Panacea

People-centric cybersecurity in healthcare

Project Title	Protection and privAcy of hospital and health iNfrastructures with smArt Cyber sEcurity and cyber threat toolkit for dAta and people
Project Acronym	PANACEA
Project Number	826293
Type of instrument	Research and Innovation Action
Topic	SU-TDS-02-2018
Starting date of Project	01/01/2019
Duration of the project	36
Website	www.panacearesearch.eu

D2.2 Human Factors, Threat Models Analysis and Risk Quantification

Work Package	WP2 Research on advanced threat modelling, human factors, resilient response and secure interconnectivity
Lead authors	Dr Dawn Branley-Bell, Prof. Lynne Coventry & Dr Elizabeth Sillence (UNAN)
Contributors	Silvia Bonomi, Claudio Ciccotelli, Simone Lenti, Alessia Palleschi, Leonardo Querzoni, Giuseppe Santucci, Mara Sorella & Florin Tanasache (UROME)
Peer reviewers	Davide Alì (AON), Emmanouil Spanakis (FORTH), Anthony Rabbitt (RINA)
Version	V0.20
Due Date	17/01/2020
Submission Date	17/01/2020

Dissemination Level:

	PU: Public
	CO: Confidential, only for members of the consortium (including the Commission)
	EU-RES. Classified Information: RESTREINT UE (Commission Decision 2005/444/EC)
	EU-CON. Classified Information: CONFIDENTIEL UE (Commission Decision 2005/444/EC)
	EU-SEC. Classified Information: SECRET UE (Commission Decision 2005/444/EC)

The work described in this document has been conducted within the PANACEA project. This project has received funding by the European Union's Horizon 2020 research and innovation programme under grant agreement No. 826293.

Version History

Revision	Date	Editor	Comments
0.1	08/10/19	Dawn Branley-Bell (UNAN)	Added ToC & initial document structure
0.2	14/10/19	Dawn Branley-Bell (UNAN)	Added written introductions to the different sections. Added methodology and results for the workshops.
0.3	8/11/19	Dawn Branley-Bell (UNAN)	Revised ToC
0.4	22/11/19	Dawn Branley-Bell (UNAN)	Added content
0.5	02/12/19	Dawn Branley-Bell (UNAN)	Added content
0.6	05/12/19	Dawn Branley-Bell (UNAN)	Added further content around document structure, objectives and specific nudges
	10/12/19	Silvia Bonomi (UROME)	Revised structure of sections 5 and 6, added written introduction and methodology
0.6	11/12/19	Dawn Branley-Bell, Lynne Coventry & Elizabeth Sillence (UNAN) & Anthony Rabitt (RINA)	Reviewed document & provided feedback
0.7	12-16/12/19	Dawn Branley-Bell (UNAN)	Added further content & revised according to feedback from UNAN & RINA
0.8	17/12/19	UROME	Added Use case scenario description, Results of Section 5, added content
0.9	20/12/19	Silvia Bonomi (UROME)	Revised structure of section 6, added content
0.10	17/12/19	Dawn Branley-Bell, Lynne Coventry & Elizabeth Sillence (UNAN)	Reviewed document progress
0.11	20/12/19	Dawn Branley-Bell (UNAN)	Updated nudges section
	20/12/19	Silvia Bonomi (UROME)	Revised structure of section 6, added content
	23/12/19	Silvia Bonomi (UROME)	Added content to Section 6, Business Impact Modelling and Attack based risk quantification
	27/12/19	UROME	Added content to section 6, Threat Agent modelling, content revision
	29/12/2019	UROME	Added content to Section 5 on Human Modelling and added Annex A.
0.12	06/01/20	Dawn Branley-Bell (UNAN)	Updated introduction, methods and discussion sections
	08/01/20	Dawn Branley-Bell, Lynne Coventry & Elizabeth Sillence (UNAN)	Reviewed & updated document
0.13-0.15	09/01/20	Dawn Branley-Bell (UNAN)	Merged document and amended formatting. Edited content throughout for consistency.
0.16	09/01/20	Lynne Coventry (UNAN)	Reviewed & edited document

0.17	09/01/20	Silvia Bonomi (UROME)	Updated Sections 5 & 6
0.16-0.18	10/01/20	UNAN	Merged all sections, amended language, improved flow
0.19	10/01/20	Dawn Branley-Bell & Lynne Coventry (UNAN)	Reviewed document
0.20-0.22	10/01/20	Dawn Branley-Bell (UNAN)	Restructured document & reviewed
0.23	12/01/20	Silvia Bonomi (UROME)	Added content and revisions required for internal review
0.24	13/01/20	Dawn Branley-Bell (UNAN)	Prepared document for internal review
0.25	14/01/20	Anthony Rabbitt (RINA) & Emmanouil Spanakis (FORTH)	Reviewed document & sent feedback
0.26	15/01/20	Dawn Branley-Bell (UNAN)	Edited document in line with feedback from internal review
0.27	17/01/20	Dawn Branley-Bell (UNAN)	Final formatting for submission

List of Contributors

The list of contributors to this deliverable are as follows:

- Sections 1-4, 7 & 8: UNAN
- Sections 5, 6 & 8: UROME
- Document internal review: FORTH, AON & RINA

Keywords

HUMAN FACTORS; BEHAVIOUR CHANGE; MODELLING; RISK PREDICTION

Disclaimer

Executive Summary

The purpose of this document is to identify and refine the list of human factors contributing to insecure behaviours, create a multilayer threat model that takes into account human and network factors contributing to the cybersecurity posture of an organisation, and identify potential behavioural nudges and associated methodology for the Secure Behaviour Nudging Tool (SBNT). The document consists of 4 core sections (Sections 4-7):

Firstly, Section 4 focuses upon the human behavioural aspects of the PANACEA project. This section begins by summarising the main findings from the first phase of workshops conducted as part of D1.4, and then details the rationale and methodology applied for the second phase of workshops conducted as part of this deliverable. This includes describing the theoretical models underpinning the methodology used to identify

problematic cybersecurity behaviours occurring within the healthcare (HC) environments. The results in this section:

a. Detail the type of insecure behaviours identified across the HC sites
b. Describe staff attitudes and motivations underlying these behaviours
c. Feed into the subsequent risk modelling (Sections 5 & 6) and development of behavioural nudges (Section 7).

Secondly, Section 5 focuses upon the threat model analysis and introduces the methodology used to expand upon the identified insecure behaviours to model human factors of HC cybersecurity. Section 6 discusses quantification of risk by means of the attack graph model presented in the previous section and a model of the attacker capabilities. Furthermore, we present an instantiation of the model in a user scenario.

Lastly, Section 7 describes the identification of potential behavioural nudges to be developed further in WP5.

Table of Contents

List of figures

List of tables

1. Introduction

1.1 Purpose

The purpose of this document is to detail the following tasks:

1. Identification of the extent of problematic (i.e., insecure) behaviours within the HC sites and underlying staff attitudes and motivations.
2. Modelling of human and network factors contributing to cyber-risk.
3. Identification, development and evaluation of behavioural nudges to help staff act more securely in the workplace; and development of a Secure Behavioural Nudging Tool (SBNT) to enable end-users to repeat this process as required.

Figure 1 illustrates how these sections are combined within this deliverables to achieve the final outputs for D2.2, and also how these to feed into subsequent deliverables.

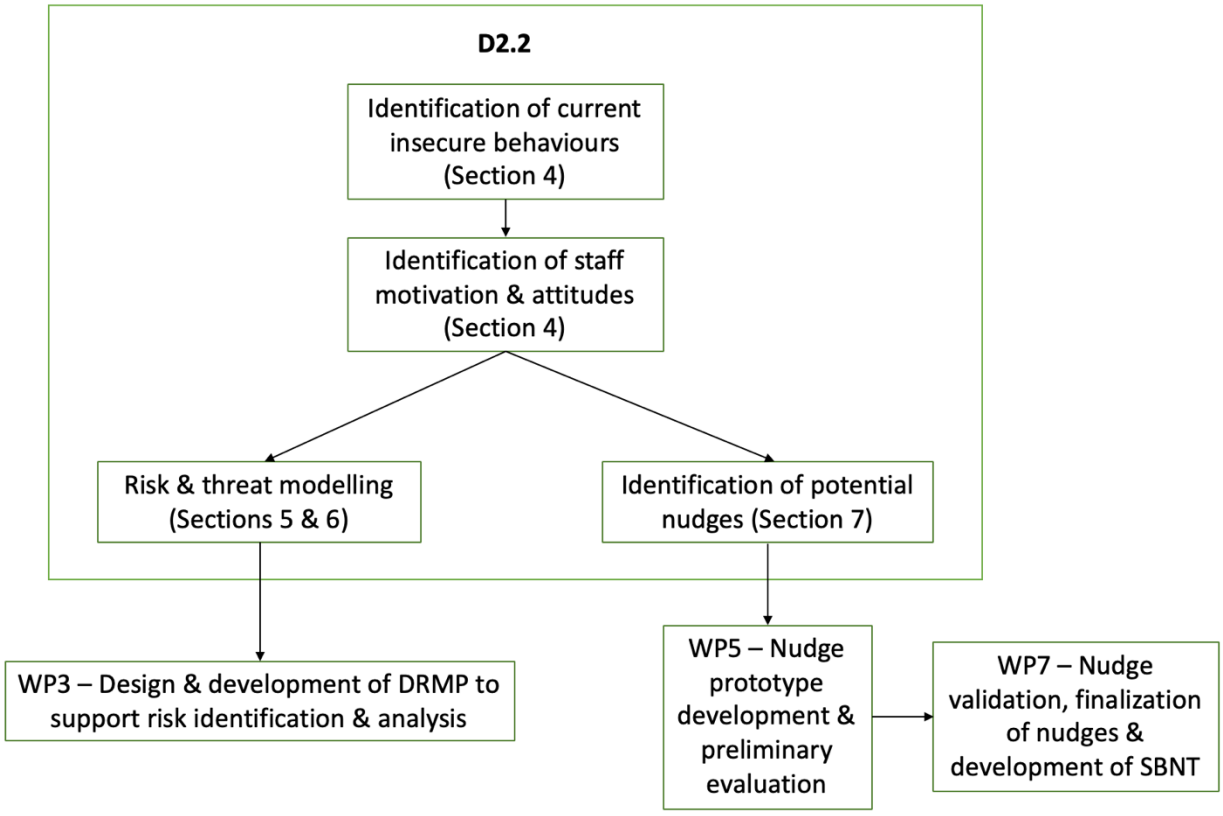


Figure 1. Map of D2.2 and associated WPs

The results from Section 4 detail the type of insecure behaviours identified across the HC sites and describe staff attitudes and motivations underlying these behaviours. This information subsequently feeds into the risk modelling (by identifying the human factors to include within the models) and nudge development (through identification of the types of behaviour and/or attitudes that could be addressed by nudging).

The threat model analysis in Section 5 and risk quantification in Section 6 will provide the modelling framework that will be used in WP3 to design and develop sub-components of the PANACEA Dynamic Risk Management Platform (DRMP); which will support risk identification and analysis.

The nudges identified in Section 7 feed into WP5 where they will be developed further into nudge prototypes and evaluated. They will then be validated further in WP7 to finalise the final nudge outputs (and SBNT) for the project.

In summary, the outputs of this deliverable are:

1. Identification of the insecure behaviours occurring across the HC sites and underlying staff attitudes and motivations.
2. Risk quantification and threat analysis models that incorporate human factors.
3. Identification of selected nudges that will be developed as part of the SBNT during WP5 and WP7.

1.2 Quality assurance

1.2.1 Quality criteria

The QA in the PANACEA project relies on the assessment of a work product (i.e. deliverable) according to lists of QA checks (QA checklists – [QAPeer]) established with the QAM, validated at a project management level and centralised in the [PMP].

For the purpose of the QA of this deliverable, it has been assessed according the following checklists:

- PEER REVIEW (PR) QA CHECKLIST [QAPeer]: this deliverable is a report, it then requires a proper peer review according to the checks defined in this checklist;

1.2.2 Validation process

For the final validation of deliverables within the PANACEA project, a final QA review process MUST be used before the issuing the final version. This QA validation process follows the Quality Review Procedure established with the QAM and validated at project management level in order to guarantee the high-quality level of work products and to validate its adequacy according to the defined quality criteria chosen and defined for each deliverable. The Quality Review Procedure itself and the selection of the QA Review Committee are described in the [PMP]. The QA validation process is scheduled in the QA Schedule [QASchedule] managed by the QAM.

1.3 Structure of the document

The structure of the document is as follows:

SECTION 1. Introduction
SECTION 2. Applicable and Reference Documents
SECTION 3. Glossary of Acronyms
SECTION 4. Human Factors of Cybersecurity
SECTION 5. Threat Models Analysis
SECTION 6. Risk Quantification
SECTION 7. Behavioural Nudges
SECTION 8. Overall Discussion, Conclusions and Next Steps

2. Applicable and Reference Documents

2.1 Applicable Documents (ADs)

The following documents contain requirements applicable to the generation of this document:

Reference	Document Title	Document Reference	Version	Date
[PMP]	PANACEA Project Management Plan		0.5	01/01/2019
[QAPeer]	PANACEA Peer Review QA Checklist		0.5	01/01/2019
[QAReqs]	PANACEA Requirements Review QA Checklist		0.5	01/01/2019
[QASchedule]	PANACEA QA Schedule		0.5	01/01/2019

Table 1. Applicable Documents

2.2 Reference Documents (RDs)

The following documents have been consulted for the generation of this document:

Reference	Document Title	Document Reference	Version	Date
[Fishbein00]	The Role of Theory in HIV Prevention	https://doi.org/10.1080/09540120050042918		2000
[Fishbein03]	Using Theory to Design Effective Health Behaviour Interventions	https://doi.org/10.1111/j.1468-2885.2003.tb00287.x		2003
[Dolan10]	MINDSPACE: Influencing Behaviour through Public Policy	https://www.instituteforgovernment.org.uk/sites/default/files/publications/MINDSPACE-Practical-guide-final-Web_1.pdf		2010
[Gollwitzer99]	Implementation Intentions: Strong Effects of Simple Plans	https://doi.org/10.1037/0003-066X.54.7.493		1999
[Criado12]	A mathematical model for networks with structures in the mesoscale	https://doi.org/10.1080/00207160.2011.577212		2012
[Kivela14]	Multilayer networks	https://doi.org/10.1093/comnet/cnu016		2014
[Granadillo18]	Dynamic risk management response system to handle cyber threats	https://doi.org/10.1016/j.future.2017.05.043		2018
[Kanoun12]	Towards Dynamic Risk Management: Success Likelihood of Ongoing Attacks	https://doi.org/10.1002/bltj.21558		2012
[D2.1]	Analysis of cyber vulnerabilities and SOA countermeasures in HCC			2019
[Bou-Harb14]	Cyber Scanning: A Comprehensive Survey	https://doi.org/10.1109/SURV.2013.102913.00020		2014
[Sametinger15]	Security Challenges for Medical Devices	https://doi.org/10.1145/2667218		2015
[Basile17]	Assessing network authorisation policies via reachability analysis	https://doi.org/10.1016/j.compeleceng.2017.02.019		2017
[CVSS]	Common Vulnerability Scoring System v3.0: Specification Document	https://www.first.org/cvss/v3.0/specification-document		2019
[CVE]	CVE Website	https://cve.mitre.org/		2019

Reference	Document Title	Document Reference	Version	Date
[NIST-SP-800-53]	NIST SP 800-53 Rev. 5	https://csrc.nist.gov/CSRC/media//Publications/sp/800-53/rev-5/draft/documents/sp800-53r5-draft.pdf		2017
[NVD]	National Vulnerability Database	https://nvd.nist.gov/vuln-metrics/cvss		2018
[OWASP]	OWASP Risk Rating Methodology	https://www.owasp.org/index.php/OWASP_Risk_Rating_Methodology		2017
[CISCO]	CISCO Hierarchical Network Design	http://www.ciscopress.com/articles/article.asp?p=2202410&seqNum=4		2014

Table 2. Reference Documents

3. Glossary of Acronyms

Acronym	Description
CIA	Confidentiality, Integrity and Availability
CyPR	Cybersecurity Professional Register
CVE	Common Vulnerabilities and Exposure (CVE) vulnerability classification
CVSS	Common Vulnerability Scoring System
DRMP	Dynamic Risk Management Platform
FSP	Full-Scale Pilot
GA	Grant Agreement
HC	Healthcare
HCO	Health Care Organisation
HRG	
IBM	Integrated Behaviour Model
ICT	Information and Communication Technologies
LAN	Local Area Network
NIST	National Institute of Standards and Technologies
NVD	National Vulnerability Database
OWASP	Open Web Application Security Project
POCT	Point Of Care Terminal
QA	Quality Assurance
SEB	Stakeholders Expert Board
SME	Small- and Medium-sized Enterprises
WP	Work Package

Table 3. List of acronyms

4. Human Factors of Cybersecurity

This section focuses upon the human behavioural vulnerabilities identified during the PANACEA project and feeds into the rest of the project in three key ways:

1. Identification of the extent of problematic, i.e., insecure, behaviours within the HC sites. This information feeds into the risk modelling and nudge development.
2. Identification and explanation of staff attitudes and motivation(s) underlying insecure behaviour. This feeds into the modelling, nudge development and informs the focus of project work more widely.
3. Identification, development and evaluation of behavioural nudges to help staff act more securely in the workplace. This helps to assist the other tools developed throughout the project by acting as a form of risk reduction.

This section addresses our findings in relation to the elements that feed into the modelling (above - points 1 & 2) by summarising the problematic behaviour identified across the sites and the attitudes and motivations driving this behaviour. The final element (point 3 above), the development of the behavioural nudges, is explained in Section 7 and proposed as a method for reducing the human vulnerabilities identified.

4.1 Human Factors Workshop Methodology

As part of the PANACEA project, the first phase of focus groups (reported in D1.4) identified eight insecure behaviours (human vulnerabilities) occurring across the three HC sites in Rome, Cork and Heraklion, within the PANACEA project. These behaviours were: Insecure computer and account behaviour (including unlocked workstations, weak passwords and sharing login credentials); Insecure e-mail use (including use of attachments to share and/or receive patient information); Use of USB devices; Use of own devices; Remote access and home working; Lack of backups, updates and encryption; Use of connected devices; and poor physical security. These behaviours are explored in more detail in this deliverable.

In order to further investigate the behaviours identified in the behavioural scenarios in D1.4 [‘Relevant user scenarios, use cases and KPIs for Panacea Toolkit validation’, Section 5.3.2] and identify facilitators of these behaviours - and conversely, barriers to secure behaviour - we conducted three more in-depth workshops across the HC sites (Rome, Cork and Heraklion). To allow us to focus upon the most prevalent and/or concerning behaviours, we asked each HC site to choose one or two of the identified behaviours based upon those that they regarded as particularly relevant to their organisation at this point in time. During this second phase of workshops, the priority behaviours chosen by the three HC sites were as follows. In order to maintain confidentiality of the HC sites, we do not identify which behaviours were chosen by each site. However, it was noted that unlocked workstations and insecure password behaviours were common across all three sites:

1. Unlocked workstations
2. Insecure password behaviours (i.e., weak passwords, writing passwords down and sharing login credentials)
3. Use of USB devices
4. Insecure sharing of patient information (i.e., by e-mail or other unofficial means such as smartphone messaging apps)

We conducted workshop sessions at each HC site. Each workshop took place as a 3-hour session with approximately 15-25 staff members taking part; with the exception of one occasion when due to logistic reasons and staff availability, the session was split into three 1-hour sessions, with approximately 5-10 staff

taking part in each session. The workshops included a range of roles and levels including doctors, nurses, administration staff, IT staff, residents/students and other HC professionals.

The aims of the workshops were twofold:

1. To identify staff members attitudes towards the identified unsecure behaviour, and their underlying motivations (e.g., what are the goals they wish to achieve from the behaviour). This includes identifying how staff behaviour differs from desirable/secure behaviour, and the barriers that may be preventing staff from changing this behaviour. This information feeds into the modelling (Sections 5 & 6) and the development of nudges (Section 7).
2. To ask staff to start to formulate ideas which they feel could help them overcome the barriers to changing the unsecure behaviour. This would supplement the expert analysis and identification of potential nudges to encourage more secure behaviour. The results are covered in Section 7.

To help provide staff with the tools to achieve these aims, two main theoretical models were introduced to the staff taking part in the workshops. Note: This methodology will also be incorporated in the final Secure Behaviour Nudging Tool (SBNT) which will be provided to end-users at the end of the project. The SBNT is designed to provide the tools required for end-users to conduct their own nudge evaluation and design. This is covered further in Section 7.

1. **The Integrated Behaviour Model (IBM [Fishbein00; Fishbein03]) - to assist understanding of current behaviour**.

This model provides a framework for explaining human behaviour and it was introduced to help prompt thoughts and discussion around the types of factors which may be influencing staff behaviour in the workplace. The model can be seen in Figure 2.

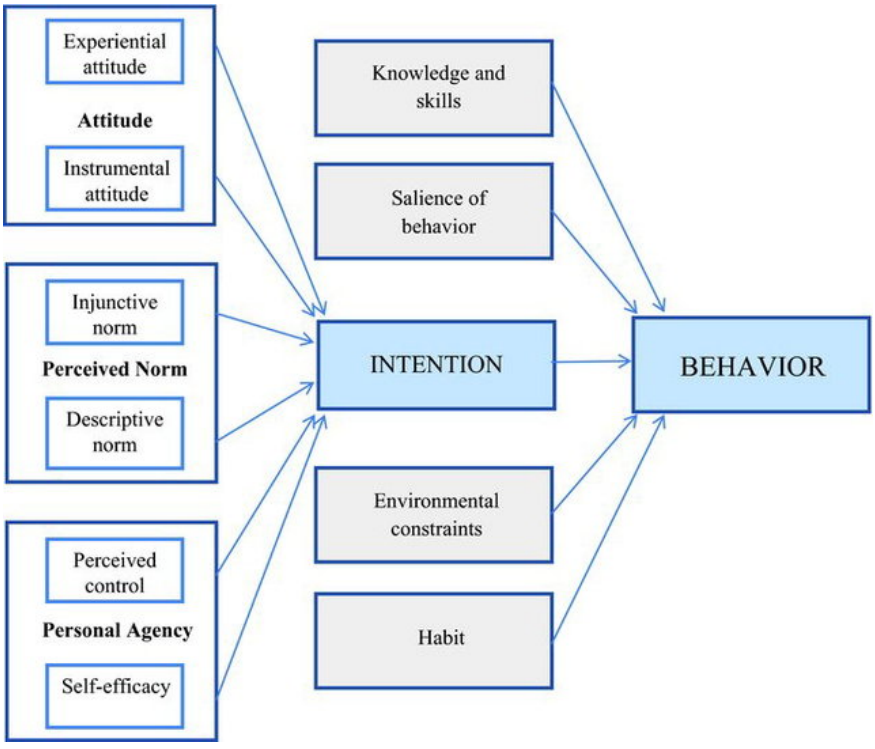


Figure 2. The Integrated Behaviour Model (IBM)

Staff were provided with a crib sheet of explanations and examples of each factor in the model (for a copy of the full crib sheet refer to Annex A).

The factors from the model were used during the workshop discussions to provide a starting point for staff to begin to identify how these factors may be influencing their behaviour at work. The IBM was also used by the expert panel when designing potential interventions to encourage behaviour change (Section 7).

2. **MINDSPACE [Dolan10] - to assist formulation of ideas to promote secure behaviour.**

The MINDSPACE framework provides a useful method to aid identification of factors which could influence behaviour change. The model is shown in Figure 3. Participants were provided with a crib sheet explaining each factor in the model (refer to Annex A).

Figure 3. MINDSPACE Framework

4.2 Workshop Results (Part 1): Attitudes and Motivations for Insecure Behaviour

For the four chosen behaviours (unlocked workstations, insecure password behaviours, use of USB devices and insecure sharing of patient information), staff identified many attitudes and motivations underlying insecure behaviour. The results for each behaviour are detailed in the following section.

4.2.1 Current Behaviour 1: Unlocked workstations

IBM Factor	Barriers
Attitude(s)	**Security as a barrier and/or burden** ⇒ Acting securely does not allow staff to work quickly & efficiently / time consuming / burdensome (makes life difficult). ⇒ Boredom (repetitiveness?) ⇒ Extended hours of work (overworked) ⇒ Saves remembering password. Entering wrong password, getting locked out, requiring resetting of password etc. – all increases time and could have a negative impact on patient care (also lead to frustration?) ⇒ Need for system to be available 24/7. Login gets in the way of this. ⇒ Not the priority – patient safety is the priority
Perceived Norms	**No culture around cybersecurity** ⇒ Common practice to leave workstations logged in/open ⇒ Trust that there is no risk as colleagues will not use for negative reasons ⇒ Managers do not care about how securely someone works, but whether they work quickly and efficiently ⇒ Some senior staff ignore the rules and set the norms
Personal Agency	**Perception that technology/IT should/does protect again risks.** Staff feel that cybersecurity is not their responsibility and/or under their control. Staff feel safe as data is controlled by the central service
Knowledge & Skills	**Risk perceptions & awareness** ⇒ Not seen as a big deal. It does not impose a perceived risk ⇒ Nothing bad has happened from this behaviour so far ⇒ Staff don't always recognise the need for passwords, they do not realise it is protecting the system in any way
Salience	There is nothing within the working environment to encourage salience of risk or the need for cybersecurity
Environmental Constraints	Staff in healthcare are overworked, fatigued, patient-focused and working in a unique environment where delays could be detrimental to patient health and wellbeing. Cybersecurity practices are perceived as in conflict with these priorities.
Habit and/or Other	Lack of enforcement - Currently no sanctions, incentives or enforcement.

Table 4. Identified factors contributing to unlocked workstations, using the IBM approach

4.2.2 Current Behaviour 2: Insecure password behaviours

Insecure password behaviours can be further categorised into 2 key areas:

a) Writing passwords down and/or choosing weak passwords
b) Sharing passwords with colleagues

Both of these behaviours are included within this section. Firstly, Table 5 looks at reasons why staff may write down password and/or choose weak passwords.

IBM Factor	Barriers
Attitude(s)	**Security as a barrier and/or burden** ⇒ Cannot remember all passwords for every system. Daily workload means staff pick short, easy to input passwords ⇒ Asked to change passwords too frequently – leads to having insecure methods for changing the password with minimal effort (e.g., Jan 1, Feb 2, March 3) ⇒ Not the priority – patient safety is the priority
Perceived Norms	**No culture around cybersecurity** ⇒ Common practice to have insecure passwords and/or write passwords down ⇒ Trust that there is no risk as only colleagues will see the passwords ⇒ Managers do not care about how securely someone works, but whether they work quickly and efficiently ⇒ Some senior staff ignore the rules and set the norms for sharing passwords
Personal Agency	**Perception that technology/IT should/does protect again risks.** Staff feel that cybersecurity is not their responsibility and/or under their control. Staff feel that they are safe and IT dept. protects them from any attacks or phishing.
Knowledge & Skills	**Risk perceptions & awareness** ⇒ Not seen as a big deal. It does not impose a significant perceived risk ⇒ Nothing bad has happened from this behaviour so far ⇒ Not informed if password is weak or strong – no feedback or salience of strength ⇒ Staff don't always recognise the need for passwords, they do not realise it is protecting the system in any way
Salience	There is nothing within the working environment to encourage salience of risk or the need for cybersecurity. Example, Staff do not always receive the reminder messages (e.g., to change passwords); and they are not informed if password is weak or strong – no feedback or salience of strength.
Environmental Constraints	Staff in healthcare are overworked, fatigued, patient-focused and working in a unique environment where delays could be detrimental to patient health and wellbeing. Cybersecurity practices can conflict with these factors.
Habit and/or Other	Lack of enforcement - Currently no sanctions, incentives or enforcement.

Table 5. Identified factors contributing to insecure password behaviours, using the IBM approach

Secondly, Table 6 details why HC staff share passwords with colleagues at work.

IBM Factor	Barriers
Attitude(s)	**Security as a barrier and/or burden** ⇒ Acting securely does not allow staff to work quickly & efficiently / time consuming / burdensome (makes life difficult). ⇒ Staff "care about privacy but share passwords for convenience" ⇒ Extended hours of work (overworked) ⇒ Not the priority – patient safety is the priority ⇒ Junior staff worry about future employment references from senior staff should they refuse to comply with the behavioural expectations held by the senior staff member (i.e. to share passwords) ⇒ It can take up to 2 weeks for new staff to be issued with their own login credentials
Perceived Norms	**No culture around cybersecurity** ⇒ Common practice to share passwords ⇒ Trust that there is no risk as colleagues will not use passwords for negative reasons ⇒ Managers do not care about how securely someone works, but whether they work quickly and efficiently ⇒ Some senior staff ignore the rules and set the norms
Personal Agency	**Perception that technology/IT should/does protect again risks.** Staff feel that cybersecurity is not their responsibility and/or under their control.
Knowledge & Skills	**Risk perceptions & awareness** ⇒ Password sharing is not seen as a big deal. It does not impose a perceived risk ⇒ No password policy ⇒ No training on how to protect the system ⇒ Nothing bad has happened from this behaviour so far ⇒ Staff don't always recognise the need for passwords, they do not realise it is protecting the system in any way
Salience	There is nothing within the working environment to encourage salience of risk or the need for cybersecurity.
Environmental Constraints	Staff in healthcare are overworked, fatigued, patient-focused and working in a unique environment where delays could be detrimental to patient health and wellbeing. Cybersecurity practices can conflict with these factors. Also: ⇒ Sometimes colleagues need help, and it is necessary to use their credentials to access their systems ⇒ Staff can feel more secure/safer when other colleagues have access to their system – so that they can help immediately when needed. ⇒ Staff can be expected to do tasks that are not their responsibility, which necessitates the need for sharing passwords due to differences in staff access rights
Habit and/or Other	Lack of enforcement - Currently no sanctions, incentives or enforcement.

Table 6. Identified factors contributing to password sharing, using the IBM approach

4.2.3 Current Behaviour 3: Use of USB devices

IBM Factor	Barriers
Attitude(s)	**Security as a barrier and/or burden** ⇒ Facilitates the job – more storage capacity, good for backup, IT too slow to get back to you, good for transporting data ⇒ Convenience & ease of access ⇒ Patient care staff priority, not commitment to cybersecurity ⇒ Staff resentment of security, e.g., "being forced to do something" they do not understand or agree with
Perceived Norms	**No culture around cybersecurity** ⇒ "It's the norm, it's habitual to use USBs" ⇒ Part of the culture at work – everyone uses them ⇒ Managers are doing it (top down norms)
Personal Agency	**Perception that technology/IT should/does protect again risks.** Staff feel that cybersecurity is not their responsibility and/or under their control. **Desensitisation to cyber risk.** Staff may think they're in control or desensitised to the use of technology as they use it every day.
Knowledge & Skills	**Risk perceptions & awareness** ⇒ No perception of risk / lack of awareness ⇒ "Just don't think" e.g., plugging in mobiles to USB ports to charge them ⇒ Lack of training ⇒ Hasn't caused any harm so far ⇒ Can prevent sharing sensitive information via the internet/e-mail (may be potentially perceived as safer by some staff) ⇒ Posters (if any) don't tell you how to act more securely – just mention the risk and not to do it, not what behaviour to do instead
Salience	There is nothing within the working environment to encourage salience of risk or the need for cybersecurity. Staff forget about the risk unless there happens to be something in the media, but this is quickly forgotten too. Security is not something most staff generally think about on a day to day basis. Also they can become 'blind' to any reminders/alerts that do exist, due to being presented with the same information time and time again.
Environmental Constraints	Staff in healthcare are overworked, fatigued, patient-focused and working in a unique environment where delays could be detrimental to patient health and wellbeing. Cybersecurity practices can conflict with these factors.
Habit and/or Other	Lack of enforcement - Currently no sanctions, incentives or enforcement. Staff don't read the cybersecurity policy (unless maybe the manager if something goes wrong). Policy only referred to when something goes wrong – reactive not proactive

Table 7. Identified factors contributing to use of USB devices, using the IBM approach

4.2.4 Current Behaviour 4: Insecure sharing of patient information

IBM Factor	Barriers
Attitude(s)	**Security as a barrier and/or burden** ⇒ Convenience & ease of access ⇒ Patient care staff priority, not commitment to cybersecurity – using official systems can take time or mean leaving the patient's bedside. Using unofficial workarounds such as WhatsApp or e-mail attachments from a smartphone can enable staff to stay close to the patient and get quicker feedback from colleagues ⇒ Staff resentment of security, e.g., "being forced to do something" they do not understand or agree with ⇒ Official system is worked around to provide better service to patients
Perceived Norms	**No culture around cybersecurity** ⇒ "It's the norm" ⇒ Managers are doing it (top down norms)
Personal Agency	**Perception that technology/IT should/does protect again risks.** Staff feel that cybersecurity is not their responsibility and/or under their control. **Desensitisation to cyber risk.** Staff may think they're in control or desensitised to the use of technology as they use it every day.
Knowledge & Skills	**Risk perceptions & awareness** ⇒ No perception of risk / lack of awareness ⇒ Lack of training ⇒ Hasn't caused any harm so far
Salience	There is nothing within the working environment to encourage salience of risk or the need for cybersecurity. Staff forget about the risk unless there happens to be something in the media, but this is quickly forgotten too. Security is not something most staff generally think about on a day to day basis. Also they can become 'blind' to any reminders/alerts that do exist, due to being presented with the same information time and time again.
Environmental Constraints	Staff in healthcare are overworked, fatigued, patient-focused and working in a unique environment where delays could be detrimental to patient health and wellbeing. Cybersecurity practices can conflict with these factors.
Habit and/or Other	Lack of enforcement - Currently no sanctions, incentives or enforcement. Staff don't read the cybersecurity policy (unless maybe the manager if something goes wrong). Policy only referred to when something goes wrong – reactive not proactive

Table 8. Identified factors contributing to insecure sharing of patient information, using the IBM approach

4.3 Section Summary

This section has recapped upon the insecure behaviours identified in D1.4, and introduced the methodology used for the WP2 workshops to enable us to focus specifically upon key *priority* behaviours. The HC organisations highlighted four priority behaviours: unlocked workstations, insecure password behaviours, use of USB devices and insecure sharing of patient information. Within this section we have identified and described a range of staff motivations underlying these behaviours. The findings consequently feed into the human factors modelling in Sections 5 and 6, and the identification of potential behaviour change 'nudges' in Section 7.

5. Threat Models Analysis

This section will introduce and specify a novel multi-dimensional model of cyber threat. The NIST define cyber threat as "*Any circumstance or event with the potential to adversely impact organisational operations, organisational assets, individuals, other organisations, or the Nation through a system via unauthorised access, destruction, disclosure, modification of information, and/or denial of service*" [NIST-SP-800-53]. As a consequence, it is necessary, when eliciting and analysing threats, to consider multiple perspectives that may have an impact on their identification. The proposed model will be able to collate multiple risk factors such as:

- The business processes that support the organisation mission and that could be impacted by an incident if existing threats materialise;
- The cyber space, i.e., the ICT part of the organisation that support the business processes providing communication, computation and storage services;
- The individuals involved in the organisation that play an active role in the business processes and interact with the cyber space in order to perform their activities;
- The connections between these factors.

Note: Threat is strictly related to the notions of *risk*, *asset* and *vulnerability*. Informally, a risk can be defined by "*the likelihood of an incident and its consequence for an asset*", i.e.,

$$risk = likelihood \times impact$$

Likelihood can be further decomposed as:

$$likelihood = threat \times vulnerability.$$

A threat itself does not represent a real issue if no vulnerability exists to allow it to be materialised, or if it does not impact a relevant business asset. As a consequence, we focus our attention on representing vulnerabilities and how they can be exploited in order to materialise a possible threat. To achieve this, we will use an attack graph model including multiple dimensions (i.e., *layers*) to capture all the relevant factors for an organisation.

The existing literature shows that attack graphs can represent possible ways via which a potential attacker can intrude into the target network by exploiting a series of vulnerabilities across various network hosts, gaining specific access privileges at each step. The use of an attack graph model allows us to focus on the vulnerabilities, on their exploits and on the sequence in which possible exploits can be launched by the attacker. From this point of view, the threat is inferred from the possible attack paths.

To further explain our rationale, this model was selected for the following reasons:

- It focuses upon the factors that are enabling for a potential attack (i.e., the vulnerabilities);
- It considers that attacks can be performed on different layers (e.g., attacks starting on the human factors layer and then progressing onto the ICT network layer);
- It supports risk evaluation and analysis associated to paths representing threats;
- It supports the definition of response plans to reduce or mitigate risk(s).

The aim is to extend the notion of attack graphs and paths to multiple layers to provide a more complete view. As a consequence, the model developed as part of the PANACEA project will support the definition of attack paths through four different layers: human, access, business and network. For example, one case depicts how

an insider obtains an employee's personal login credentials from the employees written notes. Subsequently, he can access his computer by impersonating the employee and then start a cyber-attack on the network.

This example highlights the importance of representing the interface that can be accessed by the login credentials, mediating the interactions between humans and assets. An attack could also originate from an external attacker, who violates an IT device exposed on the Internet. From a risk assessment perspective, mitigation actions for all three layers (human, access, and network) can be both technical and non-technical. When attacks lead to failure of the organisation mission, they have a disruptive impact on business processes (business layer). Understanding the dependencies of assets (and their applications) is key to being able to correctly estimate the impact of attacks. This topic will be explored in Section 6.1.2.

5.1 A Multi-Layer Attack Graph Model

An organisation can be perceived as a complex composite object, made of different facets, that we will call, in analogy with our threat model, *layers*. Such layers are fundamental to describe the various entities of the domain that play a key role in the context of cybersecurity risk assessment. We describe four layers in the following section: human, network, access and business.

The *human layer* is composed of the personnel of the organisation. Typically, these individuals are linked by relationships, e.g., co-work, co-operation or other interaction-based relationships, which are either directly observable, or can be implicitly determined by the organisation (e.g., spatial proximity).

To do their job, individuals make use of a series of assets held by the organisation, such as IT devices and medical devices. Those devices are typically networked, and linked to the organisational ICT infrastructure network, forming what we refer to as the *network layer*. The network layer is one of the primary targets of cyber attacks.

Individuals are authorised to use assets via various kinds of *access credentials*, such as badges, tokens, or user accounts, which provide, to various extents, authorisation/authentication mechanisms to the network assets. We call the set of such access credentials the *access layer*.

The *business layer* describes the set of business processes that support the organisation mission. Business processes have dependency relationships between them, and typically rely on the correct functioning of assets from the network layer, for example, medical devices, computer(s), or a set of networked equipment devices. Figure 4 provides an overview of these layers and their connections.

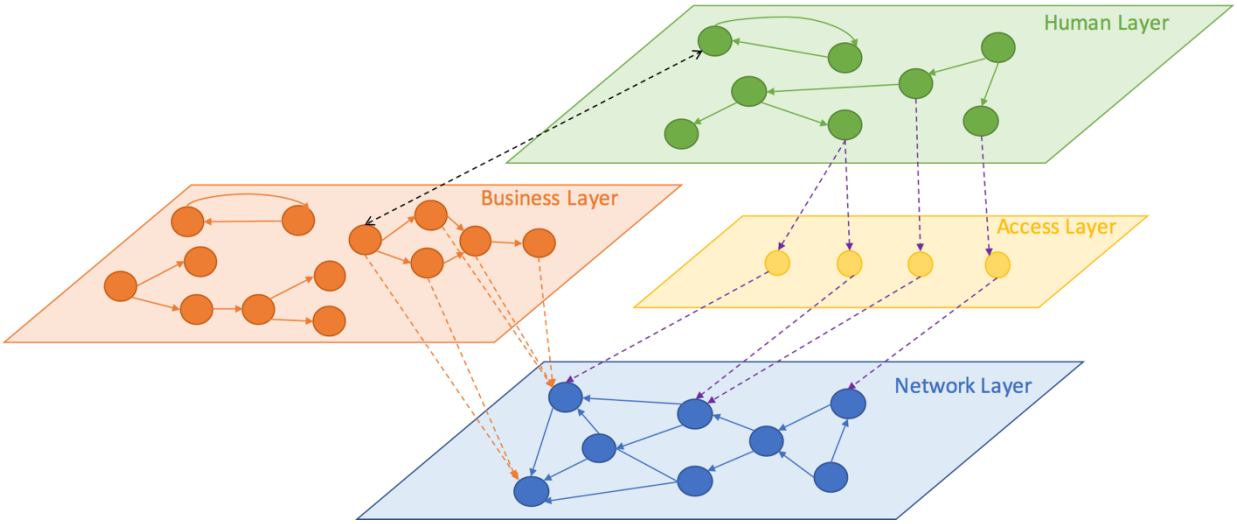


Figure 4. Multifaceted view of an organisation

5.1.1 Key Ingredients for Modelling Humans

Inside every organisation, people represent one of the main factors contributing to the successful accomplishment of the organisation mission. Staff and other parties connected to the running of the organisation are involved (in different ways) in many business processes. This is even more evident in the HC domain where people, such as medical staff, are the core component of every business process. As a consequence, in order to properly protect an organisation, we must consider humans an integral part of the system, and analyse their role in relation to cybersecurity.

To this aim, we consider every human in the organisation as a *"resource"* with her/his own characteristics and vulnerabilities (as identified in Section 4). In recognition that individuals do not operate in isolation but collaborate and interact to accomplish their goal, we consider the "human network" deriving from such social relationships. In the following paragraphs we provide an overview of these concepts before moving to a deeper formalisation.

Individual Profile

In our model, we focus on several aspects that may contribute to an individual's characterisation in relation to cybersecurity analysis. It is fundamental to represent all information that can contribute to identification of their *normal behaviour* in relation to their work duties. To this aim, we characterise every individual by considering his/her role inside the organisation and related information (e.g., time in the current role) that can contribute to their associated level of vulnerability for the environment. Table 9 summarises the most relevant attributes to every individual characterisation.

Attribute	Description
Role in the Organisation	This attribute specifies the role of the individual in the organisation
Time in the Current Role	This attribute specifies the amount of time that the individual has spent in the current role.

Table 9. Individual Profile Attributes

The Individual Profile for each individual in the organisation can be extracted by collecting information available in the HCO archives.

Cybersecurity Profile

Each individual can also be characterised by his/her *cybersecurity profile,* i.e., a set of characteristics that can describe and quantify their personal attitude to cybersecurity in the context of his/her working activities. Table 10 lists the attributes that contribute to an individual's cybersecurity profile.

Attribute	Description
Individual Security Attitude	This attribute allows to evaluate and measure the priority given to cybersecurity practices by the individual.
Security Behaviour	This attribute allows to estimate and measure the individual level of cybersecurity based on the analysis of past behaviours.
Security Culture at Work	This attribute allows to estimate and measure whether deviation from security practices is the norm in the workplace
Security Training Level	This attribute allows to estimate and quantify the overall level of training received by the individual.
Trust in Colleagues	This attribute allows to estimate and quantify the level of trust that the individual feels about her/his colleagues.
Trust of Physical Security of the Building	This attribute allows to estimate and quantify the level of trust that the individual feels about the security of the environment.

Table 10. Cybersecurity Profile Attributes

We note that the set of identified attributes are not directly and immediately available, but will require analysis of historical data within the organisation and also analysis of questionnaire results from HC employees.

Human Vulnerabilities

We consider human vulnerabilities as weaknesses that can be exploited by an attacker to:

- Obtain and/or use credentials to access ICT resources or data;
- Influence the human behaviour to circumvent a security measure.

Unlike vulnerabilities affecting hardware and software components, human vulnerabilities are not uniquely identified and catalogued, and they cannot be detected by "scanning the individual". Thus, we identified the key attributes of human vulnerability and defined a preliminary catalogue for them. This catalogue is included in Annex B.

Identification of individuals' vulnerability traits and the computation of vulnerability attribute scores can be achieved by various means, such as administering appropriate targeted questionnaires. (Note: It is also possible that fully, or semi-automatic, techniques such as traffic/app usage behaviour profiling could be useful. However, these techniques would require conducting an extensive user study involving the collection of very large historical datasets, which is out of scope for PANACEA project).

For each vulnerability, attributes that should be specified are listed in Table 11.

Attribute	Description
Id	Vulnerability identifier
Name	Vulnerability name
Description	Description of the vulnerability
Pre-Conditions	It specifies a set of conditions that must be verified to exploit the current vulnerability
Post-Conditions	It specifies a set of conditions that occur when the vulnerability has been successfully exploited
Access Vector (AV)	It specifies and evaluate the context that is needed to exploit the vulnerability as follows: • *Proximity*: the attacker and the vulnerable person do not need to know each other but the exploit can be executed by simply getting in touch; • *Knowledge*: the attacker and the vulnerable person need to know each other and have some relationship (i.e., personal, social, professional).
Attack Complexity (AC)	It captures the same aspect as the AC attribute in CVSS score [CVSS] i.e., it describes the conditions beyond the attacker's control that must exist in order to exploit the vulnerability. It can be specified as follows: • *Low*: Specialised access conditions or extenuating circumstances do not exist. An attacker can expect repeatable success against the vulnerable component; • *High*: A successful attack depends on conditions beyond the attacker's control. That is, a successful attack cannot be accomplished at will, but requires the attacker to invest in some measurable amount of effort in preparation or execution against the vulnerable component before a successful attack can be expected.
Identity Impact (II)	It measures the impact on the authentication, identification and authorisation capabilities of the vulnerable individual. In particular, it can be quantified in: • *Low*: the attacker is able to temporary impersonate the vulnerable individual; • *High*: the attacker is able to impersonate the vulnerable individual until the access credential is not reset; • *None*: the exploit does not allow to impersonate the vulnerable individual.

Table 11. Human Vulnerability Attributes

Human Reachability

As aforementioned, individuals do not operate in isolation, they are part of a complex human network. In order to support our multi-layer graph model, we need to represent connections between individuals. To this aim, we will focus on the following types of relationships:

- *Role-driven relationships* representing interactions existing amongst individuals inside the organisation in relation to working activities. Some examples of role-driven relationships are: Alice *works with* Bob or Alice *supervises/is supervised by* Bob[1].
- *Proximity relationships* representing interactions between individuals as a result of sharing the same physical working location. As an example, we can say that Alice and Bob have a *proximity* relationship if they perform their working activities in the same room.

It is important to note that:

- Relationships intrinsically introduce an "influence" factor between people (e.g., if Alice is supervising both Bob and David, it is possible that Bob may be more likely to accept Alice's requests – as his superior – in comparison to David's request).
- There may exist, inside the HCO, (internal) policies that specify and regulate interactions between people (e.g., do not share credentials with people working in a different department).

To this aim, we will define in the following section how - starting from data available in the organisation - it is possible to compute the human reachability graph.

5.1.2 Key Ingredients for Modelling Networks

To implement their mission, the organisation makes use of various IT devices and medical devices. Those devices are typically networked, and linked to the organisation ICT infrastructure network, forming thus what we call the *network layer,* which is one of the primary targets of cybersecurity attacks.

In the following we will focus on the security implications of networked devices, their services and their vulnerabilities.

In order to identify the vulnerabilities present in networked systems a key step is understanding the services exposed on the network devices and their reachability condition, i.e., if the set of devices that can connect to them. This is known as the reachability computation problem and will be detailed at the end of the section.

Devices

A network infrastructure is a set of hardware and software resources that are used to enable network connectivity, communication, operations and management of an enterprise network. In particular, network infrastructure devices are the components of a network that recognise the communication paths needed by data, applications, services, and multi-media. These devices include routers, firewalls, switches, servers, load-balancers, intrusion detection systems, domain name systems, and storage area networks.

Network infrastructure devices are key targets for malicious cyber actors, due to the majority of organisational and customer traffic passing through them. For instance, an attacker present on an organisation gateway router can monitor, modify, and deny traffic to and from the organisation. Whilst an attacker present on an

[1] These relationships also entail the possible presence of *sub-contractors* which are under control of a supervisor, and whose information is available to the organisation.

organisation internal routing and switching infrastructure can monitor, modify, and deny traffic to and from key hosts inside the network and leverage trust relationships to conduct lateral movement to other hosts.

The complexity of modern networks has been rapidly increasing due to the explosive growth of Internet connectivity expanding from end-hosts to pervasive devices and network supported applications of various scales. End-hosts are end-system devices where user applications are typically deployed, some of them expose network services. This implies an exposure of several logical TCP/IP ports. Every logical port is subject to the threat to a system, but some of the commonly used ports receive a lot of attention from cybercriminals. Cybercriminals use vulnerability scanners and port scanning techniques for identifying open ports on any system or server. Next, they can identify (from these open ports) what kind of services are running (i.e., HTTP, SMTP, FTP, DNS, SSH, Telnet or VNC) and the kind of system being used by the target victim **[Bou-Harb14]**.

Device vulnerabilities

Device vulnerabilities refer to identified implementation flaws in software or hardware of IT assets, among which also medical devices.

According to the World Health Organisation (WHO), a medical device is “an instrument, apparatus, implement, machine, contrivance, implant, in vitro reagent, or other similar or related article” intended for use in the diagnosis, prevention, monitoring, treatment, etc. of disease or other conditions. The Food and Drug Administration (FDA) uses a similar definition. Classes of medical devices have been defined differently in, e.g., the United States, Canada, Europe or Australia. The FDA has established classifications for approximately 1,700 different generic types of devices. Active devices may or may not involve software, hardware, and interfaces, which are important when considering security issues. These devices can do some processing, receive inputs from outside the device (sensors), output values to the outer world (actuators), and communicate with other devices [Sametinger15]. Such devices are characterised by a large degree of heterogeneity and may be affected by both software and hardware vulnerabilities.

In [D2.1] we have highlighted the importance of performing adequate threat identification and vulnerability assessment, in all contexts where IoT is used. This is particularly vital in the context of HCOs, where most security incidents have been found to be directly or indirectly linked to improper handling of the threat identification phase.

In our model we will rely on the Common Vulnerabilities and Exposure (CVE) vulnerability classification, i.e., a dictionary of public information about security software and hardware vulnerabilities [CVE]. In order to construct the attack graph we need to know vulnerability properties. This information can be extracted from manually analysed data or from the semi-formalised rough categorisation provided by National Vulnerability Database (NVD).

Table 12 details the attributes of interest for the model, for each vulnerability.

Table 12. Network Vulnerability Attributes

Attribute	Description
Id	Vulnerability Identifier (CVE)
Name	Vulnerability Name
Description	Description of the vulnerability
Pre-Conditions	It specifies a set of OS Level privileges (None, User, Root) that must be verified to exploit the current vulnerability
Post-Conditions	It specifies a set of OS Level privileges (None, User, Root) that the attacker might gain when the vulnerability has been successfully exploited. The semantics of a postcondition equal to $None$ is that no privileges at the OS-level is gained after exploiting the vulnerability, disregarding any impact that might be caused by the exploitation. A vulnerability with $None$ privilege post-condition might anyway impact the *confidentiality, integrity* and *availability* of the asset.
Attack Vector (AV)	The context by which vulnerability exploitation is possible. It can assume a series of categorical values *Physical, Local, Adjacent, Network*. • Network (N): A vulnerability exploitable with network access means the vulnerable component is bound to the network stack and the attacker's path is through OSI layer 3 (the network layer). Such a vulnerability is often termed "remotely exploitable" and can be thought of as an attack being exploitable one or more network hops away (e.g. across layer 3 boundaries from routers). An example of a network attack is an attacker causing a denial of service (DoS) by sending a specially crafted TCP packet from across the public Internet (e.g. CVE-2004-0230). • Adjacent (A): A vulnerability exploitable with adjacent network access means the vulnerable component is bound to the network stack, however the attack is limited to the same shared physical (e.g. Bluetooth, IEEE 802.11), or logical (e.g. local IP subnet) network, and cannot be performed across an OSI layer 3 boundary (e.g. a router). An example of an Adjacent attack would be an ARP (IPv4) or neighbour discovery (IPv6) flood leading to a denial of service on the local LAN segment. • Local (L): A vulnerability exploitable with Local access means that the vulnerable component is not bound to the network stack, and the attacker's path is via read/write/execute capabilities. In some cases, the attacker may be logged in locally in order to exploit the vulnerability, otherwise, she may rely on User Interaction to execute a malicious file. • Physical (P): A vulnerability exploitable with Physical access requires the attacker to physically touch or manipulate the vulnerable component. Physical interaction may be brief (e.g. evil maid attack1) or persistent. An example of such an attack is a cold boot attack which allows an attacker to access to disk encryption keys after gaining physical access to the system, or peripheral attacks such as Firewire/USB Direct Memory Access attacks. The categorical values are associated with corresponding numerical values that become larger the more remote (logically, and physically) an attacker can be in order to exploit the vulnerable component. The underlying assumption is that the number of potential attackers for a vulnerability that could be exploited from across the Internet is larger than the number of

	potential attackers that could exploit a vulnerability requiring physical access to a device [CVSS]. It corresponds to Access vector in CVSS v2.
Attack Complexity (AC)	This metric describes the conditions beyond the attacker's control that must exist in order to exploit the vulnerability. Such conditions may require the collection of more information about the target, or computational exceptions. It can assume two possible values. • Low (L): Specialised access conditions or extenuating circumstances do not exist. An attacker can expect repeatable success against the vulnerable component. • High (H): A successful attack depends on conditions beyond the attacker's control. That is, a successful attack cannot be accomplished at will, but requires the attacker to invest in some measurable amount of effort in preparation or execution against the vulnerable component before a successful attack can be expected. It corresponds to Access Complexity in CVSS v2, which takes into account also the interaction with the user.
Privileges Required (PR)	This metric describes the level of privileges an attacker must possess before successfully exploiting the vulnerability. It can assume three possible values. • None (N): The attacker is unauthorised prior to attack, and therefore does not require any access to settings or files to carry out an attack. • Low (L): The attacker is authorised with (i.e. requires) privileges that provide basic user capabilities that could normally affect only settings and files owned by a user. • High (H): The attacker is authorised with (i.e. requires) privileges that provide significant (e.g. administrative) control over the vulnerable component that could affect component-wide settings and files. It corresponds to Authentication in CVSS v2.
Exploit Code Maturity (E)	This metric measures the likelihood of the vulnerability being attacked, and is typically based on the current state of exploit techniques, exploit code availability, or active, "in-the-wild" exploitation. Public availability of easy-to-use exploit code increases the number of potential attackers by including those who are unskilled, thereby increasing the severity of the vulnerability. Initially, real-world exploitation may only be theoretical. Publication of proof-of concept code, functional exploit code, or sufficient technical details necessary to exploit the vulnerability may follow. The list of possible values is presented below. • Not Defined (X): Assigning this value to the metric will not influence the score. It is a signal to a scoring equation to skip this metric. • High (H): Functional autonomous code exists, or no exploit is required (manual trigger) and details are widely available. Exploit code works in every situation, or is actively being delivered via an autonomous agent (such as a worm or virus). • Functional (F): Functional exploit code is available. The code works in most situations where the vulnerability exists. • Proof-of-concept (P): Proof-of-concept exploit code is available, or an attack demonstration is not practical for most systems. The code or technique is not functional in all situations and may require substantial modification by a skilled attacker.

	• Unproven (U): No exploit code is available, or an exploit is theoretical. It corresponds to Exploitability in CVSS v2.
Report Confidence (RC)	This metric measures the degree of confidence in the existence of the vulnerability and the credibility of the known technical details. Sometimes only the existence of vulnerabilities are publicised, but without specific details. It can assume four possible values. • Not Defined (X): Assigning this value to the metric will not influence the score. It is a signal to a scoring equation to skip this metric. • Confirmed (C): Detailed reports exist, or functional reproduction is possible (functional exploits may provide this). Source code is available to independently verify the assertions of the research, or the author or vendor of the affected code has confirmed the presence of the vulnerability. • Reasonable (R): Significant details are published, but researchers either do not have full confidence in the root cause, or do not have access to source code to fully confirm all of the interactions that may lead to the result. • Unknown (U): There are reports of impacts that indicate a vulnerability is present. The reports indicate that the cause of the vulnerability is unknown, or reports may differ on the cause or impacts of the vulnerability. It is also present in CVSS v2.
Confidentiality Impact (C)	This metric measures the impact to the confidentiality of the information resources managed by a software component due to a successfully exploited vulnerability. Confidentiality refers to limiting information access and disclosure to only authorised users, as well as preventing access by, or disclosure to, unauthorised ones. The list of possible values is presented below. • High (H): There is total loss of confidentiality, resulting in all resources within the impacted component being divulged to the attacker. Alternatively, access to only some restricted information is obtained, but the disclosed information presents a direct, serious impact • Low (L): There is some loss of confidentiality. Access to some restricted information is obtained, but the attacker does not have control over what information is obtained, or the amount or kind of loss is constrained. • None (N): There is no loss of confidentiality within the impacted component. It is also present in CVSS v2.
Integrity Impact (I)	This metric measures the impact to integrity of a successfully exploited vulnerability. Integrity refers to the trustworthiness and veracity of information. The list of possible values is presented below. • High (H): There is a total loss of integrity, or a complete loss of protection. • Low (L): Modification of data is possible, but the attacker does not have control over the consequence of a modification, or the amount of modification is constrained. • None (N): There is no loss of integrity within the impacted component

Availability Impact (A)	This metric measures the impact to the availability of the impacted component resulting from a successfully exploited vulnerability. The list of possible values is presented below. • High (H): There is total loss of availability, resulting in the attacker being able to fully deny access to resources in the impacted component; this loss is either sustained (while the attacker continues to deliver the attack) or persistent (the condition persists even after the attack has completed). • Low (L): There is reduced performance or interruptions in resource availability. Even if repeated exploitation of the vulnerability is possible, the attacker does not have the ability to completely deny service to legitimate users. • None (N): There is no impact to availability within the impacted component.

Some of the characteristics considered will be involved in the computation of the network layer of the attack graph (Section 5.2.2), while others will be used in Section 6 for computing the likelihood of attack paths for the purpose of risk quantification.

Network Reachability

Computing the set of host to host reachable services in computer networks (known as the reachability matrix computation problem) is a basic step for building complex cybersecurity analyses and risk assessment; as well as many aspects of network design, monitoring and management (e.g., troubleshooting and maintenance). In this section we propose a model for network equipment (e.g., router and switches) and packet filters. Section 5.2.1 details a suite of algorithms for quantifying reachability based on Layer3 routing and packet filters configuration that also consider the impact packet transformers.

A reachability matrix is a matrix-structured data source $\mathcal{R}$ $t$hat provides information about which device can communicate with each other device of the organisation network, using which source and destination ports, or specific protocols. This information should integrate not only the logical network topology (i.e., routing-allowed communication) but should also provide the ISO OSI level 4 communication possibilities deriving from (i) available network services and (ii) the access control policy implemented in the system.

Computing the set of end-to-end reachable devices, i.e., the identification of packets that can travel from one node to another in a computer network, is a fundamental task that posits numerous challenges. There are many security and network controls that may drop, alter, or forward packets along specific paths [Basile2011]. Together with the network end-points (e.g., workstations and servers, which merely send and receive packets) there are many classes of controls that affect reachability:

- $Routing\ controls$ implement a packet forwarding policy;
- $Filtering\ controls$ permit or block specific traffic;
- $Packet\ transformation\ devices$ first modify then forward the packets.

Transformation devices include Network Address Translation (NAT) or Network Address and Port Translation (NAPT) controls, that alter the packet IP addresses and ports according to a policy. Indeed, reachability analysis can be applied to different scenarios to cope with different security related tasks.

Reachability can be computed in an active way, that is, by probing an existing network (online analysis) or performing a discovery computation considering the representation of the network (offline analysis). Online

analysis is the most used in practice (ping and traceroute), but offline analysis has many advantages and more applications. In fact, the main advantage of offline analysis is that it does not require physical access to the target network, since it relies on its model [Basile17].

In our model, network elements are mainly represented by “middleboxes”, having the role to forward a packet or act as a filter. The term “middlebox” refers to any networking device that can forward packets from one subnet to another, such as a network router, a firewall, a traffic shaper, or a L3 switch. In particular:

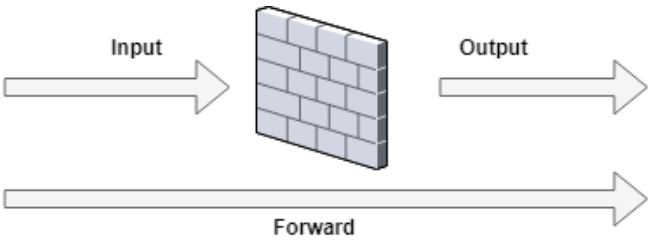
- $Router$: A router determines a hop in the path that a packet should take from subnet A to subnet B. It is designed to route data packets from one interface to another (Figure 5).
- $Firewall$:: A firewall, fundamentally, prevents traffic from reaching a protected network. It is used to provide security by controlling what types of traffic are allowed to pass through a connection (Figure 6).

Our approach must be able to model and manage all the possible different configurations present in different network components. In the following we will describe the structure of the two type of rules that could include the middleboxes.

Rule type	Destination	Gateway	Genmask	Iface
Strict	172.16.1.0	172.16.1.1	**255.255.255.0**	eth0
Direct	10.0.1.0	0.0.0.0	**255.255.255.0**	eth0
Default	0.0.0.0	10.0.1.1	**0.0.0.0**	eth0

Figure 5. Router model

Routing Rules: routing rules are used when hosts or networks are reachable through a router other than the default gateway. Each host in the network knows about the networks directly attached to it and has information on how to reach other networks in its routing table. Moreover, when where an internal router connects other remote subnets (networks), a static route must be defined for those networks to be reachable. In this case we have routers acting as gateways through which the other networks are reached. Reachability computation considers three route models: direct, static and default routing rules. It considers also the metric, if it is defined by the configuration, choosing always the route with the minimum metric value for the same destination.



Requirement	Permission	Protocol	Source	Destination	Port
1	ALLOW	IP	ANY	192.168.1.25	80
2	ALLOW	IP	ANY	192.168.1.25	80
2	ALLOW	IP	ANY	192.168.1.25	443
3	ALLOW	UDP	ANY	192.168.1.10	53
4	DENY	TCP	ANY	ANY	53
5	DENY	IP	ANY	ANY	53
6	DENY		ANY	ANY	

Figure 6. Firewall model

Firewall Rules: firewall rules define what kind of Internet traffic is allowed or blocked. Usually, a firewall implements packet filtering and thereby provides security functions that are used to manage data flow to, from and through the router. Each rule consists of two parts – the matcher which matches traffic flow against given conditions and the action which defines what to do with the matched packet. Firewall filtering rules are grouped together in chains. This mechanism allows a packet to be matched against one common criterion in one chain, and then passed over for processing against some other common criteria to another chain. For all them we need a data structure that is able to validate in an efficient way whether a packet is accepted or not for each different route path, simulating the firewall behaviour. There are three predefined chains (see Figure 7) which cannot be deleted:

- *Input Chain*: used to process packets entering the router through one of the interfaces with the destination IP address which is one of the router's addresses. Packets passing through the router are not processed against the rules of the input chain;
- *Forward Chain*: used to process packets passing through the router;
- *Output Chain*: used to process packets originated from the router and leaving it through one of the interfaces. Packets passing through the router are not processed against the rules of the output chain.

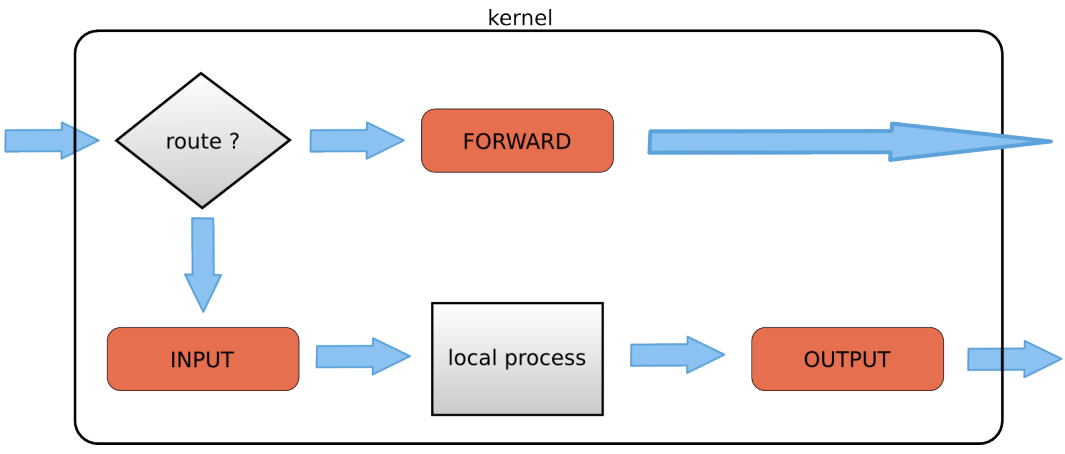


Figure 7. Firewall Chains: Input, Output and Forward

As specified above, in our model a router is defined as a network device with only routing rules, while a firewall can have associated both routing rules and firewall rules.

Depending on the type of unidirectional communications between a source and destination, our model supports the possibility to have the following two types:

- *Stateless communication:* evaluating only the possibility for a packet sent by the source to reach the destination;
- *Stateful communication:* evaluating the possibility to send packets to and receive answers from the destination.

Moreover, some of the firewalls may contain "keep-state" rules. In this case, if a return path from a destination to a source include a firewall with the rule that permits the return packet but marked as keep-state, the packet is allowed trough that firewall if and only if the forward path from source to destination transited trough the same firewall; if not, the return packet is dropped. Our solution overcomes in the following way: when a keep-state firewall rule in encountered on the return path, then we can check to see if that firewall was on the forward portion (from source to destination) of the current path.

5.1.3 Formalising the Multilayer Attack Graph Model

We use a full attack graph to allow minimal a-priori assumptions over attack entry points. The attack graph is composed of three layers: The Human Layer, the Access Layer and the Network layer.

Let us note that the Business Layer aimed at representing business processes and their dependencies relates more to the risk analysis and quantification and it is not directly involved in the threat modelling phase. Thus, we will formalise it later in Section 6.1.2.

More formally, we can define it as a directed multilayer multi-graph [Kivela14], $AG = \langle V_{AG}, E_{AG}, s, t, L \rangle$ where:

- $L = \{L_h, L_a, L_n\}$ is a set of layers (aspects) of AG, to which are associated, respectively, a set of subgraph:
  - $L_H = \langle V_H, E_H \rangle$ representing the Human Layer sub-graph,
  - $L_A = \langle V_A, E_A \rangle$ representing the Access Layer sub-graph,
  - $L_N = \langle V_N, E_N \rangle$ representing the Network Layer sub-graph;
- $s: E_{AG} \rightarrow V_{AG}$ and $t: E_{AG} \rightarrow V_{AG}$ are two functions assigning respectively to each edge its source and target node;
- $V_{AG} = \bigcup_{i \in \{H,A,N\}} V_i$ is the set of vertices of the multi-layer attack graph simply obtained by making the union of all the vertices of layers sub-graphs;
- $E_{AG} = \bigcup_{i \in \{H,A,N,HA,AN\}} E_i$ where $E_{HA} = \{e \in E_{AG}: s(e) \in V_H \wedge t(e) \in V_A\}$, and $E_{AN} = \{e \in E_{AG}: s(e) \in V_A \wedge t(e) \in V_N\}$ is the set of edges of the multi-layer attack graph, obtained by making the union of all the edges of layers sub-graphs and edges cross-layer.

In the following paragraphs, we will describe first each layer sub-graph and then how to represent cross-layers edges.

Human Layer sub-graph $L_H = \langle V_H, E_H \rangle$

This layer aims to represent attack steps that involve exploitation of human vulnerabilities. For example, these can be performed by implementing, among others, social engineering techniques on organisation staff. This

layer aims to demonstrate that, by exploiting human vulnerabilities of an individual h_i, it is possible to get access to, and/or acquire h_i's digital identities. To this aim, we associate to each individual h_i, three possible states to model the capability of h_i to own, use or execute code when using his/her digital identities.

In order to construct this layer, the following elements are correlated:

1. The human reachability graph $HRG = (V_{HR}, E_{HR})$ obtained by aggregating together work-driven relationship and proximity relationships;
2. Individual and security profiles;
3. The set of human vulnerabilities.

More formally, the human layer of the attack graph is represented by a directed subgraph $L_H = \langle V_H, E_H \rangle$ where:

- An element $x \in V_H$ represents a possible level of use that an individual may get on digital identities. In particular, for each human $h_i \in V_{HR}$ we will define three nodes in V_H namely oh_i, uh_i, eh_i representing respectively the fact that h_i *owns* a digital identity, h_i *can use* a digital identity and h_i *can execute code with* a digital identity.
- An edge e ∈ E_H , is associated with a human vulnerability v_k and is such that $s(e) = x_i$ and $t(e) = x_j$, when h_i can get level of usage x_j on some h_j's digital identity by exploiting the human vulnerability v_k on h_j.

Access Layer sub-graph $L_A = \langle V_A, E_A \rangle$

The aim of this layer is to represent credentials or, more generally, digital identities that link an individual to a device, a specific service or a specific piece of data. Thus, it represents the mediator between individuals and assets. More formally, it is represented by a directed subgraph $L_A = \langle V_A, E_A \rangle$ where:

- An element $x \in V_A$ represents a credential (e.g., a pair <username, password>, a badge, a biometric key, etc.);
- An edge e ∈ E_A has $s(e) = x_i$ and $t(e) = x_j$ if and only if a credential x_i depends on another credential x_j (e.g., if there is a two factors authentication method that requires the usage of two credential together).

Note: Each credential $x \in V_A$ is also characterised by a type (e.g., user/password pair, badge, token, etc.) and a level of robustness that can be used in order to weight associated risks when computing attack paths.

Network Layer sub-graph $L_N = \langle V_N, E_N \rangle$

This layer represents possible vulnerability exploits of the organisation's networked assets. Let $\mathcal{D}$ be the set of networked device assets (workstations, network equipment, medical devices, etc.) of interest of the organisation. Concerning cyberattacks over IT devices, an important concept to describe the network layer sub-graph is the *privilege level* that an attacker can gain on such assets. For instance, an intruder might start an attack from the Internet, i.e., with no privilege on the internal IT infrastructure of the organisation, or in the case of an insider threat, they might have an initial given privilege on a machine. As a consequence of attack steps that involve vulnerability exploits, they might raise their privilege level on the current machine (privilege escalation) or gain privileges on other machines (remote privilege gain).

To put all this together, let us describe some preliminary notation of our model:

Each $d \in \mathcal{D}$ is associated to a set of applications $A(d)$, where each $a \in A(d)$ is associated with a (possibly empty) set of network services $S(a)$. Let $S^d = \bigcup_{a \in A(d)} S(a)$ the set of network-exposed services of d and $app(s)$ a function that map each service back to its associated application. To compute attack paths, we need to identify all the vulnerabilities that can be exploited on each $d \in \mathcal{D}$ from every other node in the system. We can recover it using the reachability matrix: Let $S_i^j = S_{\mathcal{R}(i,j)} \subseteq S^j$ obtained from the (i,j) entry of the reachability matrix. Let $P = \{None, User, Root\}$ be the set of OS-level privileges. Each application is associated with a set $\mathcal{V}(a)$ of asset vulnerabilities, where each $v \in \mathcal{V}(a)$ has an identifier $v.cve$ (following the CVE naming standard), a *precondition* ($v.pre \in P$) a *postcondition* ($v.post \in P$) a vector of CIA-impacts (a triple $\langle c, i, a \rangle$, where each of the components can assume different severity levels), and an attack vector $v.av \in \{Physical, Local, Adjacent, Network\}$. For each device, let $\mathcal{V}_i^d = \bigcup_{a \in d} \{v \in V(a) : (v.av = Local) \vee (v.av = Physical)\}$ be the set of local vulnerabilities, and $V_i^j = \{v \in V(app(s)), \forall s \in S_i^j\}$ be the set of vulnerabilities that are present applications related to network services of j that are reachable from i .

$L_N = \langle V_N, E_N \rangle$ is a *state-based, condition-oriented* attack graph [D2.1], where

- Nodes $V_N \subseteq (P \times \mathcal{D})$ represent the possible *privilege states* of an attacker in the organisation's IT infrastructure, and
- Directed multi-edges between them represent attack phases that involve exploitation of vulnerabilities allowing the attacker to escalate its privilege state on a given machine, or move laterally gaining privileges on other machines in the network.

In each privilege state, i.e., each vertex, the attacker can take one of the available attack actions, corresponding to a directed edge out of that vertex. All of such edges share the same source, but may have different destinations. Furthermore, each directed edge e is associated with a single vulnerability v present on applications installed on the device associated to the destination node: we will discuss the contribution of each edge on the risk computation in Section 6.2.

Inter-layer edges

Inter-layer edges are represented by the edge set E_{HA} and connect nodes in the human layer with nodes in the network layer. They link human-layer attack paths exploiting vulnerabilities on the human layer, to the privileges that the attacker can obtain on devices in the network layer. Informally, two sets of edges exist, both of them involving the digital identities:

1. Edges connecting digital identities with their individual level of access/privileges. This set of edges is used to model the fact that an individual can *own* or just *use* a digital identity allowing him/her to access or to execute code.
2. Edges connecting a digital identity with the privilege that it guarantees on a networked asset.

More formally, the inter-layer relationships are modelled by the two sets E_{HA} and E_{AN} where:

- E_{HA} represents the relationships between individuals and credentials;
- E_{AN} represents the relationships between credentials and privileges obtained on a device.

Every edge e of E_{HA} has $s(e) = x_i$ and $t(e) = x_j$ where $x_i \in V_H$ and $x_j \in V_A$ and represents the fact that an individual h_i has capabilities x_i on the credential x_j, while every edge e' of E_{AN} is such that $s(e') = x_i$ and $t(e') = x_j$ where $x_j \in V_A$ and $x_k \in V_N$ and represents the fact that the credential x_j allows to get access to a device with privileges x_k.

5.2 Algorithms

In this section we discuss algorithmic techniques that can help in the automatic generation of the multi-layer attack graph described in the previous section. For the sake of simplicity, we split the description into layers. Firstly, we describe algorithmic approaches to generate an attack graph considering only the asset and network information. Then we demonstrate how to enrich such attack graph by analysing information related to human factors and to credentials used to access applications and devices.

5.2.1 Network Reachability Matrix Computation

The following section explains the algorithms, which allow computation of the reachability matrix. Let $n = |\mathcal{D}|$. A reachability matrix is a $n \times n$ matrix-structured data source $\mathcal{R}$ that provides reachability content for each host in the network.The input and the output of our algorithm are defined as follows:

- *Input:* it must contain the complete configuration data for each device in the target network (hosts endpoints, routers, firewalls). In particular for each of them, all network interfaces and their corresponding L3 IP addresses, the set $S^d\ of$ all network application services running on the device, with associated metadata (name, protocol and port), routing tables and the deployed filtering rules, including those relative to packet transformations (SNAT and DNAT).
- *Output:* can be represented as a reachability matrix, displayed as follows: columns and rows including the hosts in the network, each entry representing the reachability condition between two hosts on the corresponding row and column where each entry, if exist, contains the services, the corresponding interfaces and the set of ports associated with the respective protocol (TCP, UDP). Table 12 provides an example of a reachability matrix where each entry is represented with (in this order) ingoing interfaces, TCP open ports, UDP open ports and the outgoing interface. However, it can be represented also as semi-structured file (like JSON or XML).

	Host1	**Host2**	**PC1**	**Server1**
Host1	eth0:any, any:eth0	unreachable	eth0:443:eth1	unreachable
Host2	eth0:22-80:eth0	eth0:any, any:eth0	eth1:53:eth0	unreachable
PC1	eth1:443:eth0	eth1:53:eth0	eth0:any, any:eth0	unreachable
Server1	unreachable	unreachable	unreachable	eth0:any, any:eth0

Table 13. Reachability Matrix Output

For instance $\mathcal{R}(1,2)$ indicates that Host1 cannot reach Host2, while $\mathcal{R}(1,3)$ shows that Host1 from its source interface eth0 can reach a web HTTPS service (port 443) on PC1 through the destination interface eth1.

The algorithm performing reachability computation is divided in two phases, which we will call **Network** and **Content**. The two can be considered as two separate problems, corresponding to the two different problem scopes that are solved and integrated to provide the solution for the reachability analysis.

As introduced in Section **Error! Reference source not found.**, Network reachability determines, for each host, all the different possible network paths that packets can traverse to reach all of the other hosts. This was possible by modelling network information such as routing routes, simulating the behaviour of routers.

Content reachability models the network security controls (entities) that are accounted for in the computation of the reachability information, reproducing how these systems restrict the specific kind of traffic that is actually allowed between any two hosts, in terms of ports and protocols. The analysis takes into consideration the firewall rules (implemented in endpoints, routers and firewall) and the packet transformers such NAT and NAPT.

In the following we explain the algorithms used and for each of them there will be a pseudocode that help to better understand each computation phase.

Network Reachability

This phase computes a complete network reachability for all hosts proceeding in the following stages:

- For each network equipment device, called middlebox, it computes through a recursive discovery algorithm all the paths connecting the middlebox to the various network LANs considering all the information from its routing table. In particular, the algorithm takes different decisions according to the type of the static rule (default-direct-static).
- For each path of a specific middlebox, the algorithm identifies also the other security devices (middleboxes) that control the traffic keeping track of the input and output network interfaces involved in the considered path.
- Only once the algorithm has computed all the paths from all the middleboxes, does the algorithm compute the complete host-to-host reachability. In this way, the computation becomes more efficient, since usually each host in a specific network has its routing table with a possible default routing rule. This latter is responsible for the reachability of the other remote hosts. Moreover, there are some specific cases when a host has a more detailed routing table with specific static rules for well-determined network or hosts. However, also in this case these rules are most likely repeated along the path to the predefined destination. Hence, it is convenient to build firstly the middlebox-to-LAN paths, since the path from all the hosts connected to a middlebox could have the same path to reach a specific destination. More details will be defined in the Section 5.3.

In the network reachability phase the algorithm mainly takes into account the router's routing tables, which usually constitutes the "backbone" part of a network infrastructure, not considering the endpoints hosts. Practically, the problem consists of simulating the primary routers function, computing the path determination process, which is the way of determining all the possible paths that a packet must follow.

To determine the paths, the algorithm has to explore all the routing tables of each middlebox, searching for specific network address or simply continuing the discovery process until it can be possible. In a generic routing table contains various rule types that implies three main different discovery processes. As illustrated in Table 13 and detailed below there are three different routing rules:

- **Direct Network:** If the destination IP address of the packet belongs to a device on a network that is directly connected to one of the interfaces of the router, that packet is forwarded directly to the destination device. This means that the destination IP address of the packet is a host address on the same network as the interface of the router.

- **Strict Network:** If the destination IP address of the packet belongs to a remote network, then the packet is forwarded to another router trough a default gateway. Remote networks can only be reached by forwarding packets to another router.

- **Default Network:** If the destination IP address of the packet does not belong to either a connected or remote network, the router determines if there is a default gateway. If there is a default route, the packet is forwarded to the another middlebox. If the router does not have a default route, then the packet is discarded.

Rule Type	Destination	Gateway	Mask	Interface
Direct	172.16.1.0	0.0.0.0	255.255.255.0	eth0
Strict	10.0.1.0	172.16.1.1	255.255.255.0	eth1
Default	0.0.0.0	10.0.1.1	0.0.0.0	eth1

Table 14. Example of routing rules for a given middlebox

The first algorithm, described in Figure 8, computes for each routing rule of each middlebox the discovery problem according to the type of rule encountered.

Algorithm 1: Middlebox to LANs

Input: A network inventory with m as total Middleboxes and l total LANs

Result: An $(m \times l)$ matrix $\mathcal{R}$ where each entry (i, j) is filled with a (possibly empty) list of tuples of the kind $(\langle i^a_{in} : mbx : i^a_{out} \rangle)$, e.g., $(\langle eth0 : R1 : eth1 \rangle)$, meaning Middlebox i can reach LAN j via traversing such sequence of interfaces on middleboxes.

```
for each middlebox mbx do
    for each rule r in the routing table do
        reachable_networks = ∅
        ruleType = {direct, strict, default}
        r={destination, gateway, interface}
        if ruleType of r is direct then
            list_mbx = ∅
            list_mbx = list_mbx ∪ mbx with corresponding interface
            reachable_networks = reachable_networks ∪ r.destination
        if ruleType of r is strict then
            list_mbx = ∅
            forwardStrict(list_mbx, r.destination,
             r.gateway)
            reachable_networks = reachable_network ∪ r.destination
        if ruleType of r is default then
            list_mbx = ∅
            default_nets = ∅
            forwardDefault(list_mbx, default_nets,
             r.gateway)
            reachable_networks = reachable_network ∪ default_nets
    end
end
```

Figure 8. Middlebox to LANs pseudocode

In particular, for the direct rule it adds to the list of reachable network the direct connect middleboxes. For the strict and the default rules, the algorithm uses two different discovery method that takes in input some information and returns the results.

The $forwardStrict()$ method takes in input the destination network $dest$ from the rule and its gateway gw, that is the next middlebox that is responsible for the reachability. If gw has not the $dest$ as one of the direct connected networks it continues its discovery problem according to two cases:

1. If gw has information about to $dest$ in its routing table then it forwards the computation to the $next\_gw$.
2. If gw has no information about to $dest$ in its routing table then it forwards the computation to the default gateway.

In this way the final path will include all the middleboxes that are responsible to forward the packet from the initial middlebox. We remind that our algorithm does not arise the problem of inconsistency. If for a determined destination the algorithm will not find the right path because some static routing rules are missing, it will not consider that destination as reachable.

The $forwardDefault()$ method takes as information an empty list $networks$ and the $next\_gw$ in order to search all the possible networks. It is a recursive method that add to the list all the connected network of the reachable gateway and continue to explore until the $next\_gw$ has a default gateway in its routing table. However, in order to avoid cycles the algorithm will not consider those middleboxes already visited. It will return the list $networks$ where each network includes the path of the responsible middleboxes.

When the stage of Middlebox to LANs is completed the algorithm can compute the Host to Host reachability $[H\ x\ H]$. Figure 9 of the pseudocode shows the main loop. This part has in input the network inventory, since it's need to have information about the hosts, e.g. its interfaces and its routing table. It returns in output the complete reachability matrix host to host where for entry in the matrix we have the output interface eth_out of the started host, the input interface eth_in of the reached node and the list of open ports separated by protocol.

In particular, the algorithm starts the computation taking in consideration each interface and firstly it adds to the list $reached\_nodes$ all the directly connected nodes, that is, all those host inside the same interface LAN.

Each host can route its packets to other remote networks in two ways: with a strict static rule that indicates the exact next hop to follow in order to reach the destination or by the default gateway.

The method $getReachedNodesFromStrict()$ takes care of the first case. In practice, the method searches in the Middlebox to LANs matrix the strict LAN or address rs considering as middlebox mbx the gateway of the rule. The correctness of this procedure is based on the fact that mbx has necessary information about rs since it knows the paths for all the reachable remote networks. If the destination is network then the algorithm adds to the list $reached\_nodes$ all the host form this LAN, otherwise it adds to the list only the host with the specific address. If mbx has no information about rs it means that the packet sent by h will never reach that destination cause an inconsistency problem with the routing rules.

The method $getReachedNodesFromDefaulìt()$ takes care of the second case and it is simpler. In practice, the method searches in the Middlebox to LANs matrix all LANs considering as middlebox mbx the gateway of the rule. The correctness is due to the fact that each packet sent by h will certainty reach all the LANs reached by the middlebox mbx.

Algorithm 2: Reachability Host to Host

Input: A network inventory with n as total Hosts
Input: The reachability matrix Middleboxe to LANs (R_m)
Result: An $(n \times n)$ matrix $\mathcal{R}$ where each entry (i, j) is filled with a (possibly empty) list of tuples of the kind $(\langle i^a_{in} : P_1[N_1, N_2] - P_2[N_1, N_2, N_3] : i^a_{out} \rangle)$ where P_n are the protocols and N_n the open ports, e.g., $(\langle eth0 : TCP[22, 80] - UDP[2250, 2344, 3444] : eth1 \rangle)$, meaning Host i can reach Host j throuh the corresponding input and output interface having the list of opened protocol ports at the destination.

```
for each host h do
    reach_nodes = ∅
    for each interface eth do
        reach_nodes = ∅
        interface_lan = getLanFromInterface()
        for each host h in the interface_lan do
            ports = getOpenPorts()
            reach_nodes = reach_nodes ∪ (⟨eth_out : ports : eth_in⟩)
        end
        if h has strict rules in its routing table then
            for each strict rule rs do
                getReachedNodesFromStrict(reach_nodes,
                 rs.destination, rs.gateway)
            end
        if h has deafult rule rd then
            getReachedNodesFromDefault(reach_nodes,
             rd.destination, rd.gateway)
    end
end
```

Figure 9. Host to Host pseudocode

Content Reachability

Once the network reachability has been computed, the algorithm takes in consideration all the security devices that filter the traffic between all host-to-host path. In particular, the algorithm will proceed in the following way:

- The output of the first part gives in output the content for each host-to-host reachability. In particular, the algorithm list all the open ports on each reachable destination. Therefore, in order to validate all the open ports, firstly, the algorithm has to check if each host is or not an end-system firewall, that is, if the host has a firewall system installed that can influences the reachability properties of all the network. All the end-to-end hosts have only the Input and the Output chain when it has an end-system firewall installed since they usually are not capable to forwarding packets. Therefore, for each host and for each LAN the algorithm validates in the first phase all the reachability content considering corresponding Input and Output chains.
- For the middleboxes, the algorithm firstly checks the Forward chain since it is responsible for the forwarding packets. For a specific source-destination path there are more than one firewall that control the communication flow. However, the firewall rules can be deployed in different devices, then the algorithm computes an “equivalent firewall” that includes all the rules of all the firewall included in a specific path. This approach makes our algorithm very efficient since every time we need to filter the reachability content for each host-to-host the algorithm checks immediately if a host has the same path of another considering the computation of a new “equivalent firewall” if it does not already exist.

Algorithm 3: Reachability-Content Host to Host

Input: Reachability Matrix Host to Host $\mathcal{R}$ with n as total Hosts
Input: The list of firewall rules f
Result: An $(n \times n)$ matrix $\mathcal{R}$ where each entry (i, j) is filled with a (possibly empty) list of tuples of the kind $(\langle i^a_{in} : P_1[N_1, N_2] - P_2[N_1, N_2, N_3] : i^a_{out} \rangle)$ where P_n are the protocols and N_n the open ports, e.g., $(\langle eth0 : TCP[22, 80] - UDP[2250, 2344, 3444] : eth1 \rangle)$, meaning Host i can reach Host j throuh the corresponding input and output interface having the list of opened protocol ports at the destination.

```
for each host h1 do
    outputPorts ← getOutputPorts(h1)
    for each reach node h2 do
        path ← getPathFromH1toH2()
        inputPorts ←getInputPorts(h2)
        if path p include firewalls then
            fws ← getFirewalls(p)
            forwardPorts ←getforwardPorts(h1, h2)
            eq_fw ← computeEquivalentFirewallst(fws,
              inputPorts, outputPorts, forwardPorts)
        if ⟨h1.address, h2.address⟩ exists in eq_fw then
            open_ports = getOpenPorts(⟨h1.address, h2.address⟩,
              eq_fw)
            reachPorts ← open_ports
    end
end
```

Figure 10. Pseudocode Reachability-Content

The pseudocode of the algorithm is illustrated in Figure 10. It takes in input the reachability matrix calculated before and for each host $h1$ it checks all the reached ports from $h2$. In particular, the methods $getOutputPorts()$ and $getInputPorts()$ check if the hosts respectively $h1$ and $h2$ have end-firewalls and for each of them computes those ports allowed for the communication $h1.address \rightarrow h2.address$. Then, the algorithm takes the path for each specific communication $h1 \rightarrow h2$ and if this path includes some firewalls that could change the reachability in terms of reachable ports it computes the equivalent firewall eq_fw. If the source address and the destination address are present in the firewall rules considering each Forward Chain of each firewall fw in the path then it report in output only those ports allowed for the output by $h2$ ($outputPorts$), allowed for the forwarding through eq_fw ($forwardPorts$) and those allowed for the input in $h2$ ($inputPorts$). These final open porta will be the reached ports in the reachability matrix taken in input.

Analysis

Let m be the maximum number of middleboxes present in a network inventory and let r the maximum number of rules for each middlebox. The complexity of algorithm Middlebox to LANs is $O[m(r + m)] = O(r \cdot m^2)$. In practice, for each middlebox mbx the algorithm for the discovery will take in consideration at maximum m next_hops. However, this situation is quite limited and present in those cases where are present few middleboxes. Algorithm for the Reachability Host to Host has as complexity $O[n(r + l)]$ where n is the maximum number of host, r the maximum number of rules for each host and l the maximum number of host in a LAN. The cost of this algorithm depends by the complexity of the network in terms of number of host for each LAN since it need to check the reachability for each directed connected host.

The complexity for the Algorithm Reachability-Content Host to Host is $O[n(n + in + out + fw \cdot fwd)] = O(n^2 + fw \cdot fwd)$ where n is the maximum number of host in the network inventory, in the maximum number of firewall rules in the input chain for each n, out the maximum number of firewall rules in the output chain for

each n, fw the maximum number of firewalls for a path between two remote hosts and fwd the maximum number of rules in the forward chain. Also, in this case the cost of the algorithm depends by the complexity of the network topology in terms of firewalls and reachability.

5.2.2 Network Attack Graph Computation

This section describes the algorithm that allows computation of the network layer of the attack graph, whose pseudocode is reported in Figure 11. The algorithm inputs consist of the network inventory information for all networked assets and their reachability matrix. The outputs are the set E_N of edges and the set of corresponding nodes V_N. To make explicit the association between privilege levels and assets, in the code we denote nodes in V_N with the notation $a.p$ when the node refers to a privilege $p \in P$ gained on asset a (e.g., Root privilege on PC1).

In order to have a complete comprehension of the algorithm we recap some base metric of the Common Vulnerability Scoring System (CVSS) already discussed in [D2.1]. In particular, for each vulnerability v, the attack graph is created considering its pre and post conditions and the Attack Vector (AV) metric. This metric reflects the context by which vulnerability exploitation is possible. This metric value will be larger the more remote (logically, and physically) an attacker can be in order to exploit the vulnerable component. The assumption is that the number of potential attackers for a vulnerability that could be exploited from across the Internet is larger than the number of potential attackers that could exploit a vulnerability requiring physical access to a device [CVSS]. The AV metric presents the four cases listed in Table 12, i.e., *Network (N), Adjacent (A), Local (L), Physical (P)*.

The first part of the algorithm adds edges corresponding to local privilege escalations. Any of such edges is such that s(e) = $a_1.p_1$ and $t(e) = a_2.p_2$, with $a_1 = a_2$. For each vulnerability v of an asset a, if the attack vector of the corresponding vulnerability is physical, $(v.av = P)$, it adds an edge between a.N and $a.to\_priv$, as the attacker does not need any OS-level privilege to exploit the target except to be physically near to the target machine.

If the attack vector is not Physical and the post-condition of the corresponding vulnerability is Root, $v.post = R$ then the algorithm adds an edge between $a.U$ and $a.R$, meaning that the attacker need to have user-privileges in order to exploit the target.

If neither of the two cases is considered, the algorithm adds an edge between $a.priva$ and $a.priva$. This case corresponds to all the cases of privilege escalations when the attacker won't gain other privileges except those as pre-conditions. This kind of attacks lead to CIA impacts.

The second part of the algorithm takes into consideration the remote exploitability cases. This part needs of the reachability matrix since for each asset a_1 the algorithm will explore the vulnerabilities of each reachable remote device a_2. For each corresponding vulnerability v, the algorithm checks if the attack vector of v is Network or if it is Adjacent but both of assets belong to the same LAN. In this case, it adds two edges:

1. An edge between the source node a_1 with User privilege and a_2, with the Post-condition derived by the vulnerability v.
2. An edge between the source node, a with Root privilege and a_2, with the Post-condition derived by the vulnerability v.

The algorithm adds two edges because both User level or a Root level preconditions allow to exploit remote vulnerabilities.

Algorithm 1: Network Layer Attack Graph Generation

Input: Vulnerability inventory information for network devices set $\mathcal{D}$, with $|\mathcal{D}| = n$
Input: $n \times n$ reachability matrix $\mathcal{R}$
Output: $AG_N = \langle V_N, E_N \rangle$
$P \leftarrow \{N(None), U(User), R(Root)\}$
$AV \leftarrow \{P(Physical), L(Local), A(Adjacent), N(Network)\}$
$E_N = \emptyset$

for *each node* $n_1 \in \mathcal{D}$ **do**
  for *each* v *in* $V_l^{n_1}$ **do**
    if $v.av = P$ **then**
      $to\_priv = v.post$
      $E_N = E_n \cup (n_1.N, n_1.to\_priv, v)$
    else if $v_1.post = R$ **then**
      $E_N = E_N \cup (n_1.U, n_1.R, v)$
    else
      $priv = v.pre$
      $E_N = E_N \cup (n1.priv, n_1.priv, v)$
  end
  for *each* n_2 *where* $\mathcal{R}(n_1, n_2) \neq \emptyset$ **do**
    for *each* v *in* $V_{n_2}^{n_1}$ **do**
      if $(v.av = N)$ *or* $(v.av = A$ *and* `sameLan(n1,n2)`$)$ **then**
        $to\_pr = v.post$
        $E_N = E_N \cup \{(n1.U, n_2.to\_pr, v), (n1.R, n_2.to\_pr, v)\}$
    end
  end
end

Figure 11. Network Layer Attack Graph Generation algorithm

Analysis

Let k be the maximum number of vulnerabilities present on a node. The complexity of Algorithm 1 is $O[n(k + k \cdot n^2)] = O(k \cdot n^3)$. In practice, computer networks enforce segregation rules using routing and firewall rules that sensibly limit host-to-host reachability, which is never a full clique (i.e., the second component of the sum is typically very distant from n2). However, since many hosts are in the same collision domain (e.g., same LAN) complete subgraphs between subset of nodes might induce corresponding dense portions of the attack graph.

5.2.3 Computing Human and Credential Attack Layers

The following steps are necessary in order to compute the human and credential attack layers:

1. Computing the human reachability graph;
2. Assessing humans' vulnerabilities;
3. Generating the human layer of the attack graph.

Computing the Human reachability graph

The objective of the human reachability graph is to represent social and/or professional relationships that may facilitate the exploitation of human vulnerabilities. This is done by creating a directed multigraph where nodes represent people that we want to consider in our model and edges represent relationships and their corresponding attributes.

More formally, a Human Reachability Graph can be specified as follows

$$HRG = (V_H, E_H) \text{ where}$$

- V_H is the set of people (employees) considered in the model and
- E_H is the set of 4-tuples $\langle h_i, h_j, type_{i,j}, inluence_{i,j} \rangle$ where:
  - $h_i, h_j \in V_H$ are two individuals related by some social/professional relationship,
  - $type_{i,j}$ is the type of relationship existing between h_i and h_j[2] and
  - $influence_{i,j} \in [0,1]$ allows to capture the probability that the considered relationship allows h_i to influence h_j's behavior.

INPUT

- *Organisational Chart* specifying the assignment of people to departments and teams and the hierarchy between roles
- *Working Space Allocation* specifying physical locations where people typically perform their main work

OUTPUT

- The Human Reachability Graph $HRG = (V_H, E_h)$

The algorithm to construct HRG is composed by the following steps:

1. Create the set of nodes V_H by defining an element h_i for each people in the organisational chart that should be included in the model
2. For each pair of people $h_i, h_j \in V_H$ check the following conditions:
   a) If h_i and h_j share at least one working space then create *proximity* edges between them (i.e., add to $E_H$the tuples $\langle h_i, h_j, prox, inluence_{i,j} \rangle$ and $\langle h_j, h_i, prox, inluence_{i,j} \rangle$ where we assume that $influence_{i,j} = influence_{j,i} = 0$ as typically sharing a working space does not introduce any type of influence between people),
   b) If h_i supervises $h_j$according to the organisational chart then create role edges between them (i.e., add to $E_H$the tuples $\langle h_i, h_j, role, inluence_{i,j} \rangle$ and $\langle h_j, h_i, role, inluence_{j,i} \rangle$ where we assume that $influence_{i,j} > influence_{j,i}$ as typically supervisors tends to "impose" behaviors on supervised people),

[2] Currently we considered only two types of relationships that are role-driven relationships and proximity relationships (i.e., $type_{i,j} \in \{prox, role\}$). However, the model is flexible enough to be extended with other types of relationships (e.g., friendships).

c) If h_i and h_j work in the same team but none of them is supervising the other work then create role edges between them (i.e., add to $E_H$the tuples $\langle h_i, h_j, role, inluence_{i,j}\rangle$ and $\langle h_j, h_i, role, inluence_{j,i}\rangle$ where we assume that $influence_{i,j}$ and $influence_{j,i}$ are independent each other).

We acknowledge that we are interested in modeling how human behavior can be affected by external influences. However, how to estimate and quantify precisely the level of influence is still an open problem and it is currently out of scope of this document.

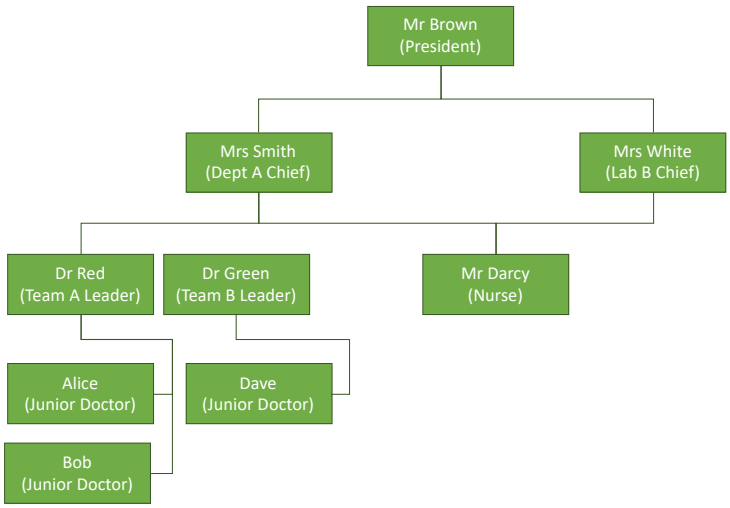


Figure 12. Sample of an Organisational Chart

As an example, Figure 12 shows a sample of an organisational chart. Assuming that Alice, Bob and Dave during their working day spend some time doing research within the same physical laboratory, we can construct the HRG depicted in Figure 13 where solid lines represent edges induced by the *role* relationship while dotted edges represent edged induced by the *proximity* relationship.

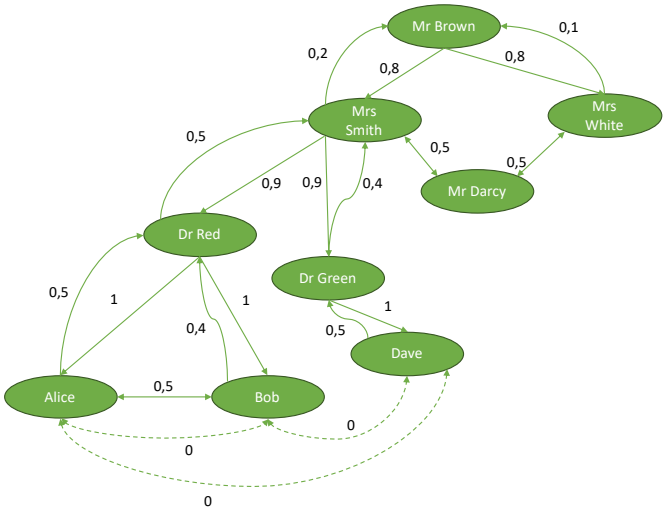


Figure 13. Example of HRG derived from Figure 12

Assessing Humans' vulnerabilities

In order to construct the human-layer attack graph, we need to evaluate human vulnerabilities associated to individuals considered in the model. Unlike the network case, human vulnerabilities cannot be detected and associated to individuals by a scanner. In Section 4 we documented the work the team has carried out to identify and understand those vulnerabilities. Next, it is necessary to identify a procedure that allows us to create the *Human Vulnerability Inventory* (HVI). The HVI is a table depicting the set of human vulnerabilities affecting each individual h_i.

In order to estimate the probability that a certain human vulnerability v_k affects an individual $h_i$we analyse his/her cybersecurity profile.

Human Vulnerability List	Individual Security attitude I.e., priority level in the workplace	Security behaviour i.e., what happened in the past	Security culture at work i.e., whether it is the norm in the workplace	Security training if and when the last train course has been received	Trust in colleagues	Trust in physical security of the building
No 'logout' when leaving the workstation	**X**	X	X	X	**X**	**X**
Disposal or reuse of storage media without proper erasure	**X**	X	X	X	X	
Sharing credential	X	X	X	X	X	
Unprotected credential	X	X	X	X	X	X
Poor password management	X			X		
Insufficient security training on *				X		
Incorrect use of software and hardware			X	X		
Lack of security awareness				X		
Unsupervised work by outside or cleaning staff						X
E-mail misuse	X	X	X	X		
Non-compliance with procedures for introducing software into operational systems			X	X		
Non-compliance to policy on mobile computer usage	X			X		
Insufficient 'clear desk and clear screen' policy	X		X	X	X	X

Table 15. Factors contributing to the human vulnerability presence estimation

Generating the human layer of the attack graph

The basic idea behind the generation of the human layer of the attack graph ($AG_H = \langle V'_H, E'_H \rangle$) is to navigate the Human Reachability Graph (HRG) to evaluate human vulnerabilities that can be exploited, and identify the types of access(es) exploitation can gain. For every human h_i we add three meta nodes to V'_H: uh_i for credentials that h_i can use, oh_i related to credentials owned by h_i and eh_i for credentials that allow h_i to execute code. Edges are computed by combining information from the Human Reachability Graph and the Human Vulnerability Inventory. For every human h_i, let be $neighbours_i$ the list of his/her neighbours (i.e., humans with a relation with him/her), obtained by the Human Reachability Graph. For every neighbour h_j let be $HVI[h_j]$ the list of her vulnerabilities obtained by the Human Vulnerability Inventory; for every vulnerability $v_k \in HVI[h_j]$, we evaluate if the probability that h_i exploits v_k on h_j is above a certain threshold. If it is, we add an edge to E'_H from oh_i and incident to oh_j, uh_j *or* eh_j depending on the post conditions of v_k. The pseudocode of the algorithm is shown in Figure 14.

Algorithm 1: Human Layer Graph Generation: main loop

Input: Human Reachability Graphs $HRG = (V_H, E_H)$
Input: Human Vulnerability Inventory $HVI = [vul\_set_i]^{|V_H|}$
Output: $AG_H = \langle V'_H, E'_H \rangle$

Init
$V'_H \leftarrow \emptyset$; $E'_H \leftarrow \emptyset$
for *each* $h_i \in V_H$ **do**
  $V'_H \leftarrow V'_H \cup < uh_i, oh_i, eh_i >$
end

Construct Human Attack Graph
for *each node* $h_i \in V_H$ **do**
  $neighbors_i \leftarrow$ get_Neighbours(HRG)
  for *each* $h_j \in neighbors_i$ **do**
    for *each* $v_k \in HVI[v_j]$ **do**
      if pre_conditions_hold(v_k, HRG) **then**
        $post_k \leftarrow$ evaluate_post_conditions(v_k)
        if $post_k = U$ **then**
          $E'_H =\leftarrow E'_H \cup \{< oh_i, v_k, uh_j >\}$
        else if $post_k = O$ **then**
          $E'_H =\leftarrow E'_H \cup \{< oh_i, v_k, oh_j >\}$
        else if $post_k = E$ **then**
          $E'_H =\leftarrow E'_H \cup \{< oh_i, v_k, eh_j >\}$
    end
  end
end

Figure 14. Human Layer Attack Graph Generation algorithm

5.3 Section Summary

In this section we have provided a new multilayer threat model that takes into consideration all the facets of an organisation that have key security implications related to cyber threat. This model is novel in its introduction of the *human layer*: Through their involvement in the successful execution of an organisation's mission,

individuals also represent a source of vulnerabilities and possible attack vectors towards business assets and processes.

To account for individual differences, we introduced cybersecurity profiles, capturing the different characteristics of personal attitudes towards cybersecurity in the context of working activities.

We also extended the concept of network reachability to the human layer; which similar to the network case, allows us to model how an attacker can exploit multiple human vulnerabilities to reach the target. Human reachability reflects the influences that individuals have on each other depending on role-driven relationships (i.e., imposed by the roles of the individuals within the organisation) and proximity relations (i.e., individual working in the same location/facilities).

Lastly, the model accounts for each individual's access privileges on assets and services of the organisation, by introducing the access layer. This layer represents the conjunction layer between the human layer and the network layer of the attack model. This multifaceted implant allows for an end-to-end analysis of threat sources that enables a more comprehensive perspective for risk estimation.

Section 7 will describe methods to reduce vulnerabilities by encouraging secure, alternative behaviours.

6. Risk Quantification

Informally, risk can be defined as “the likelihood of an incident and its consequence for/to an asset”. In this section, we explain how starting from the multi-layer attack graph model (Section 5), it is possible to evaluate the risk level. In particular, how to compute the likelihood associated to an attack path and how to correlate the likelihoods of different paths. In addition, we explain how business processes and their dependencies can be represented in order to estimate the impact of a possible incident on the organisation; and how this can be combined with the risk likelihood to obtain the overall evaluation of the risk. In addition, we provide a model that attempts to capture the capabilities of an attacker, to allow this to be accounted for in the risk quantification analysis.

6.1 Methodology

The OWASP Risk Rating Methodology suggests that in order to have a correct cybersecurity posture, there are number of factors that can be considered to determine the likelihood of risk. The first set of factors are the *threat factors*, and are related to the *threat agent* involved (i.e., the attacker) and second set of factors corresponds to the exploitation of vulnerabilities [OWASP].

In the following, we introduce the driving ideas behind our threat agent model, while in Section 6.1.2 we describe our business impact modeling approach. Finally, in Section 6.2 we describe how we use the threat agent model and the attack graph model (described in Section 5) to compute attack likelihood; and how this enables risk computation for business processes.

6.1.1 Threat Agent Modelling

Attack graphs exploit known information about monitored systems to correlate information about current security vulnerabilities and reachability conditions. This information is used within the graphs to identify all *basic attack steps* that form possible attack paths originating from every possible source, and to all targets. However, each and every attack step may or may not be feasible, depending upon the kind of *threat agent* that might try to exploit them.

Threat agents represent different kinds of attackers with differing characteristics. The likelihood of a threat agent exploiting a vulnerability(s) and converting it into successful attack depends upon their characteristics. These can including factors such as *skill level* (e.g., a naïve, script kiddie or an expert hacker), *motivation(s)* (based on possible reward), and *opportunities* (e.g., different available resources), and the possible total size of the attackers.

In particular we will consider 3 kinds of attackers profiles, namely A_n, A_a, A_p (defined formally in Section 6.2.1) corresponding to naive, advanced, and professional attackers, As these attackers have increasing capabilities and resources, their behavior will differ based on the kind of vulnerability of each basic attack step, and will thus impact the likelihood of the attacks. Following [OWASP], such profiles encode different skills and resources, based on factors among which the attack complexity, the maturity of available exploit codes and tools, and available information on vulnerabilities.

This enables to draw threat-agent dependent conclusions for risk, allowing to take decisions based on different hypotheses on the possible threat agents involved.

6.1.2 Business Impact Modelling

In this section we detail the modeling choices taken for the formalisation of the business layer of an organisation, with the aim of identifying the impact of possible threats to the organisational mission.

We start by defining the *business impact model* as a tuple $BI = \langle BDM, Impact(\cdot) \rangle$, where BDM is the business dependency model, that models the business-level entities and their interdependencies, and $Impact$ is a function that assigns an impact to each business-level entity when failing to provide their intended service level. All of these concepts will be discussed more formally in the following sections.

Business Dependency Model

The business dependency model is a tuple $M = \langle BE, DG \rangle$, where BE is the set of business entities and DG is the (business) dependency graph. The set $BE = BE_B \cup BE_S \cup BE_N, s.t. \ (BE_B \cap BE_S = \emptyset, BE_B \cap BE_N = \emptyset, BE_S \cap BE_N = \emptyset)$ is partitioned into three classes of business layer entities: businesses (BE_B), services (BE_S) and assets (BE_N). The assets of the business layer are the direct counterparts of the assets of the network layer (e.g., physical/virtual hosts, network equipment, hardware devices, etc.). The services are the direct counterparts of the services and applications running on network layer assets (e.g., software components, applications, etc.), as specified in Section 5. Businesses have no direct counterparts in the other layers of the models and represent the business processes.

Business Service Levels

Each entity in the business dependency model BDM provides a service, that, depending on the state of the entity itself and the system, may be provided at different *service levels*. In particular, we model these service levels in terms of the CIA triad, i.e., a service level is a triple corresponding to the confidentiality level, integrity level and availability level that the service is able to guarantee, as reported in Table 16.

Parameter	Levels
Confidentiality	violated, guaranteed
Integrity	corrupted, default, intact
Availability	disrupted, degraded, nominal

Table 16. Level of Service specification for CIA attributes

The confidentiality level of a service corresponds to the privacy of the information stored/transmitted/processed by the business layer entity providing that service. The confidentiality level can assume only two values: *violated*, meaning that the confidentiality of the information has been compromised, and *guaranteed*, meaning that the confidentiality is preserved.

The integrity level of a service may represent a different kind of integrity depending on the nature of the business layer entity providing it:

- Physical integrity: for a hardware component;
- Data integrity: for a business layer entity storing/transmitting/processing data;
- Functional integrity: when the integrity refers to the correctness of the function provided by a business layer entity (i.e., the correct behavior of the entity).

The integrity level can range in three possible values: *corrupted*, *default* and *intact*, but the default level is used only in the context of functional integrity. With respect to physical integrity, a component integrity level has

value *corrupted* when it is physically damaged, and *intact*, if it is not damaged. With respect to data integrity, an entity integrity level has value *corrupted* when the integrity of its data has been compromised, and *intact* when it is preserved. Finally, with respect to functional integrity, an entity whose integrity level is *corrupted* provides a service which deviates from the nominal one in an unpredictable way. Instead, an entity has a *default* integrity level if its integrity has been compromised so that it cannot provide its nominal service, but it deviates from its nominal service in a predictable way (e.g., by providing a default service). Finally, an entity with *intact* integrity level provides the nominal service.

The availability level of entities can range in a variable number of values from *disrupted*, when the service provided by the entity is completely unavailable, to *nominal*, when the entity provides the nominal service. Rather than being completely unavailable or providing the nominal service, an entity may also provide the correct service (that is, its functional integrity is preserved) but with *degraded* performance, e.g., in terms of latency and/or throughput.

Therefore, more formally a business layer entity $be \in BE$ is the set of service levels that the entity may provide (depending on its state and that of the system), where each service level $\langle c, i, a \rangle \in be$ is a triple in the set $\{violated, guaranteed\} \times \{corrupted, default, intact\} \times \{disrupted, degraded, intact\}$ representing the CIA levels ensured by that sevice level.

In order to provide a given service level a business layer entity may require that some other entities provide given service levels. This impose interdependencies between business layer entities, that are captured by the dependency graph, detailed in the next section.

Business Dependency Graph

The business dependency graph $DG = \langle V_{DG}, E_{DG} \rangle$, is a directed graph, where the set of nodes $V_{DG} = V_{SL} \cup V_{ANY}$ ($V_{SL} \cap V_{ANY} = \emptyset$) is partitioned into two sets:

- The set of "entities service levels" nodes $V_{SL} = \{\langle be, sl \rangle | be \in BE, sl \in be\}$ that has a pair for each service level of each business layer entity. We will often refer to these nodes simply as "*service level*" nodes.
- A set V_{ANY} of special nodes whose purpose will be clarified later. We will often refer to these nodes as "*any*" nodes.

The set of edges $E_{DG} = E_{ALL} \cup E_{ANY}\ s.t.\ (E_{ALL} \cap E_{ANY} = \emptyset)$ is also partitioned into two sets, where $E_{ALL} \subseteq (V_{SL} \times V_{DG})$ and $E_{ANY} \subseteq (V_{ANY} \times V_{SL})$, that is, an edge in E_{ALL} always starts from a "*service level*" node, but can end in all kind of nodes, while an edge in E_{ANY} always starts from an "*any*" node, and ends in a "*service level*" node. Both kind of edges represents dependencies, but of a different kind, as detailed in the following.

An edge $\langle s, t \rangle \in E_{ALL}$, where $s = \langle be, sl \rangle$, models the following:

- If $t = \langle be_2, sl_2 \rangle \in V_{SL}$, then the edge means that the business layer entity be, in order to provide its service at service level sl, requires that entity be_2 provides its service at service level sl_2.
- If $t \in V_{ANY}$, then the edge means that the business layer entity be, in order to provide its service at service level sl, requires that *any* (at least one) of the dependencies of the "*any*" node t is satisfied.

The dependencies of an "*any*" node a are expressed with edges $\langle a, \langle be_1, sl_1 \rangle \rangle, \ldots, \langle a, \langle be_k, sl_k \rangle \rangle \in E_{ANY}$. Note that those edges, have no particular meaning for the "*any*" node itself. Rather, the "*any*" node represents a kind of relation between them, which assumes a meaning when associated to an edge such as $\langle s = \langle be, sl \rangle, a \rangle$,

meaning that in order for entity be to provide its service at service level sl, *any* (at least one) of the entities $be_1, \dots, be_k$ must provide their services at, respectively, service levels $sl_1, \dots, sl_k$.

Given a "*service level*" node $s = \langle be, sl \rangle \in V_{SL}$, we define $Dep^+(s) = \{\langle s', t \rangle \in E_{ALL} | s' = s\}$ as the set of dependencies of s where s is the dependent entity. Note that, in order for entity be to provide its service at service level sl, *all* of its dependencies in $Dep^+(s)$ must be satisfied.

Analogously, given an "*any*" node a, we define $Dep^+(a) = \{\langle a', t \rangle \in E_{ANY} \mid a' = a\}$ as the set of dependencies of a.

For example, let us assume $s_1 = \langle be_1, sl_1 \rangle$, $Dep^+(s) = \{\langle s_1, \langle be_2, sl_2 \rangle \rangle, \langle s_1, \langle be_3, sl_3 \rangle, \langle s, a_1 \rangle \rangle\}$ and $Dep^+(a_1) = \{\langle a_1, \langle be_4, sl_4 \rangle \rangle, \langle a_1, \langle be_5, sl_5 \rangle \rangle\}$. Then, we can say that in order for be_1 to provide its service at service level sl_1, entity be_2 must provide its service at service level sl_2 *and* entity be_3 must provide its service at service level sl_3 *and* either entity be_4 provides its service at service level sl_4, *or* be_5 provides its service at service level sl_5.

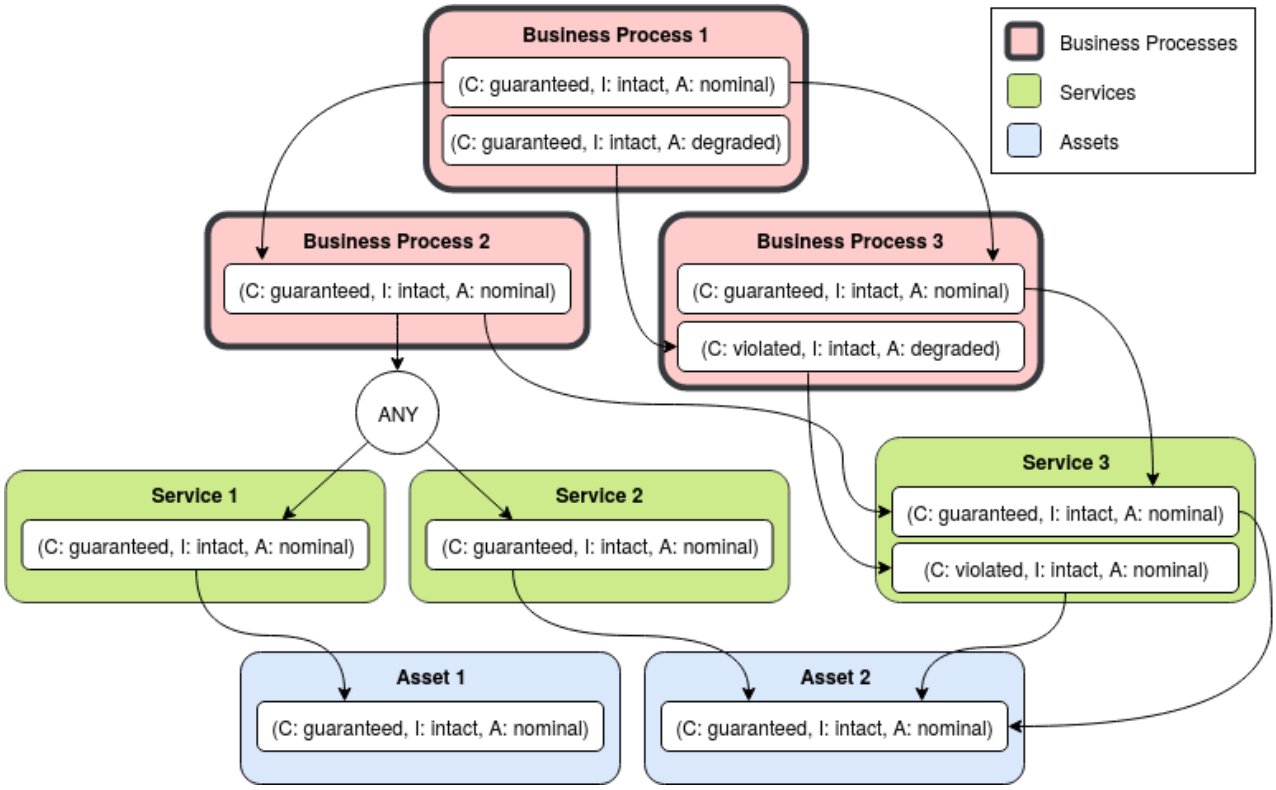


Figure 15. Example of a Dependency Graph.

From a graphical point of view, in this document, we will always represent the business dependency graph as shown in Figure 15 and detailed in the following:

Each service level is represented as a tuple with the values of the CIA triad surrounded by a rectangle. All service levels of a given business layer entity are grouped together and surrounded by a rectangle representing the entity (with a name identifying the entity itself). "*Any*" nodes are represented as circles marked with the text "ANY". Directed edges are represented with arrows as usual.

Impact

The consequences of a business entity not providing its service at a given service level is the impact associated to that "entity – service level" pair. In the business impact model, the impact is modeled by the following function:

$$Impact: V_{SL} \rightarrow I_D$$

This assigns an impact to each business layer "entity – service level" pair (i.e., each element of V_{SL}). The impact is a numerical value from the impact domain $I_D \subseteq \mathbb{R}$. In practical cases, the impact domain can be based on a discrete scale (e.g., "none", "very low", "low", "medium", "high", "very high"), describing various severity levels and mapped to appropriate numerical values.

6.2 Attack Graph-based Risk Quantification

In this section we describe how we make joint use of the multilayer attack graph from Section 5.1, and the business impact model to estimate threat-agent based attack likelihood and perform risk quantification on the business layer.

6.2.1 Attack Path Likelihood

To compute the attack path likelihood we follow an approach similar to **[Granadillo18]**, with the main difference that our approach also takes into consideration the attacker model introduced in Section 6.1.1**.** An attack path can be seen as a sequence of vulnerabilities in the human and/or network layers that an attacker has to exploit to reach the target. Each vulnerability has a different difficulty of being exploited depending on properties of the vulnerability itself and also the capabilities of the attacker.

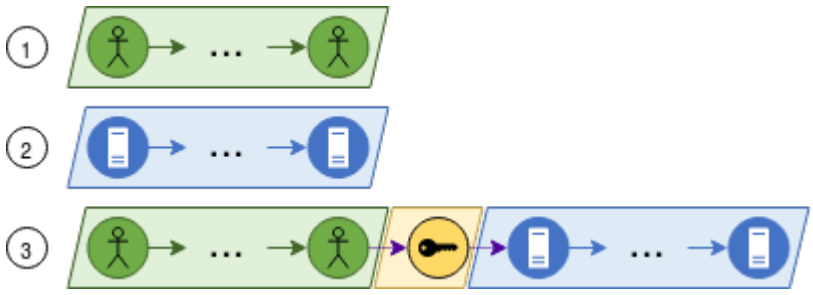


Figure 16. Attack paths structure.

Figure 16 shows the possible structure of an attack path given the multi-layer model of the attack graph. As shown in the figure, there are three possible structures for an attack path:

CASE 1. All nodes of the attack path are nodes in the human layer.
CASE 2. All nodes of the attack path are nodes in the network layer.
CASE 3. The attack path is composed of three sub-paths: (1) a leading sub-path of n human layer nodes, (2) an intermediate sub-path composed of a single node in the access layer and (3) a trailing sub-path of m network layer nodes.

Clearly, the first two cases can be seen as special cases of the third most general case, in which either the leading or the trailing sub-paths can be missing (or, equivalently, can have zero length), and if they are missing

also the intermediate sub-path (the access layer node) is missing. Therefore, each attack path can be modelled as a Markov chain in which the k-th state T_k in the Markov chain corresponds to the k-th step in the attack path. The exit rate λ_k of the sojourn time of the state T_k is set to a value which is homogeneous to the difficulty of exploitation of the vulnerability of the k-th step of the attack path, as detailed later. As noted in **[Granadillo18],** the rationale behind this choice of modelling an attack path through such a Markov chain is driven by two assumptions: (i) compromising a node by exploiting a given vulnerability at the k-th step of the attack path does not depend on the compromising of the previous nodes in the chain (i.e., stateless process), and (ii) the time spent at each node (i.e., attack path step) by the attacker is proportional to the difficulty of exploiting the corresponding vulnerability.

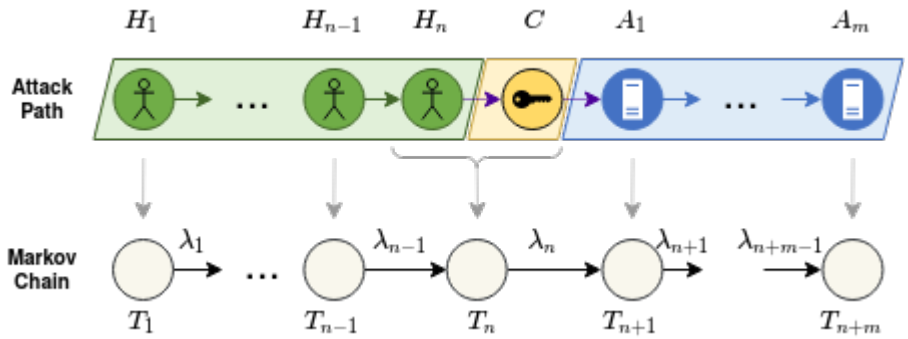


Figure 17. Markov chain associated to an attack path.

Figure 17 shows the construction of the Markov chain associated to an attack path in the most general form. Starting from the leading part of the attack path, we put a node in the Markov chain for each human-layer node of the attack path, but the last, that is, the one connected to the access-layer node. The exit rates $\lambda_1, \dots, \lambda_{n-1}$ of such nodes depend only on the difficulty of exploiting the related human vulnerabilities, as detailed in the following paragraphs. For the last human layer node and the subsequent access layer node, a single node is added to the Markov chain whose exit rate λ_k depends not only on the difficulty of exploiting the human vulnerability, but also on the robustness of the credentials related to the access layer node. Note that in case the access layer and network layer parts of the attack path structure are missing (see Figure 16 - CASE 1), λ_k is determined in the same way as $\lambda_1, \dots, \lambda_{n-1}$. Finally, for the trailing part of the attack path a node is added to the Markov chain for each network layer node, and the exit rates $\lambda_{n+1}, \dots, \lambda_{n+m-1}$ depend on the difficulty of exploiting the related vulnerabilities and also on the capabilities of the attacker. In the following, we will detail how these exit rates are computed and finally how the Markov chain is used to compute the likelihood of the attack path.

Network Layer Exit Rates

Similarly to **[Granadillo18]**, for the network layer part of the Markov chain (exit rates $\lambda_{n+1}, \dots, \lambda_{n+m-1}$ in

Figure 17 we derive the k-th exit rate λ_k by considering the metrics of the Common Vulnerability Scoring System (CVSS) **[CVSS]**. At the time of writing, the most up-to-date version of CVSS is version 3 (CVSSv3), therefore we base our formula on the metrics of CVSSv3 (which are not exactly the same set as CVSSv2, which is used in **[Granadillo18]**). In particular we consider the following CVSSv3 metrics:

CVSSv3 Metric	Metric Values	Numerical Values
Attack Complexity (AC)	Low	0.77
	High	0.44
Attack Vector (AV)	Network (N)	0.85
	Adjacent (A)	0.62

	Local (L)	0.55
	Physical (P)	0.2
Privilege Required (PR)	None (N)	0.85
	Low (L)	0.62
	High (H)	0.27
Exploit Code Maturity (CM)	Not Defined (X)	1.0
	High (H)	1.0
	Functional (F)	0.97
	Proof-of-Concept (P)	0.94
	Unproven (U)	0.91
Report Confidence (RC)	Not Defined (X)	1.0
	Confirmed (C)	1.0
	Reasonable (R)	0.96
	Unknown (U)	0.92

Table 17. CVSSv3 Metrics and associated values

As shown in the previous table, each CVSSv3 parameter has an associated set of metric values (second column) and each such value has an associated *numerical* value in $[0,1]$. These latter values are the basis for all CVSSv3 scores calculations and basically reflect a severity value of the vulnerability. The higher the value the higher the severity, as a higher value for each metric reflects a higher likelihood of attack. For example, considering the *Attack Complexity* metric the *Low* value is associated to a higher numerical value than the *High* value, indeed a vulnerability with a low attack complexity will be more easily exploited by an attacker, compared to a vulnerability with a high attack complexity (assuming all other metrics have the same values for both vulnerabilities). In the following, given a vulnerability v, we will refer to the numerical value of metric X assigned to v as $X(v)$. For example, $AC(v)$, will indicate the numerical value associated to the Attack Complexity metric for vulnerability v.

In addition to this metric, and differing from **[Granadillo18],** we also model the capabilities of different attackers and consider them in the exit rate formula.

Formally, an attacker is modelled as a tuple $A = \langle t_A^{AC}, t_A^{AV}, t_A^{PR}, t_A^{CM}, t_A^{RC} \rangle$, where $t^X \in [0,1]$ is a threshold on the value of the CVSSv3 metric X which basically limits the ability of attacker A of exploiting a vulnerability in case the value of CVSSv3 parameter X is below the threshold for that vulnerability, as clarified later.

Given the vulnerability v_k of the k-th step (in the network layer part) of an attack path and assuming an attacker $A = \langle t_A^{AC}, t_A^{AV}, t_A^{PR}, t_A^{CM}, t_A^{RC} \rangle$, the associated exit rate is computed as follows

$$\lambda_k^A = \prod_{X \in \{AC,AV,PR,CM,RC\}} H(X(v_k) - t_A^X) \cdot X(v_k)$$

Where $H(\cdot)$ is the Heaviside step function, defined as follows:

$$H(x) = \begin{cases} 0, & x < 0 \\ 1, & x \geq 0 \end{cases}$$

Therefore, each factor of the form $H(X(v_k) - t_A^X) \cdot X(v_k)$ in the exit rate formula has value 0 if $X(v_k) < t_A^X$ (that is, metrics X numerical value does not exceed the corresponding attacker's threshold) and value $X(v_k)$ otherwise. Note that, since the exit rate formula is a product, it is sufficient that any metrics value is such that $X(v_k) < t_A^X$ to make the exit rate equal to 0. An exit rate of zero means that the attacker is not able to exploit the corresponding vulnerability and thus unable to reach the target node of the attack path, meaning that the

entire likelihood associated to that attack path for that attacker is 0. So, basically, the attacker's thresholds set which vulnerabilities a given attacker is or not able to exploit. In

Attacker	t_A^{AC}	t_A^{AV}	t_A^{PR}	t_A^{CM}	t_A^{RC}
Naïve	0.77	0.0	0.0	1.0	1.0
Advanced	0.0	0.0	0.0	0.97	0.96
Professional	0.0	0.0	0.0	0.0	0.0

Table 18 we report an example of three increasing capabilities attackers.

Attacker	t_A^{AC}	t_A^{AV}	t_A^{PR}	t_A^{CM}	t_A^{RC}
Naïve	0.77	0.0	0.0	1.0	1.0
Advanced	0.0	0.0	0.0	0.97	0.96
Professional	0.0	0.0	0.0	0.0	0.0

Table 18. Attacker Type

Human Layer Exit Rates

For the human layer part of the Markov chain (exit rates $\lambda_1, \dots, \lambda_{n-1}$ in

Figure 17 we derive the k-th exit rate λ_k by considering the *Attack Complexity* (AC) and *Access Vector* (AV) attributes (see Section 5.1.1) of each human vulnerability, whose possible values are reported in Table 19 for convenience.

Human Vulnerabilities attributes	Attribute Values
Attack Complexity (AC)	Low
	High
Access Vector (AV)	Proximity
	Knowledge

Table 19. Human Layer Attributes

To compute the exit rates, numerical values in $[0,1]$ expressing the ease of exploitation of the human vulnerability must be defined. For example, considering the Attack Complexity attribute, the "*Low*" value should have an associated numerical value higher than that associated to the "*High*" value (condition that we will indicate with the notation "*Low*" $\succ$ "*High*"), since a lower attack complexity implies a higher ease of exploitation. Analogously, "*Proximity*" $\succ$ "*Knowledge*".

Similarly to the network layer case, given a human vulnerability v, we will refer to the numerical value of attribute X assigned to v as $X(v)$. For example, $AC(v)$, will indicate the numerical value associated to the Attack Complexity attribute for vulnerability v. Each numerical value $X(v)$ should be computed as a function of the attributes of the individual and security profile of the individual that has vulnerability v.

Formally, given the vulnerability v_k of the k-th step (in the human layer part) of an attack path, the associated exit rate is computed as follows:

$$\lambda_k^A = \prod_{X \in \{AC, AV\}} X(v_k).$$

Note, that differently from the network layer case the attacker A capabilities do not affect the computation of the exit rate (even though we kept the attacker label A in the superscript for the sake of notation uniformity).

When the last node of the human layer is connected to a node of the access layer, associated to credentials c, the exit rate λ_n^A of the associated node of the Markov chain is computed through the following formula:

$$\lambda_n^A = \left(\prod_{X \in \{AC, AV\}} X(v_n) \right) \cdot \left(1 - R(c)\right)$$

Where the only difference with the previous formula is the factor $R(c)$, where R is a function mapping the credentials to a numerical value in $[0,1]$ representing the robustness of the credentials (i.e., the difficulty of obtaining the credentials).

Attack Path Likelihood Computation

As suggested in **[Granadillo18, Kanoun12]**, we compute the Mean Time to Attack Object (MTAO) as the sum of the expectation of the mean sojourn time of each state in the Markov chain associated to the considered attack path p, for an attacker A:

$$MTAO^A(p) = \sum_k E\{T_k\} = \sum_k \frac{1}{\lambda_k^A}$$

Note that the previous formula is defined only when $\lambda_k^A > 0, \forall k$. If $\lambda_k^A = 0$ for some k, then MTAO is not computed and the likelihood of the entire path is set to 0, as shown in the next formula. The likelihood of the attack path p (when the attacker is A) is computed in dB as follows:

$$L^A(p) = \begin{cases} -20 \cdot \log_{10}\left(\dfrac{MTAO^A(p) - MTAO_{min}}{MTAO^A(p)}\right), & \lambda_k^A > 0, \forall k \\ 0, & otherwise \end{cases}$$

Where $MTAO_{min}$ is the lowest possible value for MTAO, that is $MTAO_{min} = \lambda_{max}^{-1}$, where λ_{max} is the maximum possible value for the exit rate (for any attacker). Basically, the likelihood increases (non-linearly) as the attacker approaches the target (that is, the closer the attacker is to the target, the higher the likelihood). Moreover, given two attack paths with the same number of steps, the higher likelihood value is assigned to the path with the easiest vulnerability exploitation considering the specific attacker.

In the following we describe how the likelihood is used to compute the risk associated to attacks. As pointed out in Section 6.1.1, we remark that being the likelihood tailored on the specific kind of attacker, it will enable to draw risk conclusions that make different hypotheses on the attacker type.

6.2.2 Computing Risk

The risk associated to any element $e = \langle be, sl \rangle \in V_{SL}$ (with respect to an attacker A) is:

$$Risk^A(e) = Likelihood^A(e) \cdot Impact(e)$$

That is the risk is the product of the impact caused by an event that makes business entity be unable of providing service level sl, multiplied by the likelihood that the attacker A is able to cause that event.

Since the impact is given by the business impact model, the only part of the formula that has not yet been defined is the likelihood associated to an element $e = \langle be, sl \rangle \in V_{SL}$, that represents the likelihood that an attacker is able to make be's unable to provide service level sl, through *some attack path* in the attack graph.

Let us define $V_{SL}^{EP} \subseteq V_{SL}$ as the set of business layer "entity – service level" pairs that represent the *entry points* of the attack graph to the business dependency graph. A pair $\langle be, sl \rangle \in V_{SL}$ belongs to V_{SL}^{EP} if and only if there is a node n in the network layer of the attack graph such that one of the following cases holds:

CASE 1. The node n represents the compromising of the business layer entity be (e.g., an asset, service, application, etc.) which impairs be ability to provide its service at service level sl.

CASE 2. The node n has an incoming edge corresponding to a vulnerability that impacts the service of the business layer entity be which impairs be ability to provide its service at service level sl.

Case 1 occurs when the network layer node n has associated privilege Root (R), i.e., the attacker could gain root privilege on the involved asset, and therefore have complete control over it. Case 2 occurs when the incoming edge e is associated to a vulnerability v affecting the application be such that the vector of CIA-impacts $v.cia$ are such to disrupt sl.

The elements of V_{SL}^{EP} represents the connection points between the business dependency graph and the attack graph.

Note that for each entry point $ep = \langle be, sl \rangle \in V_{SL}^{EP}$ there might be multiple attack graph nodes such that the above cases hold. Therefore, we define $V_{AG}^{EP}(ep)$ as the set of nodes of the attack graph such that CASE 1 or CASE 2 holds for ep. For any node $n \in V_{AG}^{EP}(ep)$ we define $AP(n, ep)$ as the set of attack paths that target ep through n. Note that if ep and n are such that CASE 1 holds, then $AP(n, ep)$ is equal to the set of all attack paths that have n as a target. However, if ep and n are such that CASE 2 holds, then $AP(n, ep)$ is equal to the set of all attack paths that ends in n through those incoming edges of n that satisfy CASE 2.

Given an entry point $ep = \langle be, sl \rangle \in V_{SL}^{EP}$, we define the set of all attack paths targeting ep as:

$$AP(ep) = \bigcup_{n \in V_{AG}^{EP}(ep)} AP(n, ep)$$

Moreover, we define the likelihood associated to the entry point ep as:

$$L_{EP}^{A}(ep) = f(\{L^{A}(p) : p \in AP(ep)\})$$

which is the likelihood that the attacker A causes ep not to provide its service level due to an attack that involves one of the attack paths to ep. The generic function $f: \mathcal{P}(\mathbb{R}) \to \mathbb{R}$, in the formula is a function that aggregates the likelihoods of all attack paths targeting ep into a single likelihood value. A possible function for f could be the max function that simply assign to the likelihood of ep the maximum of the likelihoods of its targeting attack paths. This choice is justified by the fact that for the attacker it is sufficient to follow a single attack path to compromise ep, and thus it is reasonable to consider the most likely attack path to compute the risk.

The *entry-point likelihood* $L_{EP}^{A}(ep)$ represents the basis for assigning a likelihood value to each element in the dependency graph. In the following we will describe how to compute the likelihood associated to each node of the dependency graph. Note that the dependency model that we assume admits circular dependencies (e.g. mutual dependencies), thus the dependency graph is not a DAG. However, in the following we will first

introduce how to compute the likelihood in the special case in which the dependency graph is a DAG, and then we will build upon it to discuss the general case in which the dependency graph is not a DAG.

Likelihood Computation when the Dependency Graph is a DAG

Given any node $v \in V_{DG}$, its associated likelihood is computed recursively as follows:

$$Likelihood^A(v) = \begin{cases} g(\{L^A_{EP}(v)\} \cup \{Likelihood^A(v') : v' \in Dep^+(v)\}), & if\ v \in V^{EP}_{SL} \\ g(\{Likelihood^A(v') : v' \in Dep^+(v)\}), & if\ v \in V_{SL} - V^{EP}_{SL} \\ h(\{Likelihood^A(v') : v' \in Dep^+(v)\}), & if\ v \in V_{ANY} \end{cases}$$

So, basically, if v is an *entry point*, then the likelihood is a combination of $L^A_{EP}(v)$ and the likelihoods of the dependencies (if any) of v, according to an aggregation function $g: \mathcal{P}(\mathbb{R}) \rightarrow \mathbb{R}$. Note that if v is an entry point which has no dependencies ($Dep^+(v) = \emptyset$) this is the base case of the recursion. if v is an "entity – service level" node which is not an entry point ($v \in V_{SL} - V^{EP}_{SL}$), then the likelihood is a combination of the likelihoods of the dependencies of v according to the aggregation function g (the same as the first case). Finally, if v is an "*ANY*" node, then the likelihood is a combination of the likelihoods of the dependencies of the "*ANY*" node according to an aggregation function $h: \mathcal{P}(\mathbb{R}) \rightarrow \mathbb{R}$. Note that in addition to $v \in V^{EP}_{SL} \wedge Dep^+(v) = \emptyset$, there are other two base cases of the recursion, which occurs when $v \in V_{ANY} \wedge Dep^+(v) = \emptyset$ and when $v \in V_{SL} - V^{EP}_{SL} \wedge Dep^+(v) = \emptyset$. However, these last two cases cannot happen since the nodes in V^{EP}_{SL} are the only nodes that are allowed not to have dependencies in the business dependency graph. Indeed, an "*ANY*" node without dependencies is useless and can be removed by the model. Moreover, since a node $v \in V_{SL} - V^{EP}_{SL}$ has no direct connection with nodes in the attack graph, with $Dep^+(v) = \emptyset$ there is no way to compute its likelihood (or put in another way, its associated likelihood could be considered 0 given the model). Therefore, such a node would be useless for the purposes of the risk calculation and could safely be removed by the model.

A possible function for g can be simply the max function. Setting $g = max$ means that the likelihood that an attacker compromises a given business entity's ability to provide a given service level is equal to the maximum of the likelihoods associated to its dependencies. This makes sense when computing the risk associated to a single attack, because in order to compromise an entity, for the attacker it is sufficient to compromise any of its dependencies. Thus, in a worst-case scenario it makes sense to consider the path with the highest associated likelihood. On the other hand, max is certainly not a good function for h. Indeed, in case of an "*ANY*" node, an attacker needs to compromise all of the "*ANY*" nodes' dependencies. Modelling the most appropriate function for h its not an easy task, however a good upper-bound to the likelihood of an "*ANY*" node might be the min function. Indeed, since the attacker has to compromise all of the dependencies of the "*ANY*" node, it has to compromise the one with the lowest likelihood. Thus, the likelihood is at most the minimum one, but could be even lower. However, putting $h = min$ represents a valid approximation.

Likelihood Computation when the Dependency Graph is not a DAG

The business dependency graph in general is not a DAG. However, as already stated, it does not admit cycles involving edges in E_{ANY}. That is, circular dependencies (directed cycles) only involves edges in E_{SL}. Since the dependency relation is transitive (if A depends on B and B depends on C, then A *indirectly* depends on C), every node in a directed cycle depends (directly or indirectly) on any other node in the same cycle. This implies that for an attacker it is sufficient to compromise a single node in a directed cycle to compromise every node in the cycle (note that this would not be the case if the cycle would contain edges in E_{ANY}). In other words, if we would make indirect dependencies explicit by putting edges whenever the transitive property of the dependency relation allows to do so (note that this is never needed, as the model itself implicitly captures indirect dependencies, but is always possible), each directed cycle would result in a clique. This means that the likelihood of compromising a node in a cycle is the same for all nodes in the cycle, and is equal to the

likelihood of compromising any of the nodes in the cycle. To compute this likelihood the edges within the cycles are irrelevant. The edges that are relevant for computing the likelihood of the nodes in a cycle, instead, are those outgoing from that cycle. Thus, we introduce the notion of *compressed dependency graph* as follows:

Definition. Given a business dependency graph DG, the *compressed dependency graph* of DG is a graph $\mathcal{C}(DG)$ that has a node u_c for each directed cycle of maximal length c of DG, where u_c is called the *compressed equivalent node* of cycle c in $\mathcal{C}(DG)$, and a node u_v for each node v of DG that is not contained in any directed cycle, where u_v is called the *equivalent node* of node v in $\mathcal{C}(DG)$. The edges of $\mathcal{C}(DG)$ are such that:

- The edges of each directed cycle in DG have no equivalent in $\mathcal{C}(DG)$.
- For each edge (v, v') of DG that is not contained in a directed graph, there is an edge (u, u') in $\mathcal{C}(DG)$, such that u is the *equivalent node* of v if v is not contained in a directed cycle, or, otherwise, the *compressed equivalent node* of the cycle containing v, and u' is the *equivalent node* of v' if v' is not contained in a directed cycle, or, otherwise, the *compressed equivalent node* of the cycle containing v'

Intuitively, the compressed dependency graph is a business dependency graph with each directed cycle compressed into a single node (as shown in Figure 18). Given the above definition, it is straightforward that the compressed dependency graph of a business dependency graph is a DAG.

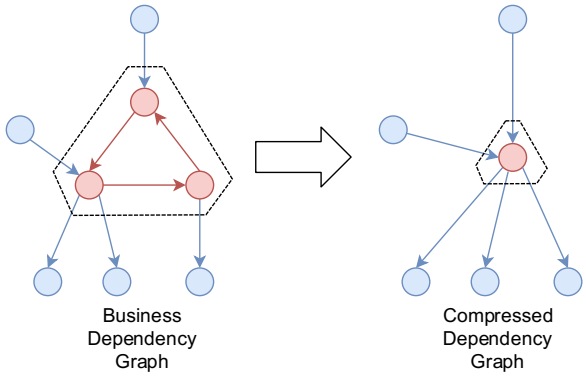


Figure 18. Example of Compressed Dependency Graph.

Therefore, the likelihood of each node in a business dependency graph DG can be computed as follows. Generate the compressed dependency graph $\mathcal{C}(DG)$ of DG and compute the likelihood for each node in $\mathcal{C}(DG)$ by using the methodology described in the previous section (DAG case). The likelihood of any node v in DG is equal to the likelihood of the equivalent node in $\mathcal{C}(DG)$ if v is not contained in a directed cycle, or to the likelihood of the compressed equivalent node in $\mathcal{C}(DG)$ otherwise.

6.3 User scenario

In this section we will show how the proposed model is able to capture relevant aspects of PANACEA domain by providing an instantiation and using it to show how it is possible to compute attack paths. The instantiation provided here is inspired by one of the use cases described in [D1.4] to demonstrate its applicability in the following tasks. However, being still in the early stages of the project, we were not able to provide the model instantiation in the exact use case as some data are not yet available (e.g., data related to the human part) and we need to create them synthetically.

The user scenario considers processes related to the use of a stand-alone medical instrument, in particular the Point of Care Terminal (POCT). The normal operational flow for the POCTs can be described by the procedure illustrated in Figure 19.

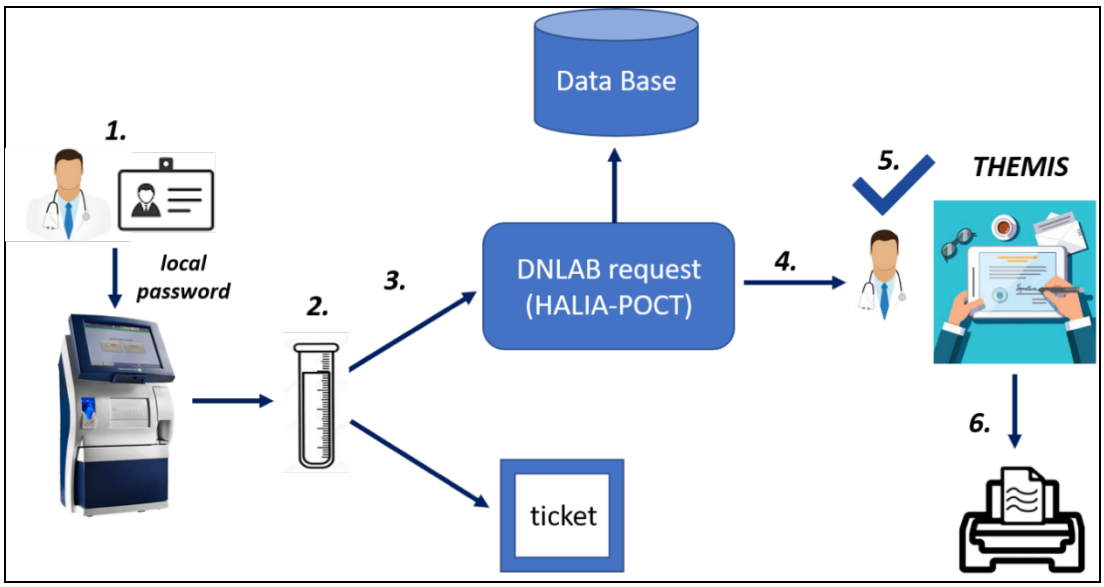


Figure 19. Data flow process for the POCT case

1. The POCT reads, through the scanner, the serial number of the operator present on the badge. To authenticate their access to the HALIA-POCT module, the operator must enter their password. If the procedure is successful, the operator is authorised to carry out necessary analysis of the biological fluid.

2. After performing the sample analysis, the POCT checks whether the results obtained are within the acceptance range. In case of emergency the POCT does not send the results for the validation, but they are automatically validated by the authenticated operator on the POCT.

3. The POCT sends the results in the form of an automatic request to the DNLAB (with the patient's information, the department of origin and the results of the exams) using HALIA-POCT. DNLAB is the client server system for the management of pre-analytical (request management) and analytical (results management) processes. HALIA is a system for connecting laboratory equipment.

4. The results are validated by the clinical personnel responsible for emergencies (clinical validation takes place by pressing the "VALIDATE" button, after the authentication of the individual on the medical device, using UserName and Password).

5. After the clinical validation, a report is generated that is both visible and printable in the different departments of the hospital and in the dispatching office through the intranet (SI). However, this is only possible if the responsible personnel affix the digital signature via Themis. (Themis can be accessible to operators by entering the username and password of the active directory and the USB with personal PINCODE, provided by the hospital or the supplier).

6. The medical report can be printed from any workstation connected to the SI.

6.3.1 Model instantiation

The model instantiation of the user scenario though the multilayer graph defines four different layers described in the following paragraphs. The network layer represents the connection of the vulnerabilities to the different devices in the network. The human layer shows the people who participates in the workflows. The access layer figures out the different ways that the people can use to authenticate them self on the different devices. The business layer shows possible dependencies between processes supported by the selected user scenario.

Network Layer

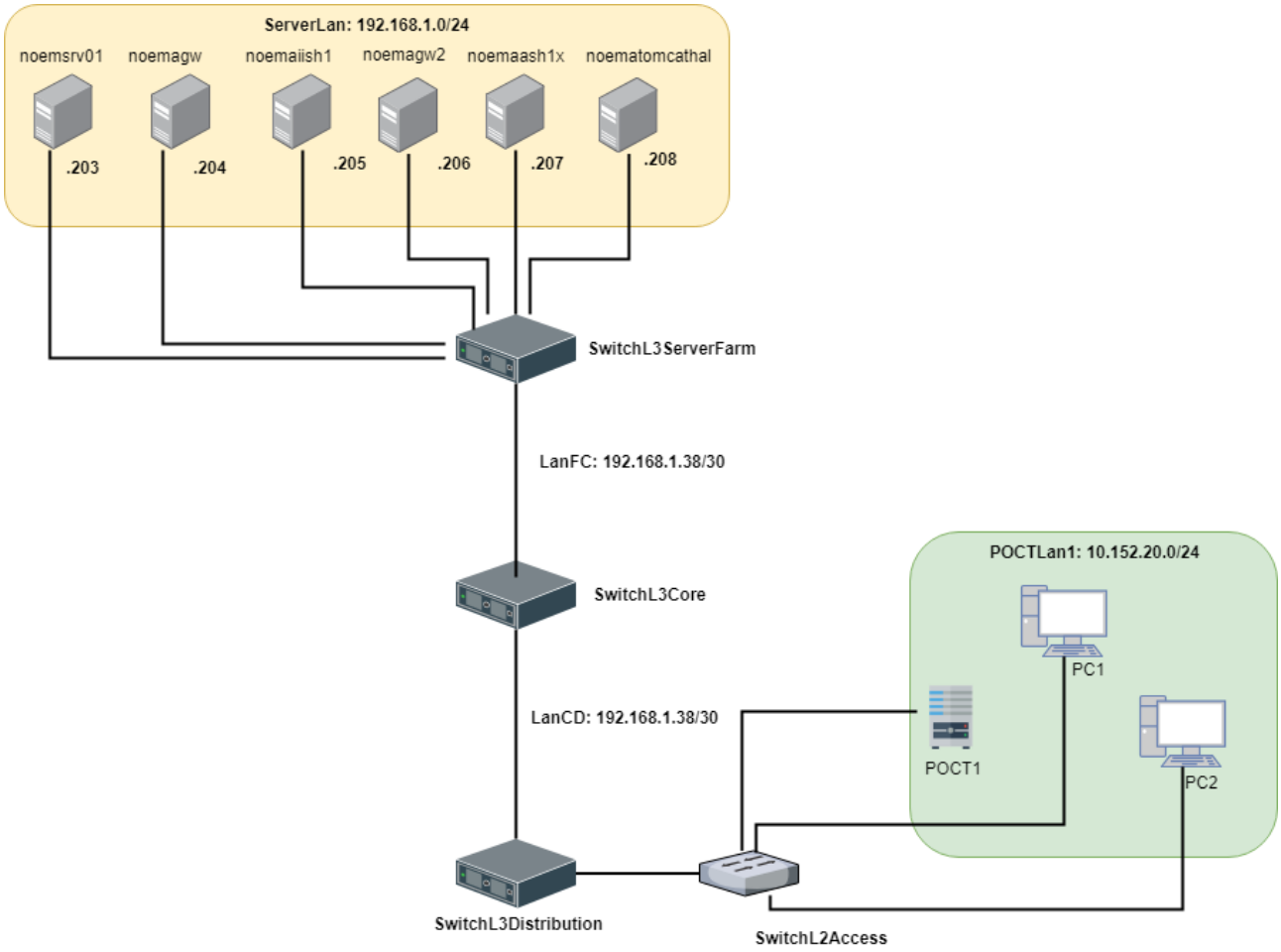


Figure 20. Network Topology User Scenario

The network topology is illustrated in Figure 20. This topology and characteristic of the user scenario have been simplified for the sake of clarity, in order to exhibit only the details needed to exemplify the model instantiation. In particular, we considered only one virtual LAN with one POCT as the medical device and two PCs as workstations. Concerning the network devices, we considered an instance of a hierarchical internetworking model with only one device for each layer. The hierarchical internetworking model is a three-layer model for network design first proposed by **[CISCO]**. It divides enterprise networks into three layers: core, distribution, and access layer. End-stations and servers connect to the enterprise at the access layer. Access layer devices are usually commodity switching platforms, and may or may not provide layer 3 switching services (SwitchL2Access and SwitchL3ServerFarm in Figure 20). The distribution layer is the smart layer in

the three-layer model. Routing, filtering, and QoS policies are managed at the distribution layer (SwitchL3Distribution). The core network provides high-speed, highly redundant forwarding services to move packets between distribution-layer devices in different regions of the network. Core switches and routers are usually the most powerful, in terms of raw forwarding power, in the enterprise (SwitchL3Core).

In this scenario, the medical devices, workstations and servers concerned do not currently have firewall or IPS, there are only routing rules in order to forward the required communications.

The following is a short description of all the devices (PCs) involved during the data flow process for the POCT case:

- N1 (noemasrv01): a database server for the applications HALIA, DNLAB and THEMIS.
- N2, N3, N4 (noemagw, noemaiish1, noemagw2): application servers for DNLAB.
- N5, N6 (noemaash1x, nematomcathal): application server for HALIA.
- N7 (POCT1): Point of Care testing analyser. Software applications:
    - DNLAB Client
    - HALIA Client
    - THEMIS Client
- N8, N9 (PC1, PC2): workstations connected to DNLAB where the operators can consult the clinical reports.
- N10, N11, N12 (SwitchL3ServerFarm, SwitchL3Core, SwitchL3Distribution): L3 switching devices.

In this section we start to describe the network layer of the multilayer attack graph considering the assets presented before. In order to have the generation of the Network Layer Graph we need in input the complete network inventory that include all the necessary information about each device and the Reachability Matrix that is illustrated in Table 20 below. For ease of explanation it contains Boolean values that represent the communication among the devices, rather than its actual content (reachable ports, protocols, input and output interfaces).

	N1	N2	N3	N4	N5	N6	N7	N8	N9	N10	N11	N12
N1	*Yes*	*Yes*	*Yes*	*Yes*	*Yes*	*Yes*	*Yes*	*No*	*No*	*Yes*	*No*	*No*
N2	*Yes*	*Yes*	*Yes*	*Yes*	*Yes*	*Yes*	*No*	*No*	*No*	*Yes*	*No*	*No*
N3	*Yes*	*Yes*	*Yes*	*Yes*	*Yes*	*Yes*	*No*	*No*	*No*	*Yes*	*No*	*No*
N4	*Yes*	*Yes*	*Yes*	*Yes*	*Yes*	*Yes*	*No*	*No*	*No*	*Yes*	*No*	*No*
N5	*Yes*	*Yes*	*Yes*	*Yes*	*Yes*	*Yes*	*Yes*	*No*	*No*	*Yes*	*No*	*No*
N6	*Yes*	*Yes*	*Yes*	*Yes*	*Yes*	*Yes*	*Yes*	*No*	*No*	*Yes*	*No*	*No*
N7	*Yes*	*No*	*No*	*No*	*Yes*	*Yes*	*No*	*No*	*No*	*No*	*No*	*No*
N8	*No*	*No*	*No*	*No*	*No*	*No*	*No*	*No*	*No*	*No*	*No*	*No*
N9	*No*	*No*	*No*	*No*	*No*	*No*	*No*	*No*	*No*	*No*	*No*	*No*
N10	*No*	*No*	*No*	*No*	*No*	*No*	*No*	*No*	*No*	*No*	*Yes*	*Yes*
N11	*No*	*No*	*No*	*No*	*No*	*No*	*No*	*No*	*No*	*Yes*	*No*	*Yes*
N12	*No*	*No*	*No*	*No*	*No*	*No*	*Yes*	*Yes*	*Yes*	*No*	*Yes*	*No*

Table 20. Reachability Matrix User Scenario

The attack graph of the Network Layer is represented in Figure 21. The edges among the assets indicate the different attack paths from source to destination. In particular, we consider two vulnerabilities:

- **CVE-2010-1883**: Integer overflow in the Embedded OpenType (EOT) Font Engine in Microsoft Windows XP SP2 and SP3, Windows Server 2003 SP2, Windows Vista SP1 and SP2, Windows Server 2008 Gold, SP2, and R2, and Windows 7 allows remote attackers to execute arbitrary code via a crafted table in an embedded font, aka "Embedded OpenType Font Integer Overflow Vulnerability".
  - Pre-condition: *None*
  - Post-condition: *Root*
  - Devices: *N2*
- **CVE-2018-4846**: A factory account with hardcoded password might allow attackers access to the device over port 5900/tcp. Successful exploitation requires no user interaction or privileges and impacts the confidentiality, integrity, and availability of the affected device.
  - Pre-condition: *None*
  - Post-condition: *Root*
  - Devices: *POCT1*

Taking into consideration these two vulnerabilities and the reachability matrix, all nodes in the same vLAN (e.g., ServerLAN and POCTLan1) of the vulnerable devices can be a step of an attack or also the starting point since it reaches the possible "target" directly. For the other remote connections the communication is allowed by the definition of specific network policies.

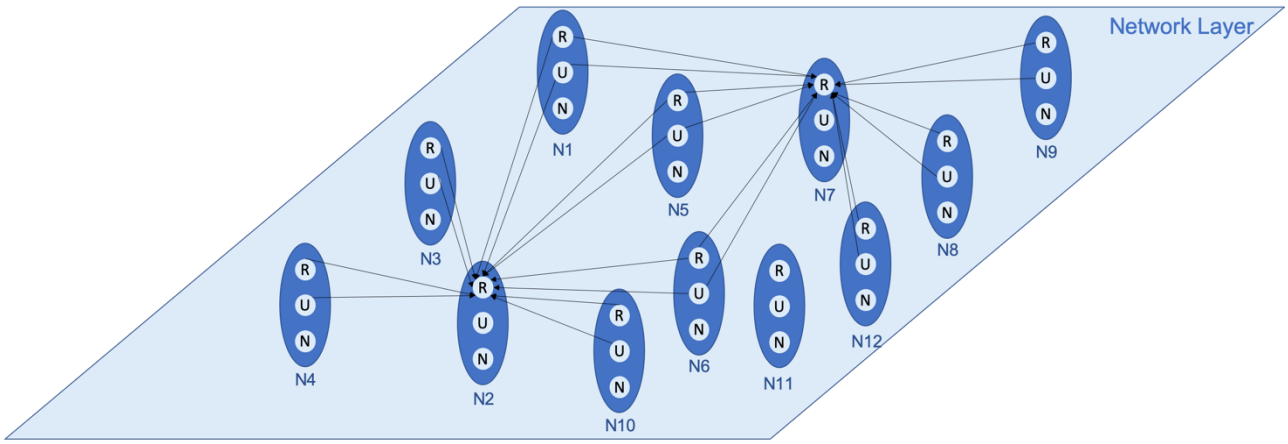


Figure 21. Attack Graph Network Layer

Human Layer

Concerning the human layer of the POCT scenario we assume the existence of seven different individuals:

- H1, H7 are operators of the laboratory;
- H2 is responsible for the validation of emergencies;
- H3 is a doctor responsible personnel;
- H4, H6 are doctors;
- H5 is a nurse.

H1 and H7 are two operators that work in the same room of the laboratory. H2 can validate the analysis in case of emergency, while H3 affix the digital signature via Themis. This step is important to allow the sharing of the medical report in the different departments of the hospital. H6 is a doctor that works in collaboration with H3. Finally, H4 is a generic doctor that can visualise and print the medical report on a workstation connected to the SI. H4 works generally with a nurse H5.

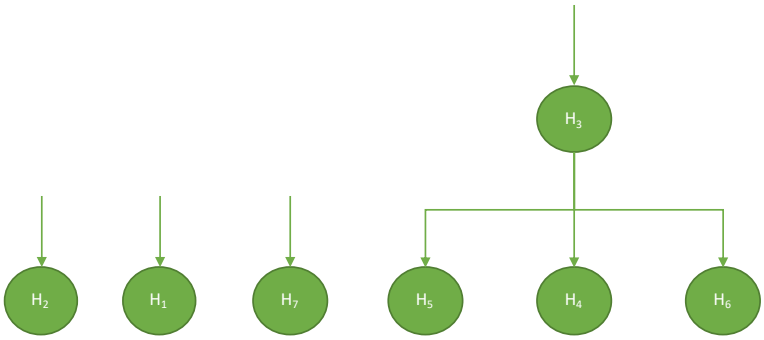


Figure 22. Fragment of the Organisational Chart

Figure 22 shows the fragment of the Organisational chart that we assumed be representative of the seven persons considered in our example while Figure 23 shows the related Human Reachability Graph HRG.

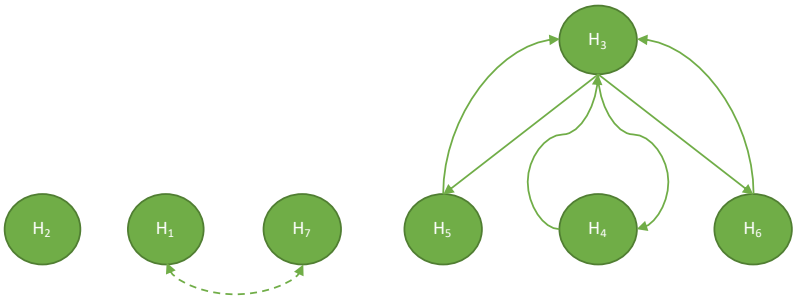


Figure 23. Human Reachability Graph $HRG = (V_H, E_H)$

After the analysis of individual profiles and cyber security profiles let us assume that each individual is affected by some human vulnerabilities (selected from those listed in Annex B) as described in Table 21.

Table 21. Human Vulnerability List for each Individual

Individual	List of Human Vulnerabilities
H_1	HVUL_01, HVUL_04, HVUL_05
H_2	HVUL_01
H_3	HVUL_03
H_4	HVUL_03, HVUL_05
H_5	HVUL_04
H_6	HVUL_03, HVUL_05
H_7	HVUL_02

Applying the reasoning reported in Section 5.2 we get the Attack graph for the Human Layer reported in Figure 24.

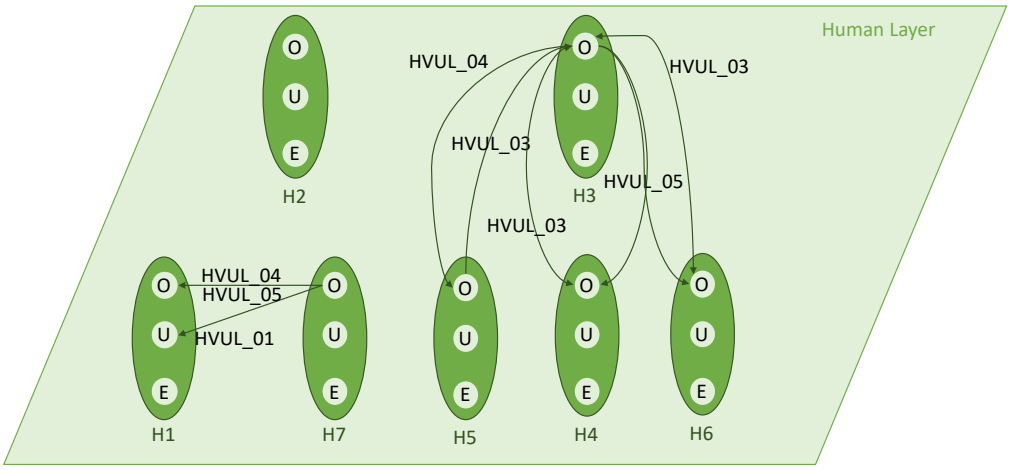


Figure 24. Attack Graph Human Layer

Access Layer

The access layer connects the human and the network layers by collecting all the user credentials for network and medical devices. In the POCT scenario, we identify four credentials used by the personnel to access the devices of interest (Figure 25):

- A1 is the authentication with badge and password used by a laboratory operator (H1) to access the HALIA client on N7;
- A2 is the authentication with username and password used by H2 to access the Point of Care Testing Analyser (N7);
- A3 is the authentication with username and password and USB with personal PINCODE used by H3 to sign medical reports through the THEMIS client of N7;
- A4 is the authentication with username and password used by a doctor (H4) to visualise and print medical reports using a workstation (N8, N9) in the SI.

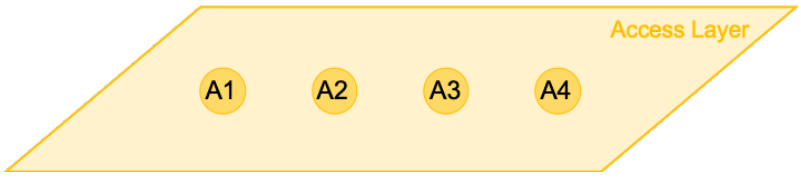


Figure 25. Attack Graph Access Layer

Business Layer

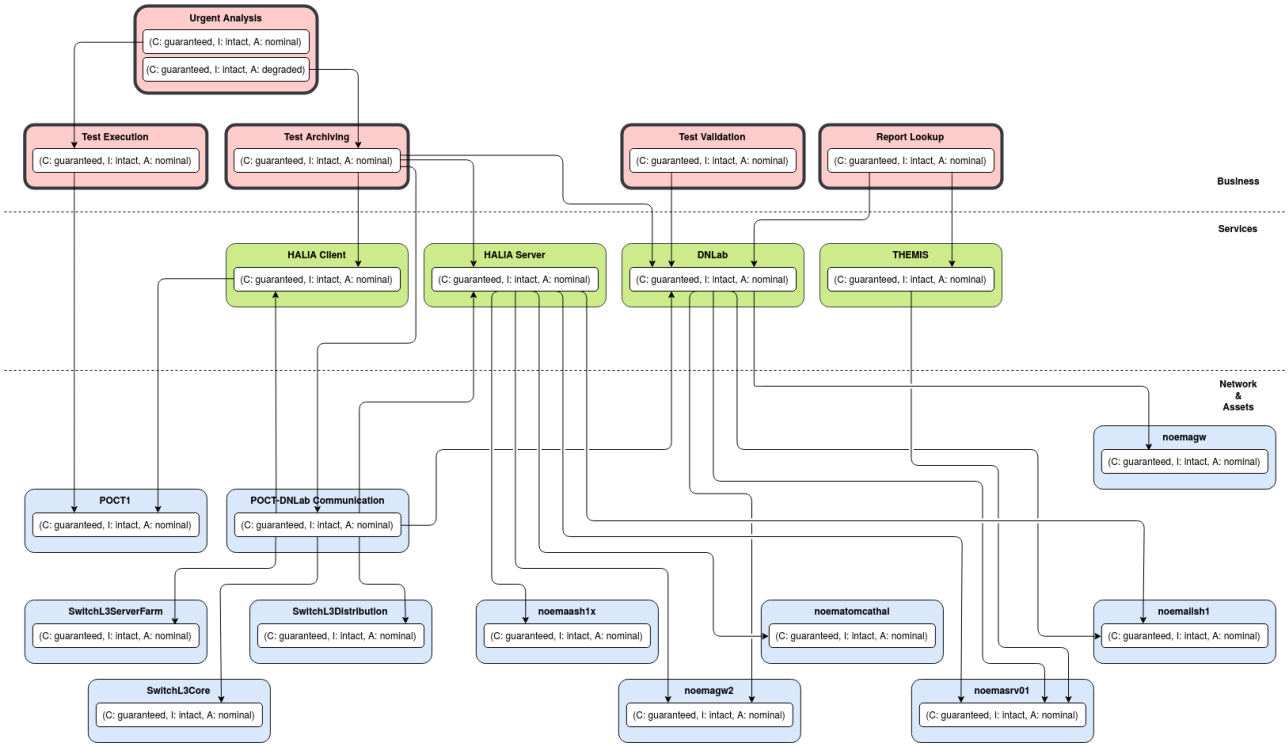


Figure 26. Business Dependency Graph

The Business Dependency Graph of the POCT scenario is depicted in Figure 26. There are five business processes:

- Urgent Analysis
- Test Execution
- Test Archiving
- Test Validation
- Report Lookup

In addition to the business processes there are four services associated to the applications of the POCT scenario, namely HALIA (with the client running on the POCT and the server distributed on the VMs), DNLab and THEMIS. Finally, there are 11 assets, including the POCT itself (N7), the database and application servers (N1-N6), the L3 switches (N10-N12) and an “abstract” asset named “POCT-DNLab Communication” (see Business Dependency Graph, Figure 26) which represents the communication between the POCT and the DNLab. This latter depends on the correct functioning of the three L3 switches, HALIA (both the client and the server) and DNLab. Note that this asset is just a convenience node and it is not necessary to add it to the business layer. Indeed, in principle we could have linked each node connected to this abstract node directly to all nodes it depends on. However, putting this abstract node reduces the number of links, making the model more readable and modular, without affecting the correctness of the risk computation.

The HALIA client, running on the POCT depends on this device, while the HALIA server, DNLab and THEMIS depend on the servers they run on.

The test execution business process depends on the POCT, while Test Archiving depends on HALIA (client/server), DNLab and the communication between the POCT and DNLab. The Urgent Analysis business process has two service levels. The nominal one (C: Confidential, I: Intact, A: Nominal) depend on the Test Execution, while the degraded one (C: Confidential, I: Intact, A: Nominal) depend on the Test Archiving. The Test Validation only depends on DNLab, while the Report Lookup depends on DNLab and THEMIS

6.3.2 Multi-Layer Attack Paths

In the previous section have been defined and instantiated the four layers: Network Layer, Access Layer, Human Layer and Business Layer. Figure 27 further shows the inter-layers connection that allow to represent the relations among humans and devices.

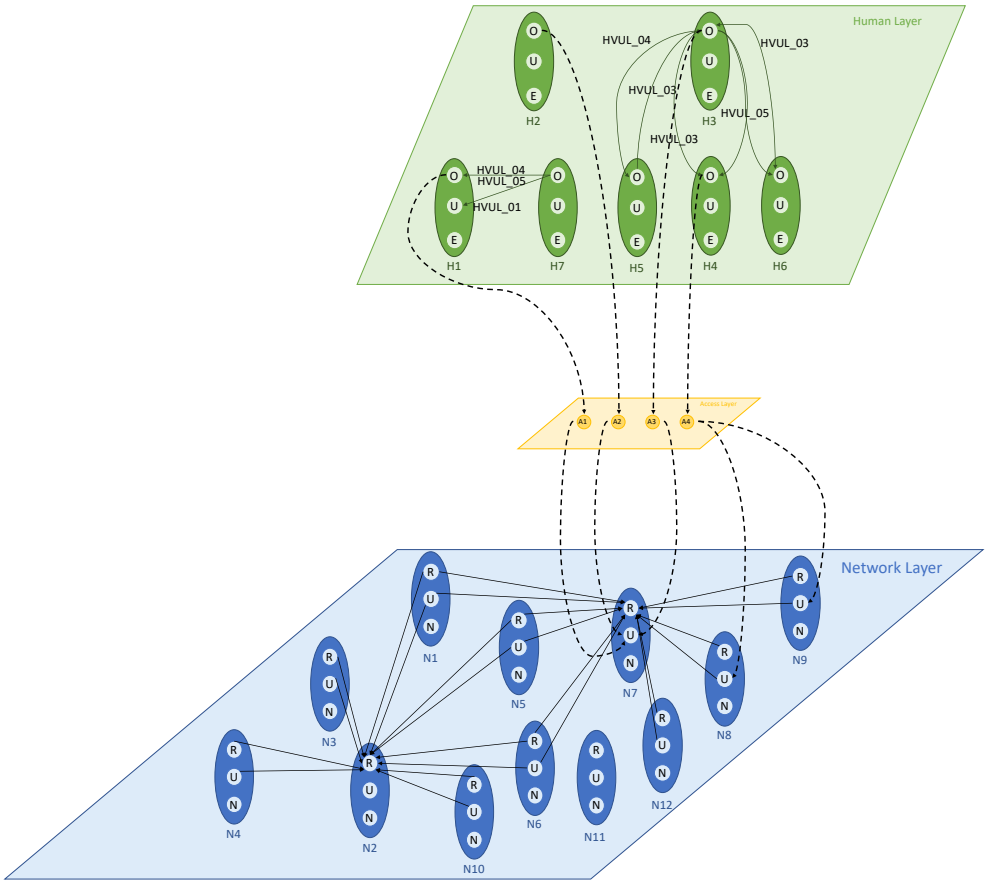


Figure 27. Human, Access and Network Layer of the Attack Graph

In this section, we report an example scenario from which we extract some examples of multi-layer attack paths. In particular, according to some kind of vulnerability, we describe the attack flow of an attacker from the human layer to the network layer targeting different devices.

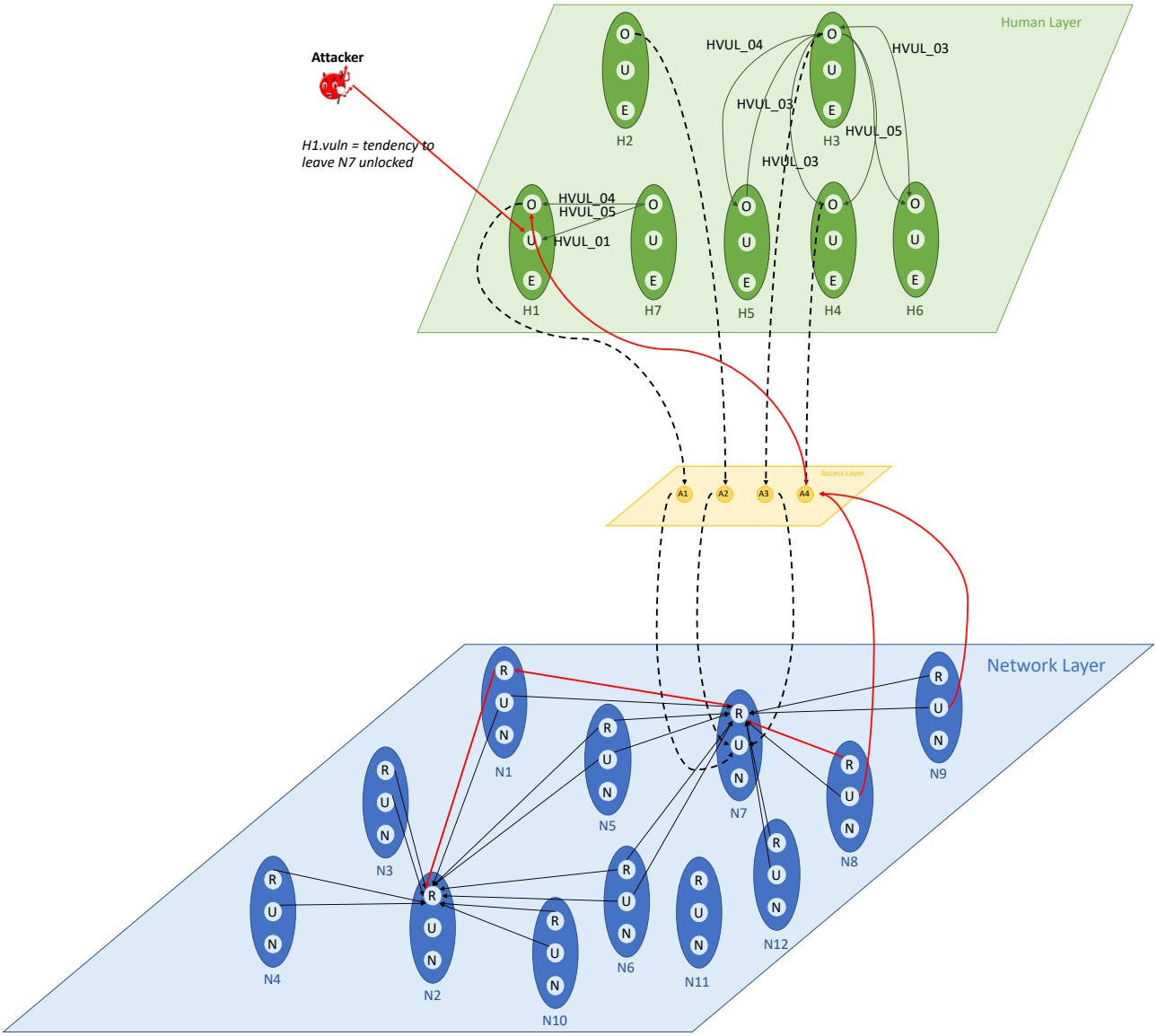


Figure 28. Multi-layer Attack Graph

In particular, this scenario (Figure 28) considers an attacker that could be in the laboratory of chemistry, biochemistry and clinical biology who aims at disrupting the data flow process with a POCT (in this case Urgent Analysis). In the laboratory there are different devices: *N7* and the two workstations (*N8* and *N9*), both of which can be used through two different access identities: *A1* and *A4*. The attacker relies on the poor security attitude of the personnel present in the laboratory. *H1* usually uses workstation *N8*, and sometimes leaves the workstation logged in but unattended.

N8 is connected to the local area network (LAN) of *N7*(PoctLan1). The other network involved in this data flow process is the *Server LAN* network (*ServerLAN*). *ServerLAN* hosts various kinds of servers, among which *N1, N5* and *N6*, machines that communicate with N7 to archive the test of the analysis. *ServerLAN* also contains servers *N2, N3* and *N4* which together guarantee the clinical validation and the generation of reports.

N7 is the only device in *PoctLan1* that reaches *N1, N5* and *N6* in the *ServerLan* due to network policies of the *N10, N11* and *N12* devices which sits between the two LANs.

However, from *N8* the attacker can only connect to *N7* and exploit the vulnerability CVE-2018-4846 on the POCT device, allowing access to the device over port 5900/TCP. In this way, the attacker can block the current data flow process in two ways:

1. **PATH 1**: Once he had Root access to *N7* **he can shut down the medical device violating all the CIA requirements**. In this case the attacker ends his attack disrupting the following business processes: *Urgent Analysis* and *Test Execution*. In the following are listed all the "points" interested by this final step:
    - Path: *H1->N8->N7*
    - Network Layer: *N7* and *N8*
    - Access Layer: *A4*
    - Human Layer: *H1*
    - Business Layer: *Test Execution*, *Test Archiving* and *Urgent Analysis*.
2. **PATH 2**: Another path can instead use *N7* as a pivot to connect to *N1* and then exploit an arbitrary execution code vulnerability of *N2*, CVE-2010-1883 to obtain root access on the machine and shut it down. In the following are listed all the "points" interested by this final step:
    - Path: *H1->N8->N7->N1->N2*
    - Network Layer: *N7, N8, N1, N2*
    - Access Layer: *A1* and *A4*
    - Human Layer: *H1*
    - Business Layer: *Test Execution*, *Test Archiving*, *Urgent Analysis*, *Test Validation* and *Report Lookup*.

The effect of this scenario is that the attacker could make unavailable those compromised devices, as well as the business processes associated to it (Figure 29).

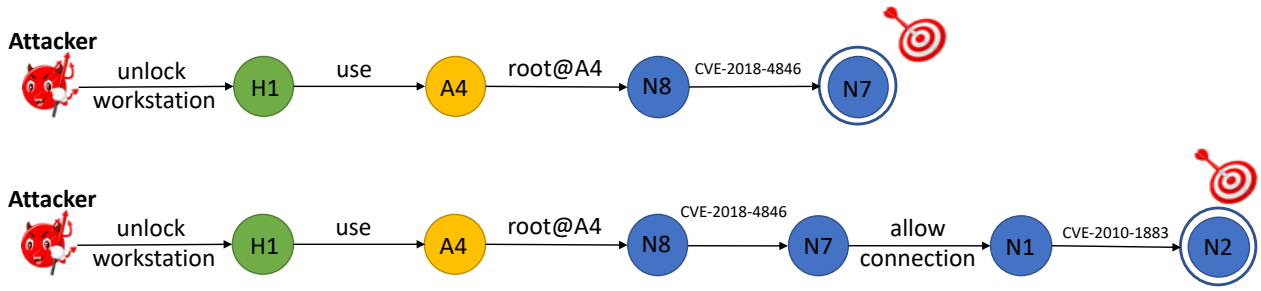


Figure 29. Attack Path Scenario 1 (Path1 and Path 2)

6.4 Section Summary

The main contribution of this section is a methodology for risk quantification based upon the threat model described in Section 5. This methodology contains the following novel aspects:

1. Threat agent modelling: When calculating the risk, in addition to characterising intrinsic complexity of the various steps required to perform an attack towards a target (depending on the difficulty of exploiting the related vulnerabilities), the model also accounts for the specific capabilities of the attacker.
2. Business impact: To add to our multilayer view of the system we introduce the business layer, which accounts for the organisation business processes and their dependency relations on other business process, services and assets of the organisation. Therefore, allowing computation of the risk at the

business level, i.e., the risk associated to the possibility of not being able to meet a specific service level requirement for a given business process.

In the next section we move on to identify possible ways of reducing the human vulnerabilities and risks associated.

7. Behavioural Nudges

As mentioned in Section 4, during the workshop, HC staff were also asked if they had any suggestions for interventions that may encourage people to give up the insecure behaviour in the workplace. Their thoughts are described in the first part of this section (7.1) in relation to the MINDSPACE approach. The second section (7.2), describes identification of potential nudges by experts in behaviour change using IBM and MINDSPACE. For methodology please refer back to Section 4.

7.1 Workshop Results (Part 2): Staff Suggestions for Behavioural Nudges

Staff found it easier to suggest potential interventions for some behaviours more so than others. For example, they found it very difficult to identify interventions to help prevent use of USB devices, due to feeling that the use of such devices is necessary for their daily work. Consequently, the only intervention they suggested was a technological intervention involved designing USB devices to be more secure, for example by using only password-protected, encrypted USB sticks. Likewise, they only identified one potential intervention for encouraging secure sending of patient information, which was again a technological intervention to automatically detect if confidential documents are being attached to an e-mail to prevent sending. It is possible that some situations would be more appropriately addressed by technological changes rather than, or in addition to, behavioural changes – this will be considered when the final nudges are being selected.

Staff found it easier to suggest several potential interventions to promote more secure behaviour in relation to open workstations and insecure password behaviours. However, they also identified that many of these proposed interventions may not be effective, particularly in relation to messenger effects where staff struggled to know which messenger could influence senior members of staff – who were widely perceived to be setting a precedent for insecure behaviour in the workplace (largely due to prioritising patient care, and managing workload, rather than due to any malicious intent). Many staff members reflected upon the need for an overall ‘culture change’ within the working environment, as they felt that security is not the norm within the workplace.

Staff also recognise that environmental constraints make change very difficult without impacting upon workload and potentially upon patient care. For example, the sharing of passwords is driven by a need for staff to complete tasks that are outside of their job role responsibilities. Therefore due to computer access restrictions on their own accounts, they are required to use the accounts of other staff members (e.g., senior doctors, managers) to enable them to access the systems required to do so. This is often a consequence of overworked senior doctors passing responsibilities onto junior doctors and administration staff, to increase the number of patients who can be seen. The doctors within the focus groups reported that they would get “half the work done” if they did not allocate some of their responsibilities to other staff members.

Note: In some countries, junior doctors are not legally permitted to carry out these tasks. Hence why they do not automatically have the same access rights as the senior doctors.

Staff suggestions in keeping with the MINDSPACE approach are summarised in Table 22.

MINDSPACE	Staff Suggestions & Comments
Messenger	• Security messages from IT Staff, peers and/or management/leaders (Note: some staff argued that management only influence insecure behaviour through norms and that positive behaviour messages would be ignored due to no sanctions or enforcement. They also struggled to identify an effective messenger for senior staff, who are largely seen as setting the norm for insecure behaviours in the workplace).
Incentive	• Incentives for secure behaviour • Penalties for insecure behaviour, e.g., financial sanctions (fines) • Regulations, e.g., need to report incidents
Norms	• Addressing norms in the workplace – may require a "culture change"
Defaults/Design	• "Design it out". For example: ⇒ Make passwords and logins easier, e.g., 4-digit pin code for access across all systems ⇒ Card or electronic 'fob' touch in/touch out system to login (similar to that used in bars/restaurants)
Salience	• Reminders about how to behave safely • Frequent reminders about potential security risks • Messages relating to data protection laws & regulations
Priming	• Raised awareness of risk/threat as currently do not understand the risk • Experience or awareness of losing important information and/or financial loss
Affect/Emotion/ Ego	• Increasing feelings of vulnerability • Increasing feelings of control over the situation as some staff reported feeling as if they had no power over security at work if everyone else continues to act insecurely
Commitment	• Some formal commitment
Environment	• Training (as currently none) • Policy made available in all departments, and in an easy to access format • Stronger user policy including new user policy for secure internet use

Table 22. Staff suggestions for behaviour change interventions, based upon the MINDSPACE framework

7.1.1 Nudge identification

The results from the workshops, including the suggestions from staff, were used to aid the expert panel in the development of a comprehensive list of potential behaviour change nudges based on factors from IBM and MINDSPACE. Six categories of nudge types were identified: Awareness & Salience nudges, Norm nudges, Planning nudges, Environmental nudges, Disincentives and Incentives. Each of these are described in more detail below:

1. **Awareness & Salience nudges.** These are nudges which aim to raise awareness and salience of potential risk and/or opportunities to minimise this risk. The following types of awareness and salience nudges were identified:

    - **Patient-focused:** highlighting potential negative impact on patients if breach/attach occurs.
    - **Risk level:** Provide an indication of risk level, this could use visual aids such as a traffic light system displaying red for high risk, green for low risk. E.g., in relation to passwords a strong password could be shown with a green indicator, whereas a weak password would show red.

Risk-related salience nudges can also include relatable information such as “Based on your current password, it would take hackers around XX seconds to break into your account”

- **Responsibility:** e.g., “what happens on your login, is your responsibility”; Display all actions conducted on the staff members computer account that day & require approval. Choose a file which is believed to be very personal or is marked confidential; “Technology alone cannot prevent all cyberattacks/cyber risk. Ensure you are taking responsibility for your own safety”.
- **Privacy:** e.g., “Are you happy for everyone to see this?” Also “technology alone cannot prevent all cyberattacks/cyber risk. Ensure you are taking responsibility for your own safety” – and bite-sized tips on how to achieve this.
- **Data laws and regulations:** e.g., detailing GDPR, & how this could impact staff personally.
- **Complacency:** e.g., “Just because something hasn’t happened yet, doesn’t mean it won’t. Don’t get complacent! XX% of healthcare organisations, like this one, have been affected by some form of cyberattack or data breach”.
- **Policy:** e.g., “Don’t wait until something has happened. Be prepared, familiarise yourself with our cybersecurity policy today”.

Note: nudges raising awareness of risk should also include task-specific, user-centric advice on how to counter or protect against this risk. It is not sufficient to amplify salience of risk, without providing the tools for users to deal with these risks.

2. **Norm nudges.** These are nudges which use social norms to encourage positive (i.e., secure) behaviour. The following types of norm nudges were identified:

   - **Messenger effect:** Poster campaign using showing senior staff acting securely and/or supporting secure behaviour
   - **Norm information:** Alert staff to positive norms in the workplace, i.e., when staff are behaving securely

3. **Planning nudges.** These are nudges which guide the user through self-generation of simple, situation-specific plans to help break insecure habits and promote secure behaviour. This is based upon implementation intentions theory:

   “When people encounter problems in translating their goals into action (e.g., failing to get started, becoming distracted, or falling into bad habits), they may strategically call on automatic processes in an attempt to secure goal attainment. This can be achieved by plans in the form of implementation intentions that link anticipated critical situations to goal-directed responses ("Whenever situation x arises, I will initiate the goal-directed response y!"). Implementation intentions delegate the control of goal-directed responses to anticipated situational cues, which (when actually encountered) elicit these responses automatically.” **[Gollwitzer99]**

   Therefore an example in a HC environment could be: ‘When I leave my workstation for whatever reason, I will lock the screen’, or ‘When I need to use a USB, firstly I will scan it for viruses on a non-networked machine’. This approach also helps to improve users’ perceived sense of control and self-efficacy (predictive factors of behaviour in the IBM).

4. **Environmental nudges.** These are nudges which use changes to the environment to encourage positive (i.e., secure) behaviour. This can include amendments to technology and/or policy. Examples of environmental nudges include:

- Introducing an improved, quick login process (such as touch in/out, and/to single login)
- Changing policy to no longer require staff to change passwords unless compromised or shared.
- Introducing auto-detection of sensitive documents to prevent sending (and/or display a 'nudge' to encourage the sender to consider a more secure method).

These nudges aim to decrease the burden of acting securely.

5. **Disincentives.** These are nudges which use disincentives to discourage negative, insecure behaviour. The following types of disincentive nudges were identified:

   - Loss framing nudge: Playing to users' sense of loss, by alerting them that the files could be corrupted or lost
   - Pop up message every X mins if the user engages in an insecure behaviour (e.g., sends an e-mail attachment, uses a USB).
   - Record number of files sent by e-mail (So staff feel this is being monitored)

6. **Incentives.** These are nudges which use incentives to encourage positive, secure behaviour. The following types of incentive nudges were identified:

   - Have a regular award within the company for appropriate behaviour.
   - Display a message thanking/acknowledging staff for using the approved, secure process.

In keeping with the theoretical framework used in the workshops, the nudges map onto the predictive factors from IBM (e.g., Norms, Personal Agency, Knowledge & Skills, Salience, Environmental Constraints, Habit) and MINDSPACE (e.g., Incentives, Norms, Defaults, Salience, Priming). These nudges can be designed to take into consideration the unique HC working environment; and have the potential to be applied across all of the identified insecure behaviours.

7.2 The Secure Behaviour Nudging Tool (SBNT)

It is important when designing nudges that a 'one size fits all' approach is not taken. Nudges should be designed so that they are specific to the working context in question, and the security behaviours that the organisation is prioritising; rather than attempting to create a general set of nudges covering all behaviours and all situations. Therefore the aim of PANACEA project is to provide HC organisations with the tools and knowledge to continuously evaluate and modify their cybersecurity approach – therefore ensuring that security evolves in line with the current working environment and associated vulnerabilities.

To assist this process, a key outcome of the PANACEA project is the delivery of the Secure Behaviour Nudging Tool (SBNT) – which provides end-users with a *methodology toolkit* for (in)secure behaviour prioritisation, nudge identification, design and evaluation.

Using this approach, key behaviours can be targeted and once cybersecurity levels have improved, the organisations can re-evaluate and design additional nudges to target their updated security priorities.

7.3 Limitations of Behavioural Nudges

It is important to recognise that behavioural nudges may not be the optimum – or only - solution in all situations. In many instances a combination of nudges and/or other solutions may be more effective. For example, some

behaviours may call for a change in policy and/or systems. For example, many of the factors underpinning shared login credentials relate to junior staff being required to undertake tasks on behalf of senior staff, without having the access rights to do so under their own login. In this instance, broadening junior staff access rights– and revising IT policy - could work but may not be appropriate (e.g., if staff are not suitably qualified to conduct the task in question). In this instance, a new system could be introduced that allows junior staff to undertake more 'senior' tasks but also requires - and provides the tools to assist - their seniors to quickly and efficiently monitor these tasks using some form of electronic 'sign off' process. This is an example of a technological intervention rather than a behavioural nudge. However, if the HC organisation decides that junior staff, in line with current policy, should not be completing these tasks and non-sharing of login credentials must be in forced, behavioural nudges may be appropriate. As a further example, the use of USB devices could be tackled in many different ways. Behavioural nudges can be used to encourage more secure use (by raising awareness of risk, nudging staff to scan before use and use only for work, encourage password protection and encryption etc.). However, should the organisation wish to ban the use of such devices, a technological intervention could be much more effective such as deactivating all USB ports. Or if some USB use is essential, using a combination of technological and behavioural interventions by deactivating unnecessary USB ports and using behavioural nudges to encourage secure usage when required.

Other aspects of the PANACEA project are already focused on the design of technological interventions and developments that will aid HC organisations. For example, the Secure Information Sharing Platform (SISP). A similar approach could be taken to addressing the sharing of patient information via unapproved methods, such as smartphone messenger apps. For example, introducing a similar messenger-style system that is approved, encrypted and private to HC staff could help to discourage the use of (and perceived need for) unofficial applications such as WhatsApp.

The decision of whether to apply behaviour nudges, technological interventions, policy changes or a combination, is one that must be decided by the HC organisation in question. This decision should consider their unique working environment, the behaviours concerned, and legal, moral and professional responsibilities.

The SBNT will include information and guidance on assessing nudge suitability, to supplement the nudge identification and design methodology.

7.4 Section Summary

This section has identified a wide range of potential behavioural nudges to influence more secure behaviour. We have also discussed other forms of intervention which may be required, such as technological interventions and/or policy changes. We also summarise the aim of the SBNT which is delivered as part of the PANACEA project and will provide users with the tools required to identify, design and evaluate potential nudges (including assessing their suitability to the situation and issue in question). Alongside the SBNT, the PANACEA project will provide the HC sites with an initial set of behavioural nudges. This allows the HC organisations to test the approach and provides them with time to apply the SBNT to develop their own subsequent nudges. In order to identify the initial set of nudges, the suggestions provided within this deliverable will be evaluated further in D5.2 to assess feasibility, predicted efficacy and appropriateness within the scope of the project. The identified nudges will be developed, with preliminary evaluations taking place in D5.2, and further validation during WP7 Methods of evaluating nudge adoption and compliance will also be considered.

8. Overall Discussion, Conclusions & Next Steps

This document details the human factors research conducted thus far as part of the PANACEA project, and illustrates the importance of recognising the human element of cybersecurity.

Firstly, in Section 4 we described how – and *why* – staff may act insecurely in the workplace and what steps could be taken to help to promote more secure behaviour.

In Sections 5 and 6, we demonstrated how the human layer of cybersecurity can be modelled using a multilayer model to provide a more comprehensive view of cybersecurity position and risk level. Section 5 described how it is possible to represent threats encompassing multiple layers i.e.., human, access, network and business and Section 6 demonstrated how to exploit this model to estimate risks.

Lastly in Section 7, we identified a wide range of behavioural nudges which could be designed to help facilitate more secure behaviour at work. Importantly, we also highlighted how other interventions may be required alongside nudges – such as technological interventions or policy changes. Often a combination of approaches will be most effective. We also introduce the project plan to deliver the SBNT and describe how it will provide end-users with the tools to identify, design and evaluate *appropriate* and *feasible* nudges tailored to their unique working environment.

Next steps relate to D5.2, which will include the evaluation of the possible nudges identified within this document. This will lead to selection of the final nudges that will subsequently be designed and prototypes created, alongside development of the SBNT. Preliminary evaluations of the nudge prototypes will take place in D5.2, with further validation during WP7.

Key findings from D2.2

- Human factors (i.e., behaviour, motivation and attitudes) are a key component of cybersecurity.
- There are many different motivations underlying insecure staff behaviour. The most successful approach to facilitating more secure behaviour is likely to involve a holistic approach combining behavioural, technological and environmental changes; PANACEA project aims to encompass this.
- Multilayer models allow the representation of deep dependencies existing between human behaviours and the related impacts on the security of ICT infrastructure. It is fundamental to analyse organisations' security from a larger perspective including capturing threats, risks and exploitation of complex attack techniques targeting human and machine vulnerabilities.
- Interventions should be able to evolve with changes to the working environment. The SBNT delivered as part of PANACEA project will provide HC organisations with the tools required to achieve this. The SBNT will allow continual assessment, identification, design and evaluation of behavioural nudges, reflective of any changes within the organisation and/or security priorities.

Annex A

IBM: Current Behaviour (Worksheet 1)

ATTITUDE:

- **Experiential Attitude –** How do you feel about the identified risky behaviour? And how do you feel about acting more securely – do you generally feel positively or negatively and why?

  Do you think acting securely at work is beneficial or a hindrance? Do you have mixed feelings about this behaviour – if so, can you explain why? How do you think your feelings affect your behaviour?

- **Instrumental attitude -** What are the costs and benefits of behaving more securely, in relation to this risky behaviour? Are the benefits generally greater than the costs?

  Can you think of any benefits of behaving more securely – for example, do you get rewarded for this behaviour at work? Would acting more securely help or hinder your day to day work?

NORMS:

- **Injunctive Norm -** Do your work colleagues ever give you the impression that they think *you* should carry out this behaviour?
  For example, do they ever expect you – or ask you - to conduct this behaviour?

- **Descriptive Norm –** How do your work colleagues behave at work?
  Do you see your work colleagues conducting the identified risky behaviour at work?

PERCEIVED CONTROL:

- How much control do you feel you have over this risky behaviour (and acting more securely) in the workplace?
  Do you feel it is within your control to act in a more secure manner, or do you think there is anything stopping you from doing so?

SELF-EFFICACY, KNOWLEDGE & SKILLS:

- Are you confident about your ability to behave in a more secure manner? Alternatively, if you are not confident, does this drive the identified risky behaviour?
- What knowledge and skills do you think are needed to enable you to be able to behave more securely, in relation to the identified risky behaviour?

  What training do you currently receive? Do you think this is sufficient? If not, how would you improve upon this? Is there anything else outside formal training, that you feel has (or could) provided you with the knowledge and skills to behave more securely?

SALIENCE:

- What prompts or reminds you to behave securely at work (in relation to the identified behaviour)?

  For example, are there any relevant posters to raise awareness? Alerts on the workstations or by e-mail?

ENVIRONMENTAL AND/OR TECHNOLOGICAL CONSTRAINTS:

- In what ways does your environment create barriers to secure behaviours? In what way does your environment and/or the technology you use encourage the identified risky behaviour?

  This could include your working environment, daily responsibilities and/or the computer systems.

HABIT AND/OR CONVENIENCE:

- Do you think the identified risky behaviour has become habitual at work?

  For example, do you feel this behaviour is part of the normal working culture amongst yourself, or other staff members? Or do you think people engage in this behaviour for convenience?

MINDSPACE: Behaviour Change (Worksheet 2)

MESSENGER: Who do you think would be the most effective person(s) to communicate security information at work, in relation to this behaviour? Who would people listen to or not listen to? Who would people believe?

Are you more likely to listen to information/advice about this behaviour from your manager or your peers? What about governmental advice?

INCENTIVES: What incentives (or sanctions) might encourage secure behaviour, and/or discourage the identified risky behaviour?

Do you think rewards (e.g., verbal praise, bonuses, or another form of positive recognition) would be effective in encouraging secure behaviour? Do you think punishments (e.g., fines, warnings, or other forms of negative recognition) would be effective in discouraging insecure behaviour. Which do you think would work best?

NORMS: How could norms be influenced to encourage more secure behaviour (and/or discourage the identified risky behaviour)? Whose security behaviour influences other peoples' security behaviour?

How do you think changing the social norms in the workplace (e.g., how colleagues are behaving, or how colleagues expect others to behave) could influence this behaviour in others? Can you think of any ways to do this? Who do you think would influence others' behaviour more (e.g., peers, senior staff, IT staff etc)?

DEFAULTS/DESIGN: Where could defaults be used to encourage secure behaviour and develop secure habits?

What defaults do you think should be offered on workstations to encourage more secure behaviour and/or discourage the identified risky behaviour? What other design changes can you think of to encourage more secure behaviour?

SALIENCE: What things could be introduced to increase your awareness of risk in relation to this behaviour? What would make people feel that cybersecurity is an important issue?

Would posters in the workplace help at all? How about alerts via e-mail or on the computer screens? What else do you think could help raise your awareness of the risk related to this behaviour?

PRIMING: What might be used to prime or prompt more secure behaviour? What can be used to keep security in peoples mind?

Can you think of any targeted prompts or security-related questions or alerts that could help while you are using the workstations? E.g., would it be helpful to have a reminder about cyber risk when you log on, or when you perform certain tasks related to this behaviour? Do you think visual indicators of safety would be helpful?

AFFECT/EGO/EMOTION: Is there any way to use emotion to encourage more secure behaviour (and/or discourage the identified risky behaviour)? What would make people feel positive or negative about security behaviours?

Do you think asking staff to reflect upon how they feel about security (or how they would feel if they were the victim of a cyberattack) would be helpful at all?

How could security be encouraged as something that people feel is important to them and their self-image? What would make people feel confident in their security behaviours?

What could encourage staff to feel pride in their secure behaviour? For example, would some kind of award/recognition system be effective? Can you think of any other suggestions?

COMMITMENT: What kind of commitment/agreement to act securely, might be helpful?

Would signing a written agreement to behave according to security guidelines encourage you to act more securely? Can you think of any other steps that staff could take to indicate – and/or encourage - their commitment to acting securely?

ENVIRONMENT: In what ways could your environment be changed to encourage more secure behaviours and/or discourage the identified risky behaviour?

Is there anything about your working environment, daily responsibilities and/or the computer systems at work, which could be changed to make it easier to behave securely (or conversely more difficult to behave insecurely)?

Annex B – Preliminary Human Vulnerability Catalogue

In this appendix we provide our initial catalogue of Human vulnerabilities, including the main vulnerabilities that we will consider in the instantiation of the model and in the design and development of the PANACEA Dynamic Risk Management Platform (DRMP).

The list has been created starting from the analysis of vulnerabilities reported in the Annex D of ISO/IEC 27005 standard.

It is important to note that this list must not be considered exhaustive, but it rather represents a starting point for the human vulnerability classification task. Thus, we believe that it can be further expanded to include other relevant vulnerabilities discovered during the project development.

Attribute	Description
Id	HVUL_01
Name	No 'logout' when leaving the workstation
Description	The person has the attitude to leave unattended his/her device(s) logged in with his/her personal credential
Pre-Conditions	The person has some credential to access a device
Post-Conditions	The attacker can impersonate (temporary or permanently) the human
Access Vector (AV)	Proximity
Attack Complexity (AC)	Low
Identity Impact (II)	High

Attribute	Description
Id	HVUL_02
Name	Disposal or reuse of storage media without proper erasure
Description	The person has the attitude to connect storage media to devices without properly check them
Pre-Conditions	the person has physical access to a device where the memory storage can be plugged in
Post-Conditions	code can be copied and/or executed on the machine according with user's accounts privileges
Access Vector (AV)	Proximity
Attack Complexity (AC)	High
Identity Impact (II)	None

Attribute	Description
Id	HVUL_03
Name	sharing credential
Description	The person has the attitude to share his/her personal credential with others
Pre-Conditions	The person has some credential to access a device
Post-Conditions	The attacker can impersonate the human
Access Vector (AV)	Knowledge

Attack Complexity (AC)	Low
Identity Impact (II)	High

Attribute	Description
Id	HVUL_04
Name	Unprotected credential
Description	The person has the attitude to store his/her personal credentials without taking too much care of their protection
Pre-Conditions	The person has some credential to access a device
Post-Conditions	The attacker can impersonate the human
Access Vector (AV)	Proximity
Attack Complexity (AC)	Low
Identity Impact (II)	High

Attribute	Description
Id	HVUL_05
Name	Poor password management
Description	The person has the attitude to use default or weak passwords or to not update them properly
Pre-Conditions	The person has some credential to access a device
Post-Conditions	The attacker can impersonate the human
Access Vector (AV)	Proximity
Attack Complexity (AC)	High
Identity Impact (II)	High

Attribute	Description
Id	HVUL_06
Name	Insufficient security training on *
Description	The person has not received any training on * or the outcome of the training evaluation are not satisfactory
Pre-Conditions	None
Post-Conditions	The attacker can impersonate the human
Access Vector (AV)	Proximity
Attack Complexity (AC)	High
Identity Impact (II)	High

Attribute	Description
Id	HVUL_07
Name	Incorrect use of software and hardware

Description	The person uses devices or software installed on them improperly. Some examples are: (i) not take care of warning and/or error messages reported, (ii) unsafe click on links or images, (iii) unsafe launching of code, etc…
Pre-Conditions	None
Post-Conditions	The attacker can be able to run arbitrary code with the human privileges or can steal his/her credentials
Access Vector (AV)	Proximity
Attack Complexity (AC)	High
Identity Impact (II)	High

Attribute	Description
Id	HVUL_08
Name	e-mail misuse
Description	The person uses institutional email address for private purposes or to register to services not related to working activities
Pre-Conditions	None
Post-Conditions	The attacker can be able to run arbitrary code with the human privileges or can steal his/her credentials
Access Vector (AV)	Proximity
Attack Complexity (AC)	High
Identity Impact (II)	High

Attribute	Description
Id	HVUL_09
Name	non-compliance with procedures for introducing software into operational systems
Description	the person bypasses policy for installing new software on the devices
Pre-Conditions	The person has privileges that allows software installation
Post-Conditions	malicious software can be installed, and a default zero-day vulnerability can be introduced on the device
Access Vector (AV)	Knowledge
Attack Complexity (AC)	High
Identity Impact (II)	None

Attribute	Description
Id	HVUL_10
Name	non-compliance to policy on mobile computer usage
Description	the person bypasses policy regulating the usage of personal mobile devices
Pre-Conditions	None
Post-Conditions	malicious software can be installed, zero-day vulnerability can be introduced on the device and sensible data can be stolen
Access Vector (AV)	Knowledge

Attack Complexity (AC)	High
Identity Impact (II)	None