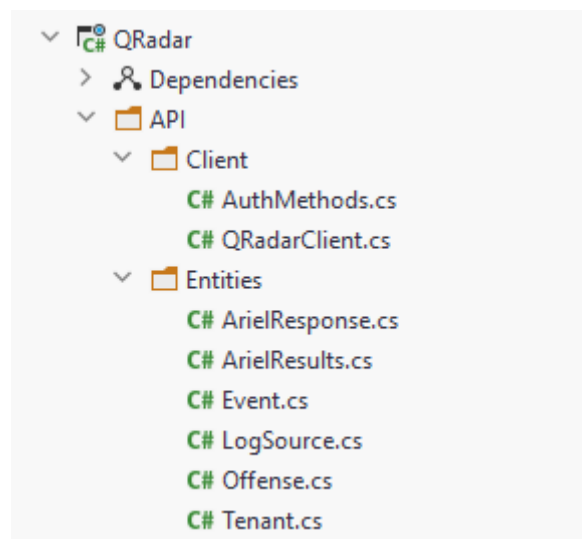


Figure 9. QRadar API project structure

On application initialization QRadar API client instance is registered as a service which can be used by other components of application. This way HTTP client instance is reused which avoids potential exhausting of available network sockets [25]. Client's default parameters such as base URL, accepted format and API token which is necessary since every request is authenticated are configured also on application initialization as shown on figure 10.

```

builder.Services.AddHttpClient<QRadarClient>(httpClient =>
{
    httpClient.BaseAddress = new
        Uri(builder.Configuration["QRadarBaseUrl"]);
    httpClient.DefaultRequestHeaders.Add(
        HeaderNames.Accept, "application/json");
    httpClient.DefaultRequestHeaders.Add(
        "SEC", builder.Configuration["QRadarAPIKey"]);
});

```

Figure 10. QRadar HTTP client initialization within application

5.3 Database

To overcome QReport's limitation of having to edit the source code to add additional report to the schedule application is going to utilize a relational database. Usage of a database will allow to dynamically add and remove entries using a graphical user interface which is provided by web application. Underlying database engine is unimportant in the context of this work, as it can be changed out for another one.

Application's domain consists of following models: "Report", "Search" and "Schedule". Based on definition of these models Entity Framework Core generates database schema which is applied to configured database instance.

"Report" model represents the authentication failure report. It has a target client's name, QRadar tenant ID which points to the client's tenant in the QRadar. Figure 11 shows model definition within application's code.

```

public class Report
{
    public int Id { get; set; }

    [Required]
    [MaxLength(128)]
    [DisplayName("Client's name")]
    public string ClientName { get; set; } = default!;

    [Required]
    [DisplayName("QRadar Domain")]
    public int QRadarTenantId { get; set; }
}

```

```

[DisplayName("Authentication failure search")]
public ICollection<Search>? Searches { get; set; }

[DisplayName("Schedule")]
public ICollection<Schedule>? Schedules { get; set; }
}

```

Figure 11. Report model in application code

"Search" model (Figure 12) contains 2 searches: one for retrieving alerts from the system, another for retrieving events. It has a foreign key that points back to parent "Report" record. "Report" and "Search" models have one-to-many relationship: report can have multiple searches, while search can only be tied to one report at a time. Having individual search objects for each report allows filtering to be defined within the search query and avoid defining it withing application logic.

```

public class Search
{
    public int Id { get; set; }

    [Required]
    [MaxLength(4096)]
    [DisplayName("Offenses' Search")]
    public string OffenseSearch { get; set; } = default!;

    [Required]
    [MaxLength(4096)]
    [DisplayName("Events' Search")]
    public string EventSearch { get; set; } = default!;

    [MaxLength(4096)]
    [DisplayName("Comment")]
    public string? Comment { get; set; }

    public int ReportId { get; set; }
    public Report? Report { get; set; }
}

```

Figure 12. Search model in application code

"Schedule" model (Figure 13) represents time period for which a report is generated. Fields marked with "Start" and "End" represent start and end times of reporting period. Fields that have 'Due' prefix stand for time by which report must be generated. "Due" fields are intended to be used by a background task that will go through schedules and generate

reports. Once report is generated it is marked as "Done" which tells the background tasks not to generate a duplicate. Models "Report" and "Schedule" also have one-to-many relationship which allows one report to have several schedules for more complex scenarios.

```
public class Schedule
{
    public int Id { get; set; }

    [Required]
    public DayOfWeek StartDay { get; set; };

    [Required]
    public TimeOnly StartTime { get; set; };

    [Required]
    public DayOfWeek EndDay { get; set; };

    [Required]
    public TimeOnly EndTime { get; set; };

    [Required]
    public DayOfWeek DueDay { get; set; };

    [Required]
    public TimeOnly DueTime { get; set; };

    public bool isDone { get; set; };

    public int ReportId { get; set; }
    public Report? Report { get; set; }
}
```

Figure 13. Schedule model in application code

Based on defined models and their relationships Entity Framework Core generated a schema for SQLite database engine (Figure 14).

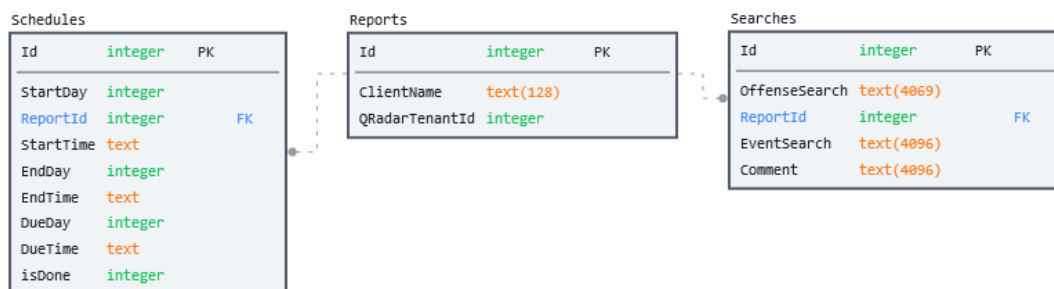


Figure 14. Application database schema

5.4 Failed authentication report generation process

To begin report generation process program retrieves "Report" object from database based on its primary key. Searches and schedules related to the report object are also retrieved. Retrieval of schedules is optional in case of on-demand report generation scenario, where report dates are supplied by user. Figure 15 shows code that retrieves the data using Entity Framework Core.

```
var report = await _context.Reports
    .Include(r => r.Schedules) // Optional, if manually
        specified
    .Include(r => r.Seaches)
    .FirstOrDefaultAsync(m => m.Id == id);
```

Figure 15. Code retrieving data from database

Next step is to compile search queries. First alert search is compiled from template which was retrieved from the database. Sections enclosed with "{}" are placeholders for parameters. Program supplies QRadar tenant id, start and end times of search. A sample of offense search is provided on Figure 16.

```
domain_id={0} and status='open' and start_time > {1} and
start_time < {2} and categories contains 'General
Authentication Failed'
```

Figure 16. Offenses search sample

Formatted search is sent by QRadar client to QRadar API, and result is all authentication related alerts for set time period. After results are returned and parsed, event search query can be compiled. Program replaces placeholders with tenant id, condition that ties events to alerts, as well as start and end times of the search. Example AQL search template is provided below (Figure 17).

```
SELECT
username AS username,
QIDNAME(qid) AS event_name,
devicetime/(4*3600*1000) AS time_span,
sum(eventcount) AS event_count
FROM events
```

```

WHERE domainid = {0} AND
hasoffense = TRUE AND
LOGSOURCETYPENAME(logsourceid) NOT IN ('Custom Rule Engine',
'Device type 63') AND
{1}
GROUP BY time_span, qid, username
ORDER BY time_span ASC
START {2}
STOP {3}

```

Figure 17. AQL search for failed authentication events

One major difference with previous solution, QReport, is that application described in this work retrieves all events with one API call, whereas QReport makes a separate request per each alert. Reduction of API call count is beneficial for performance and leaves less room for network errors [25]. One downside of this approach is that it does not allow to tie a group of events to a specific alert.

After source data is retrieved it is going to be processed to calculate metrics. Further data processing is done withing the application. While making additional searches with AQL to gather metrics like overall event and offense count, event count per user etc. would be faster than doing it within application, it could also negatively affect QRadar's performance. Searches for big tenants can be computationally expensive. QRadar is considered a mission critical component of CYBERS SOC, therefore unnecessary load is undesirable. Authentication failure reports can tolerate longer generation times, as long as they are generated before submission date.

First operation on retrieved data is grouping retrieved records by username (Figure 18). This grouping is later used for further transformations.

```

var eventsByUser = from record in Model.Events
                    group record by record.Username;

```

Figure 18. Code grouping events by username

To count how many events there were per user goes through aforementioned grouping and adds up event counts (Figure 19). Generated data is used to plot a main graph for the authentication failure report. This graph can be utilized to quickly identify users with alarming event count.


```

var eventCountByUser = from grouping in eventsByUser
    select new {
        Username = grouping.Key,
        EventCount = grouping.Sum(e =>
            e.EventCount)
    };

```

Figure 19. Counting events by user

To display more detailed information per user, events from each user grouping are further grouped by event names and by approximate time period of occurrence. Result gives overview of which events occurred at what time and in what quantities. This data is used to display events' approximate timeline per user. While there are no event payloads to see exactly what happened, giving a visual interactive overview should provide a better starting point for further investigations. Application logic is provided below (Figure 20).

```

foreach (var userGroup in eventsByUser)
{
    var eventsByNameAndByTime = from record in userGroup
        group record by record.EventName
        into eventGroups
        select new
        {
            EventName = eventGroups.Key,
            Events = from record in eventGroups
                group record by record.TimePeriod
                into timeGroups
                select new {
                    TimePeriod = (timeGroups.Key * 4 *
                        3600 * 1000),
                    EventCount = timeGroups.Sum(g =>
                        g.EventCount)
                },
        };
}

```

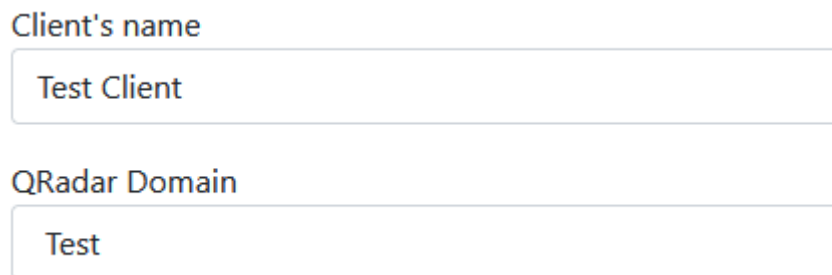
Figure 20. Grouping events by time and event type

6. Solution overview

This chapter gives an overview of the resulting solution and discusses the resulting systems benefits and negative points.

6.1 Report creation process

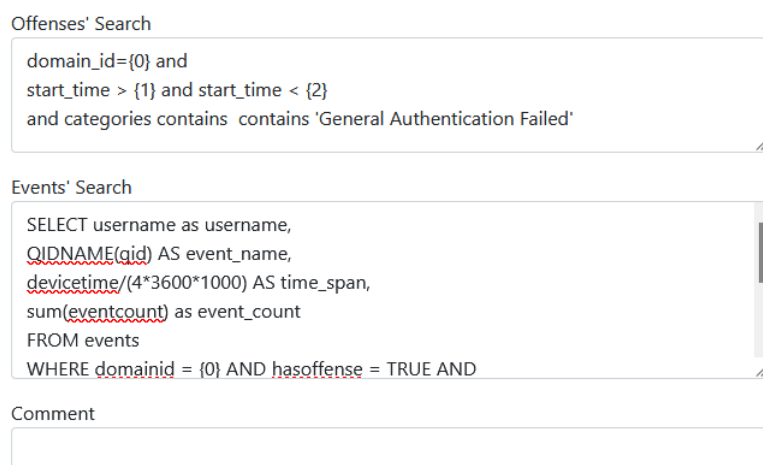
First step is to define customer's name and their corresponding tenant. Figure 21 depicts a report creation initial dialogue.



The screenshot shows a form with two input fields. The first field is labeled "Client's name" and contains the text "Test Client". The second field is labeled "QRadar Domain" and contains the text "Test".

Figure 21. Report creation initial dialogue

Next step is to define searches that will be used for retrieval of data. Additionally, user can leave a comment that explains filtering conditions within searches.



The screenshot shows a form with three sections. The first section is labeled "Offenses' Search" and contains the text: "domain_id={0} and start_time > {1} and start_time < {2} and categories contains contains 'General Authentication Failed'". The second section is labeled "Events' Search" and contains the text: "SELECT username as username, QIDNAME(qid) AS event_name, devicetime/(4*3600*1000) AS time_span, sum(eventcount) as event_count FROM events WHERE domainid = {0} AND hasoffense = TRUE AND". The third section is labeled "Comment" and is an empty text box.

Figure 22. Search definition dialogue

Final step is setting the schedule for report generation.

StartDay: Sunday

EndDay: Sunday

StartTime: --:-- --

EndTime: --:-- --

DueDay: Sunday

DueTime: --:-- --

Figure 23. Schedule setting dialogue

To generate report on-demand 'Generate' button can be used which is displayed on the 'Reports' tab landing page.

Index

[Create New](#)

Client's name	QRadar Domain	
Cybers test report	3	Generate Edit Details Delete

Figure 24. On-demand generation option

After information defined in searches had been received from QRadar and processed, application renders the final result. Report main page is shown in Appendix 3. Figure 25 illustrates details section for a particular user.

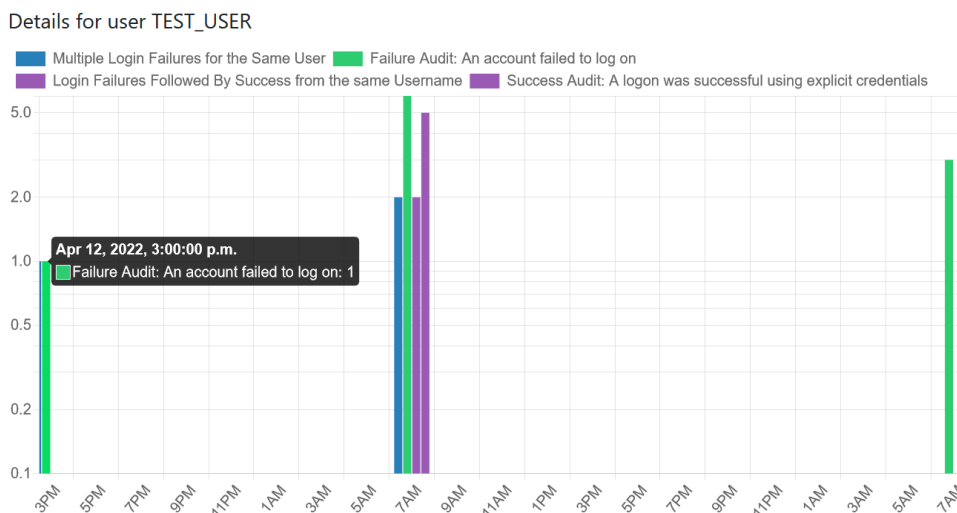


Figure 25. Event details per user

Graph on Figure 25 is interactive. It allows crossing out certain event types, so that viewer could concentrate on events that are of interest to them.

6.2 Solution analysis

Resulting application has the following benefits:

- It is fully configurable. Every aspect of the application can be tuned. Range of reports that can be generated using this system is only limited by source data. Modular design allows switching out certain components like graph plotting library, if a more suitable alternative is found and sufficiently studied.
- Solution is extensible. Through code and components reuse, other CYBERS SOC reporting procedures can also be implemented in this application.
- Solution provides a web interface for report configuration and generation.
- Database allows for more granular configuration of report generation. Searches and schedules can be adjusted on each report individually using of graphical user interface.
- Data is visualized. This enables the SOC analysts to get a quick overview of the situation. Filtering options allow viewing certain types of events. This gives a good starting point for investigation if an anomaly is detected.

However new solution also introduces some problems and points for consideration.

- System configuration and development requires considerable effort. Due to multi-component nature of the application, implementation of new features or modifying old ones require sufficient knowledge of all involved tools. Modifying graphs will require knowledge about Graph.js etc.
- Current 'schedule' model limits application to only using weekdays as start and end dates. This is unsuitable for generating monthly reports for example were start and end of the month are not fixed to certain weekdays. Database schema likely needs to be modified to allow multiple ways of scheduling
- Report does not contain event's payloads for further troubleshooting. These need to be queried separately from QRadar.

6.3 Future work

Future development would include inclusion of alerts into the graphs to illustrate correlation between alerts and events. Application was tested with auxiliary instance of QRadar which

does not include all the latest fine-tuning improvements done by CYBERS SOC team. As a result, data retrieved from this instance contains more 'noise' which complicates correlation between users, events, and alerts.

To make this application suitable for production use, data that is submitted by user from web interface must be validated for its correctness. For example, queries that are submitted to be used in the searches have to be tested against QRadar API to ensure that query syntax is correct, and that application is able to parse the result.

To make generated reports look coherent with the rest of the documentation, web development tools need to be adjusted according to CYBERS report design guidelines.

7. Summary

The goal of this work was to improve reporting capabilities of CYBERS SOC by designing a new system that would be integrated with existing monitoring system.

Author gave an overview of CYBERS SOC and its reporting process, as well as an overview of reporting systems in use. Potential of alternative reporting solutions was also discussed. Author came to a conclusion that existing reporting solutions do not easily integrate with existing processes and infrastructure.

After reviewing CYBERS SOC reporting process and existing reporting solutions, author identified shortcomings of current solutions. In proposed solution author attempted to address these weaknesses by employing technologies specifically designed to solve a certain problem. Result is a web application that provides a graphical user interface to interact with the system and allows individual configuration of reports. New solution uses modern technologies that simplifies development when compared to development "from scratch". Information is presented primarily with graphs which simplifies analysis by analysts and clients.

Focus of this work was only one of the reporting processes and proposed solution aimed to improve it. Author managed to improve CYBERS SOC reporting process and the latter approved the solution for use. In the future solution could be extended to implement other reporting processes.

Bibliography

- [1] Omar Santos Joseph Muniz Aamir Lakhani and Moses Frost. *The Modern Security Operations Center*. Addison-Wesley Professional, Apr. 2021. [E-book].
- [2] Check Point. *SOC-as-a-Service*. 2021. URL: <https://www.checkpoint.com/cyber-hub/threat-prevention/what-is-soc/soc-as-a-service/>. (visited on 16.09.2022).
- [3] Don Murdoch. *Blue Team Handbook*. Mar. 2019.
- [4] John Hubbard. *Guide to Security Operations*. 2020. URL: <https://sansorg.egnyte.com/dl/TLpDjvybnc>. (visited on 16.09.2022).
- [5] Nadhem AlFardan Joey Muniz Gary McIntyre. *Security Operations Center: Building, Operating and Maintaining your SOC*. Cisco Press, Nov. 2015. [E-book].
- [6] Splunk. *What Is SIEM?* 2022. URL: https://www.splunk.com/en_us/data-insider/what-is-siem.html. (visited on 16.09.2022).
- [7] IBM. *Why is SIEM important?* 2022. URL: <https://www.ibm.com/topics/siem>. (visited on 16.09.2022).
- [8] Inforegister. *SECURITY SOFTWARE OÜ*. 2022. URL: <https://www.inforegister.ee/en/11924368-SECURITY-SOFTWARE-OU>. (visited on 16.09.2022).
- [9] CYBERS. *Services*. 2022. URL: <https://cybers.eu/en/services>. (visited on 16.09.2022).
- [10] Martin Nystrom Chris Fry. *Security Monitoring*. O'Reilly Media, Feb. 2009. [E-book].
- [11] Elastic. *Kibana. Your window into the Elastic Stack*. 2022. URL: <https://www.elastic.co/kibana/>. (visited on 16.09.2022).
- [12] jsreport. *javascript reporting server*. 2021. URL: <https://jsreport.net/>. (visited on 16.09.2022).
- [13] IBM. *IBM QRadar Security Intelligence Platform version 7.4*. 2022. URL: <https://www.ibm.com/docs/en/qsip/7.4>. (visited on 16.09.2022).
- [14] IBM. *Ariel Query Language Structure*. 2022. URL: <https://www.ibm.com/docs/en/qradar-on-cloud?topic=aql-query-structure>. (visited on 16.09.2022).

- [15] Microsoft. *What is .NET? Introduction and overview*. 2022. URL: <https://docs.microsoft.com/en-us/dotnet/core/introduction>. (visited on 16.09.2022).
- [16] Microsoft. *Overview to ASP.NET Core*. 2022. URL: <https://docs.microsoft.com/en-us/aspnet/core/introduction-to-aspnet-core?view=aspnetcore-6.0>. (visited on 16.09.2022).
- [17] MDN Web Docs Glossary: Definitions of Web-related terms. *MVC*. 2022. URL: <https://developer.mozilla.org/en-US/docs/Glossary/MVC>. (visited on 16.09.2022).
- [18] Microsoft. *Views in ASP.NET Core MVC*. 2022. URL: <https://docs.microsoft.com/en-us/aspnet/core/mvc/views/overview?view=aspnetcore-6.0>. (visited on 16.09.2022).
- [19] Bootstrap team. *Build fast, responsive sites with Bootstrap*. 2022. URL: <https://getbootstrap.com/>. (visited on 16.09.2022).
- [20] Microsoft. *Entity Framework Core*. 2021. URL: <https://docs.microsoft.com/en-us/ef/core/>. (visited on 16.09.2022).
- [21] Andrew Rininsland Andy Kirk Simon Timms and Swizec Teller. *Data Visualization: Representing Information on Modern Web*. Packt Publishing, Sept. 2016. [E-book].
- [22] Chart.js contributors. *Simple yet flexible JavaScript charting*. 2021. URL: <https://www.chartjs.org/>. (visited on 16.09.2022).
- [23] Mike Bostock. *Learn D3: Introduction*. 2020. URL: <https://observablehq.com/@d3/learn-d3>. (visited on 16.09.2022).
- [24] IBM. *What's new in REST API Version 16.0*. 2021. URL: <https://www.ibm.com/docs/en/qradar-common?topic=160-whats-new-in-rest-api-version>. (visited on 16.09.2022).
- [25] Microsoft. *ASP.NET Core Performance Best Practices*. 2022. URL: <https://docs.microsoft.com/en-us/aspnet/core/performance/performance-best-practices?view=aspnetcore-6.0#pool-http-connections-with-httpclientfactory>. (visited on 16.09.2022).

Appendix 1 - Non-exclusive licence

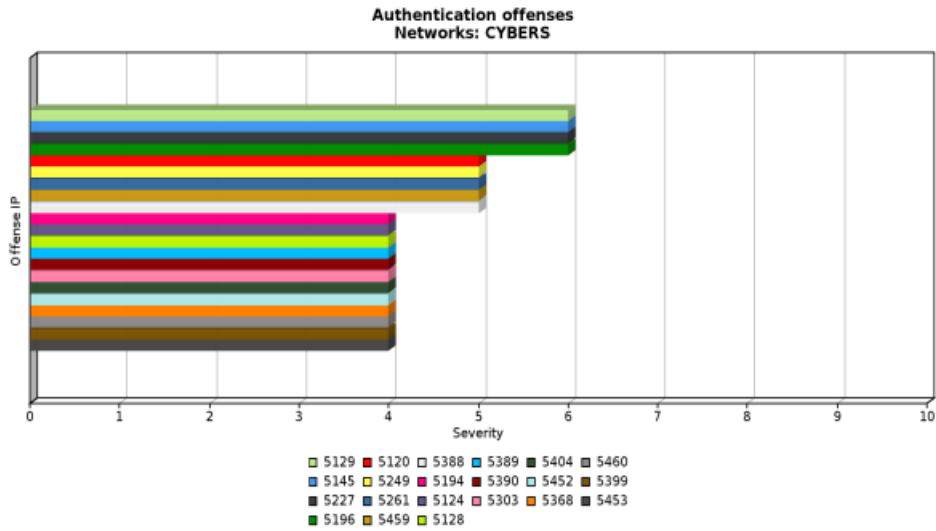
I Vjatšeslav Rukavišnikov

1. Grant Tallinn University of Technology free licence (non-exclusive licence) for my thesis "Improving Reporting Process in CYBERS Security Operations Center", supervised by Jürgen Erm
 - (a) to be reproduced for the purposes of preservation and electronic publication of the graduation thesis, incl. to be entered in the digital collection of the library of Tallinn University of Technology until expiry of the term of copyright;
 - (b) to be published via the web of Tallinn University of Technology, incl. to be entered in the digital collection of the library of Tallinn University of Technology until expiry of the term of copyright.
2. I am aware that the author also retains the rights specified in clause 1 of the non-exclusive licence.
3. I confirm that granting the non-exclusive licence does not infringe other persons' intellectual property rights, the rights arising from the Personal Data Protection Act or rights arising from other legislation.

Appendix 2 - QRadar report example

Authentication failure report

Generated: Apr 8, 2022, 7:05:30 PM



Authentication failures events by user

Top Authentication Failures by User
Apr 1, 2022, 1:00:00 AM - Apr 3, 2022, 1:30:00 AM

Username	Log Source (Unique Count)	Event Name (Unique Count)	Low Level Category (Unique Count)	Source IP (Unique Count)	Destination IP (Unique Count)	Geographic Country/Region (Unique Count)	Event Count (Sum)	Count
admin	Multiple (2)	Multiple (4)	Multiple (4)	Multiple (13)	Multiple (7)	Multiple (7)	100	89
user	Multiple (2)	Multiple (4)	Multiple (3)	Multiple (15)	Multiple (7)	Multiple (7)	84	77
oracle	Multiple (2)	Multiple (2)	Multiple (2)	Multiple (15)	Multiple (7)	Multiple (7)	57	56
postgres	Multiple (2)	Multiple (3)	Multiple (3)	Multiple (16)	Multiple (5)	Multiple (5)	53	53
test	Multiple (2)	Multiple (3)	Multiple (2)	Multiple (15)	Multiple (6)	Multiple (6)	45	44
pi	Multiple (2)	Multiple (4)	Multiple (3)	Multiple (11)	Multiple (8)	Multiple (8)	33	33
ansible	LinuxServer@	Bad Username	SSH Login Failed	Multiple (11)	Multiple (5)	Multiple (5)	30	30
testuser	LinuxServer@	Bad Username	SSH Login Failed	Multiple (7)	Multiple (5)	Multiple (5)	20	20
ubnt	Multiple (2)	Multiple (2)	Multiple (2)	Multiple (6)	Multiple (4)	Multiple (4)	17	17
server	Multiple (2)	Multiple (2)	Multiple (2)	Multiple (8)	Multiple (5)	Multiple (5)	17	17

Figure 26. QRadar report example (obfuscated)

Appendix 3 - New solution report example

Summary [Details](#)

Authentication Failure Report for Cybers

18.04.18 12:25 - 24.04.2022 12:25

Executive summary:

Total Alerts	17
Total Users	32
High severity alerts	1
High risk users	2

Suggested actions:

- Review user 133

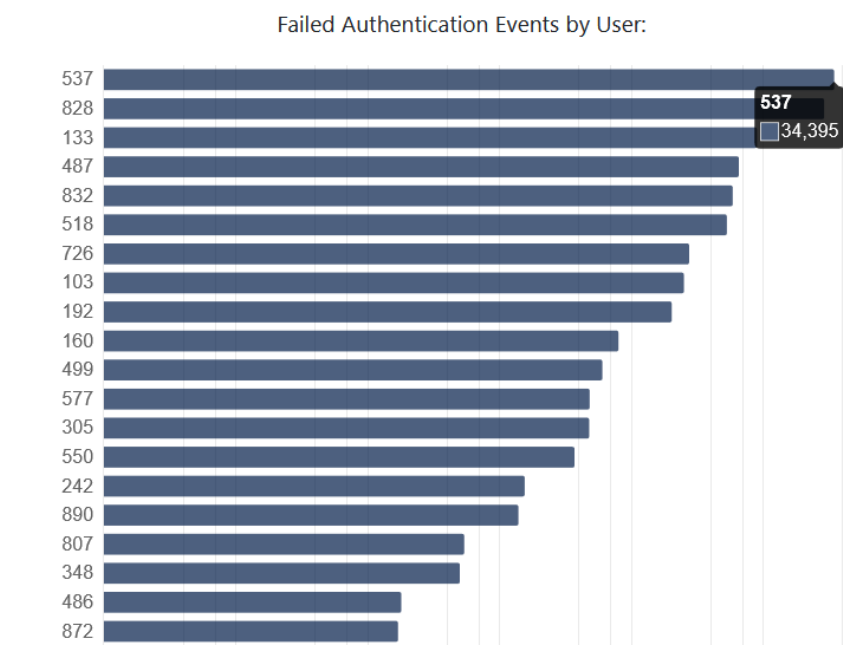


Figure 27. New solution report example (obfuscated)