VAASAN AMMATTIKORKEAKOULU
UNIVERSITY OF APPLIED SCIENCES

Juha Luukkanen

CYBER SECURITY INTELLIGENCE NEWS AGGREGATION AND THREAT ANALYSIS

School of Technology
2022

VAASAN AMMATTIKORKEAKOULU
Tietotekniikka

TIIVISTELMÄ

Tekijä	Juha Luukkanen
Opinnäytetyön nimi	Tietoturvallisuuden uhkatiedustelun koostaminen ja analysointi
Vuosi	2022
Kieli	englanti
Sivumäärä	30 + 3 liitettä
Ohjaaja	Jukka Matila

Opinnäytetyön tavoitteena on tutkia, kehittää ja arvioida, kuinka uutiskoostajapalvelu Feedlyn avulla voitaisiin rikastaa Wärtsilän kyberuhkatiedustelun raportointia ja muodostaa uhkatiedustelun kanavointi. Tämä mahdollistaa nopeampia sekä informoidumpia, datan tukemia turvallisuuspäätöksiä ja muuttaa puolustustoimet uhkateikijöitä vastaan ennakoivimmiksi.

Uhkatiedustelu on tietoa, jota kerätään, käsitellään ja analysoidaan, jotta voitaisiin ymmärtää uhkatekijöiden motiiveja, kohteita sekä hyökkäysmenetelmiä. Tätä tietoa kerätään yleisesti manuaalisesti käyden läpi eri uutislähteitä. Feedlya voi muokata keräämään tietynlaisia artikkeleita, erillisten työryhmien tarpeiden mukaisesti.

Automatisoitu uutisten koostaminen nopeuttaa vakavien haavoittuvuuksien huomaamista. Useiden uutislähteiden selaamisen sijaan asiantuntija saa tällä tavoin kaikki oleelliset artikkelit automaattisesti yhteen paikkaan kerättynä päivittäin ja mahdollistaa asiaan liittyvää keskustelua muiden asiantuntijoiden kanssa. Tämä saavutetaan luomalla kanavointi uutislähteiden ja käyttöliittymän välillä, artikkeleita jäsentäen näiden välissä.

Avainsanat tietoturva, tehokkuus, uutiset

VAASAN AMMATTIKORKEAKOULU
UNIVERSITY OF APPLIED SCIENCES
Information Technology

ABSTRACT

Author	Juha Luukkanen
Title	Cyber Threat Intelligence News Aggregation and Threat Analysis
Year	2022
Language	English
Pages	30 + 3 Appendices
Name of Supervisor	Jukka Matila

The goal of the thesis is to research, develop and evaluate how Feedly, a news aggregator service, could be used to enrich the cyber threat intelligence reporting at Wärtsilä and to create a threat intelligence pipeline. This enables faster, more informed, data-backed security decisions with an objective to change behavior from reactive to proactive in the fight against threat actors.

Threat intelligence is data that is collected and analysed to understand a threat actor's targets and methods, usually collected by manually going through different news sources. Feedly can be specified to gather certain kinds of articles according to the needs of individual teams.

Automated news aggregation increases the speed in which critical vulnerabilities are noticed. Instead of an expert sifting through all their news sources manually and collecting articles to publish in a weekly newsletter, they can have all relevant articles collected automatically into one place daily. This is achieved by creating a pipeline between a news source and a user interface, parsing the articles in-between.

Keywords Data security, stimulation, and news

CONTENTS

TIIVISTELMÄ

ABSTRACT

1 THREAT INTELLIGENCE 6

1.1 Wärtsilä 6

1.2 Cyber Security in Wärtsilä 6

1.3 Gathering Threat Intelligence 8

1.4 Thesis Objectives 10

2 FEEDLY 11

2.1 Tuning Feedly 11

2.2 Team Boards 15

2.3 Subscription Levels 16

2.4 Use Cases 17

3 DEVELOPMENT 19

3.1 CI/CD Pipeline Structure 19

3.2 Code Structure 20

3.3 Testing Locally and in Development 25

3.4 Pipeline and PowerBI 26

4 RESULTS 30

4.1 Expectations 30

4.2 Result and Challenges 30

4.3 Improvement and Future 31

REFERENCES 32

LIST OF FIGURES

Figure 1. Cyber security teams generalised. 7

Figure 2. Different layers of threat intelligence. 8

Figure 3. Web alert containing a custom topic -list and search logic (Feedly, 2022, online) 12

Figure 4. View of what a feed looks like (Feedly, 2022, online). 13

Figure 5. Vulnerability Leo found from the article (Feedly, 2022, online). 14

Figure 6. CVE insight card (Feedly, 2022, online). 15

Figure 7. Process of generating articles with Feedly. 16

Figure 8. Feedly's different subscription services (Feedly, n.d, online). 17

Figure 9. AWS structure. 20

Figure 10. General structure of the code. 21

Figure 11. Establishing a connection to a Feedly board in a get() -function (screenshot from VS Code). 21

Figure 12. Invoking all the API-calls (screenshot from VS Code). 22

Figure 13. Inserting new data in to the database in a create() -function (Screenshot from VS Code). 23

Figure 14. Modeling a table (screenshot from VS Code). 23

Figure 15. Example of what data is written into the database (screenshot from TablePlus). 25

Figure 16. Testing method. 26

Figure 17. Pipeline structure. 27

Figure 18. News dashboard created for the articles (screenshot from Power BI). 28

Figure 19. How a saved article appears in the Slack channel (screenshot from Slack). 29

1 THREAT INTELLIGENCE

The world of cyber security is a constant battle between cyber criminals and cyber defenders. Knowing threat actors' methods and next move is crucial for proactively tailoring defenses and pre-empting future attacks and breaches, which is why threat intelligence is crucial for every company in the digital world.

With more services and data storage methods having turned completely digital, a cyber-attack stealing and even completely wiping all data can potentially result in devastating financial losses, stolen information (concerning both company and personnel info), damage to equipment and potentially company losing face and trust in the eyes of the public. Also having the company's customer data compromised also causes mistrust and may also lead to further legal consequences and fines. In some cases, some smaller companies have even gone completely bankrupt, for they did not have appropriate data backups and permanently lost all information and data (Hashim, 2022, online).

1.1 Wärtsilä

Wärtsilä is a Finnish company focusing on innovative technologies and lifecycle solutions for the marine and energy markets. It emphasizes on sustainable technology to improve environmental and economic performance. Wärtsilä employs around 18000 people in 68 countries all over the world and had a net sale of 4,8 billion euros in 2021. Its head office is in Helsinki, Finland (Wärtsilä, 2022, online).

1.2 Cyber Security in Wärtsilä

Cyber security in Wärtsilä is branched among multiple teams focusing on different areas. All cyber-focused teams work together in similar tasks and even in same projects but have focus on different areas of operations (Figure 1).

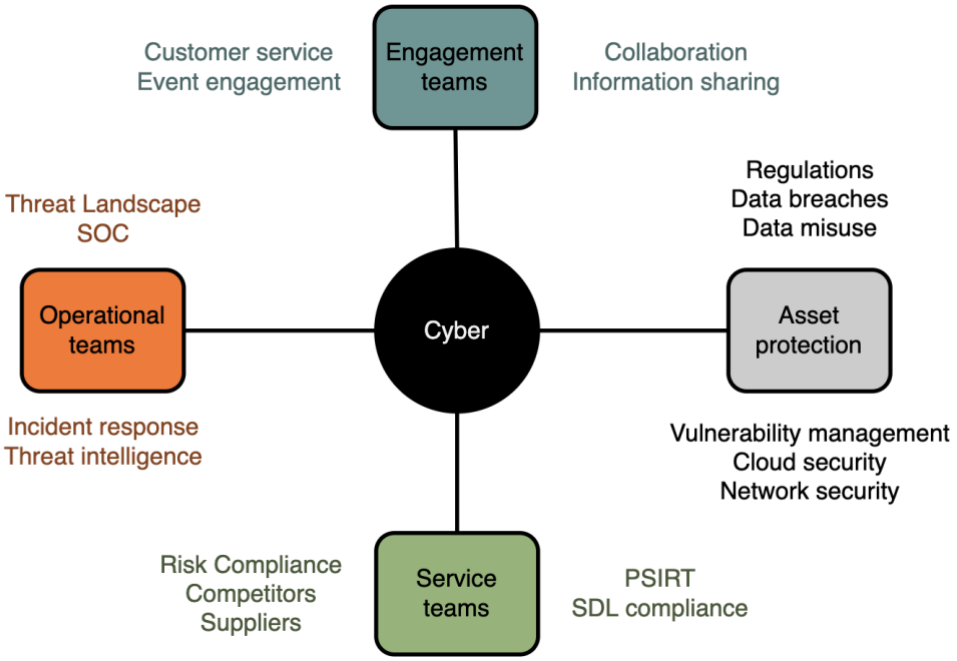

Figure 1. Cyber security teams generalised.

Security Operations Center (SOC) is a major part of Cyber Operations. Its purpose is to respond to cyber security incidents and to identify, analyze, isolate, remediate and report them. SOC is the main contact point for internal and external personnel to report if they believe an incident is going on (Wärtsilä, 2022, online).

The asset protection teams focusing on information security and data protection aims to protect Wärtsilä's information and assets by designing security and privacy implementations. They also evaluate and update policies and risks involving them. They guide all Wärtsilä's employees on data protection, privacy and everybody's responsibility within. They make sure the company's data protection policies comply with GDPR within the Wärtsilä companies in EU. While the GDPR does not affect Wärtsilä companies outside the EU, these teams ensure the same policies are followed in these geographical areas, as well (Wärtsilä, 2022, online).

The team working on security services focuses more on interacting with customers and products, both from marine and energy side. It provides centralized approach

to vulnerability management across Wärtsilä's products and offers commercial cyber services to Wärtsilä's customers (Wärtsilä, 2022, online).

Engagement teams focus on first- and third-party risk management, policy compliance, cyber security training and cyber security strength assesments. Some teams and members also focus more on engaging with other teams and companies for collaboration on various projects and incidents (Wärtsilä, 2022, online).

1.3 Gathering Threat Intelligence

To be able to prevent compromise of important resources, threat intelligence must be gathered. Cyber threat intelligence (CTI) is generally formed in multiple ways, with the three most common being strategic, tactical and operational threat intelligence (Figure 2) (Correa, 2022, online).

Figure 2. Different layers of threat intelligence.

Strategic threat intelligence (STI) is focused on the general threat landscape for an organization. To be able to focus on specific threats, a broader understanding of assets and risks along with attackers' motives and capabilities must be assessed. STI focuses on figuring out **who** is a potential threat and **why.** The purpose of STI is to help non-technical, higher-level executives to understand the situation and impact of the current threat landscape and help them make decisions on policies and guidance. STI is mainly composed of open-source intelligence, such as news articles, blog posts, social media posts, comments and intelligence reports (Recorded Future, 2018, online).

Operational threat intelligence (OTI) is more detailed on analyzing threats posed by threat actors and their methods. This requires people who have a deeper understanding of the operating environment and tools used in cyber operations and answers questions like **how** attacks occur and **where**. The **how** helps understanding the tools and methods of attackers and the **where** relates to identifying where the incident would take place and preventing that. This information is mostly used by information technology (IT) service managers and SOC managers. OTI sources come also mainly from open sources[1]; them being malware analysis data, incident reports, attack group and campaign reports and human intelligence in the dark web (SOCRadar, 2021, online).

Tactical threat intelligence (TTI) is specific knowledge on cyber-attacks and campaigns. TTI is real-time information utilized to help in responding to active threats and trying to figure out the threat actors' next moves. This information can help organizations improve their security measures and answers the question of **what** is the current threat. TTI is mostly collected by infiltrating hacker chat groups and other, often private discussions within the dark web (SOCRadar, 2021, online).

[1] Open-source intelligence, also known as OSINT, is collecting and analysing data that is gathered from public sources, such as news, public websites, blogs, social media posts etc.

1.4 Thesis Objectives

Much of threat intelligence online must be searched manually through countless number of sources online and each of the articles and posts have to be assessed and analyzed to evaluate if the information is valid and of use or not. Many cyber security experts spend a lot of their time weekly to go through their own sources and compile wanted information into a weekly newsletter, for example.

The objective of this thesis project is to automate collecting OSINT and increase the speed at which the information can be acquired and processed with the help of Feedly, which is a news aggregation service. It will not replace the experts doing their job, but instead makes gathering information much faster and easier, granting more time for experts to focus on dealing with the other work related to dealing with cyber threats. The articles are also to be put in a dashboard in PowerBI, a reporting tool, which can then be embedded into other software and web pages.

2 FEEDLY

Feedly is a news aggregator released in 2008 and is currently the most popular RSS-reader with over 14 million users as of 2018 (Barrett, 2018, online). It incorporates an AI, Leo, that collects information based on the configured preferences from around the web into one place, be it a computer or a mobile device. It can be used to view news on any selected topic, whether the information comes from the news, blogs, social media posts, RSS-feeds[2], or other sources. Feedly tuning needs to be focused on Wärtsilä, the tools that Wärtsilä uses and the suppliers and customers that it works with.

Information gathered by Feedly improves the rate of speed, by which for example vulnerabilities can be found. Whenever vulnerabilities or exploits are found within hardware or software, experts can check whether it applies to any company assets or not. Without Feedly, experts have to go manually through their preferred news sources, instead of having everything readily collected into one place.

2.1 Tuning Feedly

Feedly's artificial intelligence (AI), Leo[3], needs to be tuned to search interesting and topic-related articles to cut through the noise. Every organization and team withing have different interests in what kind of information they are looking for, and since Leo looks for information all around the web, it needs to also be filtered to make assessing the articles easier and less time consuming. For example, the operational teams have a higher interest regarding vulnerabilities and threats than the engagement teams.

The feeds can be tuned to gather related articles in multiple ways. "Custom topics" can be created in Feedly, which is a list containing keywords and concepts regarding assets, organisations, events and activity that Leo will look for. These

[2] RSS (Really Simple Syndication) is a web feed that can be allows keeping track of updates on websites with news aggregators.
[3] Leo is the name Feedly gave to its AI.

lists can be further refined in “web alerts” to apply certain logic to which articles should be pulled, such as defining whether to include articles that only concerns certain operating systems or versions, has high enough severity or articles that include a proof of exploit (Figure 3). Additional and specific sources can also be set as searching points for Leo. It can be set to go through specified social media accounts, subreddits, blogs, databases and RSS-feeds.

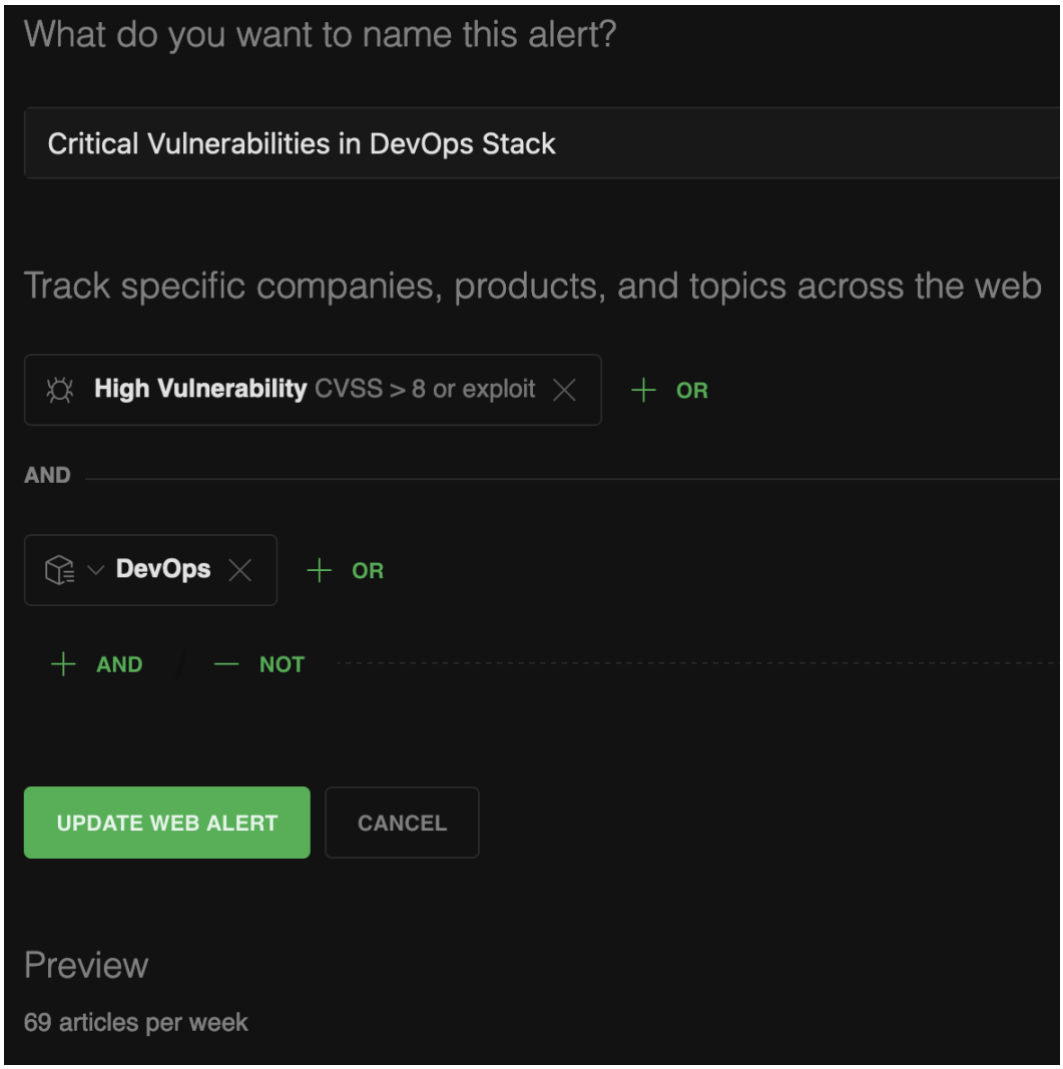

Figure 3. Web alert containing a custom topic -list and search logic (Feedly, 2022, online)

Once the feed has produced articles, they can be viewed from their respective feed tabs. The user can change how the articles are listed on the page, but by default they are shown as in digital news magazines (Figure 4).

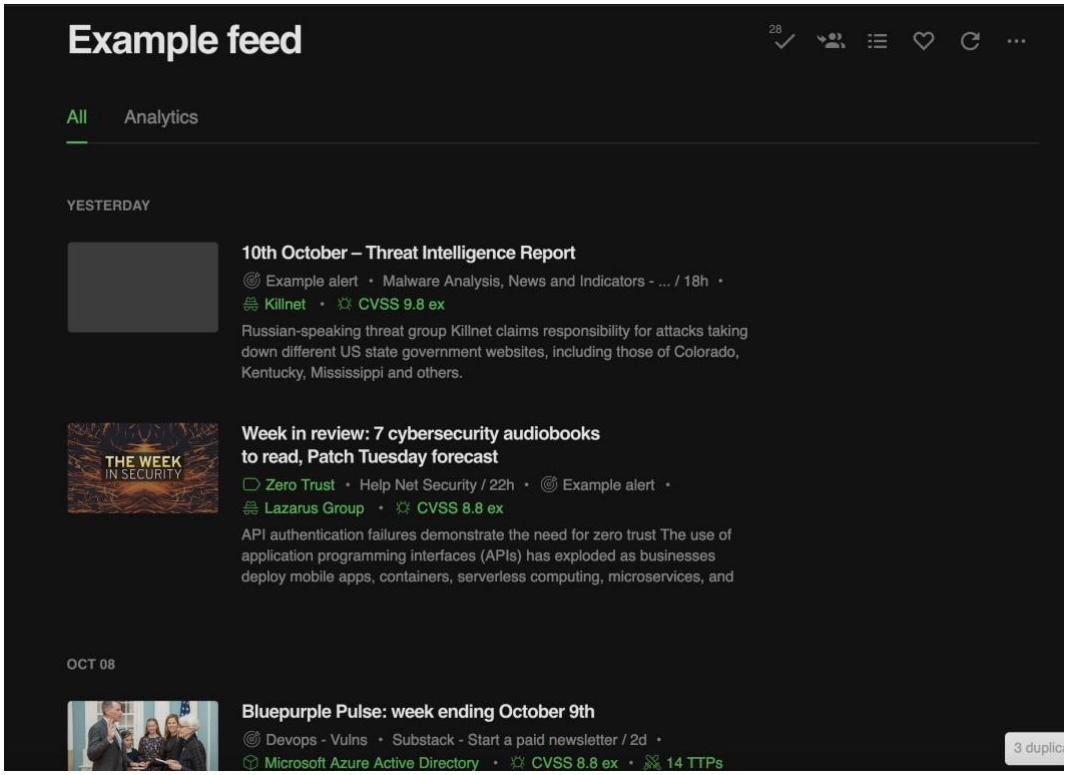

Figure 4. View of what a feed looks like (Feedly, 2022, online).

Articles in Feedly can also be expanded for more information. For example, Leo will search, and list all found vulnerabilities concerning the article (Figure 5), show the given common vulnerability score (CVSS)[4], or give its own estimation if a score is not given. It also highlights the most important parts of the article, automatically forming a summary. The text within can also be manually highlighted by users and comments can also be added to give more insight or to ping someone from

[4] CVSS (common vulnerability scoring system) is a standardized score given to vulnerabilities to measure its severity, with 0.0 being the lowest severity and 10.0 being the highest. The assessment is based on how many systems are potentially affected and how easily the vulnerability is exploited (i.e., how much does it require user interaction to be exploited).

messaging platforms, such as Slack or Teams. Links to the article sources are also provided and the article can be shared practically anywhere.

Figure 5. Vulnerability Leo found from the article (Feedly, 2022, online).

"CVE[5] insight cards" can also be expanded in Feedly to gain more insight of the vulnerability (Figure 6). The CVE insight cards also provide statistics on awareness levels over time and show all related articles to it for further reading.

[5] CVE (common vulnerabilities and exposures) is a collection of known security issues. Every vulnerability and exploit is given its own CVE and a unique identifier number.

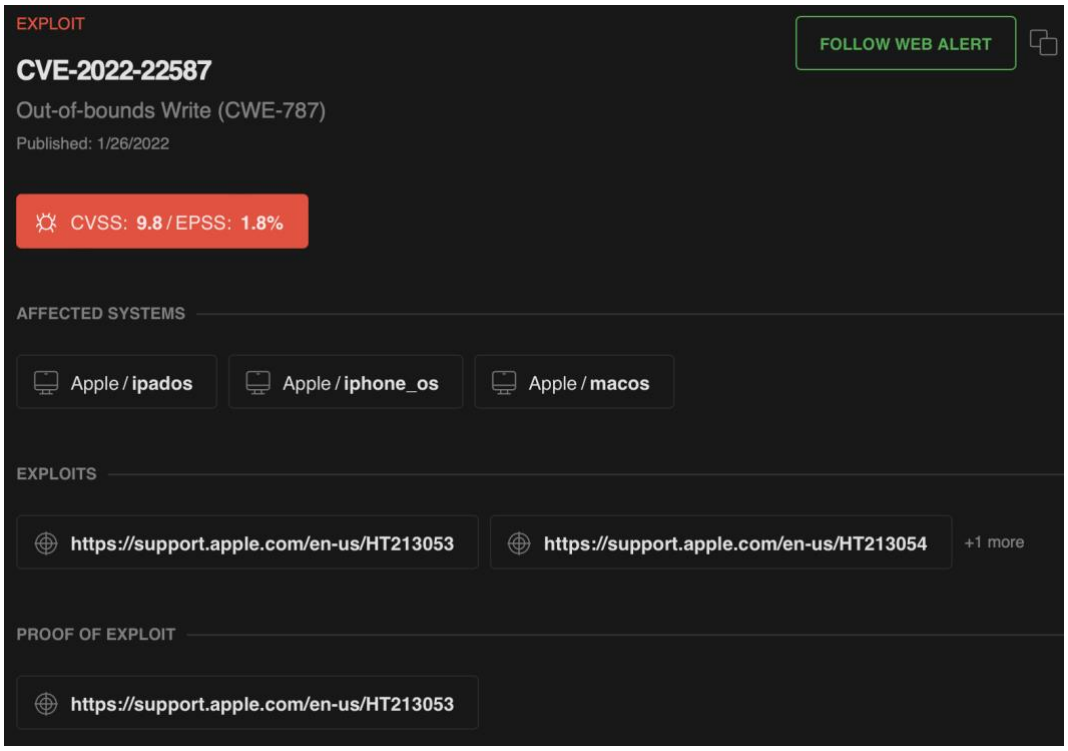

Figure 6. CVE insight card (Feedly, 2022, online).

2.2 Team Boards

To fully utilize and further improve the machine learning aspect of Leo, users need to approve or disapprove Leo's search choices to train it. Irrelevant articles that do not actually meet the criteria set for the feed or do not simply fulfil the interests can be downvoted to discourage Leo for looking that kind of articles. Wanted articles on the contrary can be saved to "boards" created by the user. These boards act as a sort of a folder, where permitted users can read view all the articles saved into it. Articles from multiple feeds can be saved onto a single board. In this project, each team is created their own respective board.

In summary, a feed is formed by custom topics and concepts created by the user and Leo will look for articles that abide those search parameters. Articles from these feeds can then be saved to boards, which can then be pulled by API-calls (Figure 7).

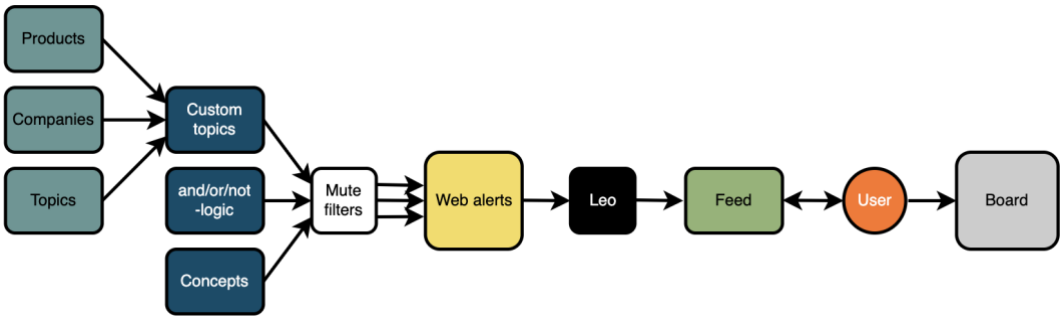

Figure 7. Process of generating articles with Feedly.

2.3 Subscription Levels

Feedly is by default a free service. However, most of the important features requires either a pro, pro+ or enterprise level subscriptions, with varying prices. The pro subscriptions are more geared towards individual users, while the enterprise is meant for companies interested in gathering intelligence (Feedly, n.d, online).

With the free version, the user can create feeds and boards for themselves, but only add websites as sources for the feeds and are not able to refine any search logic to them and they are not able to search for articles with keywords via power search. The free version is limited to 100 feeds and three folders.

The Pro plan is $6 a month, where the number of feeds is raised to 1000. This version features power search, sharing capabilities, noting and highlighting, faster news aggregation, no advertisements and premium support. This plan still does not contain Leo as a service.

The Pro+ plan with $12/month, however, enables using Leo for tuning the feeds. With this the user can create web alerts, follow newsletters, get the RSS-builder and follow certain topics in feeds. Intelligence regarding markets, threats and competitors, however, is reserved for the enterprise plan.

The enterprise subscriptions cost varies and is not listed. The users get up to 7500 feeds, can create team feeds and boards (whereas the cheaper subscriptions are

limited for personal use), has access to Feedly's API, get Slack and Teams integration and gathering intelligence with Leo is enabled. Feedly also offers onboarding and training for companies regarding using Feedly and its API (Figure 8).

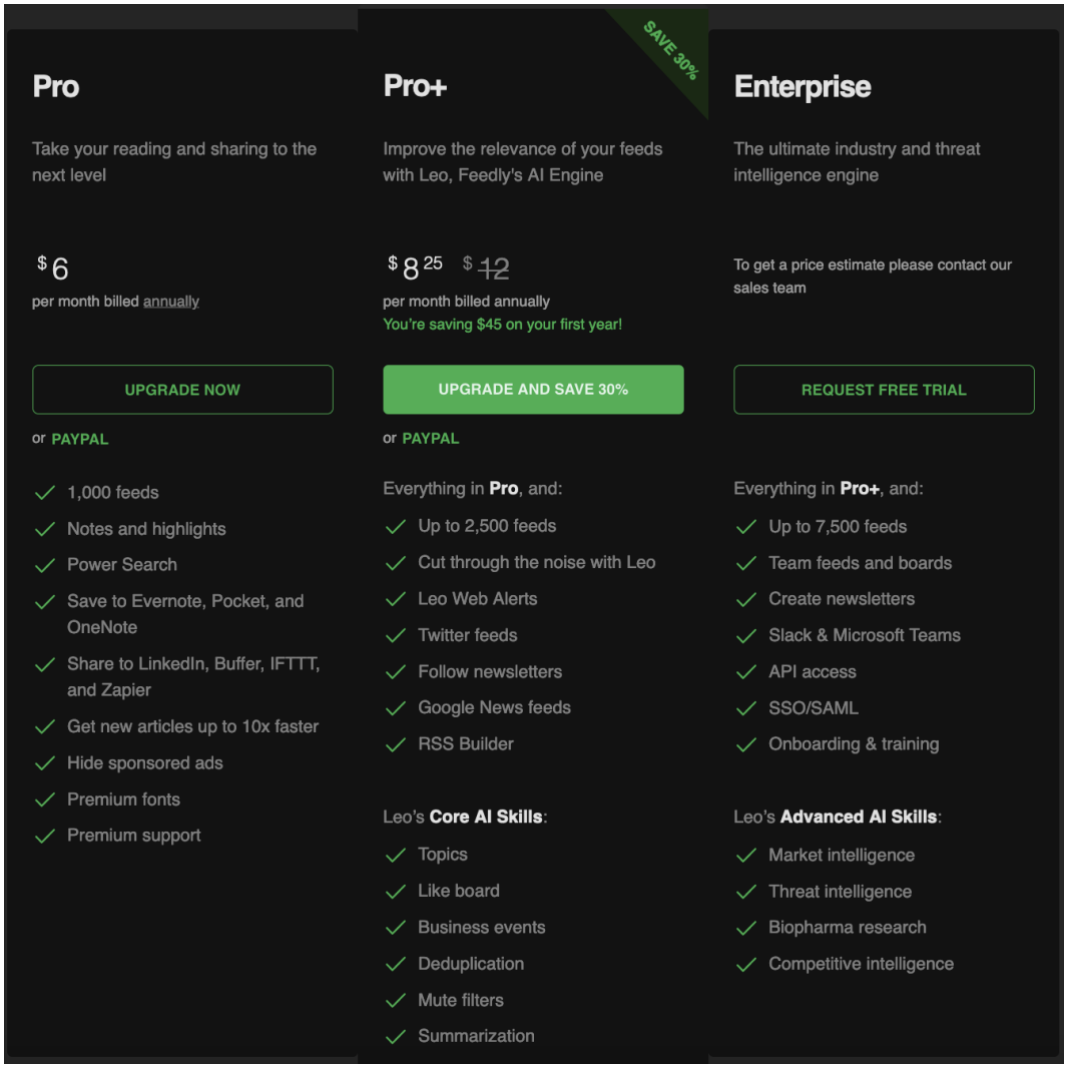

Figure 8. Feedly's different subscription services (Feedly, n.d, online).

2.4 Use Cases

Use cases for Feedly varies greatly between companies and even between teams within companies. Some teams are more interested in marketing news and tracking competitors rather than threat intelligence.

A use case example in Wärtsilä is when the infamous Log4j vulnerability (CVE-2021–44228) was discovered in December 2021. It allowed the attacker to execute code remotely and steal information and was one of the most critical vulnerabilities in history of the internet, as it affected a third of the world's web servers (Catanese, Beppe. Medium. Accessed 07.10.2022). An expert in Wärtsilä, however, saw an article in Feedly the day it was published, regarding the founding of this vulnerability, and the cyber teams were able to patch this vulnerability before threat actors were able to exploit it.

3 DEVELOPMENT

Gathering threat intelligence is only part of the pipeline. The information needs to also be processed before it is forwarded to the user-end of the pipeline. Feedly allows for API calls to fetch articles from web alerts and boards along with other data, such as information from Feedly's own dashboards. First, the articles need to be pulled with an API-call from a link provided by Feedly, then parsed in order to gather only relevant information about them, such as title, summary and publish date. After properly modelling the data and inserting the articles to database tables, they are uploaded to a server, where the tables can then be visualised in a reporting tool or something similar. This process is to be completely automated.

3.1 CI/CD Pipeline Structure

For running backend programs, Wärtsilä has chosen Amazon Web Services (AWS) as its service platform, which is a cloud-based web service platform. It offers a multitude of different services, of which many are needed to successfully run this backend application (Amazon Web Services. Accessed 11.10.2022).

First the whole application is uploaded to Bitbucket, which is a git-based source code repository. Bamboo, an Atlassian tool for setting up continuous integration and deployment (CI/CD), is set up so that any code uploaded to Bitbucket is automatically checked for any errors and faulty formatting. If the build is successful, the code is uploaded to AWS CloudFormation, which is an infrastructure as a code (IaC) and manages third-party resources. From there it's uploaded to S3 which stands for Simple Storage Service. The code is stored there as objects, both the serverless version and the development and production versions. The code is also uploaded to AWS Lambda, which is an event-based computing service, where the code can be set up to be run automatically on certain events or timed intervals. In this project the Lambda is set up to run the application every 24 hours midnight, so the news dashboards at the end of the pipeline are readily refreshed for each morning.

PostgreSQL was chosen as a database solution to store the data. When the code is run in Lambda, it pulls and parses the articles from Feedly and stores them into RDS Postgres. From there the tables can be integrated into any wanted software or program (Figure 9).

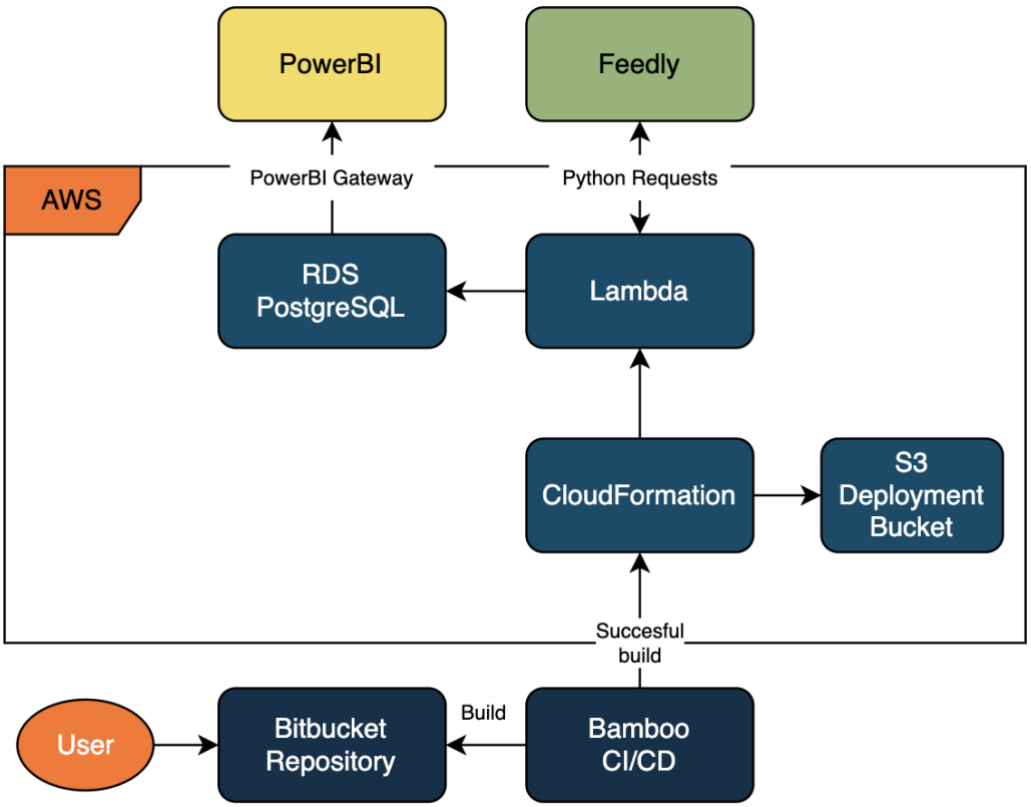

Figure 9. AWS structure.

3.2 Code Structure

The articles from Feedly are pulled with a scheduled running backend API caller application (Figure 10). It is written in Python and utilises Pydantic models. Pydantic modeling is a way of parsing data and ensuring it conforms the defined field type (Pydantic docs, n.d. online).

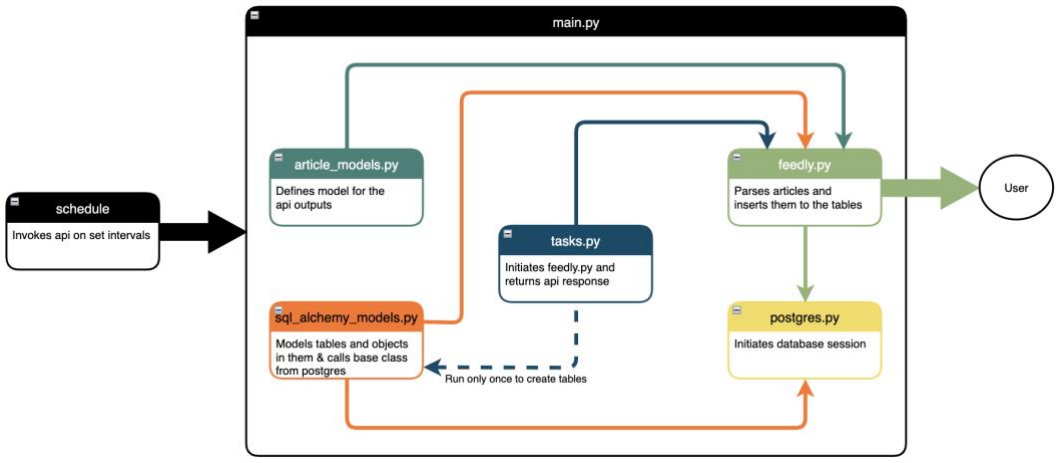

Figure 10. General structure of the code.

First, the articles are pulled from Feedly with an API-call directed to chosen address in feedly.py (Figure 11).

```
11  FEEDLY_BOARD_API_URL = os.environ["FEEDLY_BOARD_API_URL"]
12  FEEDLY_BEARER_TOKEN = ssm_store.get("FEEDLY_BEARER_TOKEN")
13  FEEDLY_BOARD_ID = ssm_store.get("FEEDLY_BOARD_ID")
14
15
16  def get_dashboard_articles(db: Session):
17
18      header = {"Authorization": "Bearer " + FEEDLY_BEARER_TOKEN}
19      response = requests.get(url=FEEDLY_BOARD_API_URL + "?streamId=" + FEEDLY_BOARD_ID, headers=header, verify=False)
```

Figure 11. Establishing a connection to a Feedly board in a get() -function (screenshot from VS Code).

All the get() -functions are set up to run every 24 hours in AWS Lambda, where the interval can be configured. In this project Lambda invokes tasks.py every 24 hours, which initiates all the API-calling functions (Figure 12).

```
@router.get("")
def feedly_poller(event=None, context=None):
    session = None
    try:
        session = SessionLocal()
        get_dashboard_articles(db=session)
        get_trending_vulnerabilities(db=session)
        get_trending_attackers(db=session)
        get_new_malware(db=session)
        get_trending_articles(db=session)
```

Figure 12. Invoking all the API-calls (screenshot from VS Code).

All the get() -functions also call for create() -functions, which inserts parsed data into their corresponding columns, in the correct table and inserts the new data in to the database (Figure 13). A single article contains a lot of unneeded data which can be viewed in JSON-format[6]. So only necessary fields are parsed and inserted into the database, such as the title of the article, date released and summary.

[6] JSON is a human-readable data format that stores data in objects and arrays (JSON. Accessed 14.10.2022).

```
50      db_article = sql_alchemy_models.FeedlyArticles(
51          article_title=new_data.title,
52          article_summary=sentences,
53          article_url=new_data.canonicalUrl,
54          article_banner=banner,
55          article_date=converted_date,
56      )
57      db.add(db_article)
58
59      try:
60          db.commit()
61          db.refresh(db_article)
62          return db_article
63      except Exception:
64          db.rollback()
```

Figure 13. Inserting new data in to the database in a create() -function (Screenshot from VS Code).

When the code is run for the first time, it is set to also run sql_alchemy_models.py, which models the tables within the database (Figure 14).

```
6   class FeedlyArticles(Base):
7       __tablename__ = "Dashboard articles"
8
9       article_id = Column(Integer, primary_key=True)
10      article_title = Column(String, unique=True)
11      article_summary = Column(String)
12      article_url = Column(String)
13      article_banner = Column(String)
14      article_date = Column(String)
```

Figure 14. Modeling a table (screenshot from VS Code).

Functions initialising the creation of the tables can be disabled (commented out) after the first run, as all future data will be stored in already existing tables (unless new ones are to be created).

The articles title acts as a unique identifier for the PostgreSQL database, so a check is run automatically, where if the title already exists, it will not be re-added to the database. This is done because while the ID also acts as a primary key, sometimes the same article can appear as a duplicate, but with a different ID. Since every single character within the title is accounted for, it is highly unlikely that two different articles would have the exact same title and thus would be excluded from the list, so this is a fairly fool-proof method.

After parsing the articles and inserting them into the tables, feedly.py also calls for postgres.py, which then pulls the database session from AWS with proper credentials. As this application is back-end software, the data can not be viewed within the command console but instead must be viewed in a table tool of some sort (Figure 15).

article_id	article_title	article_summary	article_url	article_banner	article_date
761	Malicious file...	PEframe is an o...	https://se...	https://i0.wp.co...	09 Aug. 20...
781	Microsoft Pa...	Microsoft patch...	https://w...	https://eu-imag...	09 Aug. 20...
782	10 Credentia...	The packages "i...	https://th...	https://thehack...	09 Aug. 20...
783	Twilio Suffer...	Customer enga...	https://th...	https://thehack...	09 Aug. 20...
801	How to Prev...	Attackers often...	https://be...	https://miro.me...	11 Aug. 20...
802	APT-C-35: N...	In this post we d...	https://bl...	https://lh4.goo...	11 Aug. 20...
821	Cisco Patch...	The issue, assig...	https://th...	https://thehack...	12 Aug. 20...
881	Thousands o...	Exposing VNCs...	https://w...	none	15 Aug. 20...
882	Russian hac...	Threat analysts...	https://w...	https://www.ble...	15 Aug. 20...
883	Oil and Gas...	EMPTY	https://w...	https://www.tre...	15 Aug. 20...
884	RT @S1r1u5...	EMPTY	NULL	https://blog.ele...	14 Aug. 20...
901	What Will De...	Most of the ICS...	https://da...	EMPTY	09 Aug. 20...

Figure 15. Example of what data is written into the database (screenshot from TablePlus).

3.3 Testing Locally and in Development

For local testing purposes several different software are needed. The code is also set up to simulate the cloud structure locally. The local backend is run from a console with a command, which initiates the serverless-offline configuration. Docker needs to be run, which is a set of platform-as-a-service products, that allows for delivering software in containers with the help of an OS-level virtualisation. The application is then initiated with a local GET-call to defined local address (which in this case is http://localhost:3000/local/tasks) with an API platform (in this project Postman is used). The API-call then initiates the application, and the articles are pulled and parsed into a local database, which can then be viewed with TablePlus for example.

The testing method consists of deleting any already existing tables related to this project, then running the program to see whether it added articles properly and with all the data that was set up to be gathered. When that is confirmed, the program should be run again after adding new articles to the corresponding board

in Feedly to see effects. The program should not add any duplicate articles and it should add any new articles previously not found in the table (Figure 16). The function for creating new tables should be commented out before adding articles to existing tables.

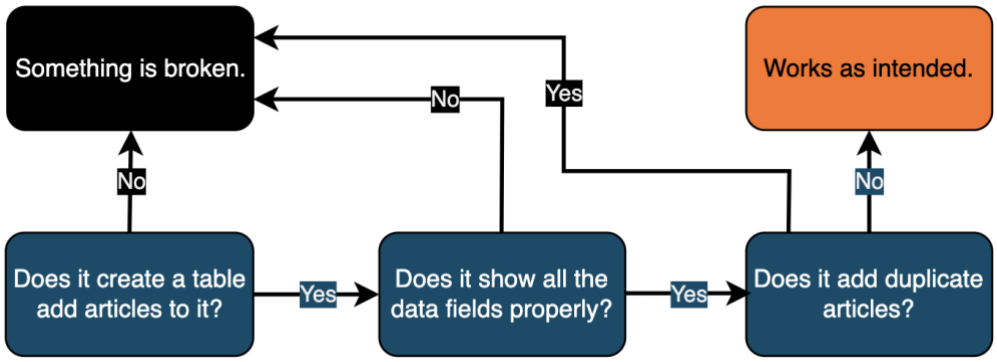

Figure 16. Testing method.

The application has to be tested without errors locally before uploading to development. For testing in development, the application can simply be run from AWS Lambda, where the user can manually test run the selected file or files, which in this case needs to be tasks.py only.

3.4 Pipeline and PowerBI

The main feature that needed to be included in this project was to create a pipeline that allows easy transfer of data to user endpoints. Feedly interactions are to be shown in communication software such as Slack and Microsoft Teams and the parsed articles are to be brought to a reporting tool, Microsoft PowerBI (Figure 17).

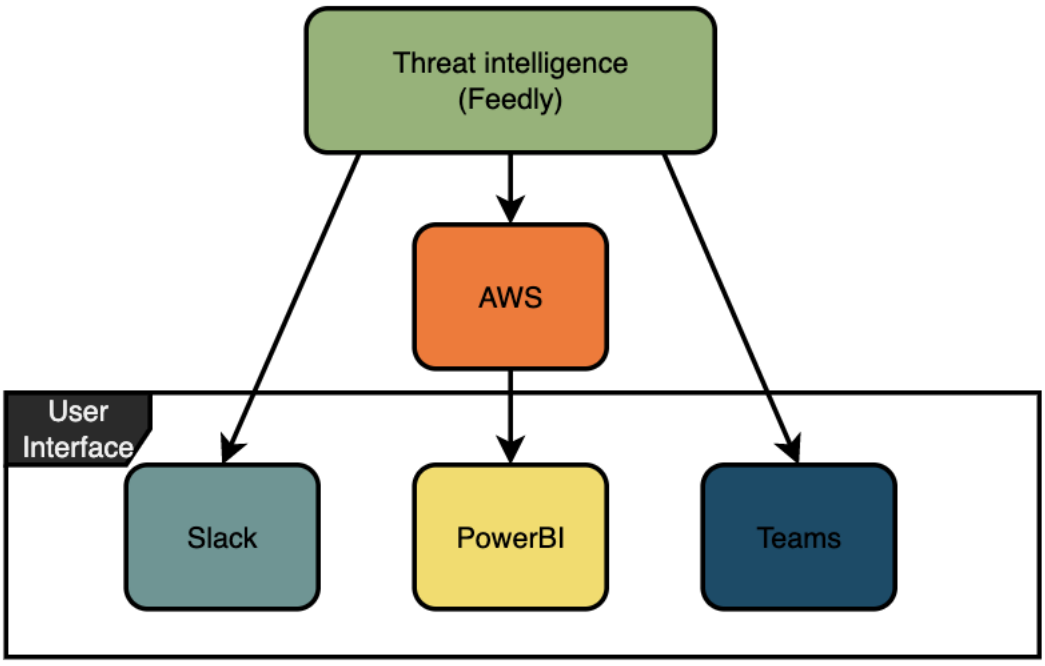

Figure 17. Pipeline structure.

Microsoft PowerBI is an interactive data visualization software used for making visualised reports based on data, either imported from a file or directly from the data source. It allows for faster and more powerful data visualisation and customization than Microsoft Excel for example. PowerBI is favored by the company for it is part of the Microsoft program stack used by the company. It does not require any large-scale integrations or user management. However, a network gateway must be configured by the company, which allows secure data transfer between on-premises data, to be able to get and see data contained within the published report.

The news dashboard was designed so that it shows the banner image, title and publish date. When an article is highlighted by clicking it, it also shows the summary created by Leo and a link to the full article (Figure 18).

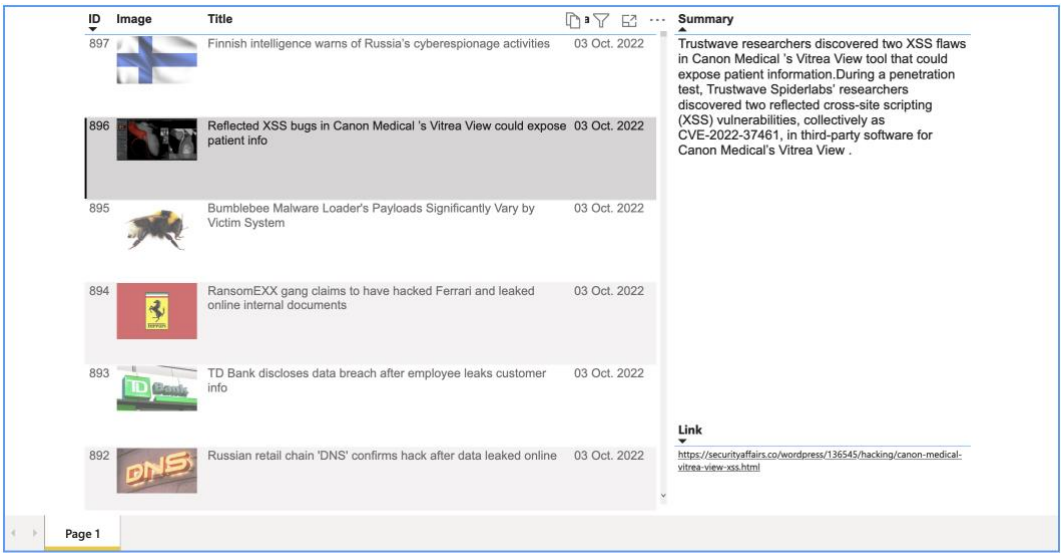

Figure 18. News dashboard created for the articles (screenshot from Power BI).

Feedly also offers Slack and Microsoft Teams integrations. Slack is a messaging platform designed for office and personal use. A channel can be created in it which will then be connected to Feedly with a web link. Whenever a user saves an article to the connected board, it appears to the slack channel too, which helps with quickly sharing information with everyone involved (Figure 19). A similar feature is available for Teams, which is also a business communication platform.

Figure 19. How a saved article appears in the Slack channel (screenshot from Slack).

4 RESULTS

The goal of this thesis was to tune Feedly to gather relevant cyber security information related to Wärtsilä, Wärtsilä's customers and Wärtsilä's suppliers and to develop a pipeline, which will visualise the results in a dashboard, for example, PowerBI.

4.1 Expectations

Expectations and purpose of this project was not to replace the expert working on gathering threat intelligence but to increase the speed and ease of gathering threat intelligence and reduce the workload of having to go through every individual source one at a time. The related articles selected by experts would be easily available for reading for those, who need to see them. With this pipeline, the news about new exploits, for example, could be reached a lot faster with daily or even hourly updates on news, rather than waiting for a weekly newsletter hand-crafted by somebody.

4.2 Result and Challenges

The technical results were good regarding the objectives of the thesis. The pipeline is working and is easily customizable for future parsing from different sources. The Feedly board used for this thesis was tuned successfully so that it provides appropriate articles. This project is truly running in production, so it is not a mere proof of concept, but rather a working service that the company can utilize.

The biggest challenge, however, is in getting people to use any new services and software. In a company as big as Wärtsilä, there are plenty of tools already in use and it is always difficult to get workers to change their routines. While Feedly would greatly increase the ease and speed of gathering threat intelligence as it can bring all sources to one place, some people still prefer going through the sources manually. Others may want to have the process fully automated without any sort

of interaction on Feedly, but the AI needs to be trained first by going through the articles it finds.

Throughout this project, meetings with various different teams and their member have been held to show the capability and potential of this project. Companywide presentations for cyber teams were also held once the pipeline was ready, to raise awareness and this sparked up more interest into the project. Collaboration has also been done with other companies, such as Feedly itself and AEMO, Australian Energy Market Operator.

On a technical note, the most challenging part has not been learning Python as a new language for me, nor eventually having my code setup to run properly on company systems, but rather in getting PowerBI to work. A lot of extra work had to be done to be able to run PowerBI virtually, since my working laptop has been a MacOS and to get the live data flowing in properly to the virtual machine, a lot of networking and permission sets had to be configured.

4.3 Improvement and Future

The development and onboarding still continues and improvement on what data can be gathered and how it is presented is actively being worked on both from Feedly's side and from my projects point of view. The onboarding is an ongoing process within the company, with some teams showing more interest than others. Not every team needs to be following news to be able to operate and improve, but those who do should be introduced to this concept.

The main services used in this project were Feedly, AWS and PowerBI, but the concept of this pipeline can be utilized in any other environment as well. The articles can be pulled from any chosen RSS-feed, the application does not need to necessarily run on AWS and the user end can be some other tool than PowerBI. PowerBI is not the actual preferable tool to work on such a project, as other more fitting tools exist for creating dashboards with better visuals.

REFERENCES

Amazon. AWS services. Accessed 04.10.2022. https://aws.amazon.com/.

Baker, Kurt. Crowdstrike. What is threat intelligence. Accessed 22.08.2022. https://www.crowdstrike.com/cybersecurity-101/threat-intelligence/.

Brian Barrett. Wired. It's Time for an RSS Revival. Accessed 15.09.2022. https://www.wired.com/story/rss-readers-feedly-inoreader-old-reader/.

Catanese, Beppe. Medium. The Log4j Incident Explained. Accessed 07.10.2022. https://medium.com/geekculture/the-log4j-incident-explained-ed0ce6d36df2.

Correa, Andre. Malwarepatrol. Three types of cyber threat intelligence. Accessed 25.08.2022. https://www.malwarepatrol.net/three-types-of-cyber-threat-intelligence/.

Feedly. More signal, less noise. Accessed 07.10.2022. https://feedly.com/i/pro/landing-pro.

Hashim, Abeerah. PrivacySavvy. 6 times when hackers forced companies to go bankrupt and shut down. Accessed 22.08.2022. https://privacysavvy.com/security/business/6-times-hackers-forced-companies-to-go-bankrupt-shut-down/.

JSON. Introducing JSON. Accessed 14.10.2022. https://www.json.org/json-en.html.

Jokela, Petri. Wärtsilä Sharepoint. Security Operations in Wärtsilä. Accessed 29.07.2022.

Petterson, Dan. Wärtsilä Sharepoint. Welcome to Wärtsilä Corporate Security. Accessed 29.07.2022.

Pydantic docs. Accessed 03.10.2022. https://pydantic-docs.helpmanual.io/.

SOC Radar. What is Operational Cyber Threat Intelligence. Accessed 09.09.2022. https://socradar.io/what-is-operational-cyber-threat-intelligence/.

SOC Radar. What is Tactical Cyber Threat Intelligence and How to Use it. Accessed 09.09.2022. https://socradar.io/what-is-tactical-cyber-threat-intelligence-and-how-to-use-it/.

Suerz, Erika. Wärtsilä Sharepoint. Welcome to Cyber Security. Accessed 29.07.2022.

The Recorded Future Team. How Strategic Threat Intelligence Informs Better Security Decisions. Accessed 25.08.2022. https://www.recordedfuture.com/strategic-threat-intelligence.

Wärtsilä. This is Wärtsilä. Accessed 13.10.2022. https://www.wartsila.com/about.