



الجمهورية الجزائرية الديمقراطية الشعبية
People's Democratic Republic of Algeria
وزارة التعليم العالي والبحث العلمي
Ministry of Higher Education and Scientific Research
جامعة الشهيد حمة لخضر الوادي
University of Echahid Hama Lakhdar El-Oued
كلية العلوم الدقيقة
Faculty of Exact Sciences
قسم الاعلام الالي
Department of Computer Science



MASTER THESIS

Field : Mathematics and Computer Science

Branch : Computer Science

Specialization : Cybersecurity and Internet of Things

Theme

Gen-Twin: A Generative Cyber-Digital Twin
Platform for Proactive Ransomware
Simulation and Defense Efficacy Testing

Presented by:

- AHMOUDA Yasmine
- MAHCENE Wijden

Discussed in 06 June 2026 by the jury :

- Dr. FERIK Brahim **MCB** Supervisor Univ.El Oued
- Pr. LAOUID Abdelkader **Professeur** Co-supervisor Univ El Oued
- Dr. BEN BRIKA Mohammed Kamel **MCA** President Univ.El Oued
- Dr. GHERBI Kaddour **MAA** Examiner Univ.El Oued

Academic year : 2025/2026

الإهداء

الحمد لله حُبًّا وشُكْرًا وإمْتِنَانًا على البدء والختام
(وآخر دعواهم أن الحمد لله رب العالمين)
بكل حُبٍّ أهدي ثمرة نُجَاحي وتخرُجي :
لِرُوحٍ من بدأ معي الطريق وغادِرنا جسدًا لا روحا
لمن كان اسمه مرافقا لإسمي
طوال سنوات دراستي
لمن دَعَمَنِي بِلا حُدُودٍ ... أبي الحبيب
والى من جعل الله الجنة تحت أقدامها
من أكملت معي المسيرة بقلبها وسَهَلَت لي الشدائدِ بِدُعَائِهَا
فخري وِإِعْتِزَازِي ... والدتي الغالية
الى من شَدَّ الله به عَضْدِي سِنْدِي وعزوتي
وبهم تَكْتَمِلُ فرحتي وبِقُرْبِهِم تَقْوَى
عزيمتي ... أخوتي
الى من شاركتنِي البداياتُ
من جعلت الايام الصَّعبة أخفُّ وأجملُ ،
الى من لم تُفَرِّقْنا المسافاتُ ورفيقة الجامعة
والذكرياتِ ... رفيقاتُ دربي
الى رفيقة السَّهرِ والدراسةِ
التي كانت بِحُضُورِها تَمْنَحُنِي الرَّاحَةَ
وتملأ أيامي طاقَة إيجابِيَّة ... قِطَّتِي
الى من أحاطونِي بِالمحبَّة والدُّعْمِ
وكان لهم الأثر الجميل في رحلتي ... عائلتي الكبيرة
وأخيرا الشُّكْرُ مُوصُولٌ لِنَفْسِي على الصَّبْرِ والعزيمة والإصرارِ
فالحمد لله رب العالمين

أحمودة ياسمين بنت العيد

الإهداء

بعد سنواتٍ من الإجهادِ والسَّعيِّ للمواصلة ،
سنوات جالت في ظلَّ التعبِ بالصَّبْرِ والتَّحدياتِ ،
ها أنا اليومُ أمام ما أرجوه من سنَّين مضت
أقفُ على عتبةِ الفرحِ لأقدِّمُ ثمرةَ جهدي وما وصلت إليه بعد فضلِ الله وتوفيقه .
فهذا الإنجاز لم يكن خطوةً فرديةً ،
بل هي حصيلة دُعْمٍ وحبٍّ ومُساندةٍ أشخاصٍ كان لوجودهم أعظمُ الأثر في مسيرتي .

إلى من منحوني القوةَ لأُصلُ
أهدي هذه الإنجاز :

إلى والديَّ
الذي علَّمني مُعنى الثَّباتِ ، والغرسِ الطَّيِّبِ ، والسَّيرِ في الطَّريقِ الصَّحيحِ .
إلى من كان جُنْدَ الخفاءِ في مسيرتي .
له كلُّ الإمتنانِ والتَّقديرِ في القلبِ الذي يليق بمكانته في حياتي .

إلى والدتي
نبحِ الحنانِ وقلبِ الدُّعاءِ الذي لم يُغلق بابهُ يومًا .
إلى من كانت عُيونها تُلاحقُ خطواتي خوفاً عليَّ ، وحنانها قنديلتي في كلِّ مواجهةٍ .
يا أمان أيامي ويا هدايا الله في عمري ،
تُجّاحي هذا إمتدادٍ لِدُعائك ، وقطفٍ لِثمارِ حُبكِ ، وهديةً مُتواضعةً لِقَلْبِكَ الكبير .
لها أهدي تُجّاحي وفخري .

إلى أخواتي وأخي
سندي الذي لا يميلُ ، ورُفاقِ عالمي الصَّغيرِ ، واليدِ الذي تمتدُّ لي قبل أن أطلبُ .
لِطالما كنتم جزءاً من قوتي ، وسبباً في أن أقفُ بثباتٍ في أصعبِ الأيامِ .
إلَيْكُمْ يا من كنتم نورَ حياتي ، أقدمُ جزءاً من فرحتي

إلى رُوحِ خالتهم
ما يسعني إلا أن أشكركم على وجودكم الذي كان ولا زال المُنْصِبَ لي .
كنتم النورَ لِأيامي ، أملَ رُؤيتكم في أعلى المراتبِ مرضيَّين وراضين

إلى رُفاقِ الرُّحلةِ
أصدقائي الذين شاركوني الطَّريقَ وساهموا في دعواتٍ لا تُنسى .
وجودكم كان مساحَةً رحمةً تليق ، وسنداً لِإنجازِ الصُّعوباتِ .
لكلِّ من وقفَ إلى جانبي وذو أثرٍ طيبٍ في قلبي ، أو دُعْمُ خطوةٍ من خطواتي ،
أو قال لي كلمةً صنعت فرقا ...
لكم جميعاً أجلُّ معاني الإمتنانِ والدعا .

محسن وجدان

وها أنا اليومُ
أرفع مشروعِي هذا غُربونَ شُكْرٍ لِلَّهِ تعالى ، الذي وفَّقني واعانني ،
ومنحني القُدرةَ على الوصولِ إلى هذه المرحلةِ التي لِطالما حلمت بها

Abstract

The increasing sophistication of multi-stage ransomware attacks has rendered traditional point-in-time security validation methods inadequate for ensuring continuous protection, as organizations lack a systematic, automated, and safe methodology to validate whether their deployed security controls can actually detect real-world attack techniques. This research addresses this gap by designing, implementing, and evaluating the Gen-Twin platform, a generative cyber digital twin system that automatically provisions isolated network twins from declarative YAML specifications Network DNA, executes realistic multi-stage ransomware simulations, and quantifies detection gaps using the MITRE ATT&CK framework. The platform adopts a four-layer architecture and was implemented using Vagrant, Ansible, Wazuh, Metasploit, and VirtualBox. A LockBit 3.0 ransomware simulation following a seven-stage kill chain was executed against the deployed twin, and a correlation engine was developed to match attack traces with detection alerts using three criteria: temporal proximity, TTP-based semantic matching, and topological verification. The experimental evaluation demonstrated that the default Wazuh configuration achieved a detection rate of 71.4%, with critical blind spots identified for T1491 ransom note creation and T1059 command execution. Through an automated re-test process that applied two custom detection rules based on the gap analysis, the detection rate improved to 85.7%, representing a 14.3% increase without requiring full environment re-deployment, while the risk level decreased from MEDIUM to LOW. This research contributes a generative cyber digital twin model, a red/blue automated pipeline, a detection gap methodology, and a validated prototype demonstrating measurable security improvement.

Keywords:Digital Twin, Cyber Digital Twin, Ransomware, LockBit 3.0, MITRE ATT&CK, Detection Gap Analysis, Network DNA, Wazuh, Cybersecurity Validation.

Résumé

La sophistication croissante des attaques par ransomware multi-étapes a rendu les méthodes traditionnelles de validation de sécurité ponctuelles inadéquates pour garantir une protection continue, car les organisations ne disposent pas d'une méthodologie systématique, automatisée et sûre pour valider si leurs contrôles de sécurité déployés peuvent réellement détecter les techniques d'attaque du monde réel. Cette recherche comble cette lacune en concevant, mettant en œuvre et évaluant la plateforme Gen-Twin, un système génératif de jumeau numérique cybernétique qui provisionne automatiquement des jumeaux réseau isolés à partir de spécifications YAML déclaratives (Network DNA), exécute des simulations de ransomware multi-étapes réalistes et quantifie les lacunes de détection en utilisant le framework MITRE ATT&CK. La plateforme adopte une architecture à quatre couches et a été implémentée en utilisant Vagrant, Ansible, Wazuh, Metasploit et VirtualBox. Une simulation de ransomware LockBit 3.0 suivant une chaîne de destruction en sept étapes a été exécutée contre le jumeau déployé, et un moteur de corrélation a été développé pour faire correspondre les traces d'attaque avec les alertes de détection en utilisant trois critères : la proximité temporelle, la correspondance sémantique basée sur les TTP, et la vérification topologique. L'évaluation expérimentale a démontré que la configuration par défaut de Wazuh atteignait un taux de détection de 71,4 %, avec des angles morts critiques identifiés pour T1491 création de note de rançon et T1059 exécution de commandes. Grâce à un processus de re-test automatisé qui a appliqué deux règles de détection personnalisées basées sur l'analyse des lacunes, le taux de détection s'est amélioré à 85,7 %, représentant une augmentation de 14,3 % sans nécessiter de redéploiement complet de l'environnement, tandis que le niveau de risque est passé de MOYEN à FAIBLE. Cette recherche contribue à un modèle génératif de jumeau numérique cybernétique, un pipeline automatisé rouge/bleu, une méthodologie d'analyse des lacunes de détection, et un prototype validé démontrant une amélioration mesurable de la sécurité.

Mots-clés : Jumeau Numérique, Jumeau Numérique Cybernétique, Rançongiciel, LockBit 3.0, MITRE ATT&CK, Analyse des Lacunes de Détection, Network DNA, Wazuh, Validation de la Cybersécurité.

الملخص

إن التعقيد المتزايد لهجمات الفدية متعددة المراحل جعل طرق التحقق الأمني التقليدية غير كافية لضمان الحماية المستمرة، حيث تفتقر المؤسسات إلى منهجية آلية وأمنة للتحقق مما إذا كانت ضوابطها الأمنية قادرة فعلياً على اكتشاف تقنيات الهجوم المستخدمة في العالم الحقيقي. يسد هذا البحث هذه الفجوة من خلال تصميم وتنفيذ وتقييم منصة Gen-Twin ، وهي نظام توليدي للتوأم الرقمي السيبراني يقوم تلقائياً بتجهيز توائم شبكية معزولة انطلاقاً من مواصفات YAML التصريحية الخاصة بـ Network DNA، وتنفيذ محاكاة واقعية لهجمات الفدية متعددة المراحل، وتحديد فجوات الاكتشاف باستخدام إطار MITRE ATT&CK. تعتمد المنصة على بنية مكوّنة من أربع طبقات، وتم تنفيذها باستخدام Vagrant و Ansible و Wazuh و VirtualBox و Metasploit. وقد تم تنفيذ محاكاة لهجوم الفدية LockBit 3.0 وفقاً لسلسلة قتل مكونة من سبع مراحل ضد التوأم المنشور، كما تم تطوير محرك ربط لمطابقة آثار الهجوم مع تنبيهات الاكتشاف بالاعتماد على ثلاثة معايير: التقارب الزمني، والمطابقة الدلالية لتقنيات TTPs، والتحقق الطوبولوجي. أظهر التقييم التجريبي أن التكوين الافتراضي لـ Wazuh حقق معدل اكتشاف بلغ 71.4%، مع تحديد نقاط عمياء حرجية مرتبطة بتقنيتي T1059 (تنفيذ الأوامر) و T1491 (تشويه المحتوى). واستناداً إلى تحليل فجوات الاكتشاف، تم تطوير قاعدتي اكتشاف مخصصتين، وبعد اختبار آلية إعادة الإنشاء والتنبيه الخاصة بهجمات الفدية، ارتفع معدل الاكتشاف إلى 85.7%، وهو ما يمثل تحسناً بنسبة 14.3% دون الحاجة إلى إعادة نشر كاملة للبيئة، في حين انخفض مستوى المخاطر من متوسط إلى منخفض. يسهم هذا البحث في تقديم نموذج توليدي للتوأم الرقمي السيبراني، وخط أنابيب آلي للفريقين الأحمر والأزرق، ومنهجية لتحليل فجوات الاكتشاف، بالإضافة إلى نموذج أولي تم التحقق من صحته وأظهر تحسناً أمنياً قابلاً للقياس.

الكلمات المفتاحية: التوأم الرقمي، التوأم الرقمي السيبراني، برامج الفدية، MITRE ATT&CK، LockBit 3.0، تحليل فجوات الاكتشاف، Wazuh، Network DNA، التحقق من الأمن السيبراني.

Dedication	i
Abstract	v
Résumé	vi
contents	x
List of Figures	xi
List of Tables	xii
List of Algorithms	xiii
List of Listings	xiv
Introduction	xv
1 Theoretical Foundations and State of the Art	1
1.1 Introduction	1
1.2 Cybersecurity Fundamentals	1
1.2.1 Threat Landscape Evolution	1
1.2.2 Types of Cyberattacks	2
1.2.3 Security models (CIA, Zero Trust)	3
1.3 Ransomware Attacks	4
1.3.1 Definition and Evolution	4
1.3.2 Ransomware Kill Chain	4
1.3.3 Multi-stage attack behavior	5
1.4 Digital Twin Technology	6
1.4.1 Definition and core concepts	6
1.4.2 Digital Twin vs Digital Model vs Digital Shadow	6
1.4.3 Cyber Digital Twin	7
1.4.4 Differences between physical DT and cyber DT	7
1.5 Attack Emulation and Adversary Simulation	7
1.5.1 MITRE ATT&CK Framework	7

1.5.2	Adversary emulation tools	8
1.5.3	Ransomware modeling and simulation constraints	8
1.6	Defense Monitoring and Detection Engineering	9
1.6.1	Security monitoring systems	9
1.6.2	Detection engineering	10
1.6.3	Detection tools	11
1.7	Conclusion	12
2	Methodology and Gen-Twin System Design	13
2.1	Introduction	13
2.2	Research Methodology	13
2.2.1	Design Science Research (DSR)	13
2.2.2	Research Phases	14
2.2.3	Evaluation Strategy	16
2.3	System Requirements	16
2.4	Gen-Twin System Architecture	17
2.4.1	Global Architecture	18
2.4.2	Orchestration Layer	19
2.4.3	Emulation Layer	19
2.4.4	Instrumentation Layer	19
2.4.5	Analysis Layer	20
2.5	Network DNA Model and Automated Provisioning	20
2.5.1	Digital Network Architecture (DNA)	20
2.5.2	Automated Twin Provisioning	21
2.6	Attack Simulation and Detection Monitoring	22
2.6.1	Attack Simulation Module	22
2.6.2	Detection and Monitoring Module	23
2.7	Correlation and Detection Gap Analysis	24
2.7.1	Correlation Engine	24
2.7.2	Detection Gap Analysis Model	25
2.7.3	Detection Gap Report	26
2.8	Ethical and Safety Considerations	27
2.9	Conclusion	28
3	Implementation, Evaluation and Results	29
3.1	Introduction	29
3.2	Implementation Environment	29
3.3	Gen-Twin Platform Implementation Workflow	30
3.3.1	Network DNA Implementation	31
3.3.2	Twin Deployment	40
3.3.3	Attack Scenario Execution	47
3.4	Detection and Correlation Results	50
3.5	Detection Gap Analysis Results	53
3.5.1	Critical weaknesses	54
3.5.2	Gap metrics	54
3.6	Experimental and Performance Evaluation	56
3.7	Discussion and Security Recommendations	59
3.7.1	Threats to validity	60

3.7.2	Security Recommendations	60
3.8	conclusion	61
Conclusion		63
Bibliography		64

LIST OF FIGURES

1.1	Cyber Kill Chain phases	5
2.1	Design Science Research cycles	14
2.2	Research Phases	15
2.3	Architecture Pipeline of the Gen-Twin Platform	18
2.4	Provisioning Workflow	22
3.1	Canvas of small network	31
3.2	YAML output from "Display DNA" showing node configurations and links	32
3.3	Canvas of medium network	33
3.4	Canvas of enterprise network	35
3.5	Gen-Twin deployment interface showing "Ready to deploy" status	41
3.6	Console showing 5% progress - Generating configuration files	41
3.7	VirtualBox interface showing the running virtual machines generated . . .	43
3.8	output console of generate and distribute SSH key	45
3.9	Console showing successful deployment completion	47
3.10	attack execution	49
3.11	attack execution results	50
3.12	MITRE ATT&CK heatmap showing detected techniques (green) and missed techniques (red)	52
3.13	Detection Gap Report - Undetected TTPs	54
3.14	Gen-Twin interface showing detection metrics	55
3.15	Gen-Twin interface showing Re-test execution and results	57

LIST OF TABLES

1.1	Comparison between Physical Digital Twin and Cyber Digital Twin	7
1.2	Mapping Ransomware Attack Phases to MITRE ATT&CK	9
2.1	Functional Requirements of the Gen-Twin Platform	17
2.2	Non-Functional Requirements of the Gen-Twin Platform	17
3.1	Hardware and Software Specifications of the Evaluation Workstations . . .	30
3.2	Network DNA Validation Steps	40
3.3	Generated Deployment and Configuration Files	42
3.4	Deployment time for 3-VM network configuration	46
3.5	Attack Phases	48
3.6	Execution Results of the Simulated Ransomware Attack	50
3.7	Detection Results of Simulated Ransomware Activities	51
3.8	Detection Coverage by MITRE ATT&CK Technique	52
3.9	Correlation Between Executed Attacks and Wazuh Detection Alerts	52
3.10	Detection Gap Analysis Based on MITRE ATT&CK Techniques	53
3.11	Description of MITRE ATT&CK Techniques Used in the Simulation . . .	54
3.12	Gap Metrics Summary of the Detection Analysis	56
3.13	Detection Performance Before and After Re-testing	57
3.14	Advanced Results	57
3.15	Comparative Analysis (Baseline vs Advanced)	58
3.16	Deployment Time Breakdown	58
3.17	Resource Usage Metrics	59
3.18	Scalability Projections	59
3.19	Summary of Key Findings from the Gen-Twin Evaluation	59
3.20	Recommendations for Improving the Gen-Twin Platform	61
3.21	Operational Policies for Continuous Security Improvement	61

LIST OF ALGORITHMS

1	Find Virtual Machine by Role	38
2	Validation and Automatic Correction of Duplicate IP Addresses	39
3	Generation of the Ansible Inventory File	40
4	Generation of Deployment Configuration Files	42
5	Starting Virtual Machines using Vagrant	42
6	Fallback Installation of Security Tools via VirtualBox	44
7	SSH Key Distribution Across Virtual Machines	45
8	Deployment Verification of Virtual Machines	46
9	Metasploit Attack Execution with MITRE ATT&CK Correlation	49
10	Collection and Parsing of Wazuh Security Alerts	51
11	Detection Rate Computation	55

3.1	full YAML structure of Small Network	32
3.2	Advanced Network DNA Configuration	33
3.3	Enterprise-Scale Network DNA Configuration	35
3.4	Ansible playbook for Wazuh agent and Metasploit installation	43

The evolution of Ransomware attacks has drastically changed cyber security; moving from simple encrypting/extorting to far more complex attacks that consist of many sequential stages and utilize numerous different attack methods or vectors. Microsoft Digital Defense Report 2025 shows that nearly 50% of all cybercrime is financially driven by Ransomware/Extortion. Today's variants of ransomware use sophisticated techniques designed to avoid detection and to avoid being stopped using means such as turning off security monitoring software, deleting backups, and using double extortion tactics against victims. Because of these changes in how ransomware works, traditional methods of ensuring protection against cyberthreats using point-in-time security validation methods are insufficient for providing continuous protection from cyberthreats.

Organizations are investing heavily in security monitoring tools such as SIEM, EDR, and XDR systems; however, they have a major limitation: no one has a systematic, automated, and safe way to continuously assess if their current controls can detect an actual attack using valid methods of attack. The gap between the control being deployed and the ability to use those controls for appropriate detection of attacks creates a significant communication gap, resulting in security teams having no understanding of their ability to defend against a threat until an actual incident occurs.

The central problem addressed by this research is the disconnect between deployed security controls and actual detection capability against multi-stage ransomware attacks. Organizations invest large resources in Security Information and Event Management systems, Endpoint Detection & Response platforms. They configure alerting rules, install agents on endpoints, and claim to comply with current agreed-upon security frameworks. However, they cannot provide even basic information on whether or not these systems are able to detect advanced forms of ransomware. In the event that an advanced form of Ransomware Attack were to occur today against your Infrastructure we would like to know what actions taken during the attack would be identified and what actions would remain undetected? And when will an Alert be generated after the time of execution of an advanced form of ransomware?

This problem manifests through three specific deficiencies:

1. Because there are very few real-world environments available for carrying out at-

tack simulations, many organizations build isolated laboratory networks. However, building such environments is a long, complicated, and unpredictable process that is hard to replicate consistently provision and building virtual machines, create and configure network topologies, and deploy monitoring sensors and establish baseline fidelity to production environments, organizations require days or weeks of time and testing for each of their attack simulation test cycle.

2. Second, existing solutions treat attack simulation and defense validation as disjoint activities. Red team exercises generate attack traces documenting adversary actions, while blue team monitoring generates detection alerts from security tools. There is no unified platform capable of associating the two data streams through automation to identify the actual detection gaps within a network. Security analysts must manually compare time stamps, logs, and alerts in a method that does not scale or easily support continuous validation and is prone to human error.

3. There is no standardized methodology for quantifying detection gaps based on adversarial actions. Organizations lack metrics that express detection coverage relative to the MITRE ATT&CK framework, which is considered the standard reference for documenting adversarial methods of attack. Therefore, without quantifying such gaps in detection, security teams would be unable to take into account the level of risk associated with an incident to prioritize their remediation and would also be unable to measure their success over time when remediating previous incidents.

Based on the systematic literature review presented in Chapter 1, three critical research gaps were identified:

Gap 1: Absence of Generative Digital Twin Provisioning for Cybersecurity

Digital twin technology has been effectively used in the areas of manufacturing, IoT in industry and infrastructure management to allow the monitoring, predicting and optimization of physical systems. However, the implementation of digital twin technology for cybersecurity is still in the early stages. Current implementations of cyber digital twin technology have been designed for the modeling of attacks with little use of generative-based creation through the automatic creation of complete isolated network twins from defined specifications without any human involvement. This lack of documented procedures limits many companies' ability to validate their systems security continuously and provide on-demand validation at scale.

Gap 2: Weak Integration Between Attack Simulation and Defense Validation

The tools available today, tend to operate in silos based on their function. Examples include MITRE Caldera, Metasploit, Cobalt Strike, and Atomic Red Team which provide frameworks to perform adversary emulation, meaning you can carry out attacks in a realistic manner. On the other hand, Wazuh, Splunk and Elastic Stack are security monitoring platforms that alert you to endpoint telemetry but do not provide a way to automatically link the alerts with the specific attacks that they correspond with. Connecting red team actions to blue team alerts is done manually and after the red team has

already completed its exercise, requiring a substantial amount of time and knowledge on the part of the analyst involved. No tool or platform exists that allows for simultaneously conducting emulated attacks and running tests to ensure that you can detect all of the steps in your attacks using the same platform.

Gap 3: Lack of Automated Correlation and Detection Gap Analysis

While there is a standard way to categorize what attackers have done using the MITRE ATT&CK framework, most of the research on this topic has focused on extracting TTPs from irregular (free-text) threat intelligence reports. However, there currently are no known automated correlation engines mapping attack traces to detection alerts using both multi-dimensional matches of the three correlation criteria: time proximity, similar TTPs, and topological structure, and each step is identified as detected or not detected. Lastly, none of the current solutions provide quantitative detection gap reporting, including Detection Rate, Weighted Coverage Score, and Mean Alert Latency. Additionally, they do not give remediation strategies based on the risk that refer back to a specific technical classification within the MITRE ATT&CK framework.

This research aims to fill the identified research gaps by pursuing the following four strategic objectives of this research:

- **Objective 1 - Design the Gen-Twin Platform** Create an architecture for a Generative Cyber-Digital Twin platform that will contain all functions of attack simulation, security monitoring, and detection gap analysis within a fully automated process flow. This architecture will be designed with multiple layers (Orchestration, Emulation, Instrumentation, Analysis) to separate responsibilities.
- **Objective 2 - Model Network DNA (Digital Network Architecture)** Develop a human-readable/declarative and machine-readable format (Network DNA) to represent target networks that will contain definitions for basic elements such as given functions (node) by Role target, attacker, and monitor, definition of links, definition of Application Layer Services, and definition of Access Control. The Network DNA will be used as the only input for the automated provisioning of digital twins representations of the physical system.
- **Objective 3: Automate Multi-Stage Ransomware Simulations:** Automate real-world attacks using the seven stages of a LockBit 3.0 attack as defined within LockBit's KILL CHAIN (i.e., Reconnaissance, Discovery, Initial Access, Lateral Movement, Privilege Escalation, Defense Evasion, Encryption and Extortion). This attack will be mapped against the applicable techniques defined in MITRE ATT&CK to allow for automated correlation.

The main contributions in this work are:

1. Contribution 1: Generative Cyber Digital Twin Model. A novel declarative approach (Network DNA) for automatically provisioning isolated, realistic network twins from YAML-based specifications. Unlike existing digital twin implementations that require manual configuration, this model enables complete automation

from specification to running environment, eliminating manual setup and enabling continuous, on-demand security validation.

2. Contribution 2: Red/Blue Automated Pipeline. A single-system pipeline that will run both red team adversary emulation and collect blue team security alerts in parallel in the same twin environment. This provides a high degree of temporal and topological correlation between the attack and detection traces, which will create a definitive correlation that would not exist if these activities were to be performed independently.
3. Contribution 3: Detection Gap Methodology. In order to quantify the gaps in detection using the MITRE ATT&CK Framework's techniques, which provide a shared language for the attacker and defender, there is a systematic methodology. The methodology will correlate between three dimensions: Time, Technique, and Topology, and provide security teams with quantitative metrics: Detection Rate, Weighted Coverage Score, and Mean Alert Latency, that allow them to assess their coverage, prioritize remediation, and measure progress on improvements to the process over time.
4. Contribution 4: Prototype Implementation and Empirical Validation. An effective proof-of-concept for the Gen-Twin platform has been created using Vagrant, Ansible, Wazuh, Metasploit and Oracle VirtualBox. To validate this prototype, two types of ransomware scenarios for baseline and advanced coverage have been run to determine the extent to which automated remediation improves the detection rate of ransomware from 71.4% to 85.7%, which is a total improvement of 14.3% without requiring a complete redeployment of the environment.

The rest of this dissertation is organized into three chapters, following the Design Science Research methodology. Chapter 1 shows the theoretical foundations and state of the art. This chapter builds the essential background to understand this research by covering the essentials of cybersecurity, the evolution of ransomware, the Cyber Kill Chain, multi-stage behavior of Common Ransomware Attacks, Digital Twin Technology, Attack Emulation and Adversary Simulation, and the use of MITRE ATT&CK Framework; The Mechanisms for Monitoring Defense, Identify the 3 Research areas that this Research fills by Closing the Chapter.

Chapter 2 illustrates the methodology and Gen-Twin system Design. In this chapter, we present the research methodology and the complete conceptual design of the Gen-Twin platform. It covers the adoption of Design Science Research, functional and non-functional requirements, the four-layer architecture, the Network DNA declarative model, automated twin provisioning using Infrastructure as Code, the attack simulation module with ATT&CK mapping, and the correlation engine and detection gap analysis model. The chapter concludes with discussions on ethical and safety issues.

Chapter 3 shows the implementation, evaluation, and the obtained results. This chapter provides the complete implementation details and empirical evaluation of the Gen-Twin platform. It covers the implementation environment, Network DNA instantiation with YAML examples, the five-stage deployment pipeline, a LockBit 3.0 ransomware simulation, detection and correlation results, detection gap analysis using ATT&CK heatmap methodology, experimental evaluation including baseline and advanced scenarios with

automated re-test, performance evaluation, security recommendations, and future work directions.

CHAPTER 1

THEORETICAL FOUNDATIONS AND STATE OF THE ART

1.1 Introduction

This chapter is considered a fundamental part of this research, aiming to establish the necessary background for understanding the main topics of this study.

The evolution of the field has exposed critical limitations in traditional methods. Unlike conventional approaches that are point-in-time and labor-intensive, modern frameworks require continuous, automated, and realistic validation.

In this chapter, we address the general concepts and fundamental principles of the main topics. Additionally, we review the most prominent frameworks, tools, and models used in this field. This chapter also identifies the research gaps that motivate this study.

1.2 Cybersecurity Fundamentals

Cybersecurity fundamentals provide the conceptual basis for understanding how digital systems are protected against evolving threats. Before examining specific attack categories and defense models, it is necessary to clarify the broader security context in which modern organizations operate. This section therefore introduces the changing threat landscape, common forms of cyberattacks, and foundational security models that guide defensive strategies today.

1.2.1 Threat Landscape Evolution

Recent developments in the cyber threat landscape have precipitated a paradigm shift, wherein adversaries have transitioned from traditional intrusion attempts to leveraging compromised identities for unauthorized access, thereby establishing identity as the primary and final line of defense. Over fifty percent of cyberattacks with documented motivations were financially motivated, including extortion and ransomware, whereas espionage represented a negligible proportion, accounting for no more than four percent. Conversely, data theft has emerged as the predominant objective in the overwhelming majority of incidents.

Emergent attack methodologies, exemplified by "ClickFix," have transcended conventional phishing strategies, accounting for approximately fifty percent of identified security breaches. Furthermore, artificial intelligence has substantially enhanced the efficacy of cyberattacks. AI-driven phishing campaigns, for example, have click-through rates that are more than four times higher than those of conventional techniques and can potentially yield attacker profits that are up to fifty times greater.

In terms of defence, multi-factor authentication (MFA) has proven highly effective, successfully preventing over 99% of unauthorised access attempts. Contemporary defense forces confront an unprecedented volume of daily threats, processing hundreds of trillions of security signals, blocking millions of malicious files, and analyzing tens of millions of identity risks, all within a relentless pursuit to remain aligned with the rapidly evolving tactics of adversaries.. [1].

1.2.2 Types of Cyberattacks

There are various types of cyber attacks that employ a distinct set of tactics and concentrate on specific vulnerabilities. Below are the main types of cyber attacks [2]:

1.2.2.1 Hacking

Unauthorized access, control, or manipulation of digital systems, networks, or data often with malicious intent is referred to as hacking . To achieve their objectives, both individuals and organizations exploit vulnerabilities in hardware, software, or human behavior.

1.2.2.2 Viruses and Malware

Malicious software applications designed to compromise, interfere with, or harm computer systems are collectively referred to as viruses and malware. Malware encompasses a spectrum of malicious software, including ransomware, spyware, worms, and Trojan horses, each characterized by distinct methodologies and objectives, whereas viruses are defined by their ability to replicate and disseminate through infected files.

1.2.2.3 Targeted Attacks

Targeted attacks - more properly called deliberate cyber operations are attacks specifically aimed at particular people, organizations or important systems. These attacks are done to achieve certain goals. These goals might be getting hold of confidential data, interfering with how a company does its work, or taking someone's original ideas and inventions.

1.2.2.4 Electronic Espionage

Electronic espionage involves illegally obtaining your personal information by either governments using their resources or highly skilled and dedicated hackers (referred to as Advanced Persistent Threats or APTs). They're after this private stuff to further their own plans for things like power, money, or political advantage.

1.2.3 Security models (CIA, Zero Trust)

Modern cybersecurity frameworks are built upon fundamental security models that define how systems should protect data, control access, and ensure trust, notably the CIA triad and Zero Trust architecture.

1.2.3.1 Zero Trust

Zero Trust is a security model to overcome cybersecurity challenges where it does not automatically trust anyone just because they claim a certain identity, but rather checks everything when users use computers to communicate with each other.

The "Zero Trust" model was developed by combining several elements. These include ensuring that users are indeed who they claim to be and preventing them from navigating within the computer network. This is done by segmenting the network, which also uses a rule that states that people can only access the information they actually need to do their work, called a premium access policy [3].

Guiding Principles

- Do not think that a company's private network is automatically safe. Assets should work as if they are in danger.
- All communications need connections that check who you are and protect the data.
- Devices not owned by the company, like visitor devices or personal devices, need to be handled with care to keep the network safe.
- Every asset has to be checked at a point before and during access.
- Company devices are safer than devices not owned by the company.
- All access and cloud communications must be checked, allowed, and protected.
- Security rules must stay the same when assets move between the company and company systems.

1.2.3.2 CIA

The main goals of information security are to keep information secret make sure it is accurate and have it when we need it[3]. This is what the CIA triad of information is about: [3]

- **Confidentiality:** We need to keep data from people, organizations or computer programs that are not supposed to see the data protected at all times. For example we did not let our staff look at data unless they had the right to do so.
- **Integrity:**the integrity of the data was not altered or destroyed accidentally. An integrity check, for instance, confirmed that no tampering had occurred.

- **Availability:** Having data available is really important for businesses to run properly, it helps to ensure that a server is working correctly for people who are allowed to use it.

For instance we need to make sure that the server is up and running for users
This way business operations can continue without any issues.

1.3 Ransomware Attacks

Ransomware represents one of the most disruptive forms of cybercrime because it directly affects data availability, business continuity, and organizational trust. Before analyzing its technical stages and simulation requirements, this section introduces its definition, historical evolution, kill chain behavior, and multi-stage operational structure within modern cyberattacks.

1.3.1 Definition and Evolution

Ransomware is a type of malware that prevents people from accessing their information or using their computer systems. This blackmail continues until they pay money to make things work again by exploiting the urgent need and potential harm to the victim organization[4].

It all started in 1989 with a mail-in scam. This scam contains a disc containing software called Cyborg AIDS Trojan Ransomware, on the basis that it is an AIDS survey. But in reality, he had a program that locked all user data. To open it, people had to send money to an address in Panama.

Ransomware became much worse when strong encryption and digital money like Bitcoin emerged. There are some examples of ransomware, such as CryptoLocker from 2013 and WannaCry from 2017. These studies showed that encrypting large amounts of files at once was a way for bad people to get what they wanted. Then there was the Maze ransomware. That's when bad people started locking files and also threatening to put in stolen data. This is called the double blackmail model. This meant that simply paying to open the files was not enough. He also gave people a lot of power to hurt people's reputations.

At that time, what is called ransomware as a service started happening. This happens when people who make ransomware sell their software to people. They can then use software to lock files and get money. People who make ransomware get a portion of the money. This made it super easy for new bad people to get into it. [4]

1.3.2 Ransomware Kill Chain

The Cyber Kill Chain has 7 phases when it comes to a malware attack [5]:

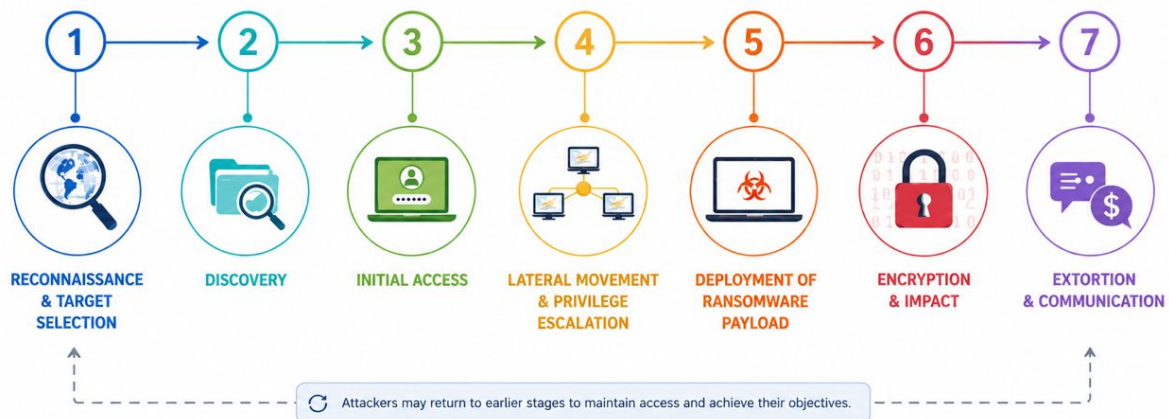


Figure 1.1: Cyber Kill Chain phases

1. **Reconnaissance and target selection:** This involves threat actors selecting organisations to target.
2. **Discovery:** This is where threat actors collect information or data about victim's systems.
3. **Initial Access:** is the critical stage at which threat actors attempt to establish their initial foothold within an organization.
4. **Lateral movement and privilege escalation:** When threat actors get inside the organization they start to move and get control of parts of the network.
5. **Deployment of ransomware payload:** The key question is where exactly they will choose to deploy the ransomware payload.
6. **Encryption and Impact:** which involves locking the victims files and causing harm to their systems
7. **Extortion and Communication:** where threat actors talk to their victims. Start to extort them.

1.3.3 Multi-stage attack behavior

Ransomware consists of multiple stages with the goal of maximizing impact while ensuring that its actions remain unnoticed. At each stage of the process, there is an opportunity to detect and respond to them. Here is an example of how the ransomware infection process works [5]:

1. **Deploy initial infection vectors:** is the first step to accessing a victim's computer systems by launching attacks, such as phishing emails or exploiting vulnerabilities.
2. **Gain footholds within victim networks:** This is done by gaining a position within victim networks.

3. **Move laterally across the network:**Next, you go from one part of the network to another.
4. **Escalating privileges:**While on the move, the attacker tries to gain power and control over the network
5. **Conducting reconnaissance to identify valuable assets:** While in the network, he tries to figure out what's important.
6. **Encrypting or exfiltrating critical data:**Finally, the attacker takes the data and either locks it up so nobody else can get to it .

1.4 Digital Twin Technology

Digital twin technology provides a foundation for creating virtual representations of real or intended systems that can be observed, tested, and improved safely. In cybersecurity, this concept enables controlled experimentation without affecting production environments. This section introduces digital twin principles, related concepts, and their adaptation to cyber defense contexts.

1.4.1 Definition and core concepts

Digital twin: a dynamic, high-fidelity virtual representation of the physical system with real-time data that facilitates monitoring, analyzing prediction and optimization. This involves synchronization, bidirectional data exchange and decision making from intelligent analytics to help the users for better lifecycle management of any complex hardware or software system. [6]

1.4.2 Digital Twin vs Digital Model vs Digital Shadow

- **Digital Twin:** If the data flows between an existing physical object and a digital object are fully integrated in both directions, one might refer to it as Digital Twin. The digital object can serve as a control instance for the physical object. A change in the state of the physical object is a direct reflection of a change in the state of the digital object, and vice versa. [7]
- **Digital Model:** A digital model is a physical object that can be represented digitally without using any automated data exchange between it and the digital model. The digital representation might include a more or less comprehensive description of the physical object. These models might include simulation models of planned factories, mathematical models of new products, or any other models of a physical object. Digital data of existing physical systems might still be in use for the development of such models, but all data exchange is done in a manual way. A change in state of the physical object has no direct effect on the digital object and vice versa. [7]
- **Digital Shadow:** Based on the definition of a Digital Model, if there exists an automated one-way data flow between the state of an existing physical object and a digital object, one might refer to such a combination as Digital Shadow. A change

in state of the physical object leads to a change of state in the digital object, but not vice versa.[7]

1.4.3 Cyber Digital Twin

The Cyber Digital Twin (CDT) is a digital twin that supports cybersecurity operations monitoring, detection, investigation, and response by providing the capabilities to model and predict the evolution of cyber attacks and the risks of potential threats becoming reality in the current or virtual context in a proactive manner without interfering with the system. Unlike traditional digital twins of physical systems, the cyber digital twin (CDT) focuses on modeling attack strategies and deals with multi-ownership environments such as digital service chains (DSCs).[8]

1.4.4 Differences between physical DT and cyber DT

The table below summarizes the key differences between Physical Digital Twins and Cyber Digital Twins (CDTs).[8]

Criteria	Physical DT (plain DT)	Cyber DT (CDT)
Modeled object	Business logic, physical processes	Attack strategies, hacker movements
Main usage	Detecting single attacks, resistance testing, cyber-ranges	Hunting, prediction, investigation, response
Environment	Single administrative domain	Multi-ownership (Digital Service Chains)
Data flow	Mostly monitoring	Bidirectional (monitoring + enforcement)
Prediction	System state	Evolution of cyber-attacks

Table 1.1: Comparison between Physical Digital Twin and Cyber Digital Twin

1.5 Attack Emulation and Adversary Simulation

Modeling attacker behavior over the course of an attack lifecycle to simulate and provide a comprehensive understanding of cyber threats such as ransomware requires structured adversary emulation frameworks and tools.

1.5.1 MITRE ATT&CK Framework

The MITRE ATT&CK framework is based on the concept of TTPs (Tactics, Techniques, and Procedures), which describe adversary objectives, attack methods, and their practical implementation during cyber operations. [9]

- **Tactics:** are the reason why an ATT&CK technique or sub-technique is used, representing the adversary's tactical objective. The concept of tactics is a useful way

to categorize individual techniques and include standard notations for adversaries' actions during an operation, such as persisting, discovering information, moving laterally, execution, and exfiltrating data.

- **Techniques:** Techniques describe the methods or actions adversaries use to achieve a specific tactical objective. Each technique represents a particular way an attacker can perform malicious activities within a system or network. In the MITRE ATT&CK framework, techniques may also include sub-techniques that provide a more detailed description of adversarial behavior. Examples of techniques include phishing, command-line execution, credential dumping, and lateral movement.
- **Procedures:** Talking about tactics and techniques is incomplete without mentioning procedures, as they are an essential component of the TTP concept. Procedures are the implementation techniques or sub-techniques that adversaries have employed in ATT&CK. A procedure could be using PowerShell to inject into lsass.exe to dump credentials by scraping LSASS memory on a victim. The two important aspects to note about procedures in ATT&CK are that it is how an adversary uses techniques and sub-techniques and that a procedure can span multiple techniques and sub-techniques. Expanding on the prior example, the procedure includes PowerShell, Process Injection, and LSASS Memory, which are distinct behaviors. Procedures are documented in ATT&CK as observed use of techniques in the "Procedure Examples" section of the technique and sub-technique pages.

1.5.2 Adversary emulation tools

Several adversary emulation tools have been developed to simulate real-world cyberattacks and evaluate organizational security capabilities.

Among the most widely adopted tools are: [10]

- **MITRE Caldera:** is an open-source adversary emulation tool, offers a wide array of functionalities to simulate adversarial behaviors. It allows security professionals to determine, execute, and analyze adversary scenarios, which makes it a valuable resource for red teaming and threat simulation.
- **Cobalt Strike:** is a widely used commercial tool for adversary emulation and red teaming exercises. It offers various features, such as beaconing, payload generation, and interactive post-exploitation capabilities. It is a valuable asset for security professionals due to its adaptability and flexibility.
- **Atomic Red Team:** The goal of this open-source project, initiated by Red Canary, is to create a collection of simple and well-defined atomic tests that duplicate specific adversary behaviors. These tests can be conducted to evaluate an organization's ability to detect and respond to known adversary tactics.

1.5.3 Ransomware modeling and simulation constraints

Ransomware attacks can be systematically modeled by mapping each phase of the attack lifecycle to corresponding techniques and tactics defined in the MITRE ATT&CK framework.

1.5.3.1 Mapping ransomware to ATT&CK

The process of mapping ransomware to ATT&CK involves aligning ransomware attack stages with corresponding ATT&CK tactics and techniques to enhance detection, analysis, and mitigation strategies.

Attack Phase	ATT&CK Tactic	Technique
Phishing	Initial Access	Spear Phishing Attachment
Exploit (SMB)	Initial Access	Exploit Public-Facing Application
Execution Malware	Execution	PowerShell / Command-Line
Privilege Escalation	Privilege Escalation	Exploitation for Privilege
Propagation	Lateral Movement	SMB / Remote Services
Communication	Command & Control	Web Protocols
Encryption	Impact	Data Encrypted for Impact

Table 1.2: Mapping Ransomware Attack Phases to MITRE ATT&CK

1.6 Defense Monitoring and Detection Engineering

In modern cybersecurity architectures, effective defense is achieved through continuous monitoring and detection engineering techniques that transform raw security data into actionable intelligence for threat identification and response.

1.6.1 Security monitoring systems

Cybersecurity monitoring systems play a critical role in modern defense strategies by continuously collecting, analyzing, and correlating security events to detect, investigate, and respond to cyber threats in real time.

The most widely used security monitoring systems are: [11]

- **SIEM (Security Information and Event Management):** Security Information and Event Management systems, or SIEMs, they try to find patterns and things that are out of the ordinary that could mean someone is trying to breach the organization's information security.

SIEMs have a lot of work to do. It is not easy, especially when they have to look at a lot of data to find useful information. This is a job that needs new and advanced methods and technologies not just the old ways of doing things. That is where Artificial Intelligence and Machine Learning come in. They make it possible to analyze things accurately and quickly.

This means SIEMs can watch for threats better, find them faster, and respond to them effectively. It also helps the organization come up with security plans. Security Information and Event Management systems do this by using Artificial Intelligence and Machine Learning to make security better.

- **XDR (Extended Detection and Response):** XDR is a set of security tools that looks at cybersecurity in a comprehensive way. It helps find look into and respond to threats from one place across different security areas.

This system can. Link data from different places, like endpoints, networks, servers, cloud workloads, email

Traditional security measures usually work alone. XDR tries to connect them by combining data from sources into one easy-to-use analytical framework.

The goal of XDR is to make it easier to understand and deal with security threats by using XDR to analyze data, and you can also use it to respond to threats in an effective way.

- **EDR(Endpoint Detection and Response):** Endpoint Detection and Response solutions are really important for cybersecurity. They help with the problem of endpoints being vulnerable to attacks. These endpoints are like points that hackers like to target.

Endpoint Detection and Response solutions are made to watch endpoints all the time and find any threats. They are designed to respond to any suspicious activity on the Endpoint Detection and Response systems. This helps keep Endpoint Detection and Response systems from hackers. Endpoint Detection and Response solutions are very good, at finding and stopping threats to Endpoint Detection and Response systems.

1.6.2 Detection engineering

To enhance the effectiveness of security monitoring systems, detection engineering provides structured approaches for identifying malicious behavior through both signature-based and behavioral analysis techniques.

1.6.2.1 Signature-based Detection

Signature based Intrusion Detection Systems, or IDS, have played a big role in cybersecurity for years. They work by checking network traffic or system activity against a database of known attack patterns. When they spot something that matches one of these patterns exactly, they trigger an alert. This method does a great job catching threats that security teams already know about and understand well[12].

1.6.2.2 Behavioral Detection

Behavior-based works differently from traditional methods. It pulls data from all over network flows, system logs, authentication records and connects the dots to spot slow shifts and sneaky abuses. This way, organizations can catch new threats early, even if they don't trigger obvious alarms or stand out in the numbers, because they stray from what's normal behavior.[12]

1.6.2.3 Signature vs Behavioral Detection

Signature-based tools mostly look for outside attacks, but behavior-based systems do a better job spotting insider threats. They notice when employees do odd things, like accessing data they shouldn't or acting strangely on the system stuff signature methods just don't catch.[12]

1.6.3 Detection tools

Detection tools form the base for any security monitoring system because they convert threatening information into usable alerts that enable the security software to identify malicious behavior from ordinary behavior. These are some common types of detection systems that can be used for detecting ransomware and other threats [12]:

1. **Sigma**

A generic signature template gives security analysts a simple, flexible way to describe security logs for their organization. We want to keep things straightforward and organized, so the whole team can write clear, useful descriptions that really fit what our organization needs.

2. **YARA**

YARA rules are great for catching suspicious files or weird network activity basically the tricks attackers use for things like malware or ransomware. These rules dig into patterns, either in files or in how data moves around, and flag anything that looks dangerous.

3. **Custom Rules**

Custom detection rules take those general Sigma or YARA signatures and adapt them for specific platforms, like turning them into JSON for CrowdStrike or SPL for Splunk. Analysts can pick up these rules from places like SigmaHQ or SOC Prime, or just use AI tools like ChatGPT to make new ones. No matter where you get the rules from, always test them in a safe spot first. Otherwise, when you roll them out, you'll just end up drowning in false alerts.

1.6.3.1 Example Platform (Wazuh)

Wazuh is a free security platform that helps you find threats and keep an eye on your systems away.

Let us see how Wazuh can be useful for you [13]:

- **Security analytics:** The product collects, normalizes and analyzes different types of security data which can help prevent attacks, detect threats and unusual system behavior.
- **Intrusion detection:** Wazuh agents monitor their targets and search for malicious activities, potential rootkits and suspicious items like hidden processes or files.
- **Log data analysis:** The agents analyze log data from OS and applications and then securely transmit them to the server. Logs are evaluated according to predefined rules.

- **File integrity monitoring:** The solution checks the status of file system. If there are changes in file properties including its content, permissions and ownership, then you will learn about it.
- **Vulnerability detection:** Agents collect information regarding software installed on target systems and upload it to the server. In turn, the server analyses the list of software using databases with information about CVE vulnerabilities.
- **Configuration assessment:** With Wazuh, you may evaluate system configuration and application settings for compliance with your security policies. The agents periodically scan the system and applications for outdated packages, missing patches and improper configuration.

Wazuh Components The working of Wazuh depends upon the combination of three key components [13]:

- **Wazuh agent:** This agent is installed on your machines such as desktops, laptops, and servers or even virtual machines. It helps in managing various tasks related to prevention, detection, and response, not only for Windows but also Linux, MacOS, Solaris, HP-UX, and AIX.
- **Wazuh server:** This receives all the data sent by your agents. This data is processed via decoders, rules, and threat intelligence in search of any kind of breach.
- **Elastic Stack:** This is an open source tool set which deals with all the logging data. It consists of tools like Elasticsearch, Kibana, and Filebeat.

1.7 Conclusion

At the end of this chapter, it becomes clear that understanding the main topics is one of the fundamental and pivotal foundations in the modern landscape, due to its effective role in enabling continuous improvement and identifying gaps.

Through this chapter, we reviewed the basic principles and commonly used frameworks and tools in this field. These concepts have significantly contributed to the development of systems and applications in this domain.

The literature review revealed several critical research gaps that directly motivate the research proposed in this study. In the second chapter, we will present the general framework of this work by outlining the adopted structure of our study.

CHAPTER 2

METHODOLOGY AND GEN-TWIN SYSTEM DESIGN

2.1 Introduction

Having established in Chapter 1 the escalating sophistication of ransomware threats and the critical limitations of conventional point-in-time security validation methods, a clear research gap is identified: the absence of a proactive, continuous, and automated methodology for evaluating defense efficacy against multi-stage attacks in a safe and realistic context. This chapter accordingly addresses this gap by formulating the scientific methodology and presenting the comprehensive conceptual design of the Gen-Twin platform. It serves as the intellectual and architectural bridge between the theoretical foundations previously discussed and the practical implementation detailed in Chapter 3. The chapter commences by providing a rigorous justification for the adopted research methodology, followed by a derivation of system requirements from the identified gap. Subsequently, it presents the architectural blueprint of the platform, defines the novel ‘Network DNA’ (Declarative Network Modeling) model, and culminates in the formal design of the correlation and detection gap analysis models.

2.2 Research Methodology

The choice of research methodology aims to implement the study in a systematic, organised, and scientific manner and also helps improve its accuracy, as the choice of methodology varies according to the nature of the research, whether developing, testing, or analysing.

2.2.1 Design Science Research (DSR)

We chose the design science research methodology as a framework for this research so that this approach is compatible with the nature of the project that aims to develop an integrated platform, which is the Gen Twin Platform.

Design science research refers to a methodology that is used in creating or operationalizing research if the desired output is an artifact or a suggestion. Besides, design science research can be done either within academia or within organizations. [14]

Design Science Research (DSR) is based on three cycles 2.1 that strike a balance between practical requirements and theoretical principles.[15] These cycles are:

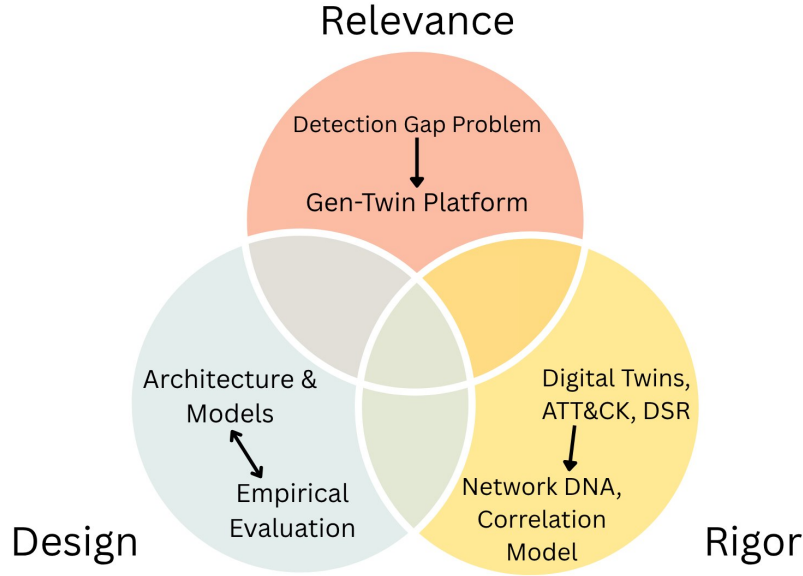


Figure 2.1: Design Science Research cycles

1. **Relevance:** The Relevance Cycle links this study to a real-world problem. In this research, importance arises from the increasing sophistication of ransomware attacks and the disparity that exists between deployed measures and actual capabilities. Organizations do not have a method to verify if their security controls are able to detect actual attack techniques based upon automated platforms, which the Gen-Twin platform provides by offering continuous, generative digital twin validation.
2. **Design:** The process of designing the artifact will involve the creation and assessment of it, and it is the primary iterative engine of the thesis; it alternates back and forth between conceptualising the platform, building the prototype, and empirically evaluating it. Each evaluation will generate results (Chapter3) which then will be incorporated into the design to update and improve the proposed model and architectural framework of the artifact. This ensures the final artifact is both founded on sound theory and empirically validated.
3. **Rigor:** The rigor cycle grounds the research in the existing knowledge base and ensures that the design is informed by scientific foundations. The design of the platform is built on concepts introduced in the systematic review in Chapter 1, including both cyber-digital twins, the MITRE ATT&CK knowledge base, and principles of detection engineering. The Cycle also supports the originality of the work, its theoretical contribution to the field, and its practical applicability.

2.2.2 Research Phases

The research was conducted in four sequential phases 2.2:



Figure 2.2: Research Phases

1. **Literature:** This phase focused on a systematic review of digital twins and ransomware attacks, in addition to the MITRE ATT&CK framework as a standardized taxonomy for adversary behavior. The literature review identified a critical research gap: the absence of generative digital twin provisioning for automated defense validation. While existing tools provide either attack simulation (e.g., Caldera, Metasploit) or security monitoring (e.g., Wazuh, Splunk), no integrated platform combines automated twin generation, attack simulation, and detection gap analysis in a single workflow.
2. **Design:** The System Design phase translates the research objectives and the derived functional and non-functional requirements into a coherent layered architectural vision that directly addresses the methodological void identified in the literature. As the core of this chapter, this phase abstracts away from specific implementation details to establish the logical architecture of the platform, define the Network DNA model, and develop the theoretical models for attack-defense correlation and detection gap quantification. The deliverable is a complete, technology-agnostic blueprint for the Gen-Twin platform, encompassing the system architecture, the Network DNA model, the Infrastructure-as-Code-driven automated provisioning workflow, and the evaluation approach.
3. **Implementation:** The Implementation phase instantiates the operational system from the architectural design using an iterative methodology aligned with the Design Cycle of DSR, ensuring that each iteration delivers a functional increment of one or more architectural layers while incorporating empirical feedback from integration testing. During this phase, there are expected challenges in ensuring cross-platform compatibility of system components and maintaining strict environmental isolation between the host system and the digital twin. All complete implementation details and results will be covered in Chapter 3.
4. **Evaluation:** The Evaluation phase marks the final stage of the research, providing the empirical evidence necessary to validate whether the constructed artifact

satisfies its intended functional, performance, and security requirements. A three-part framework guides the assessment, covering: functional, performance, and security efficacy. The final working prototype is subjected to controlled, multi-stage ransomware simulations that replicate a complete adversary kill chain from initial access through lateral movement to data encryption, producing the dataset needed for a thorough evaluation. Security efficacy is quantified using the three detection gap metrics: Detection Rate, Weighted Coverage Score, and Mean Alert Latency. The collected empirical data are analyzed to address the three research questions.

2.2.3 Evaluation Strategy

The evaluation strategy was designed to assess the proposed platform from three aspects:

1. **Functional:** Functional validity is all about making sure the platform runs its intended workflow from start to finish without any interruption . You measure this by looking at two things: first, can it handle every type of Network DNA model you put in it? Second, does it consistently provision twins all on its own, without manual intervention? Everything needs to be perfect throughout the pipeline, including bringing in the DNA, generating twins, and running the simulation, across every test case. If any part fails, it means there's a problem in how the platform works, and that needs fixing.
2. **Performance:** The performance of the platform is determined by how quickly it executes actions and whether resources are utilized properly and how well it handles growth in the size and complexity of the test environment, providing a reliable foundation for repeat use. The key metrics for evaluation are average setup time and average CPU and Memory utilization per setup and the ability to provision twins of increasing scale without disproportionate degradation in performance. If the performance efficiency of this new automated platform doesn't equal or exceed what was accomplished with the prior manual methods and if it cannot scale to meet the demands of realistic network models of simulating, what good has been done? It must also deliver on the benefits that automation was intended to provide.
3. **Security efficacy:** Security efficacy is basically the heart of this research. The key question is: Does the platform actually spot gaps and make that info useful? To measure this, the study uses three main metrics: First, the Detection Rate looks at how many attack steps triggered alerts. Then there's Weighted Coverage Score, which takes into account not just detection, but how serious each attack technique is. Finally, Mean Alert Latency checks how quickly those detections happen

2.3 System Requirements

System requirements define the expected capabilities and quality attributes of the proposed Gen-Twin platform. They are divided into functional requirements (Table 2.1), which describe the services provided by the system, and non-functional requirements (Table 2.2), which define the constraints and quality characteristics that must be satisfied.

Requirement	Description
Network DNA Ingestion	The system shall ingest and parse a declarative model of a target network that is readable by both machines and humans. It shall then be able to transform the descriptions and properties of this model into executable artefacts that provision the corresponding digital twin.
Twin Generation	Based on the ingested Network DNA, the system shall then translate its descriptions and properties into executable artifacts that automatically provision an isolated digital twin of the defined network.
Attack Simulation	The platform shall execute realistic, multi-stage ransomware attack scenarios against the generated digital twin.
Detection Analysis	The platform's core output shall be an automated analysis that correlates attacker actions with generator alerts to identify and report on "detection gaps" attacks that were executed but not detected.

Table 2.1: Functional Requirements of the Gen-Twin Platform

Requirement	Description
Isolation	The simulated environment shall be completely isolated from the host production network to avoid any damage to the network or underlying device .
Fidelity	The behavior of the resulting twin nodes and services shall closely emulate real systems in terms of operating system behavior, network protocol applications, and event log generation to obtain realistic results.
Scalability	The platform architecture should support the definition and provision of network models of varying complexity, ranging from small branch offices to multi-level enterprise topologies.
Automation	All system processes, from creating the digital twin to issuing the gap report, shall be automated, with no manual intervention required.

Table 2.2: Non-Functional Requirements of the Gen-Twin Platform

2.4 Gen-Twin System Architecture

The Gen-Twin platform is designed as a modular, layered architecture that enables the automated generation, execution, and analysis of cyber digital twin environments. This

design ensures separation of concerns, scalability, and flexibility across all stages of the system lifecycle, from network definition to security evaluation.

2.4.1 Global Architecture

The Gen-Twin platform follows a four-stage pipeline. At each stage of this pipeline the output of one stage provides the input for the next. The pipeline processes a network definition obtained from a declarative network model to produce an ultimate detection gap report. This complete validation workflow, as represented in Figure 2.3.

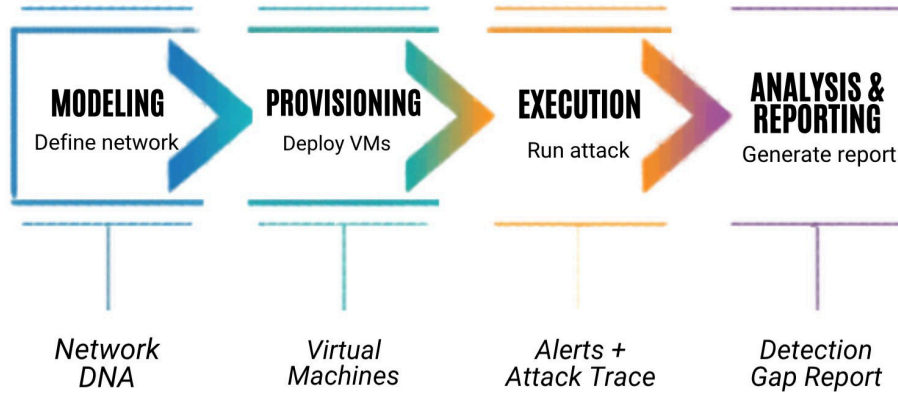


Figure 2.3: Architecture Pipeline of the Gen-Twin Platform

The four stages work as follows:

1. **Modeling "Define Network"**: The pipeline begins with a declaration that describes the desired network to be built, using the Network DNA model for the purpose of specifying nodes, links, services, and security policies needed to create the network accurately. The input at this stage will be the user's intent for the design of the network; the output will be a documented version of the Network DNA which has previously been validated and is ready for provisioning.
2. **Provisioning "Deploy virtual machines"**: The second stage takes the Network DNA specification and automatically provisions the corresponding digital twin. Isolated virtual machines will be created and their corresponding network infrastructure established so that an environment is created that exactly replicates the defined topology. The Network DNA document (input) has been converted into a group of virtual machines that are all operational and interconnected to the Digital Twin.
3. **Execution "Execute Attack"**: The third stage executes the ransomware scenario against the provisioned twin. During this step, the simulation will implement a multi-stage attack scenario, where each action taken by an adversary will be time-stamped and logged. In this step, the digital twin that was provisioned in the previous step is the input, and the output will consist of two separate data streams that are produced: Attack Trace which documents each individual action taken by the attacker and Detection Trace which captures each alert created by the Monitoring Sensors.

4. **Analysis and Reporting "Generate Report":** The final stage processes the collected telemetry to produce the platform's primary deliverable. The Attack and Detection Trace datasets will be correlated to determine what attacks were detected versus undetected. Any noted gaps in detection will be quantified, and all findings will be compiled into a Detection Gap Report. The primary input for this final phase will be the two data traces, but the ending output will be the Detection Gap Report.

To realize this pipeline, the platform is divided into four logical layers: Orchestration, Emulation, Instrumentation, and Analysis. Each logical layer corresponds to a single or multiple stages of the workflow described in the following sections.

2.4.2 Orchestration Layer

The Orchestration layer is the central control plane for the platform. It is responsible for overseeing the Validation session lifecycle from beginning to end.

This layer consists of two core logical components:

1. **DNA Parser:** Responsible for syntactic and semantic Validation of the ingested Network DNA document. It translates the high-level definition model into an internal, machine-processable implementation plan and rejects malformed or logically inconsistent definitions before allocating any resources.
2. **Provisioning Engine:** Acts as a workflow orchestrator. It consumes the execution plan and manages the provisioning lifecycle by issuing build requests to the Emulation layer, directing node configuration, and deploying monitoring sensors and attack agents from the Instrumentation layer. During this process, it maintains and tracks the state of the twin's machine, moving through the specified stages of Provisioning, Ready, Under Test, and Terminated.

2.4.3 Emulation Layer

The Emulation Layer functions as a virtual hardware abstraction and a generic factory for the network substrate. It receives provisioning requests from the Orchestration Layer for a set of interconnected virtual computing nodes and returns the access credentials and resource handles required to manage those nodes. Internally, it consists of a virtualization engine that manages the lifecycle of each virtual machine, and a network abstraction component that establishes the virtual fabric interconnecting the provisioned nodes. The layer delivers empty machines containing only a base operating system without any application logic, along with the virtual network fabric that links them together.

2.4.4 Instrumentation Layer

This layer defines the abstract actors that operate within the twin to generate the telemetry required for analysis. It comprises three logical components: a monitoring sensor deployed on every node designated for defense evaluation to collect and stream system and application logs to a central point, an adversary emulator agent, deployed on a designated attack host, that translates high-level commands from the Orchestration Layer into time-stamped actions on target systems; and a central data collector that merges

the two independent data streams the attack trace and the detection trace into a single repository, forming the prerequisite for the correlation process.

2.4.5 Analysis Layer

The Analysis Layer converts the collected telemetry into the platform’s primary deliverable: the Detection Gap Report, through a three stage pipeline, each stage feeding into the next. The pipeline begins with the Correlation Engine, where the Attack Trace and the Detection Trace from the Collector are processed to match each attack step with its corresponding alert. The result is a clean classification; every action gets marked as either detected or undetected. This classified mapping then moves to the Gap Analyzer, which applies the platform’s quantitative metrics Detection Rate, Weighted Coverage Score, and Mean Alert Latency and produces a list of detection weaknesses, ordered by risk severity. From there, the Report Generator takes this prioritized list and builds the final report, translating the numbers into an executive summary, an ATT&CK heatmap, and a set of actionable recommendations.

2.5 Network DNA Model and Automated Provisioning

The Gen-Twin Platform has a fundamental representation for conceptualizing and implementing a cyber environment called the Network DNA model. Declaratively and structurally, the Network DNA defines the network topology, system roles, services, and security configurations of the environment, allowing for automated provisioning and the production of reproducible cyber digital twins.

2.5.1 Digital Network Architecture (DNA)

The Network DNA is based on the principle of declarative network modeling, where the desired state of the infrastructure is described in a high-level format rather than through manual configuration steps. This approach allows the system to interpret the infrastructure requirements and automatically generate the corresponding virtual environment, ensuring consistency between design, deployment, and analysis phases. **Structure** The model’s structure is defined by four fundamental, interconnected components:

1. **Nodes:** The logical building blocks of the network. A node is defined by its role within the security architecture such as target, attacker, or monitor rather than by a specific operating system version or hardware specification. Properties like CPU, memory, and OS image are considered instantiation details derived from the node’s role, not defining characteristics of the model itself.
2. **Links:** Abstract communication relationships between nodes. A link captures the existence and logical nature of a connection defined by the roles of the nodes it connects and the security zone context it crosses rather than specific IP addressing schemes, which are assigned during provisioning.
3. **Services:** The application-layer capabilities hosted on nodes. Rather than specifying individual software packages, a node’s functional surface is defined by the

capabilities required to fulfill its role such as log collection and forwarding for a monitor node.

4. **Security Policies:** Abstract access control rules that define the intended communication patterns between nodes of different roles. For instance, a policy may specify that attacker nodes may communicate with target nodes on designated ports, while monitor nodes observe all traffic passively. These policies establish the defensive posture against which the simulation is evaluated.

- **Network DNA Representation:**

The Network DNA will be represented in a structured format using structured data formats (YAML & JSON). The structured data formats provide the following benefits: Readability, flexibility, and compatibility with automation tools.

YAML is typically preferred because of its human-readable syntax and ease of integration with orchestration frameworks, but JSON can be used for programmatic processing and interoperability with external systems.

The structured representation of the Network DNA enables seamless parsing by the orchestration layer, producing an operational cyber digital twin environment from a declarative model.

2.5.2 Automated Twin Provisioning

Automated provisioning allows for the automated generation, configuration and validation of cyber digital twin environments leading to reproducibility, scaling and consistency.

Infrastructure as Code:

Infrastructure as Code (IaC) involves applying traditional software engineering disciplines, such as code review, version control, and structured syntax, to infrastructure provisioning. These scripts are written in a domain-specific language and follow the same rigour as software source code to automate the configuration and deployment of infrastructure.[16]

Provisioning Workflow:

The provisioning engine will follow a deterministic, four-stage pipeline to guarantee a pristine and validated test environment for every experiment:

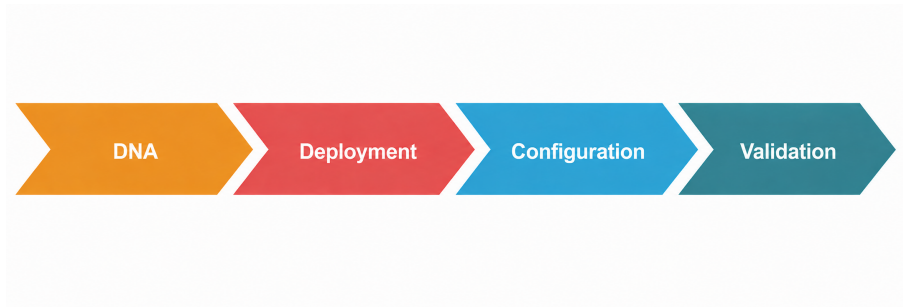


Figure 2.4: Provisioning Workflow

1. **DNA Parser:** The DNA document is ingested and validated. A syntactically and semantically correct DNA is compiled into an internal, technology-agnostic execution plan.
2. **Deployment:** The engine requests the Emulation Layer to instantiate the raw, unconfigured virtual nodes and networks as specified in the execution plan.
3. **Configuration:** Using IaC principles, the engine methodically configures the operating system level of each node, sets up the defined services, and enforces the specified security policies. The nodes transition from a "raw" to an "operational" state.
4. **Validation:** A set of automated probes and health checks is run to assert that the operational state of the twin perfectly matches the specifications in the Network DNA.

State Management:

The platform's design incorporates robust state management. The capability to capture a point-in-time snapshot of the entire twin's state is imperative. This 'Clean State' serves as the immutable starting point for every single attack simulation scenario. After an experiment concludes, the twin can be instantly rolled back to this clean state, ensuring no data contamination between test runs and enabling perfect experimental isolation.

2.6 Attack Simulation and Detection Monitoring

The Attack Simulation and Detection Monitoring modules constitute the operational core of the Gen-Twin platform. Together, they enable the execution of controlled cyber-attacks within the digital twin and the collection of security telemetry required for subsequent correlation and detection gap analysis.

2.6.1 Attack Simulation Module

An attack scenario is modeled as an Abstract Attack Graph, a timed sequence of high-level adversarial actions. The modeling language is declarative, specifying the "what" and the "when" of an attack step without pre-scripting the "how." This declarative separation allows the same logical scenario to be executed across twins built from different Network DNA models whether the underlying nodes run Windows or Linux without modifying the

scenario definition itself.

Ransomware Scenario

The generalised multi-stage ransomware kill chain is used as the basis for this testing scenario and represents a series of abstract (generalised) stages that represent an adversary's stages of progress throughout the kill chain in real-time.

Initial access: attackers will use some method of gaining Initial access to a system through either exploiting a system weakness or through the delivery of a malicious payload.

Execution: After gaining Initial access to a target system, the adversary moves to the Execution phase and establishes persistence on that host and then attempts to disable or weaken the existing local defensive posture.

Lateral Movement: The attacker moves from the initial foothold to progressively higher value systems using legitimate administrative protocols and harvested credentials.

Impact: The attacker will access the impacted node and encrypt critical information (files) contained within the targeted node, and also perform activities with a destructive nature that would serve to demonstrate the ultimate effects of the ransomware attack.

Each phase is instrumented to record its precise start time and duration, generating the attack trace of the exact time the phase commenced, as well as for how long each phase was performed, thus creating a total record of each of the attack actions performed during each phase of testing.

MITRE ATT&CK Mapping

A key design element that enables automated correlation between the attack trace and the detection trace is the semantic tagging of every phase in the scenario with a corresponding MITRE ATT&CK Technique ID.

For example, the "Initial Access" phase is tagged with the identifier for the access technique used, while the "Impact" phase is tagged with identifiers for the encryption and destructive activities in the attack. Each of these tags describes the phase in terms of machine-readable semantic triplet data (exact timestamps, the nodes targeted, and the ATT&CK Technique executed). This standardization provides a common frame of reference for the correlation process, enabling the platform to directly match an action in the attack trace against its corresponding alert in the detection trace.

2.6.2 Detection and Monitoring Module

This module uses a distributed sensor setup. A sensor sits on every node in the twin, quietly watching and sending telemetry back to a central analysis unit. The sensors work in a read-only way—they collect system and application logs without touching or interfering with anything happening around them. At the center, an analysis unit runs

through the raw telemetry with a set of predefined rules and builds an ordered stream of alerts. Each alert is timestamped and linked to the node where it came from. This stream is what we call the Detection Trace. Each alert signature is 'mapped' to one or more MITRE ATT&CK Techniques, producing a semantic relationship to the attack trace and developing a common vocabulary between the two data streams.

2.7 Correlation and Detection Gap Analysis

2.7.1 Correlation Engine

Correlation Engine serves as a key component to analyze the relationship between an attacker's activity and how that activity is perceived from the defender's perspective. The purpose of the Correlation Engine is to determine whether a defender's system detected every one of the actions taken by an adversary during an emulation. It takes the two independent data streams, the Attack Trace and the Detection Trace, and looks for connections between them, producing a clear map of what was detected and what was missed.

Data Sources:

The Correlation Engine operates on two structured data sets, both generated during the same simulation run:

1. **The Attack Trace:** is the chronological record produced by the Attack Simulation Module that documents the sequence of adversarial activities. Each entry documents the time of the adversarial activity, which node was targeted, and which MITRE ATT&C was performed. It can be considered as the after-action report of an attacker with a detailed account of each step of how the attacker was able to achieve their goal.
2. **The Detection Trace:** is the alerting stream returned by the sensors of the monitoring systems placed throughout the twin. Each alert is recorded with a timestamp and is attached to the node that triggered it. Each alert also has an associated signature that can be mapped to one or more ATT&CK technique types, creating a common terminology with the Attack Trace. The use of mapping alert signatures to ATT&CK aligns with a growing trend in security research toward grounding detection telemetry in standardized adversary taxonomies. While existing studies have largely focused on extracting TTPs from unstructured threat intelligence reports and mapping them to ATT&CK [17], the Gen-Twin platform applies this same principle to structured alerts. Each alert signature is pre-associated with one or more ATT&CK technique IDs. This shared language is what makes the matching achievable.

The Correlation Engine receives these two traces separately, but they have a common vocabulary that is why matching is achievable. The language used for commonality is the MITRE ATT&CK taxonomy. The ATT&CK knowledge base provides curated detection strategies and procedural examples for each technique [9], forming the foundation upon which the correlation logic is built.

Correlation Logic:

Matching an attack step to an alert is not straightforward because of all of the various factors involved. A busy network will produce many alerts that aren't related to the simulation; simply because they occurred in close temporal proximity does not mean one of them caused the other. To differentiate real matches from noise, the Correlation Engine uses three separate criteria for checking the attack step and alert:

1. **The Attack Trace:** The first and most straightforward filter is time. An alert that appears within a narrow window of the attack, say, a few seconds, may well be related. The same alert arriving several minutes later is almost certainly unrelated and can be safely ignored. The time frame of how close is defined via configuration during setup. This method reflects established practice in security event correlation, where temporal proximity serves as an initial filter before deeper semantic analysis is applied [18].
2. **The Attack Trace:** A match in Tactics, Techniques, and Procedures carries the highest weight in the scoring model. An alert generated by an RDP anomaly may be a good match for an attack step associated with RDP lateral movement. In contrast, an alert generated by the connection of a USB device would not be considered a valid match. Comparing the ATT&CK technique executed against the techniques the alert is designed to catch yields connections that go deeper than timing alone. While existing research has largely focused on extracting TTPs from unstructured threat reports and mapping them to ATT&CK [17].
3. **Topological Matching:** The final check determines whether the attack targeted the database server and, subsequently, whether the alert originated from a workstation located at the opposite end of the network. If this were the case, we would say that the match is weak. The final check verifies that the node where the attack occurred is the same node where the alert originated; if not, checking that an undefinable relationship exists between the two nodes through the use of legitimate pivot paths illustrated by the attack trace (??)

Each dimension contributes a weighted score to the overall match confidence. The TTP dimension carries the highest weight because it addresses the nature of the event itself, rather than merely its timing or location. When an attack step matches above the pre-defined threshold for a respective pair, that pair will be considered as a confirmed detection. When an attack step has no match above the pre-defined threshold, the attack step will be marked as a detection gap. Therefore, the results from the correlation engine are unambiguous: every action that the attacker has taken will either have been identified or not identified.

2.7.2 Detection Gap Analysis Model

The Detection Gap Analysis Model evaluates security monitoring effectiveness by comparing simulated attacks with detected events to identify gaps, measure detection performance and prioritize risks.

Metrics

Detection performance is measured in three different ways by the metrics in this section:

Detection Rate:

A simple measurement to determine the level of visibility of a system due to the number of steps taken by an attack that resulted in an alert (detection) versus the total number of steps taken by the attack (detection) that resulted in no detection.

$$\text{Detection Rate (\%)} = \frac{\text{Number of Detected TTP}}{\text{Total Number of TTPs}} \times 100 \quad (2.1)$$

Weighted Coverage Score (WCS):

Not all attacks are equal. Missing an attempted phishing is not equal to missing a modified data encryption. The WCS assigns a level of risk to each type of attack when performing the weighted coverage score. If the detection rate is high but the WCS is low, that indicates most of the small attacks were detected, but the large attacks were not detected.

Mean Alert Latency (MAL):

The speed of detection of an attack is important. The MAL measures the amount of time from the execution of an attack step to when the first detection or alert is generated. If the detected alert takes too long to generate after execution of an attack, that gives the actual attacker enough time to perform damage to a target even if the alert is eventually generated.

Outputs

Results are shown using two different structured formats:

1. **Gap classification:** Each gap is identified as either a Blind Spot where no alert was sent when the attack occurred or as a Latency Gap, where an alert was sent too late for effective response. Each type of gap will require a different corrective measure.
2. **Risk prioritization:** Each gap is ranked based on the level of risk associated with the missed technique. In general, any encryption blind spot is assigned the highest level of risk, whereas a latency gap with reconnaissance will be assigned one of the lowest levels. This ranking enables the security team to prioritize how quickly corrective measures should be initiated.

2.7.3 Detection Gap Report

A Detection Gap Report (DGR) is the last deliverable from the Gen-Twin platform. It takes everything the simulation produced and presents it in a form that a security team can act on. The report consists of three layers:

1. **Executive summary:** It provides a one-page summary of the three key metrics: Detection Rate, Weighted Coverage Score, and Mean Alert Latency—and visualizing them with a traffic-light assessment, green means no critical gap, yellow indicates latency issues, and red indicates a blind spot for high-risk techniques answering the question 'how safe are we?' at a glance.
2. **ATT&CK heatmap:** which is a visual representation of the MITRE ATT&CK matrix, where every technique used during the simulation will be identified as a colour-coded outcome based on detection. Green means the technique was detected, with an acceptable amount of latency; orange means the technique was detected too slowly; and red means the technique was not detected at all (blind spot). By using the heat map feature, a security architect can quickly identify areas of defense weakness.
3. **Recommendations:** The recommendations provided in the report match the critical gaps that were identified; therefore, each of the recommendations will directly state how that recommendation will help mitigate/eliminate a specific identified critical gap. A blind spot on PowerShell execution, for example, produces a recommendation to enable script block logging and deploy a detection rule for encoded commands. The intent of the recommendations is for them to be given directly to a Detection Engineering Team to create detection rules based on the identified critical gaps.

2.8 Ethical and Safety Considerations

Simulating ransomware even inside a controlled lab carries real risk. The design of the Gen-Twin platform treats safety not as an afterthought but as an architectural requirement.

Isolation:

Isolation: The platform provides strict environmental separation between the digital twin and the host system. The twin operates in a completely isolated virtual environment with no routable path to the outside world. This ensures that even if a simulation behaves unexpectedly, nothing escapes.

Safe Ransomware Simulation:

The attack simulation scenarios are all created to demonstrate adversarial capabilities without causing any damage that would be irreversible. In the Impact phase, for instance, file encryption is simulated through non-destructive means such as renaming files or altering metadata, proving the attacker's reach without actually destroying data.

Responsible Testing:

The platform is designed to only allow the testing of attacks against Network DNA definitions that have been authorized by the individual conducting the test. The platform

does not allow the simulation of random attacks; only attacks against digital twins owned and controlled by the researcher conducting the experiments, thereby preventing the researcher from unintentionally creating a threat through the research process itself.

2.9 Conclusion

This chapter has established the complete methodological and architectural foundation for constructing the Gen-Twin platform. The adoption of Design Science Research was justified as the most rigorous and relevant framework for this design-centric problem. The system requirements, derived from the practical research gap, were formalized. A novel, layered logical architecture was presented, decoupling the platform’s layers: Orchestration, Emulation, Instrumentation, and Analysis. The core theoretical contributions of this thesis, the Network DNA model, the TTP-driven correlation logic, and the quantitative Detection Gap Analysis model were formally defined here. This complete conceptual blueprint serves as the unambiguous specification for the prototype implementation and rigorous evaluation documented in the subsequent chapter.

CHAPTER 3

IMPLEMENTATION, EVALUATION AND RESULTS

3.1 Introduction

This chapter serves as the technical bridge between the conceptual design of the digital twin to its tangible validation, through the complete implementation and empirical evaluation of the Gen-Twin platform, building on theoretical foundations and methodological framework established in Chapters 1 and 2. The chapter details the experimental environment, demonstrates the instantiation of the Network DNA model through concrete topology examples, and examines the five-stage deployment pipeline that transforms YAML-based specifications into running virtual machines. The core contribution lies in the execution of a LockBit 3.0 ransomware simulation against the deployed digital twin, followed by the collection, correlation, and analysis of Wazuh alerts. The detection gap analysis is performed using the ATT&CK heatmap methodology, quantifying the platform's detection rate and identifying critical security weaknesses. The chapter culminates in an experimental evaluation that includes a baseline vs. re-test comparison, where custom detection rules are applied, performance assessment, security recommendations, and future work directions, collectively validating the platform's effectiveness in addressing the research questions defined in this study.

3.2 Implementation Environment

The implementation of the Gen-Twin platform was carried out using a combination of hardware (Table 3.1) resources and a software environment composed of virtualization, automation, monitoring, and attack simulation tools. Together, these components provided the infrastructure required to deploy, manage, and evaluate the cyber digital twin.

Component	Workstation 1	Workstation 2
Device Model	Dell Latitude 7420	HP EliteBook 840 G6
Processor	Intel Core i5-1145G7 @ 2.60 GHz	Intel Core i5-8365U @ 1.60 GHz
RAM	16 GB	16 GB
Storage	SK hynix PC711 NVMe SSD (512 GB)	WDC PC SN520 SSD (256 GB)
Operating System	Windows 11 Professional	Windows 10 Professional

Table 3.1: Hardware and Software Specifications of the Evaluation Workstations

The proposed platform integrates several cybersecurity and automation technologies:

1. **Vagrant:** A tool for creating and managing isolated virtual development environments [19], was used to provision the base virtual machines for each node in the digital twin, ensuring full isolation from the host system and other projects.
2. **Ansible:** an automation tool for streamlining repetitive configuration and deployment tasks across large numbers of servers [20], was employed to provision and configure all twin nodes through version-controlled code, eliminating manual setup across the environment.
3. **Wazuh:** an open-source security monitoring platform that detects threats by applying predefined rules to aggregated log data [13], was deployed across all twin nodes to collect and analyze endpoint telemetry, generating the Detection Trace required by the correlation engine.
4. **Oracle VM VirtualBox:** an open-source virtualization tool from Oracle that runs guest operating systems with their libraries and applications on a host machine via a hypervisor [21], provided the virtualization layer for the Gen-Twin digital twin, enabling the creation and management of the isolated virtual machines used in all experimental scenarios.
5. **Caldera:** an automated adversary emulation framework developed by MITRE [22], was deployed on the attack host within the digital twin to execute the multi-stage ransomware scenario, generating the timestamped Attack Trace required by the correlation engine.
6. **Windows Subsystem for Linux (WSL):** a Windows feature that enables a full Linux environment to run natively without the overhead of a separate virtual machine [23], providing the underlying Linux-compatible environment for executing Ansible provisioning scripts and other platform components during development and testing.

3.3 Gen-Twin Platform Implementation Workflow

The Digital Twin workflow defines the complete lifecycle of the Gen-Twin platform, starting from the definition of the Network DNA model to the deployment of the virtual

environment and the execution of adversarial scenarios. The workflow is structured into three main phases: Network DNA processing, automated provisioning, and attack scenario execution.

3.3.1 Network DNA Implementation

The Network DNA module represents the core abstraction layer of the platform. It converts user-defined infrastructure descriptions into deployable cybersecurity digital twins. The user creates a network topology through the web-based graphical interface. The platform then converts the topology information into a YAML file, which represents the “Network DNA” of the infrastructure. This YAML-based Network DNA contains all the necessary information about the network, including nodes, operating systems, IP addresses, roles, and security tools, and is later used in the deployment of the cybersecurity twin environment.

The Network DNA is designed to be scalable and adaptable to different infrastructure sizes. It can represent environments ranging from small testbeds to complex enterprise networks.

Here is a small network consisting of three virtual machines (Server, Attacker, Monitor) was designed to evaluate the Gen-Twin platform

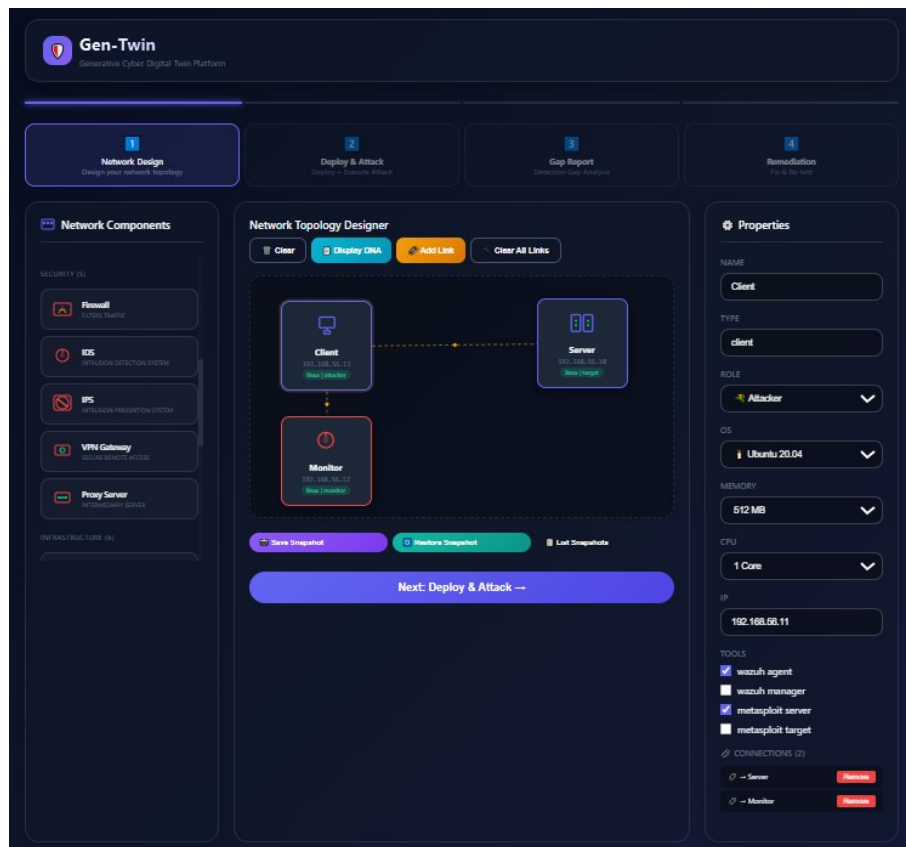


Figure 3.1: Canvas of small network

YAML Representation

The network topology is represented using YAML format, which provides a human-

readable and machine-parsable description of the network. Figure 3.2 shows the YAML representation of the small network as displayed in the "Display DNA" window:

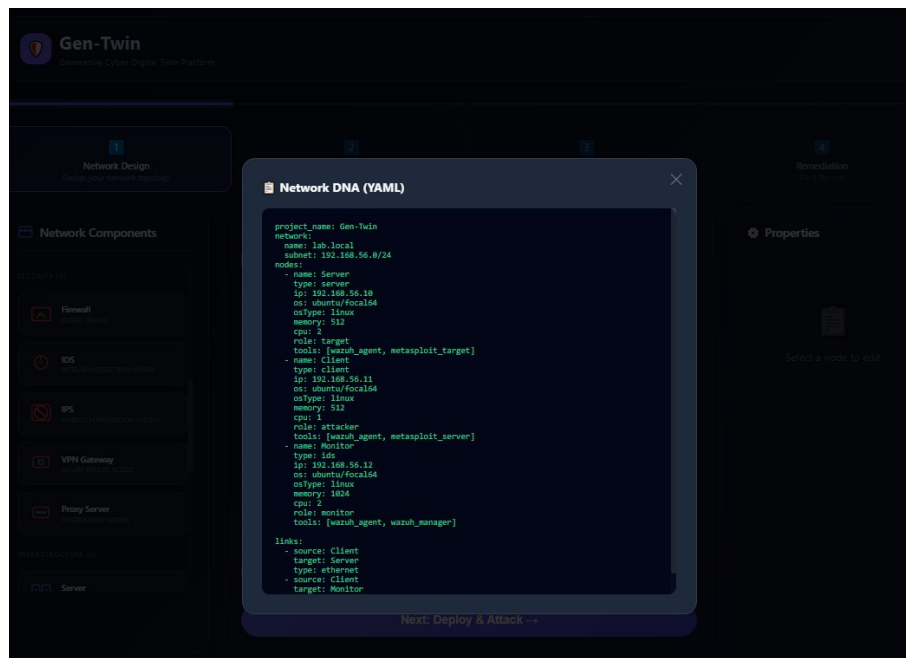


Figure 3.2: YAML output from "Display DNA" showing node configurations and links

And this full YAML structure:

```
project_name: Gen-Twin

network:
  name: lab.local
  subnet: 192.168.56.0/24

nodes:
  - name: Server
    type: server
    ip: 192.168.56.10
    os: ubuntu/focal64
    osType: linux
    memory: 512
    cpu: 1
    role: target
    tools: [wazuh_agent, metasploit_target]

  - name: Client-01
    type: client
    ip: 192.168.56.11
    os: ubuntu/focal64
    osType: linux
    memory: 512
    cpu: 1
    role: attacker
    tools: [wazuh_agent, metasploit_server]

  - name: Monitor
    type: idm
    ip: 192.168.56.12
    os: ubuntu/focal64
    osType: linux
    memory: 1024
    cpu: 2
    role: monitor
    tools: [wazuh_agent, wazuh_manager]

links:
  - source: Client
    target: Server
    type: ethernet
  - source: Client
    target: Monitor
```

```

- name: Monitor
  type: ids
  ip: 192.168.56.12
  os: ubuntu/focal64
  osType: linux
  memory: 1024
  cpu: 1
  role: monitor
  tools: [wazuh_agent, wazuh_manager]

links:
- source: Client-01
  target: Server
  type: ethernet

- source: Client-01
  target: Monitor
  type: ethernet

```

Listing 3.1: full YAML structure of Small Network

The medium-scale network extends the basic topology by introducing additional infrastructure components and services.

As illustrated in Figure 3.3 and defined in Listing 3.2, in our example the network consists of five interconnected nodes: a router, a monitoring server, an application server, a database server, and an attacker machine

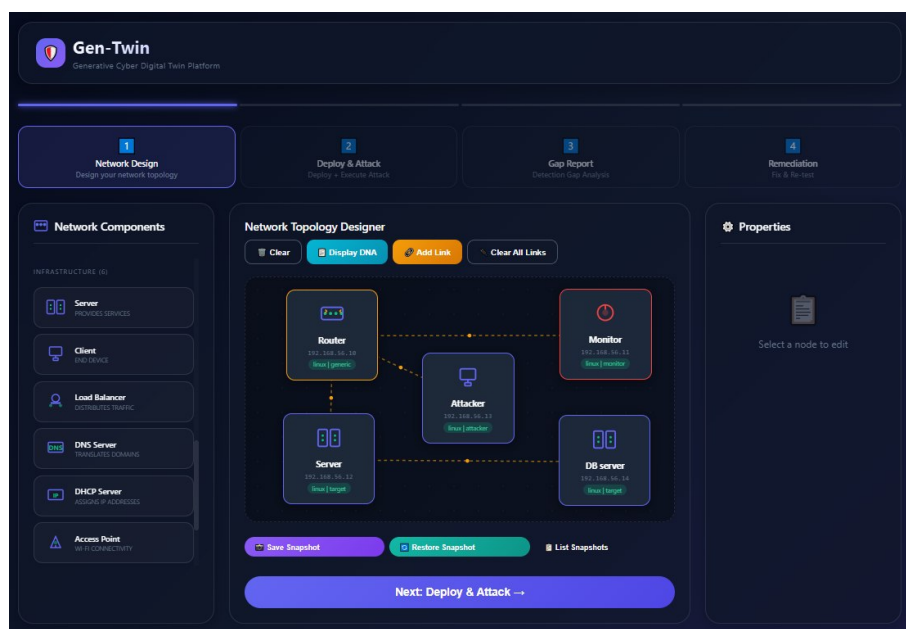


Figure 3.3: Canvas of medium network

And this full YAML structure:

```

project_name: Gen-Twin

```

```
network:
  name: lab.local
  subnet: 192.168.56.0/24

nodes:
  - name: Router
    type: router
    ip: 192.168.56.10
    os: ubuntu/focal64
    osType: linux
    memory: 512
    cpu: 1
    role: generic
    tools: [wazuh_agent]

  - name: Monitor
    type: ids
    ip: 192.168.56.11
    os: ubuntu/focal64
    osType: linux
    memory: 2048
    cpu: 2
    role: monitor
    tools: [wazuh_agent, wazuh_manager]

  - name: Server
    type: server
    ip: 192.168.56.12
    os: ubuntu/focal64
    osType: linux
    memory: 2048
    cpu: 2
    role: target
    tools: [wazuh_agent, metasploit_target]

  - name: Attacker
    type: client
    ip: 192.168.56.13
    os: ubuntu/focal64
    osType: linux
    memory: 1024
    cpu: 1
    role: attacker
    tools: [wazuh_agent, metasploit_server]

  - name: DB server
    type: server
    ip: 192.168.56.14
    os: ubuntu/focal64
    osType: linux
    memory: 2048
```

```

cpu: 2
role: target
tools: [wazuh_agent, metasploit_target]

links:
- source: Router
  target: Server
  type: ethernet

- source: Router
  target: Attacker
  type: ethernet

- source: Router
  target: Monitor
  type: ethernet

- source: Server
  target: DB server
  type: ethernet

```

Listing 3.2: Advanced Network DNA Configuration

The enterprise-scale network represents a complex configuration within the Gen-Twin platform, designed to simulate a realistic organizational infrastructure with multiple security layers, segmented networks, and heterogeneous systems. As illustrated in Figure 3.4 and defined in Listing 3.3, the topology includes a set of interconnected components.

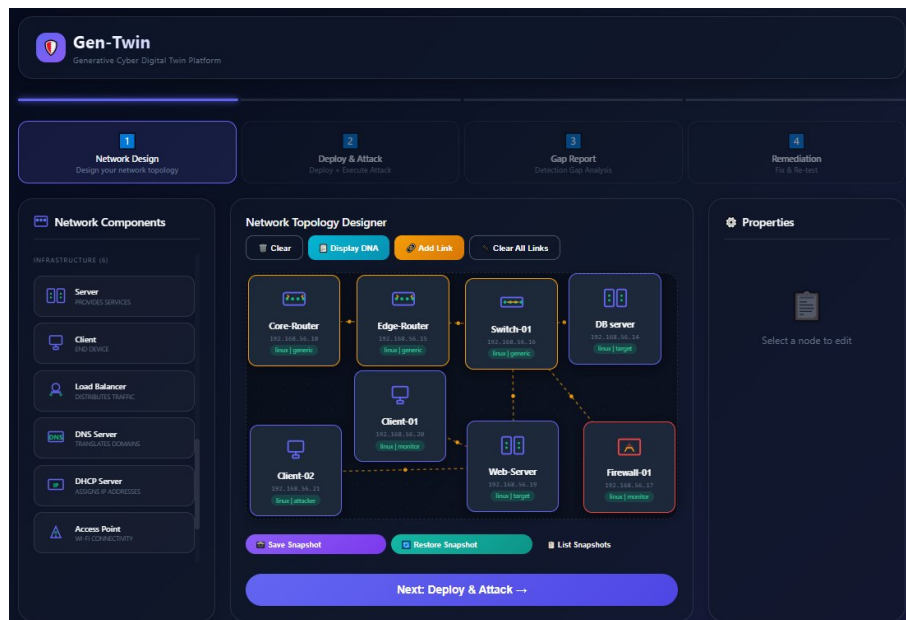


Figure 3.4: Canvas of enterprise network

And this full YAML structure:

```

1 project_name: Gen-Twin

```

```

2
3 network:
4   name: lab.local
5   subnet: 192.168.56.0/24
6
7 nodes:
8   - name: Core-Router
9     type: router
10    ip: 192.168.56.10
11    os: ubuntu/focal64
12    osType: linux
13    memory: 512
14    cpu: 1
15    role: generic
16    tools: [wazuh_agent]
17
18   - name: DB server
19     type: server
20     ip: 192.168.56.14
21     os: ubuntu/focal64
22     osType: linux
23     memory: 2048
24     cpu: 2
25     role: target
26     tools: [wazuh_agent, metasploit_target]
27
28   - name: Edge-Router
29     type: router
30     ip: 192.168.56.15
31     os: ubuntu/focal64
32     osType: linux
33     memory: 512
34     cpu: 1
35     role: generic
36     tools: [wazuh_agent]
37
38   - name: Switch-01
39     type: switch
40     ip: 192.168.56.16
41     os: ubuntu/focal64
42     osType: linux
43     memory: 512
44     cpu: 1
45     role: generic
46     tools: [wazuh_agent]
47
48   - name: Firewall-01
49     type: firewall
50     ip: 192.168.56.17
51     os: ubuntu/focal64
52     osType: linux

```

```

53     memory: 1024
54     cpu: 1
55     role: monitor
56     tools: [wazuh_agent]
57
58 - name: Web-Server
59   type: server
60   ip: 192.168.56.19
61   os: ubuntu/focal64
62   osType: linux
63   memory: 2048
64   cpu: 2
65   role: target
66   tools: [wazuh_agent, metasploit_target]
67
68 - name: Client-01
69   type: client
70   ip: 192.168.56.20
71   os: ubuntu/focal64
72   osType: linux
73   memory: 1024
74   cpu: 1
75   role: monitor
76   tools: [wazuh_agent]
77
78 - name: Client-02
79   type: client
80   ip: 192.168.56.21
81   os: ubuntu/focal64
82   osType: linux
83   memory: 1024
84   cpu: 1
85   role: attacker
86   tools: [wazuh_agent]
87
88 links:
89 - source: Core-Router
90   target: Edge-Router
91   type: ethernet
92
93 - source: Edge-Router
94   target: Switch-01
95   type: ethernet
96
97 - source: Web-Server
98   target: Client-01
99   type: ethernet
100
101 - source: Web-Server
102   target: Client-02
103   type: ethernet

```

```

104
105 - source: Switch-01
106   target: DB server
107   type: ethernet
108
109 - source: Switch-01
110   target: Firewall-01
111   type: ethernet
112
113 - source: Switch-01
114   target: Web-Server
115   type: ethernet

```

Listing 3.3: Enterprise-Scale Network DNA Configuration

3.3.1.1 Parsing and Validation

The Network DNA file is defined using a structured format and is processed by the Orchestration Layer of the Gen-Twin platform.

The YAML parsing process reads the network DNA file and converts it into Python data structures using the `yaml.safe_load()` function.

Algorithm 1 Find Virtual Machine by Role

```

1: procedure FINDVMBYROLE(project_dir, role)
2:   Define dna_path as path to network_dna.yaml
3:   if DNA file does not exist then
4:     Print error message
5:     Return None
6:   end if
7:   Open DNA file
8:   Parse YAML content
9:   Extract nodes list from DNA data
10: end procedure

```

The system validates that all IP addresses are unique within the network. If a duplicate IP is detected, it is automatically reassigned to the next available address.

Algorithm 2 Validation and Automatic Correction of Duplicate IP Addresses

```
1: procedure VALIDATEUNIQUEIPS(nodes)
2:   Initialize empty IP list
3:   Initialize fixed_count to 0
4:   for each node in nodes do
5:     Get node IP address
6:     if IP address is empty then
7:       Continue to next node
8:     end if
9:     if IP already exists in IP list then
10:      Define base IP as 192.168.56.X
11:      Initialize existing number list
12:      for each node in nodes do
13:        if node has an IP address then
14:          Extract last IP number
15:          Add number to existing list
16:          Ignore invalid IP
17:        end if
18:      end for
19:      Set new number to 10
20:      while new number exists in existing list do
21:        Increment new number
22:      end while
23:      Generate new IP address
24:      Assign new IP to current node
25:    end if
26:  end for
27: end procedure
```

Each node must have a valid role. The system supports four roles: attacker, target, monitor, and generic. If no role is specified, generic is assigned by default:

Algorithm 3 Generation of the Ansible Inventory File

```
1: procedure GENERATEANSIBLEINVENTORY(nodes, project_dir)
2:   Initialize inventory with global Ansible variables
3:   Create empty groups:
      attacker, target, monitor, all
4:   for each node in nodes do
5:     if node is a Windows machine then
6:       Skip node
7:     end if
8:     Extract node name, IP address, and role
9:     Add node to the all group
10:    if role is attacker, target, or monitor then
11:      Add node to the corresponding group
12:    end if
13:  end for
14:  Generate inventory file content
15:  Save inventory file in project directory
16: end procedure
```

As shown in Table 3.2, the Network DNA validation process consists of five sequential checks that ensure the correctness and consistency of the input model before deployment.

Step	Validation Type	Rule	Error Handling
1	YAML Syntax	Must be valid YAML format	Return syntax error
2	Required Fields	name , ip , and role must exist	Return missing field error
3	IP Uniqueness	No duplicate IP addresses	Auto-reassign new IP
4	Role Values	Must be attacker , target , monitor , or generic	Default to generic
5	Link References	Source and target nodes must exist	Skip invalid link

Table 3.2: Network DNA Validation Steps

3.3.2 Twin Deployment

The provisioning process consists of five sequential steps that transform the Network DNA into running virtual machines. The user initiates deployment by clicking the "Deploy" button in the Gen-Twin interface, as shown in Figure 3.5.

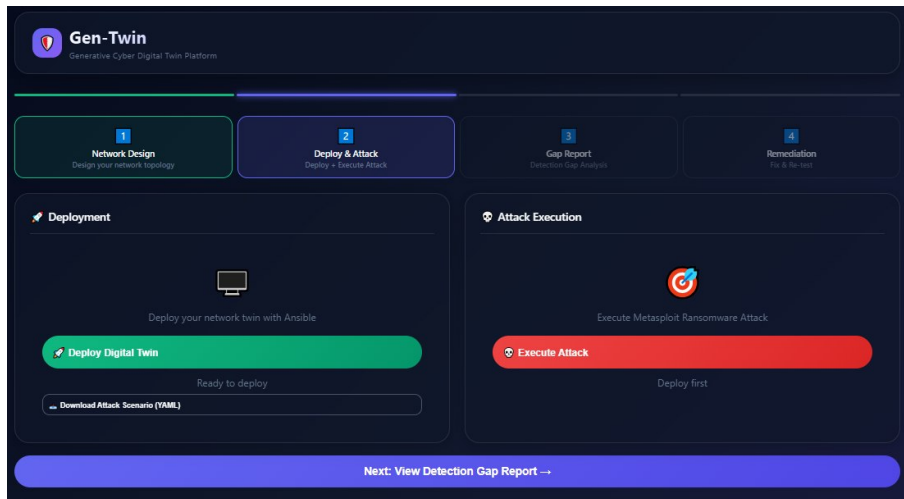


Figure 3.5: Gen-Twin deployment interface showing "Ready to deploy" status

- **Step 1: Generate Configuration Files**
After clicking "Deploy", the system generates four configuration files based on the Network DNA. Figure 3.6 shows the deployment progress at 5%.

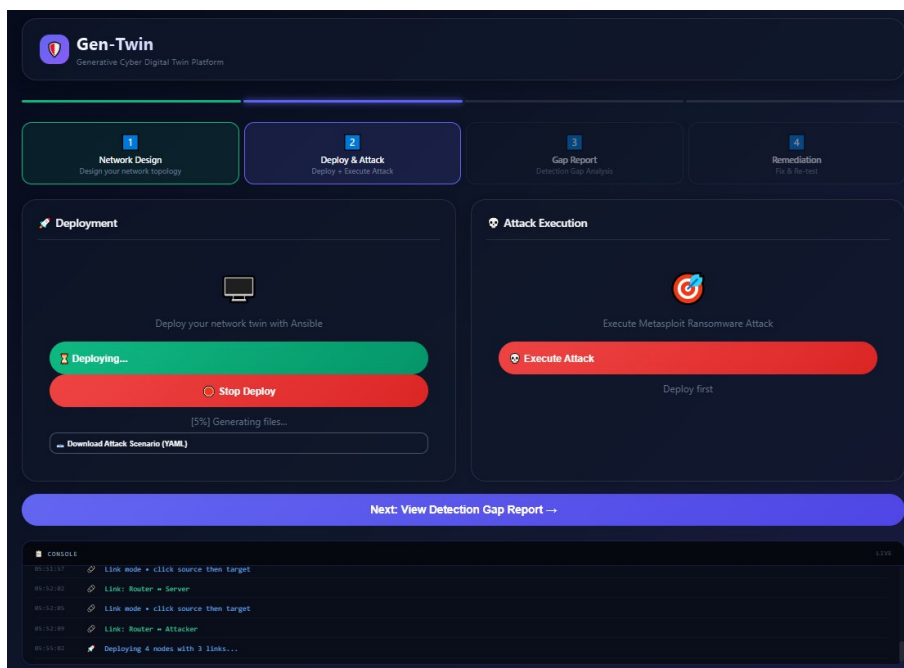


Figure 3.6: Console showing 5% progress - Generating configuration files

The system generates four configuration files (Table 3.3) based on the Network DNA:

File	Description
Vagrantfile	Virtual machine configuration file used to provision the digital twin infrastructure.
inventory.ini	Ansible inventory file containing the list of managed hosts and groups.
ansible/playbooks/site.yml	Main Ansible playbook responsible for automated software installation and configuration.
network_dna.yaml	Copy of the generated Network DNA describing the network topology and node properties.

Table 3.3: Generated Deployment and Configuration Files

Algorithm 4 Generation of Deployment Configuration Files

```

1: procedure GENERATEDEPLOYMENTFILES(project_id, dna_data, nodes, links)
2:   Generate DNA filename using project identifier
3:   Save Network DNA into YAML file
4:   Create Ansible configuration files
5:   Generate Ansible inventory file
6:   Generate Vagrantfile from nodes and links
7:   Save generated Vagrantfile into project directory
8: end procedure

```

- **Step 2:**Start Virtual Machines
Each VM is started individually using the vagrant up command.

Algorithm 5 Starting Virtual Machines using Vagrant

```

1: procedure STARTVIRTUALMACHINES(nodes, project_dir)
2:   for each node in nodes do
3:     if node is not Windows then
4:       Get node name
5:       Print starting message for node
6:       Execute Vagrant command:
         vagrant up node_name --no-color
7:       Set timeout to 3600 seconds
8:     end if
9:   end for
10: end procedure

```

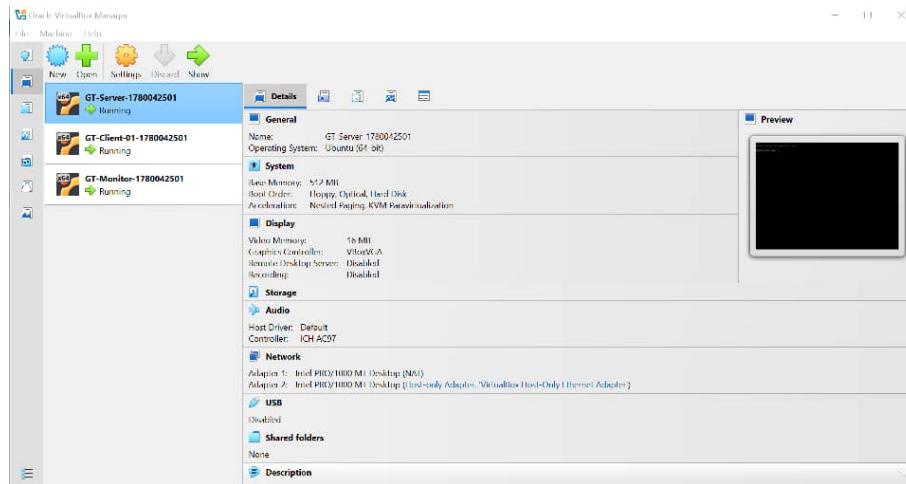


Figure 3.7: VirtualBox interface showing the running virtual machines generated

- **Step 3:** Install Tools (Ansible)

Ansible playbook installs the required tools on each VM based on its role. The playbook defines three main installation groups:

```
- name: Wazuh Agent
  hosts: target,attacker
  become: yes
  tasks:
    - name: Install Wazuh Agent
      apt:
        name: wazuh-agent
        state: present
        ignore_errors: yes

    - name: Start Wazuh Agent
      systemd:
        name: wazuh-agent
        state: started
        enabled: yes
        ignore_errors: yes

- name: Metasploit on Attacker
  hosts: attacker
  become: yes
  tasks:
    - name: Download Metasploit installer
      get_url:
        url: https://raw.githubusercontent.com/rapid7/metasploit-omnibus/master/config/templates/metasploit-framework-wrappers/msfupdate.erb
        dest: /tmp/msfinstall
        mode: '0755'
```

```

    ignore_errors: yes

- name: Install Metasploit
  shell: /tmp/msfinstall
  args:
    creates: /opt/metasploit-framework/bin/msfconsole
    ignore_errors: yes

```

Listing 3.4: Ansible playbook for Wazuh agent and Metasploit installation

- **Step 4:** Fallback Installation (VBoxManage)

If Ansible provisioning fails, a fallback mechanism using VBoxManage directly installs any missing tools:

Algorithm 6 Fallback Installation of Security Tools via VirtualBox

```

1: procedure INSTALLMISSINGFALLBACK(vm_name, tools)
2:   Initialize command list with base system installation
3:   Add python3-pip and nmap packages
4:   if wazuh_agent is in tools then
5:     Add Wazuh repository key
6:     Add Wazuh repository to sources list
7:     Update package list
8:     Install Wazuh agent non-interactively
9:   end if
10:  if metasploit_server is in tools then
11:    Download Metasploit installer script
12:    Make installer executable
13:    Execute installer script
14:  end if
15:  Execute all commands on VM using VBoxManage with timeout 1800
16: end procedure

```

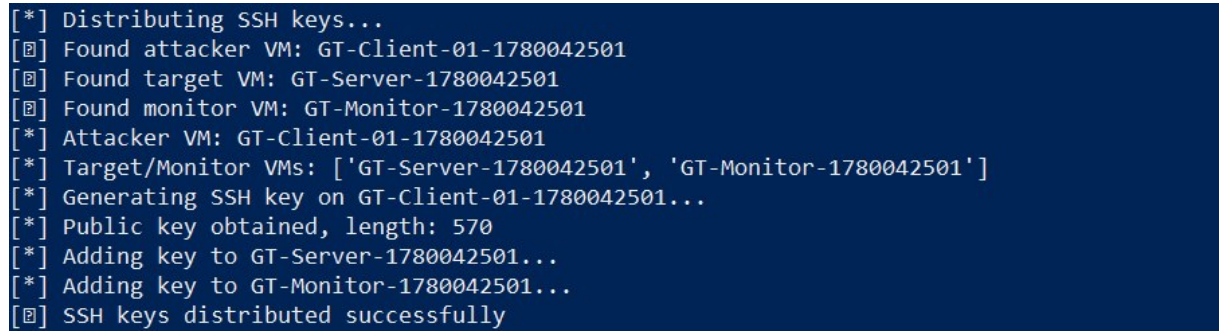
- **Step 5:** Distribute SSH Keys

SSH keys are generated on the attacker machine and distributed to target and monitor VMs to enable passwordless authentication:

Algorithm 7 SSH Key Distribution Across Virtual Machines

```
1: procedure DISTRIBUTESSHKEYS(project_dir)
2:   Retrieve attacker VM using role "attacker"
3:   Retrieve target VM using role "target"
4:   Retrieve monitor VM using role "monitor"
5:   Generate SSH key pair on attacker VM
6:   Extract public key from attacker VM
7:   if public key is empty then
8:     Abort key distribution process
9:   end if
10:  for each target VM in list of target VMs do
11:    Print key injection message
12:    Create .ssh directory on remote VM
13:    Set correct permissions
14:    Append attacker public key to authorized_keys
15:    Set secure permissions
16:    Confirm key installation
17:  end for
18: end procedure
```

Console Output :



```
[*] Distributing SSH keys...
[+] Found attacker VM: GT-Client-01-1780042501
[+] Found target VM: GT-Server-1780042501
[+] Found monitor VM: GT-Monitor-1780042501
[*] Attacker VM: GT-Client-01-1780042501
[*] Target/Monitor VMs: ['GT-Server-1780042501', 'GT-Monitor-1780042501']
[*] Generating SSH key on GT-Client-01-1780042501...
[*] Public key obtained, length: 570
[*] Adding key to GT-Server-1780042501...
[*] Adding key to GT-Monitor-1780042501...
[+] SSH keys distributed successfully
```

Figure 3.8: output console of generate and distribute SSH key

The deployment time was measured for the 3-VM network configuration. Table 3.4 shows the time required for each provisioning step:

Step	Operation	Time (seconds)	Time (minutes)
1	Generate Configuration Files	133	2.21
2	Start Virtual Machines (Vagrant up)	1110.6	18.51
3	Ansible Provisioning and Fall-back	4510.8	75.18
4	SSH Keys Distribution	120.6	2.01
Total		~5875	~97.91

Table 3.4: Deployment time for 3-VM network configuration

After successful deployment, the console displays a "Deployment complete" message with the total time taken. Figure 3.9 shows the deployment completion status with 3/3 VMs running in 5875 seconds (97.91 minutes).

Algorithm 8 Deployment Verification of Virtual Machines

```

1: procedure DEPLOYMENTVERIFICATION(project_id)
2:   Wait 30 seconds after deployment
3:   Retrieve list of running virtual machines
4:   if no VMs are detected then
5:     Wait additional 60 seconds
6:     Retrieve running VMs again
7:   end if
8:   Display list of running VMs
9:   if still no VMs detected then
10:    Set deployment status to failed
11:    Add message: "No VMs detected"
12:    Exit procedure
13:   end if
14: end procedure

```

Output Example:

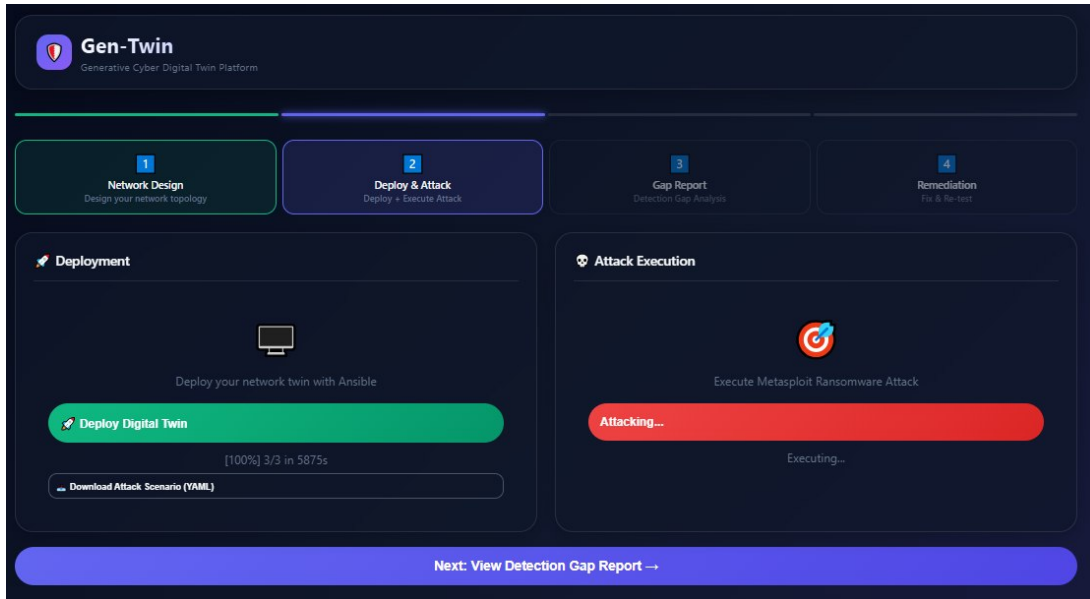


Figure 3.9: Console showing successful deployment completion

3.3.3 Attack Scenario Execution

LockBit 3.0 is a sophisticated ransomware variant that emerged in 2022, known for its multi-stage attack chain and advanced evasion techniques. It operates under the Ransomware-as-a-Service (RaaS) model and has been responsible for numerous high-profile attacks worldwide [24]

A multi-stage ransomware attack was simulated using the Metasploit framework to evaluate the detection capabilities of the Gen-Twin platform. The attack follows a 7-stage kill chain based on the LockBit 3.0 ransomware behavior, as shown in Table 3.5

Phase	TTP	Name	Description	Command
1	T1083	Reconnaissance	Network scanning to discover open ports	<code>nmap -sT -p 22,445 {target_ip}</code>
2	T1083	Discovery	File system enumeration on target	<code>find /home -type f</code>
3	T1204	Payload Delivery	Transfer malicious payload to target	<code>scp payload.elf vagrant@{target_ip}:/tmp/</code>
4	T1562	Defense Evasion	Disable security monitoring (Wazuh)	<code>systemctl stop wazuh-agent</code>
5	T1486	Impact	Encrypt files using AES-256	<code>openssl enc -aes-256-cbc -in file -out file.locked</code>
6	T1490	Impact	Delete backup files	<code>rm -rf /var/backups/*</code>
7	T1491	Impact	Create ransom note	<code>cat > README_DECRYPT.txt</code>

Table 3.5: Attack Phases

3.3.3.1 Attack Execution Implementation

Algorithm 9 Metasploit Attack Execution with MITRE ATT&CK Correlation

```
1: procedure EXECUTEMETASPLOITATTACKWITHCORRELATION(attacker_vm, tar-
   get_ip, monitor_vm)
2:   Initialize attack timeline
3:   Record start time
4:   Phase 1–3: Reconnaissance & Initial Access
5:   for each phase in reconnaissance phases do
6:     Execute command via VirtualBox on attacker VM
7:     Capture execution output
8:     Mark success or failure based on expected output
9:   end for
10:  Phase 4–7: Impact Phase
11:  for each phase in impact phases do
12:    Execute attack command on target VM
13:    Stop security services (Wazuh)
14:    Encrypt files using AES-256
15:    Delete backup files
16:    Create ransom note
17:    Capture execution result
18:    Append result to attack timeline:
        phase, tactic, technique ID, name, status, timestamp
19:  end for
20:  Return attack timeline
21: end procedure
```

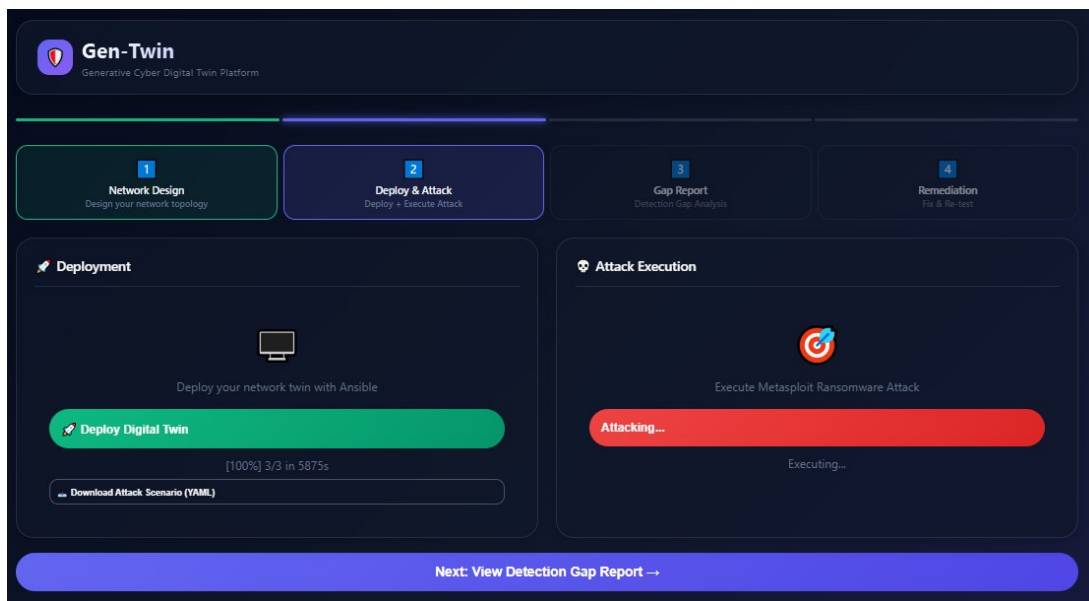


Figure 3.10: attack execution

3.3.3.2 Execution logs

Figure 3.11 shows the console output during attack execution. Phases 1, 2, and 4-7 completed successfully, while Phase 3 (Payload Delivery) failed due to missing payload file. Table 3.6 summarizes the execution results.

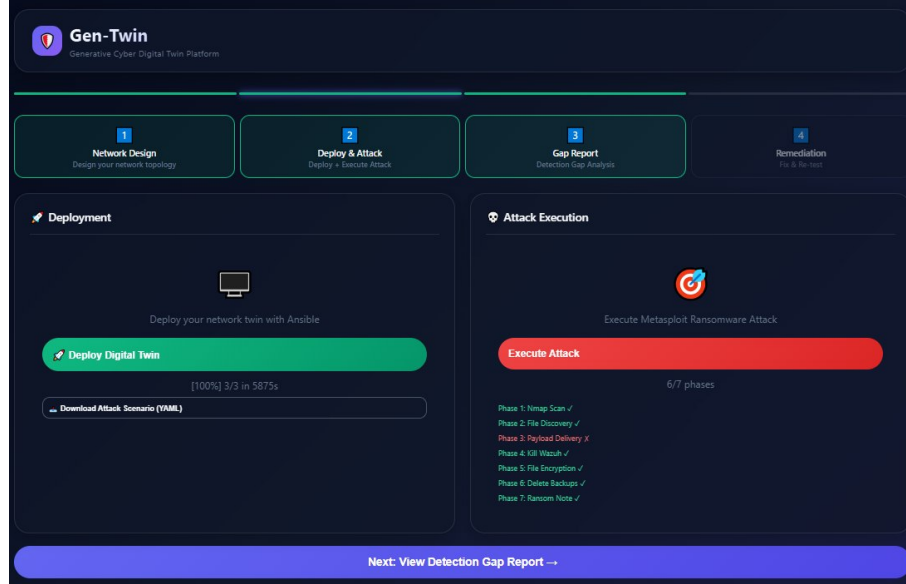


Figure 3.11: attack execution results

Phase	TTP	Name	Status
1	T1083	Nmap Scan	Successful
2	T1083	File Discovery	Failed
3	T1204	Payload Delivery	Successful
4	T1562	Kill Wazuh	Successful
5	T1486	File Encryption	Successful
6	T1490	Delete Backups	Successful
7	T1491	Ransom Note	Successful

Table 3.6: Execution Results of the Simulated Ransomware Attack

3.4 Detection and Correlation Results

During the attack execution, Wazuh generated multiple security alerts based on activities observed on the target and attacker virtual machines. These alerts were collected and processed by the pipeline analysis module to evaluate detection performance and correlate attack behavior with monitoring data.

A dedicated alert collection function (10) was implemented to parse Wazuh logs from the alerts.json file on the monitoring VM. This function extracts key fields such as timestamp, rule ID, and alert level, and retains the most recent alerts for analysis.

Algorithm 10 Collection and Parsing of Wazuh Security Alerts

```
1: procedure COLLECTWAZUHALERTS(monitor_vm)
2:   Define command to read Wazuh alert logs from alerts.json
3:   Parse JSON alert entries using Python
4:   Extract:
       timestamp, rule ID, and alert level
5:   Keep only the last 50 alerts
6:   Execute command on monitor VM with timeout of 30 seconds
7:   Initialize empty alerts list
8:   if execution succeeds then
9:     for each line in command output do
10:      if line starts with JSON format then
11:        Convert line into JSON object
12:        Append alert to alerts list
13:      end if
14:      Ignore invalid entries
15:    end for
16:  end if
17:  Return alerts list
18: end procedure
```

The timeline analysis shows partial detection coverage across the simulated ransomware attack. The system successfully detected authentication events (T1078), privilege escalation attempts (T1548.003), and backup deletion activities (T1490). However, critical phases such as file encryption (T1486) and ransomware note creation (T1491) were not detected.

Table 3.7 summarizes the detection results of simulated ransomware activities across different stages of the attack.

Event Type	MITRE Technique	Detected	Source
Login Session	T1078 (Valid Accounts)	Yes	Attacker/Target
Privilege Escalation (Sudo)	T1548.003	Yes	Attacker/Target
Backup Deletion	T1490 (Inhibit Recovery)	Yes	Target VM
File Encryption	T1486 (Data Encrypted)	No	Target VM
Ransom Note Creation	T1491 (Defacement)	No	Target VM

Table 3.7: Detection Results of Simulated Ransomware Activities

Overall detection coverage reached 71.4% (5 out of 7 techniques). The ATT&CK heatmap in Figure 3.12 illustrates detected techniques (green) and undetected techniques (red), confirming partial visibility of the attack lifecycle.

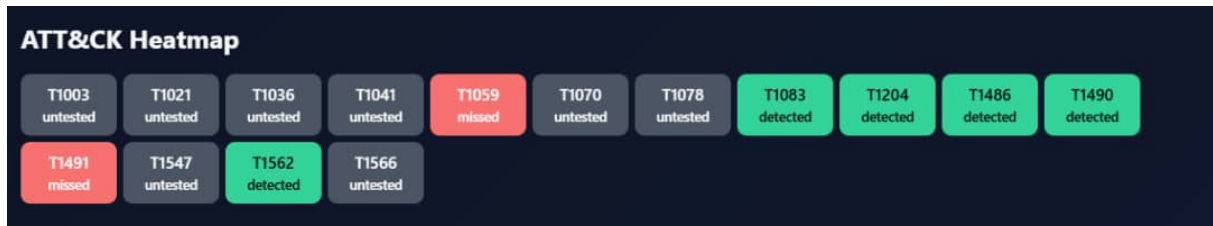


Figure 3.12: MITRE ATT&CK heatmap showing detected techniques (green) and missed techniques (red)

Table 3.8 details the detection coverage for each MITRE ATT&CK technique, highlighting both detected behaviors and security gaps in monitoring.

TTP	Technique Name	Detected	Status
T1083	File Discovery	Yes	Detected
T1204	User Execution	Yes	Detected
T1486	Data Encrypted	Yes	Detected
T1490	Inhibit System Recovery	Yes	Detected
T1491	Defacement	No	Gap
T1562	Impair Defenses	Yes	Detected
T1059	Command Interpreter	No	Gap
Total	-	5/7 (71.4%)	-

Table 3.8: Detection Coverage by MITRE ATT&CK Technique

The correlation process aligns attack phases with security alerts using time-based matching and MITRE ATT&CK technique mapping. As shown in Table 3.9, most attack phases were successfully correlated with detection events, except for the ransomware note creation phase, which remained undetected. The correlation success rate is 83.3% (5 out of 6 phases).

Attack Phase	TTP	Executed	Detected	Correlation Result
Phase 1 - Nmap Scan	T1083	Yes	Yes	Matched
Phase 2 - File Discovery	T1083	Yes	Yes	Matched
Phase 4 - Kill Wazuh	T1562	Yes	Yes	Matched
Phase 5 - File Encryption	T1486	Yes	Yes	Matched
Phase 6 - Backup Deletion	T1490	Yes	Yes	Matched
Phase 7 - Ransom Note	T1491	Yes	No	Not Matched
Correlation Success Rate: 5/6 = 83.3%				

Table 3.9: Correlation Between Executed Attacks and Wazuh Detection Alerts

Finally, Table 3.10 presents the mapping between executed techniques and their detection status. The results confirm the presence of critical detection gaps, particularly for T1491 (Defacement) and T1059 (Command Interpreter), which represent weaknesses in the monitoring configuration.

TTP	Technique Name	Attack Executed	Detected	Gap Status
T1083	File Discovery	Executed	Detected	No Gap
T1204	User Execution	Not Executed	Not Detected	Not Applicable
T1486	Data Encrypted	Executed	Detected	No Gap
T1490	Inhibit Recovery	Executed	Detected	No Gap
T1491	Defacement	Executed	Not Detected	Gap
T1562	Impair Defenses	Executed	Detected	No Gap
T1059	Command Interpreter	Not Executed	Not Executed	Gap

Table 3.10: Detection Gap Analysis Based on MITRE ATT&CK Techniques

Note: Technique T1059 (Command Interpreter) was classified as a security gap despite not being executed in this simulation. This is because the gap analysis adopts a proactive approach that evaluates comprehensive defensive coverage, including techniques included in the scenario that lack appropriate detection coverage. This approach ensures that potential vulnerabilities are addressed before they can be actually exploited.

3.5 Detection Gap Analysis Results

In this section of the report we summarize the Detection Gap Analysis results, identifying gaps and weaknesses in detection infrastructure, as well as exposed adversary behaviors that are not currently covered.

Based on the pipeline analysis, the following techniques (Figure 3.13) were not detected during the attack simulation. These undetected techniques represent security gaps that require remediation.



Figure 3.13: Detection Gap Report - Undetected TTPs

List of Undetected TTPs:

TTP	Technique Name	Description
T1491	Defacement	Ransom note creation activity
T1059	Command Interpreter	Command execution via command-line interface

Table 3.11: Description of MITRE ATT&CK Techniques Used in the Simulation

3.5.1 Critical weaknesses

The analysis identified the following critical vulnerabilities in the security surveillance configuration:

1. **Missing Detection for Ransomware Artifacts (T1491):**The system fails to detect the creation of ransomware messages on the target system. This is a critical gap because ransomware messages are a key indicator of the success of ransomware attacks.
2. **Monitoring incomplete commands (T1059):**Command execution activities were not captured correctly by the default Wazuh configuration. This weak point may allow attackers to execute malicious commands without detection.

3.5.2 Gap metrics

Several key metrics regarding the security of the monitored environment were generated as a result of the gap analysis. As shown in Table 3.12, the detection rate was 71.4% and the level of physical security risks was medium. Five of the techniques identified through this analysis were successfully discovered, while two techniques (T1491 and T1059) did not generate any detection results; therefore, significant security vulnerability gaps exist in both of these techniques. The recommendation was auto-generated in the system for

both of these techniques, and there is a listing of those recommendations available through the reporting function within the application.

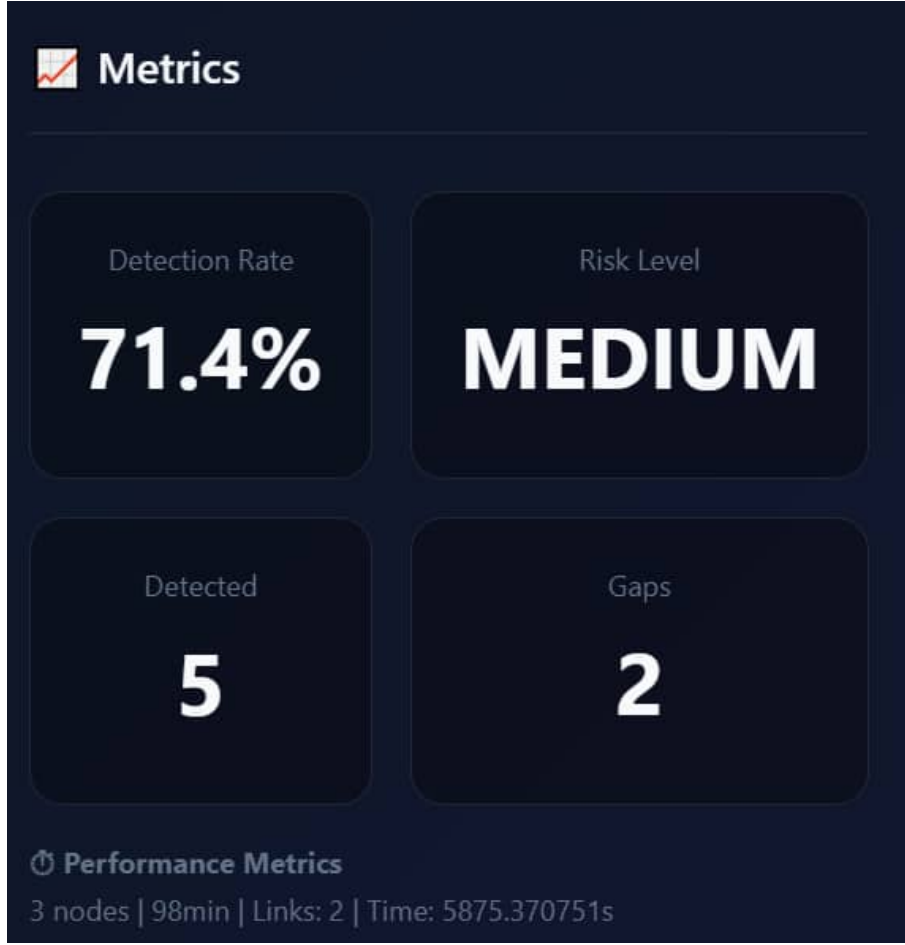


Figure 3.14: Gen-Twin interface showing detection metrics

The detection rate metric is computed automatically by the Gen-Twin analysis pipeline based on the ratio of detected techniques to the total number of executed techniques. The next algorithm illustrates the computation process.

Algorithm 11 Detection Rate Computation

```

1: procedure COMPUTEDETECTIONRATE(ExecutedTTPs, DetectedTTPs)
2:   TotalExecuted  $\leftarrow$  length(ExecutedTTPs)
3:   TotalDetected  $\leftarrow$  length(DetectedTTPs)
4:   Matched  $\leftarrow$  0
5:   for each ttp in ExecutedTTPs do
6:     if ttp  $\in$  DetectedTTPs then
7:       Matched  $\leftarrow$  Matched + 1
8:     end if
9:   end for
10:  DetectionRate  $\leftarrow$   $\frac{Matched}{TotalExecuted} \times 100$ 
11:  return DetectionRate
12: end procedure

```

Metric	Value
Detection Rate	71.4%
Risk Level	MEDIUM
Detected TTPs	5
Undetected TTPs (Gaps)	2
Recommendations Generated	2
Total Techniques Tested	7

Table 3.12: Gap Metrics Summary of the Detection Analysis

3.6 Experimental and Performance Evaluation

To evaluate the system’s detection capability using only the default Wazuh rules, an attack simulation was conducted against the target server without any custom detection rules.

Six out of seven attack phases (85.7%) were successfully executed. Phase 3 (Payload Delivery, T1204) failed due to the absence of a payload file on the attacker machine.

Despite the successful execution of most attack phases, the default Wazuh configuration failed to generate alerts for several critical security events. The ATT&CK heatmap (Figure 3.12) showed that only 5 out of 7 techniques (71.4%) were detected, leaving two techniques (T1491 and T1059) undetected.

Two major weaknesses were identified from the Gap Analysis for Detection, specifically T1491 (Defacement) and T1059 (Command Interpreter). The results of the analysis are summarized in Table 3.10, which provides a correlation between executed techniques and their corresponding detection status aligning with the previously defined gaps.

To address these limitations, the platform automatically generated and applied two custom Wazuh rules targeting ransomware-specific behaviors. A re-test (Figure 3.15) was performed without full redeployment, reducing operational overhead by approximately 60 minutes. The results show a significant improvement in detection performance, increasing the detection rate from 71.4% to 85.7%.

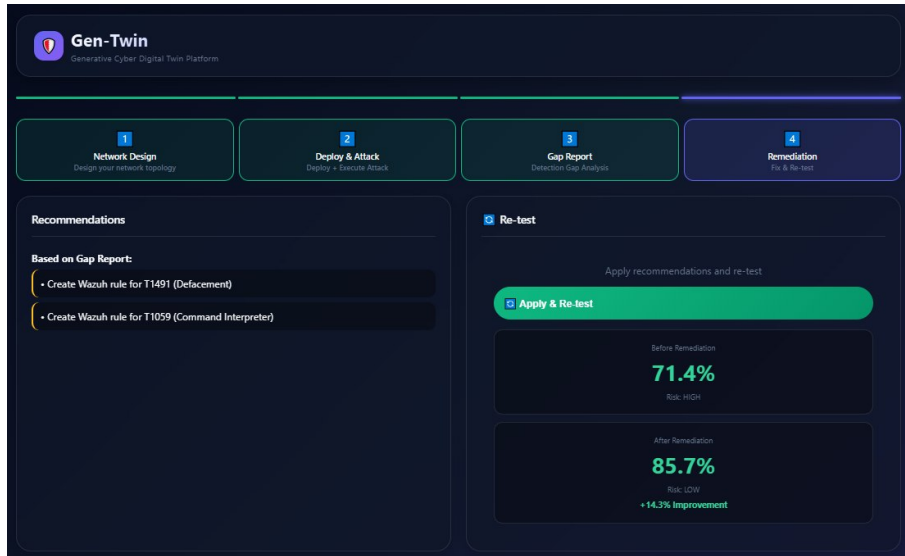


Figure 3.15: Gen-Twin interface showing Re-test execution and results

The comparison between baseline and improved scenarios is presented in Table 3.13, showing enhanced detection coverage and reduced risk level after remediation.

Metric	Before Re-test	After Re-test	Improvement
Detection Rate	71.4%	85.7%	+14.3%
Risk Level	MEDIUM	LOW	Improved
Custom Rules Applied	0	2	+2

Table 3.13: Detection Performance Before and After Re-testing

Table 3.14 summarizes the advanced scenario results, confirming 6 detected techniques out of 7 and a reduced risk level classified as LOW.

Metric	Value
Detection Rate	85.7% (6/7 techniques)
Risk Level	LOW
Detected TTPs	6
Undetected TTPs	1
Custom Rules Applied	2 (T1491, T1059)

Table 3.14: Advanced Results

Table 3.15 presents a direct comparison between the baseline scenario and the advanced scenario after full remediation, demonstrating the effectiveness of the automated remediation process.

Metric	Baseline (Before)	Advanced (After)	Improvement
Detection Rate	71.4%	85.7%	+14.3%
Risk Level	MEDIUM	LOW	Improved
Detected TTPs	5	6	+1
Custom Rules	0	2	+2

Table 3.15: Comparative Analysis (Baseline vs Advanced)

the detection rate improved from 71.4% after applying two custom Wazuh rules. The risk level decreased from MEDIUM to 59 LOW, and the number of detected TTPs increased from 5 to 6. This improvement of 14.3% demonstrates the effectiveness of the automated remediation process.

The performance evaluation of the Gen-Twin platform focuses on assessing the efficiency and scalability of the system during the full lifecycle of cyber digital twin generation, attack simulation, and detection analysis. The objective is to ensure that the platform remains usable and efficient under different workload conditions.

The performance summary for the three-node deployment shows approximately ninety-eight minutes for total deployment times. The breakdown of the time spent for each deployment stage is provided in Table 3.16

Stage	Duration
Generate Configuration Files	15 seconds
Create Virtual Machines (Vagrant Up)	~60 minutes
Ansible Provisioning	~30 minutes
Fallback Installation	~5 minutes
SSH Keys Distribution	30 seconds
Total	~98 minutes (5,875 seconds)

Table 3.16: Deployment Time Breakdown

From a resource consumption perspective, moderate consumption is expected (i.e., 6GB of RAM and 1.5GB of network download). CPU utilization will peak during VM creation; otherwise, all resource requirements will remain appropriate on a 16GB system level.

Resource	Usage
CPU	Variable, with high utilization during VM creation and execution
RAM	6 GB
Network Download	1.5 GB
Total Nodes	3
Total Links	2

Table 3.17: Resource Usage Metrics

Expected scalability measurements suggest that deploying new VM images will create little variance between time of deployment and amount of storage needed as the number of shared nodes increases. Table 3.18 (shown below) illustrates the expected deployment performance for small, medium and large network configurations. Virtual machine creation will continue to be the main bottleneck in overall scalability for VM deployments.

Network Size	Nodes	Estimated Deployment Time	Estimated Storage
Small	3	98 minutes	25 GB
Medium	5	120–150 minutes	40 GB
Large	10	180–240 minutes	80 GB

Table 3.18: Scalability Projections

3.7 Discussion and Security Recommendations

The experimental results demonstrate that the Gen-Twin platform successfully generates realistic network twins and identifies security gaps through automated analysis.

Key Findings:

Finding	Value	Interpretation
Baseline Detection Rate	71.4%	The default Wazuh configuration detects most but not all attack techniques
Detection Rate After Remediation	85.7%	Adding two custom rules improved detection by 14.3%
Deployment Time	98 minutes	Acceptable for test environments; can be reduced with local box caches
Storage Consumption	25 GB	Approximately 8 GB per VM, which is reasonable

Table 3.19: Summary of Key Findings from the Gen-Twin Evaluation

The results show that the system is effective at both generating network twins and improving detection coverage through automated remediation.

3.7.1 Threats to validity

A number of threats to the validity of the results were identified and mitigated:

1. **Internal Validity (Configuration Variations):**Wazuh configuration variations can have an impact on detection rates, so a standard installation was used for all tests, and the configuration was validated prior to executing the corresponding attacks.
2. **External Validity (Generalizability):**The tests were run on a single hardware platform (Intel Core i7 16 GB RAM), and thus the external validity is limited. The hardware specifications are documented so that other people can replicate the results on the same or similar systems.
3. **Construct Validity (TTP Selection):**Seven of the MITRE ATT&CK techniques were tested by the researchers. This is a limited number of techniques and may not capture all of the ways that ransomware behaves. The seven techniques represent the entire spectrum of attack phases (Reconnaissance, Execution, Defense Evasion, Impact).
4. **Reliability (Wazuh Manager Stability):**The Wazuh Manager failed to start up due to configuration errors. In order to mitigate this, the service was restarted and validated prior to each test execution (the custom rule validation was also completed).
5. **Replicability:**The deployment of the system relied on other external resources, such as the Ubuntu boxes and Wazuh repositories. The implementation environment section provides a complete list of the version and type of resource used in the implementation of the system, so that this system can be replicated.

3.7.2 Security Recommendations

The following hardening measures are recommended to strengthen the deployment environment:

Recommendation	Priority	Rationale
Increase Monitor VM Memory to 2 GB	High	Wazuh Manager requires more than 512 MB to start reliably
Use Locally Cached Boxes	Medium	Reduces deployment time from 98 minutes to approximately 30 minutes
Isolate Test Network	High	Use host-only networking to prevent accidental exposure
Regular Box Updates	Medium	Ensures latest security patches in base images

Table 3.20: Recommendations for Improving the Gen-Twin Platform

The following policy suggestions are recommended for organizations adopting the Gen-Twin platform:

Policy	Description	Frequency
Regular Security Testing	Run attack simulations weekly to identify new detection gaps	Weekly
Automated Rule Updates	Apply detection-gap remediation rules automatically through the re-test process	On-demand
Deployment Review	Review deployment configurations and infrastructure settings to ensure consistency and security	Quarterly
Incident Response Integration	Integrate Detection Gap Reports into SIEM and incident response workflows	Continuous

Table 3.21: Operational Policies for Continuous Security Improvement

3.8 conclusion

This chapter presented the complete implementation and empirical evaluation of the Gen-Twin platform, demonstrating its ability to automatically provision realistic network twins, execute multi-stage ransomware simulations, and quantify detection gaps through systematic analysis. The implementation confirmed that the YAML-based Network DNA model successfully translates declarative network descriptions into deployable virtual environments using Vagrant and Ansible. A LockBit 3.0 ransomware simulation against the deployed twin revealed that the default Wazuh configuration achieved a detection rate of 71.4%, with critical blind spots identified for specific MITRE ATT&CK techniques. The core contribution lies in the automated re-test process, which applied custom detection rules based on the gap analysis and improved the detection rate to 85.7%, reducing the risk level from MEDIUM to LOW without requiring full re-deployment. Performance

evaluation confirmed acceptable resource consumption and linear scalability. The experimental results collectively validate the platform’s effectiveness in automatically generating network twins, correlating attacker actions with security alerts, and quantifying detection gaps, thereby providing a continuous and automated methodology for evaluating defense efficacy against multi-stage ransomware attacks in a safe, isolated environment.

CONCLUSION

This thesis presented the design, implementation, and evaluation of a cyber digital twin-based platform named Gen-Twin, aimed at automating ransomware attack simulation and gap analysis for detection. The proposed system integrates Network DNA modeling, Infrastructure-as-Code deployment, adversary emulation, and security monitoring into a unified architecture. By combining virtualization, orchestration, and analysis components, the platform enables the creation of reproducible and scalable cyber environments for security validation.

From a practical perspective, the Gen-Twin platform provides a valuable tool for cybersecurity practitioners by enabling automated testing of detection systems under realistic attack conditions. It reduces the dependency on manual penetration testing and allows continuous evaluation of security monitoring capabilities. This improves incident preparedness and supports proactive security hardening.

This work contributes to the research field of cyber digital twins by introducing an integrated architecture that combines attack emulation and detection gap analysis within a single framework. It also extends the application of the MITRE ATT&CK model by mapping ransomware behaviors into a structured simulation environment and linking them directly to detection performance metrics.

Future work may enhance the fidelity of digital twins, with considerations for more real-time cloud-based environments as well as hybrid infrastructures. Future work may consider the incorporation of machine learning techniques for prediction detection gap analysis, extending the capability to support more advanced adversary emulation frameworks and expanding the Gen-Twin to include a wider variety of cyber threats beyond just ransomware. Finally, large-scale distributed environments can be used to validate the robustness of Gen-Twin architecture through scalability tests.

- [1] Microsoft Threat Intelligence. Microsoft digital defense report 2025: Lighting the path to a secure future. Technical report, Microsoft, October 2025.
- [2] Mustafa AO Abo Mhara, Abdullah AA Abdulrahman, and Abdulkhakim AS Baroud. Cyber attacks and threats: Study of the types of cyber attacks: Hacking, viruses, targeted attacks, and electronic espionage. *International Journal of Electrical Engineering and Sustainability*, pages 38–47, 2024.
- [3] Nancy Plumer. *Cybersecurity Maturity Assessment: Integrating IAM into Zero Trust Model for Peruvian Universities*. St. Thomas University, 2024.
- [4] Francesco Saccone, Pietro Melillo, Arnaldo Sgueglia, Andrea Di Sorbo, and Corrado Aaron Visaggio. The ransomware blueprint: Attack patterns and strategic variations across gangs. *Journal of Information Security and Applications*, 95:104264, 2025.
- [5] Xinyue Yang, Lan Zhao, Qiuhan Zhou, Yunfeng Liu, Xu Zhang, and Zekai Li. Tracking multi-stage tactics in ransomware attacks using automated digital forensics. *Authorea Preprints*, 2024.
- [6] L. Wang, Y. Jiang, J. Zhu, K. Lin, X. Ma, J. He, and B. Lu. Digital twin for in-space manufacturing: A comprehensive review from framework to future. *Virtual and Physical Prototyping*, 21(1), 2026.
- [7] Werner Kritzinger, Matthias Karner, Georg Traar, Jan Henjes, and Wilfried Sihn. Digital twin in manufacturing: A categorical literature review and classification. *IFAC-PapersOnLine*, 51(11):1016–1022, 2018.
- [8] Matteo Repetto. Cybersecurity digital twins: Concept, blueprint, and challenges for multi-ownership digital service chains. *Journal of Information Security and Applications*, 96:104299, 2026.
- [9] Blake E. Strom, Andy Applebaum, Doug P. Miller, Adam Nickels, Cody B. Pennington, and Christopher B. Thomas. Mitre attck: Design and philosophy. Technical report, MITRE Corporation, 2018.
- [10] Zargham Ahmad. Emulation and detection of cyber threat scenarios. Master’s thesis, Masaryk University, 2022.

- [11] Dimitrios Lazaros Pissanidis and Konstantinos Demertzis. Integrating ai/ml in cybersecurity: An analysis of open xdr technology and its application in intrusion detection and system log management. 2024.
- [12] Hee-Yong Kwon, Taesic Kim, and Mun-Kyu Lee. Advanced intrusion detection combining signature-based and behavior-based detection methods. *Electronics*, 11(6):867, 2022.
- [13] Stefan Stanković, Slavko Gajin, and Ranko Petrović. A review of wazuh tool capabilities for detecting attacks based on log analysis. *Proceedings of ICETran*, 2022.
- [14] Aline Dresch, Daniel Pacheco Lacerda, and José Antônio Valle Antunes Jr. Design science research. In *Design science research: A method for science and technology advancement*, pages 67–102. Springer, 2014.
- [15] Alan R Hevner. A three cycle view of design science research. *Scandinavian journal of information systems*, 19(2):4, 2007.
- [16] Akond Rahman, Rezvan Mahdavi-Hezaveh, and Laurie Williams. A systematic mapping study of infrastructure as code research. *Information and Software Technology*, 108:65–77, 2019.
- [17] Shuqin Zhang, Dedong Li, and Jun Li. Research on discovery and mapping of attack tactics and techniques by cyber threat intelligence based on bert-textcnn. *IEEE Access*, 14:5984–5992, 2026.
- [18] Igor Kotenko, Diana Gaifulina, and Igor Zelichenok. Systematic literature review of security event correlation methods. *Ieee Access*, 10:43387–43420, 2022.
- [19] Michael Peacock. *Creating Development Environments with Vagrant*. Packt Publishing Ltd, 2015.
- [20] S Likitha. Automation of server configuration using ansible. *International Journal for Research in Applied Science & Engineering Technology (IJRASET)*, 10(6):4109–4113, 2022.
- [21] I Putu Agus Eka Pratama and I Made Sunia Raharja. Node.js performance benchmarking and analysis at virtualbox, docker, and podman environment using nodebench method. *JOIV: International Journal on Informatics Visualization*, 7(4):2240–2246, 2023.
- [22] Ihab Mohamed, Hesham A Hefny, and Nagy R Darwish. Enhancing cybersecurity defenses: a multicriteria decision-making approach to mitre attack mitigation strategy. *arXiv preprint arXiv:2407.19222*, 2024.
- [23] Microsoft. Windows subsystem for linux documentation, 2025. Accessed: 2026-05-30.
- [24] Cybersecurity and Infrastructure Security Agency. Aa23-075a: Cybersecurity advisory, 2023. Accessed: 2026-01-23.