HAW HAMBURG

BACHELORARBEIT

Social Engineering und die Simulation selbstreplizierender Schadsoftware: Eine Analyse von Abwehrstrategien in virtuellen Netzwerken

vorgelegt am 04. Februar 2026
von Sophia Bosnak

Erstprüfer: Prof. Dr. Nils Martini
Zweitprüfer: Sebastian Sünkler

HOCHSCHULE FÜR ANGEWANDTE WISSENSCHAFTEN HAMBURG
Department Medientechnik
Finkenau 35
20081 Hamburg

Zusammenfassung

Die vorliegende Arbeit untersucht die Entstehung und Entwicklung zentraler Schadsoftware-Arten wie Viren, darunter Würmer und Trojaner. Die Rolle von Social Engineering zur Verbreitung dieser Programme innerhalb von IT-Netzwerken wird ebenfalls analysiert. Ziel der Arbeit ist es, zu untersuchen, inwiefern gezielte Manipulation menschlichen Verhaltens die Verbreitung von Schadsoftware begünstigt. Hierzu wurde eine simulationsbasierte Umgebung in einem virtualisierten Netzwerk aufgebaut. In dieser wurde ein selbstreplizierendes Programm über eine manipulativ gestaltete E-Mail und eine modellierte Nutzerinteraktion verbreitet.

Die Ergebnisse der Simulation zeigen, dass bereits eine einfache Nutzerhandlung ausreicht, um die initiale Infektion und die Weiterverbreitung der Schadsoftware innerhalb des Netzwerks zu ermöglichen. Die Arbeit verdeutlicht damit die potenzielle Relevanz von Social Engineering als Angriffsvektor und unterstreicht die Notwendigkeit, den menschlichen Faktor auch in technisch ausgerichteten Sicherheitskonzepten systematisch zu berücksichtigen.

Abstract

This paper examines the emergence and development of key types of malware, such as viruses, including worms and Trojans. The role of social engineering in spreading these programs within IT networks is also analyzed.

The aim of the thesis is to investigate the extent to which targeted manipulation of human behavior promotes the spread of malware. For this purpose, a simulation-based environment was set up in a virtualized network. In this environment, a self-replicating program was spread via a manipulatively designed email and a modeled user interaction.

The results of the simulation show that even a simple user action is sufficient to enable the initial infection and further spread of the malware within the network. The thesis thus highlights the potential relevance of social engineering as an attack vector and underscores the need to systematically consider the human factor even in technically oriented security concepts.

Inhaltsverzeichnis

Abkürzungsverzeichnis

BS	Betriebssystem
MBRC	Master-Boot-Record Code
SE	Social Engineering
APT	Advanced Persistent Threat
NB65	Network Batallion 65
BSI	Bundesamt für Sicherheit in der Informationstechnik
ILY	ILOVEYOU
RAT	Remote Administration Trojaner
ATD	AIDS TROJAN DISK
DDoS	Distributed Denial-of-Service
SMTP	Simple Mail Transfer Protocol
SPF	Sender Policy Framework
DNS	Domain Name System
DKIM	DomainKeys Identified Mail
DMARC	Domain-based Message Authentication, Reporting and Conformance
GANs	Generative Adversarial Networks

Abbildungsverzeichnis

Tabellenverzeichnis

1 Einleitung

1.1 Problemstellung

Rund jeder fünfte Sicherheitsvorfall in IT-Infrastrukturen wird durch die gezielte Manipulation menschlichen Verhaltens ermöglicht [1]. Social-Engineering-Techniken dienen dabei als zentrales Einfallstor, um unwissende Nutzer gezielt dazu zu bringen, Schadsoftware wie Viren unbemerkt auf ihre Systeme zu übertragen und Netzwerke zu infizieren [2]. Social Engineering (SE) bezeichnet dabei besondere Angriffsstrategien, bei denen durch psychologische Beeinflussung sicherheitskritische Handlungen ausgelöst oder sensible Informationen preisgegeben werden.

Trotz der bekannten Bedrohungslage mangelt es an ausreichender Aufklärung und Sensibilisierung der Nutzer. Eine Studie zeigt exemplarisch, dass ein Großteil der Befragten grundlegende Schutzmaßnahmen hinsichtlich Schadsoftware und manipulativer Inhalte nicht einsetzt (78 %) oder deren Relevanz unterschätzt (72 %) [3]. Selbstreplizierende Programme können durch solche Fehlannahmen ganze Netzwerke infizieren und erhebliche Schäden verursachen.

1.2 Ziel der Arbeit

Das Ziel dieser Bachelorarbeit ist es, die Rolle von SE bei der erfolgreichen Verbreitung von Schadsoftware, insbesondere bei selbstreplizierenden Schadprogrammen, zu analysieren. Daraus sollen umfassende Anforderungen für wirksame Schutzmaßnahmen abgeleitet werden, die technische als auch menschliche Faktoren berücksichtigen.

1.3 Vorgehensweise

Zur Untersuchung des dargestellten Sachverhalts wird eine Simulation mit drei virtuellen Maschinen eingesetzt, in der ein Angreifer-Rechner Bestandteil des Netzwerks ist. Dieser versendet manipulierte E-Mails mit einem als PDF getarnten ausführbaren Anhang. Der Anhang stellt ein selbstreplizierendes Programm dar, das sich nach einer Nutzerinteraktion automatisiert

weiterverbreitet. Anhand dieses Szenarios wird das Zusammenspiel von SE und selbstreplizierenden Programmen exemplarisch untersucht.

1.4 Aufbau

Die vorliegende Arbeit gliedert sich wie folgt: Zu Beginn wird die historische Entwicklung von Malware dargestellt sowie auf verschiedene damit einhergehende Bedrohungsszenarien eingegangen.
Danach werden Viren als übergeordnete Schadsoftwarekategorie sowie Würmer und Trojaner eingeordnet, um deren Funktionsweisen, Ausbreitungsmechanismen und Schadwirkung zu verdeutlichen.
Aufbauend darauf wird ein SE-Szenario entworfen und in einer virtualisierten Netzwerkumgebung umgesetzt. In diesem Kontext soll sich ein selbstreplizierendes Programm über E-Mail-Versand ausbreiten. Anhand dieser Simulation wird analysiert, inwiefern Nutzerinteraktionen zu einer Infektion beitragen können und welche Auswirkung dies auf die Ausbreitung innerhalb eines Netzwerks hat.
Danach stellt die Arbeit technische und organisatorische IT-Sicherheitsmaßnahmen dar und ordnet deren Anwendbarkeit gegenüber SE-basierten Angriffen und selbstreplizierenden Programmen ein. Zusätzlich werden mögliche zukünftige Weiterentwicklungen dargestellt.
Abschließend werden die Ergebnisse der Simulation diskutiert und mit aktuellen Forschungsergebnissen verglichen. Auf dieser Grundlage wird im Fazit die Wirkung von SE in Verbindung mit Schadprogramm-Angriffen reflektiert und ein Ausblick auf mögliche zukünftige Forschungsansätze gegeben.

2 Malware-Entwicklung

Die Entwicklung moderner Malware zeigt Ähnlichkeiten mit traditionellen Methoden der militärischen und politischen Einflussnahme. Klassische Spionage-, Sabotage- und Täuschungspraktiken wurden im Zuge der Digitalisierung zunehmend ins Internet verlagert und technisch erweitert [4]. Das Ziel dieser Maßnahmen ist die Gewinnung von Informationen, die Beeinflussung von Akteuren sowie die Störung gegnerischer Systeme. Die zunehmende Vernetzung

und Digitalisierung gesellschaftlicher sowie technischer Strukturen begünstigen die Anwendung solcher Strategien auf IT-basierte Infrastrukturen. Vor diesem Hintergrund lässt sich Malware als Produkt dieser Entwicklung betrachten, dessen Komplexität, Verbreitung und Wirkung durch den digitalen Wandel stetig zugenommen haben.

Dieser Wandel steht im engen Zusammenhang mit der zunehmenden Verbreitung privater Rechner und der Erschaffung des Internets. In den 1960er-Jahren waren große Rechenzentren ausschließlich für Universitäten und Forschungseinrichtungen zugänglich [5]. Zusätzlich war der Zugang zu ihnen nur physisch möglich.
Dies erschwerte es Außenstehenden erheblich, Zugriff auf diese Systeme und ihre Funktionsweisen zu erhalten. Nur wenige Personen kannten die genauen Standorte dieser Technologie und noch weniger wussten über ihre konkrete Arbeitsweise Bescheid.
Trotz der anfänglich begrenzten Verbreitung und Verfügbarkeit von Computern entwickelten sich Programme, die sich eigenständig innerhalb der damaligen Netzwerke verbreiten konnten. Damit gehörten selbstreplizierende Programme zu den ersten experimentellen Formen von Schadsoftware.
Der *Creeper*, aus dem Jahr 1971, gilt als eine der ersten bekannten schädlichen Programme [6]. Es wurde so entworfen, dass es sich selbstständig über das ARPANET, einem Vorläufer des heutigen Internets, verbreiten konnte. Die Ausführung des *Creepers* führte lediglich zu geringfügigen Leistungseinbußen und hatte ansonsten keine gefährlichen Auswirkungen.
Stattdessen zeigte das Programm die Nachricht: „I'M THE CREEPER: CATCH ME IF YOU CAN." [6, S. 2] auf betroffenen Systemen an. Als Reaktion darauf wurde mit *Reaper* ein weiteres Programm entwickelt, das gezielt nach *Creeper*-Kopien suchte und diese entfernte. Damit galt es als eines der ersten erschaffenen Antivirusprogramme.

Die Entwicklung von vereinzelten Rechenzentren hin zu massenproduzierten Personal Computern (PC) ermöglichte erstmals einen weitreichenden Einsatz digitaler Systeme im Arbeitsalltag. Mit der Einführung des World Wide Webs ab 1989 verlagerte sich zudem der Einsatz von PCs in die private Freizeitgestaltung [5]. Diese Entwicklungen führten zu einer erstmals großflächigen, vernetzten digitalen Welt, in der Daten über weite Distanzen hinweg ausgetauscht werden konnten. Informationen und Ressourcen wurden dadurch für viele Parteien in großer Menge zugänglich.

Wo solche Datenbestände entstehen, ergeben sich zugleich wirtschaftliche und strategische Interessen. Infolgedessen rückten digitale Systeme verstärkt in den Fokus krimineller und staatlicher Organisationen, was zu einem fortlaufenden Wettbewerb zwischen diesen Beteiligten führte. Insbesondere verfolgen Unternehmen sowie staatliche Institutionen das Ziel, den Zugang zu sensiblen Informationen zu beschränken und ihren Missbrauch zu verhindern. Während sich dieser Wettlauf intensivierte und digitale Umgebungen immer stärker ausgebaut wurden, kam es zu einer grundlegenden Veränderung der Organisation von Angreifern. An Stelle von einzelnen Hackern oder kleineren Gruppen traten zunehmend organisierte und professionell arbeitende cyberkriminelle Netzwerke in Erscheinung [7]. Ihre Schadprogramme werden seitdem nicht mehr nur experimentell entwickelt, sondern zunehmend gezielt eingesetzt, um wirtschaftliche sowie politische Ziele zu erreichen.

Ab den frühen 2000er-Jahren kamen zunehmend sogenannte *Advanced Persistent Threats* (APTs) zum Einsatz [6]. Dabei handelt es sich um langfristige Angriffe, bei denen sich unbemerkt Zugang zu einem System verschafft wird, dort über längere Zeit verbleibt und schrittweise vertrauliche Daten abgreift oder manipuliert. Ein bekanntes Beispiel hierfür ist der *Stuxnet*-Wurm, der 2010 entdeckt wurde. Er griff iranische Industrieanlagen an, um das Atomprogramm des Landes gezielt zu beeinträchtigen [6], [8]. *Stuxnet* gilt daher als eine der ersten Cyberattacken mit Kriegspotenzial, da sie verdeutlichte, wie verwundbar staatliche Infrastrukturen gegenüber digitalen Angriffen sein können [9].
APTs kennzeichnen somit den Übergang von vereinzelten Angriffen zu hochgradig professionellen Attacken.
Politische Gruppen setzen solche Aktionen oft für einen anderen Zweck ein. Diese nutzen Cyberangriffe dazu, Aufmerksamkeit zu erzeugen oder politische Botschaften zu verbreiten. Ein Beispiel hierfür ist die Gruppe *Network Battalion 65* (NB65), die in Verbindung mit Anonymous steht. Im März 2022 erklärten sie öffentlich, dass sie die staatliche russische Weltraumagentur "ROSCOSMOS" angegriffen haben sollen [10]. Im Zuge dieses Angriffs wurde zeitweise die Fähigkeit der Behörde, Satellitenaufnahmen zu generieren, außer Kraft gesetzt. Zudem wurden über eine ausgenutzte Schwachstelle im System verschiedene schädliche Programme eingeschleust, um der staatlichen Infrastruktur nachhaltig zu schaden. Die vermeintlich erfolgreiche Operation wurde im Anschluss durch veröffentlichte Bildbeweise auf Twitter dokumentiert [11].

Abbildung 1 zeigt einen Screenshot eines manipulierten Terminals der Weltraumagentur und stellt die Motivation von NB65 hinter dem Angriff dar: die Störung der militärischen Nutzung von Satelliten im Angriffskrieg gegen die Ukraine sowie die Erzeugung internationaler Aufmerksamkeit für den Konflikt.

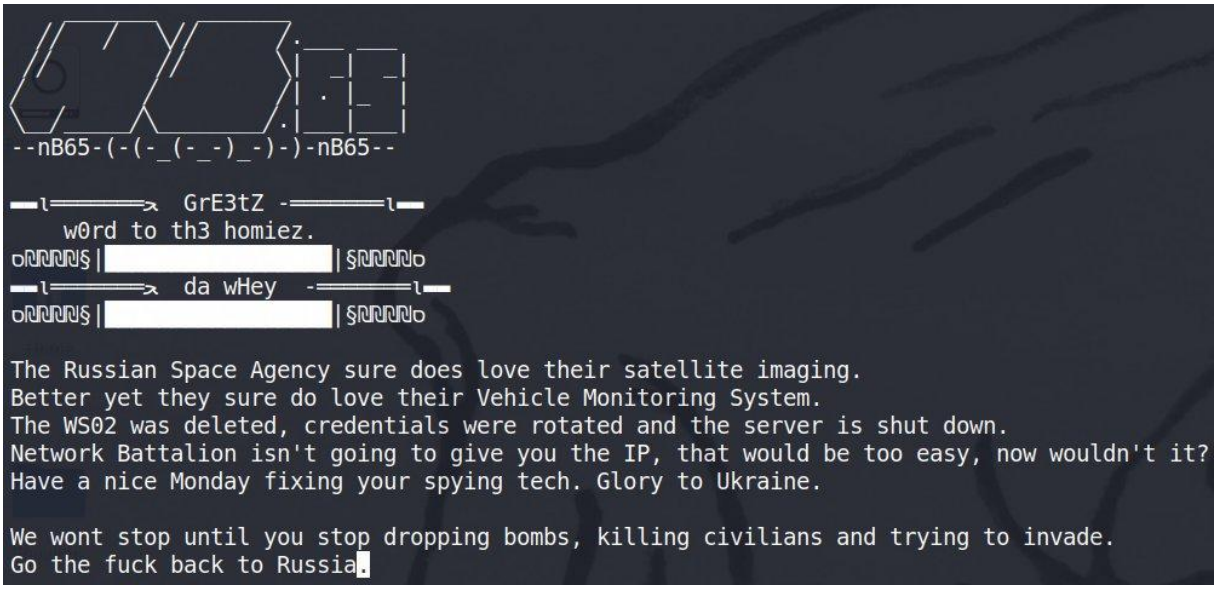


Abbildung 1: NB65 Twitter-Screenshot
Quelle: [11]

Während solche Angriffe auf die Erregung öffentlicher Aufmerksamkeit abzielen sollen, verfolgt die digitale Spionage in der Regel langfristige und verdeckte Ziele. Spionage findet heutzutage vor allem online statt und richtet sich auf den dauerhaften Zugriff übernommener Systeme und Daten, der durch Schadsoftware ermöglicht wird.

Ein Beispiel hierfür ist der koordinierte Cyberangriff auf die ukrainische Stromversorgung im Jahr 2015. Schadsoftware wurde hierbei gezielt eingesetzt, um Teile der Infrastruktur lahmzulegen und rund 225.000 Menschen zeitweise den Zugang zu Elektrizität zu verwehren [4]. Der Angriff auf die US-amerikanische Personalverwaltung, bei dem 21,5 Millionen Datensätze manipuliert wurden, verdeutlicht zusätzlich die Rolle von Malware bei Spionageoperationen gegen staatliche Infrastrukturen [4].

Gleichzeitig gewinnt die systematische Verbreitung falscher oder manipulierter Informationen zunehmend an Bedeutung. So schätzte die Internet Research Agency, dass rund 126 Millionen Facebook-Nutzer im Zusammenhang mit der US-Wahl 2016 gezielt mit widersprüchlichen po-

litischen Inhalten konfrontiert wurden [4]. Solche Einflussoperationen kombinieren technische Mittel, beispielsweise beeinträchtigte Systeme oder automatisierte Software, mit psychologischen Manipulationstechniken, um ihre Erfolgschancen zu verstärken.
Insgesamt zeigen moderne Cyberoperationen eine Kombination aus technischen Angriffen und sozialer Manipulation. Schadsoftware bildet dabei häufig die Grundlage, um Systeme zu kontrollieren, Informationen zu sammeln und gezielte Einflussmaßnahmen über soziale Medien vorzubereiten. [4].
Moderne digitale Angriffe beschränken sich damit nicht mehr ausschließlich auf die Manipulation technischer Systeme. Vielmehr wird sie mit Informations- und Beeinflussungsstrategien kombiniert. Dazu zählen das Veröffentlichen sensibler Daten, die Fälschung von Dokumenten oder die gezielte Verbreitung emotionalisierender Inhalte. Diese Vorgehensweisen gelten als kostengünstig, gut erweiterbar und schwer nachzuverfolgen, was sie sowohl für staatliche als auch für nichtstaatliche Gruppen strategisch attraktiv macht [4].

Während Malware in staatlichen und politischen Kontexten häufig als Mittel zur Spionage oder Einflussnahme eingesetzt wird, spielt sie im zivilen Bereich vor allem eine wirtschaftliche Rolle. Malware wird dabei nicht nur zur Verfolgung strategischer Ziele genutzt, sondern zunehmend als direkte Einnahmequelle.
Mit der Monetarisierung schädlicher Programme ist die Motivation zur Entwicklung und zum Einsatz von Schadsoftware erneut gestiegen. Seit den 2000er-Jahren sanken dabei die Einstiegshürden deutlich, während der wirtschaftliche Schaden durch diese Entwicklung weltweit auf über eine Milliarde US-Dollar geschätzt wird [7].
Allein in Deutschland fanden 2020 täglich schätzungsweise *16 Millionen Angriffe* mit Schadsoftware statt [12]. Dies entspricht einem Anstieg von über 320 % im Vergleich zum Vorjahr, in dem täglich etwa 3,6 bis 3,8 Millionen Angriffe verzeichnet wurden [12]. Eine Studie des Bundesamt für Sicherheit in der Informationstechnik (BSI) aus dem Jahr 2025 schätzt, dass täglich bis zu 280.000 neue Malware-Varianten entstehen [13].
Zusätzlich werden die weltweiten jährlichen Schäden durch Malware bis 2025 auf etwa 10,5 Milliarden US-Dollar geschätzt [6]. Zudem beliefen sich die weltweit gezahlten Lösegeldsummen infolge von Verschlüsselungsangriffen im Jahr 2020 auf über 400 Millionen US-Dollar, was einem Anstieg von rund 300 % gegenüber dem Vorjahr entspricht [7].
Die dabei erzielten finanziellen Gewinne erhöhen somit den Anreiz zur Entwicklung neuer und zunehmend aggressiver Schadsoftware.

Zusammenfassend zeigt sich, dass Malware im Zuge technologischer, wirtschaftlicher und politischer Entwicklungen zu einem zentralen Angriffswerkzeug geworden ist. Mit der fortschreitenden Digitalisierung ist davon auszugehen, dass ihr potenzielles Schadensausmaß weiter zunehmen wird und sich ihre Angriffswege fortlaufend verändern werden.
Um diese Entwicklung einordnen zu können, ist als theoretische Grundlage die Betrachtung der verschiedenen Arten von Schadsoftware und ihrer Funktionsweisen erforderlich.
In den folgenden Abschnitten werden daher die drei sicherheitsrelevantesten und am weitesten verbreiteten Typen von Schadsoftware vorgestellt: Viren als Hauptkategorie, Würmer und Trojaner als Unterkategorie. Für jeden Typ werden die grundlegenden Funktionsprinzipien, typische Verbreitungswege sowie ausgewählte historische Beispiele erläutert.

2.1 Viren

Die ersten erschaffenen digitalen Schädlinge waren Viren. Ihre Bezeichnung leitet sich von dem gleichnamigen lateinischen Begriff ab, welcher für Gift, Saft und Schleim steht [14].
Laut Cohen lässt sich die Funktionsweise eines solchen Schadprogramms wie folgt definieren: „We define a computer ‘virus’ as a program that can ‘infect’ other programs by modifying them to include a possibly evolved copy of itself. With the infection property, a virus can spread throughout a computer system or network using the authorizations of every user using it to infect their programs. Every program that gets infected may also act as a virus and thus the infection grows. “ [15, S. 23]. Es agiert damit ähnlich wie ein biologisches Virus, welches den Wirt mit seinem Zellgift angreift, sich innerhalb seines Körpers vermehrt und sich auf andere Wirte ausbreitet.
Aus der Biologie ist außerdem bekannt, dass sich Virusinfektionen nicht vollständig verhindern lassen. Antivirus-Programme funktionieren dabei ähnlich wie Impfungen. Sie können durch ihre algorithmischen Erkennungsmuster nur bereits bekannte Viren außer Gefecht setzen. Laut grundlegender Forschung zu Computer-Viren gibt es nämlich keinen Algorithmus, der jedes Virus erfolgreich ermitteln kann [15] und es gibt Viren, die nie durch Algorithmen definiert werden können [16]. Aufgrund dieser Einschränkungen der Virusbekämpfung ist es notwendig, grundlegende Funktionsweisen und Verbreitungsmechanismen detailliert zu betrachten. Daraus abgeleitete neuartige Virustypen können dadurch besser erkannt werden. Im Folgenden werden daher verschiedene Virustypen vorgestellt.

Den Anfang machen die *Boot-Virus-Typen*, die Anfang der 1980er/90er Jahre auftauchten. Sie ersetzten den für den Systemstart verantwortlichen *Master-Boot-Record-Code* (MBRC) durch ihren eigenen Softwarecode [17]. Dabei nutzten sie aus, dass die damals verbreiteten PCs das Betriebssystem (BS) nicht von einer verbauten Festplatte, sondern von einer externen Diskette (Floppy Disk) starteten [17]. Der MBRC wurde somit nach dem Virus-Code ausgeführt und der PC wurde erfolgreich infiziert.
Eines der ersten Boot-Viren war *Brain*, welches 1986 auf einem *IBM PC* erschaffen wurde [18], [19]. Dieses hatte jedoch keine konkrete Schadabsicht, sondern sollte nur auf eine nicht lizenzierte Version des installierten BS hinweisen [19]. Es forderte die Betroffenen zusätzlich dazu auf, sich für eine gültige Nutzung an die Entwickler zu wenden.
Ein anderes Boot-Virus war *Elk Cloner,* welches sich 1982 auf Heimcomputern verbreitete. Wie *Creeper* und *Brain* war es aus heutiger Sicht ein harmloses Virus. Nach der Infektion wurde beim 50. Hochfahren des Rechners eine humoristische Nachricht für das „Opfer“ angezeigt [6]. *Elk Cloner* führte somit zu Leistungseinbußen sowie zu einer erhöhten mentalen Belastung des Nutzers. In Einzelfällen musste mit dem Verlust der Bootfähigkeit der Floppy Disk gerechnet werden [6].
Heutzutage ist die Relevanz solcher Boot-Viren jedoch eher gering, da der Schwachpunkt des Systems, der MBR, mehr vor solchen Attacken geschützt ist. Dies wird beispielsweise durch das erzwingende Starten des BS von der Festplatte bewirkt [20].

Ab den späten 1980er Jahren wurden dateiinfizierende Viren vermehrt in den Umlauf gebracht. Ihr Hauptmerkmal bestand darin, dass sie ausführbare Dateien verändern konnten, indem sie ihren eigenen Schadcode in diese integrierten [17]. Dadurch verbreiteten sie sich jedes Mal, wenn eine infizierte Datei geöffnet wurde.
Zuerst suchten diese Viren im Dateisystem nach geeigneten Zieldateien, typischerweise mit einer *.exe*- oder *.com*-Endung [17]. Anschließend integrierten sie ihren eigenen Code, indem sie ihn an die Datei anhängten, Teile des bestehenden Datei-Codes überschrieben oder ihn in ungenutzte Dateibereiche kopierten. Anders als Boot-Viren replizierten sich diese Viren bei jeder Ausführung und konnten bei vielgenutzten Dokumenten den ganzen Computer infizieren [17].
Ein praktisches Beispiel eines dateiinfizierenden Virus stellt *Cascade* dar.

Bei Ausführung der von *Cascade* manipulierten Dateien „fielen“ die dargestellten Zeichen kaskadenartig zum unteren Bildschirmrand [17]. Bei lang bestehenden Infektionen konnten sogar jegliche Dateien der betroffenen Systeme gelöscht oder unausführbar werden.

Durch die Verschlüsselung seines Inhalts konnte das Virus die Erkennung und Entfernung durch Antivirenprogramme erschweren , da diese überwiegend nur auf feste Byte-Muster reagierten [17], [21].

Diese frühen Verschlüsselungsverfahren bildeten die Grundlage für spätere Virustypen, die gezielt darauf ausgelegt waren, der Detektion zu entgehen.

Seit den 1990ern gelten polymorphe Viren als Erweiterung des Verschlüsselungsversuchs von Viren wie Cascade. Neben der Verschlüsselung ihres Schadcodes wird bei jeder Weiterverbreitung ein neuer Entschlüsselungsschlüssel verwendet [17], [21]. Darüber hinaus schufen Mutationsmodule wie *Dark Avenger* die Möglichkeit, Schlüssel unendlich oft automatisiert austauschen zu können [21]. Dadurch kam derselbe Entschlüsseler nie zweimal zum Einsatz, was die Erkennung basierend auf festen Mustern deutlich erschwerte.

Eine Strategie, um diese Virustypen zu bekämpfen, war, die Viren in einer isolierten Umgebung auszuführen. Hierbei sollte der Zeitpunkt der Schadcode-Entschlüsselung erfasst und der freigelegte Code zur Unterbindung der Virusverbreitung analysiert werden [21].

Das *Marburg-Virus*, das sich 1998 verbreitete, benutzte diese polymorphe Verschlüsselungstechnik. Der Schadcode wurde dabei nicht unmittelbar aktiviert, sondern zeitverzögert. Erst drei Monate nach der Erstinfektion führte das Virus seinen Payload aus [19]. Dies führte dazu, dass die graphische Nutzeroberfläche mit zahlreichen Windows-Fehlersymbolen überladen wurde (siehe Abbildung 2).

Darüber hinaus setzte *Marburg* den lokalen Virenschutz außer Kraft, indem es gezielt die Datenbanken der Antivirenprogramme löschte [19].

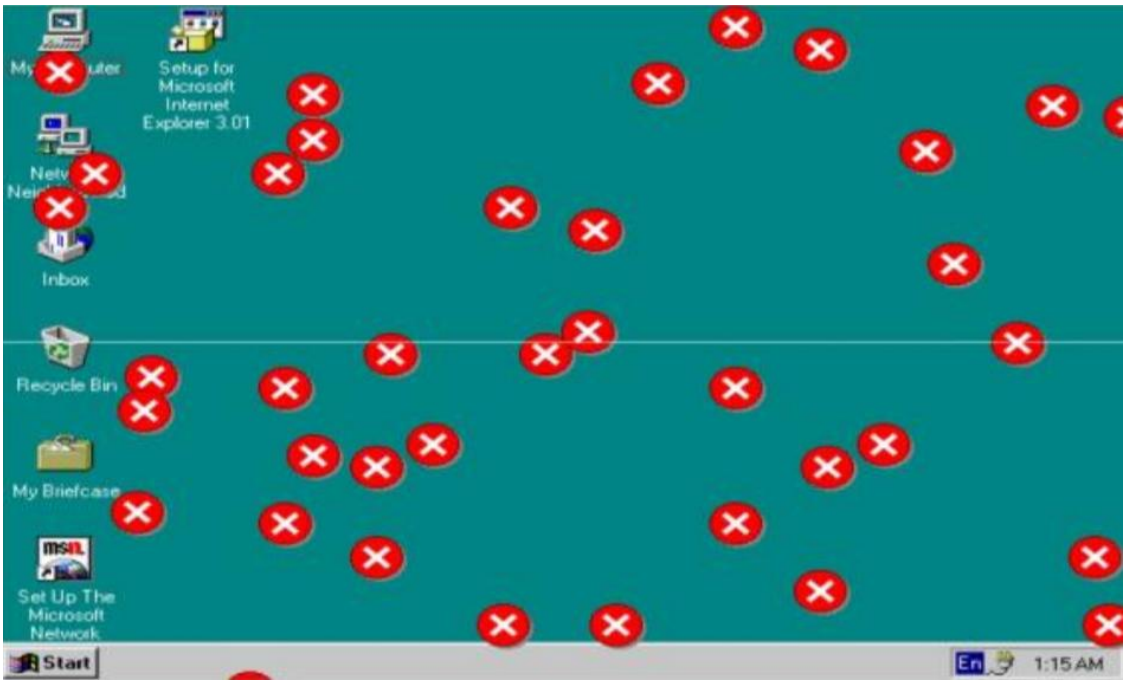


Abbildung 2*: Marburg-Virus*
Quelle: [19]

Metamorphe Viren stellen die Erweiterung von polymorphen Viren dar. Im Gegensatz zu ihren Vorgängern verändern sie nicht nur ihre Verschlüsselung, sondern generieren bei jeder Infektion vollständig neuen funktionsgleichen Code [21], [22]. Dadurch entsteht kein konsistenter Byte- oder Musterabdruck mehr, was verhaltensbasierte Erkennungsverfahren erschwert [22].

Beispiele wie *Zmist,* aus dem Jahr 2000, verdeutlichten die Komplexität dieser Techniken. Es nutzte eine anspruchsvolle Technik, bei der es das Wirtprogramm zunächst zerlegte, dessen Codeblöcke neu anordnete und dann seinen eigenen Code in den Ablauf einfügte [17], [22]. Anschließend wurde die Datei wieder zusammengesetzt, sodass sich das Virus spurlos innerhalb des Wirts tarnen konnte. Es kombinierte veränderten, sich ständig umschreibenden Code mit einer zusätzlichen polymorphen Verschlüsselungsschicht.

Zmist gehört damit zu den komplexesten Viren, die je erschaffen wurden [17] und zeigt beispielhaft, warum metamorphe Viren bis heute als eine der technisch anspruchsvollsten Virus-Kategorien gelten.

Parallel zu dieser Entwicklung entstand in den 1990er-Jahren mit der zunehmenden Verbreitung von Office-Programmen eine weitere Virusklasse. Sie nutzte gezielt die in Anwendungen wie Microsoft Word, Excel oder PowerPoint integrierten Makrosprachen aus: das Makrovirus

[22]. Im Gegensatz zu dateiinfizierenden Viren spezialisieren sich Makroviren nicht auf ausführbare Programme, sondern auf Office-Dokumente und verbreiten sich durch deren Öffnen und Austausch.
Ein Makrovirus verändert in der Regel Dokumente oder Vorlagen, um sich weiterzuverbreiten [22]. Wird ein infiziertes Dokument geöffnet, wird der Schadcode automatisch ausgeführt. Zusätzlich ersetzt es die zentrale Standardvorlage des Systems (z. B. *NORMAL.DOT*) durch eine manipulierte Kopie [17]. Dadurch wird jedes neue Dokument, das auf dem Computer erstellt wird, automatisch infiziert.
Concept, aus dem Jahr 1995*,* war das erste weit verbreitete Makrovirus. Es markierte den Beginn zahlreicher Infektionen, die durch Word-Dokumente übertragen wurden [17], [19]. Dadurch wurde erstmals deutlich, dass selbst Office-Dokumente zu Infektionsträgern werden können.
Makroviren zeigten, dass sich Schadprogramme an neue Softwareumgebungen anpassen können und dabei neue Angriffsmöglichkeiten entstehen.

Weitere Entwicklungen wie die breite Verfügbarkeit von Skriptsprachen wie VBScript, JavaScript und PowerShell führten zu einer weiteren Anpassung von Schadsoftware. Es entstanden skriptbasierte Viren, die sich über Skripte und E-Mail-Anhänge verbreiten konnten [17]. Da diese Sprachen ohne vorherige Kompilierung ausgeführt werden, konnten Änderungen am Quellcode schnell umgesetzt werden, was die Anpassung entsprechender Viren begünstigte.
Sie können auf Systemfunktionen, Dateien und E-Mail-Programme zugreifen und so Inhalte verändern oder weitere Dokumente infizieren.

Zusammenfassend zeigt die Entwicklung der Computerviren, wie sich Schadsoftware über die Jahrzehnte hinweg zunehmend an technische Fortschritte und neue Softwareumgebungen angepasst hat. Von frühen Boot-Viren, die Schwachstellen der Disketten-Ära ausnutzten, über dateiinfizierende Varianten bis hin zu polymorphen und metamorphen Viren, die gezielt Antivirenprogramme umgingen, zeigt sich eine kontinuierliche Weiterentwicklung. Mit dem Aufkommen von Makro- und skriptbasierten Viren wurden neben ausführbaren Programmen auch Dokumente, Vorlagen und skriptbasierte Laufzeitumgebungen als neue Infektionswege genutzt. Insgesamt lässt sich eine klare Tendenz erkennen: Viren wurden im Laufe der Zeit flexibler, komplexer und schwerer zu identifizieren.

Trotz all dieser Weiterentwicklungen blieb jedoch ein grundlegendes Merkmal unverändert: Für die Verbreitung von Viren ist eine aktive Ausführung infizierter Dateien erforderlich.
Genau an diesem Punkt setzt die nächste Entwicklungsstufe von Malware an.
Würmer überwinden die genannte Einschränkung, indem sie sich ohne Benutzerinteraktion selbstständig über mehrere Systeme weiterverbreiten können.
Ihre Funktionsweise und historische Bedeutung wird daher im folgenden Unterkapitel näher betrachtet.

2.2 Würmer

Zu den Formen selbstreplizierender Schadsoftware zählen *Würmer*. Die Bezeichnung ist vom biologischen Bandwurm inspiriert, einem „[…] parasitären Organismus, welcher im Inneren eines Wirts lebt und dessen Ressourcen aufzehrt, um sich selbst zu erhalten.“ [23, S. 3]. Als Computerwurm bezeichnet man daher ein Programm, das sich eigenständig über Netzwerke oder andere Kommunikationskanäle verbreitet, ohne dass eine Interaktion durch den Nutzer erforderlich ist [24].
Im Gegensatz zu den klassischen Computerviren, die sich an bestehende Dateien oder Programme anhängen und eine bewusste Interaktion voraussetzen, agieren Würmer weitgehend selbstständig. Sie nutzen vorhandene Netzwerkverbindungen, Sicherheitslücken oder unsichere Systemeinstellungen aus, um sich ohne menschliches Zutun auf weitere Systeme zu übertragen [17]. Trotz eigenständiger Ausbreitungsmechanismen werden Würmer häufig den Viren untergeordnet, da viele Varianten dateibasierte Infektionen mit netzwerkbasierter Verbreitung kombinieren [17], [24].
Der grundlegende Ablauf eines Wurm-Programms wird durch den folgenden Lebenszyklus beschrieben:

1. **Zielfindung:** Der Wurm scannt das Zielnetzwerk; listet E-Mail-Adressbücher, gemeinsame Laufwerke oder IP-Bereiche auf, um potenzielle Opfer ausfindig zu machen [17].
2. **Infektionsausbreitung:** Nachdem ein Opfer gefunden wurde, überträgt der Wurm seinen Schadcode [17]. Dies kann entweder durch das Versenden eines binären Dateianhangs, das Hochladen über eine Netzwerkfreigabe oder über eine bereits vorhandene Schwachstelle im System erfolgen. Viele Würmer nutzen dazu E-Mail-Angriffe oder injizieren Shell-Code in Netzwerkdienste mit bekannten Sicherheitslücken.

3. **Exploit-Phase:** Um Zugriffs- und Ausführungsrechte auf dem Hostrechner zu erlangen, nutzen Würmer oft bekannte Schwachstellen aus, etwa Windows-Dienste ohne aktuelle Sicherheitsupdates [17], [22]. Bei erfolgreicher Ausführung kann eine Hintertür (Backdoor) eingerichtet werden, womit weiterer Schadcode hinzugefügt werden kann.
4. **Payload-Aktivierung:** Hier führt der Wurm seinen Schadcode aktiv aus. Der jeweilige Inhalt kann dabei sehr variabel ausfallen. Dieser reicht vom Installieren einer Backdoor über Datendiebstahl bis hin zur gezielten Überlastung von Diensten oder Systemressourcen, etwa in Form von *Distributed-Denial-of-Service-Angriffen* (DDoS) [17], [25].
5. **Update/Instandhaltung:** Einige Würmer sind dazu in der Lage durch bestimmte Update-Mechanismen neue Schadfunktionen zu ihrem Code hinzuzufügen, bzw. zusätzliche Module nachzuladen [17]. Zwei Beispiele, die sich solche Verfahren zunutze machten, sind *Conficker* (2008) [26] und *Stuxnet* (2010) [27].
6. **Wiederholung:** Nachdem die Schritte 1 bis 5 durchlaufen sind, kopiert sich der Wurm innerhalb des Netzwerks und startet den Kreislauf erneut [22], [25], [28].

Die dargestellte Funktionsweise verdeutlicht die effiziente Ausbreitung solcher Schadprogramme in verteilten Netzwerken sowie deren Schadenspotenzial. Anhand der folgenden historischen Beispiele wird die Entwicklung von Würmern seit den 1980er-Jahren anschaulich dargestellt.

Als erste wurmähnliche Software gilt das zuvor erwähnte *Creeper*-Programm. Es wird häufig den Viren zugeordnet, ähnelt jedoch aufgrund seiner netzwerkweiten Verbreitung einem Wurm [6]. *Creeper* wurde als Experiment entwickelt, um die Übertragbarkeit selbstreplizierender Programme über Netzwerke hinweg zu untersuchen [6].
Aufbauend auf weiteren experimentellen Untersuchungen entstand 1988 der *Morris*-Wurm. Dieser Wurm nutzte verschiedene Schwachstellen in UNIX-Systemen, um sich zu verbreiten [23]. Ein eigentlich zur Prüfung von Nutzerzugängen vorgesehener Dienst wurde genutzt, um Schreibzugriffe auf betroffene Systeme zu erlangen. Zusätzlich wurden diverse Netzwerkdienste sowie der lokale E-Mail-Dienst kompromittiert. *Morris* veränderte zwar keine Systemdaten oder Dateien, führte jedoch aufgrund seiner unkontrollierten Verbreitung zu einer massiven Überlastung von System- und Netzwerkressourcen.
Der *Morris*-Wurm erlangte aufgrund seiner schnellen Ausbreitung große Aufmerksamkeit. Bereits innerhalb von 24 Stunden nach seiner Veröffentlichung wurden erste Notfallmaßnahmen

eingeleitet, was auf eine hohe Anzahl infizierter Systeme schließen lässt [23], [29]. Schätzungen zufolge entstand ein Schaden von mindestens 100 000 US-Dollar, wobei einige Quellen aufgrund länger andauernder Systemausfälle von deutlich höheren Kosten ausgehen [31]. Der Wurm zeigte, dass selbst harmloser Programmcode durch unkontrollierte Verbreitung erhebliche Auswirkungen haben kann. Er trug somit zur Weiterentwicklung selbstreplizierender Schadsoftware bei.

Im weiteren Kontext zeigte sich, dass Würmer neben technischen Schwachstellen zunehmend auch den menschlichen Faktor als Angriffsfläche nutzen. Während sich der Morris-Wurm hauptsächlich über die Ausnutzung von Softwarefehlern verbreitete, verfolgte der ILOVEYOU-Wurm (ILY) einen anderen Ansatz. Er nutzte menschliches Verhalten gezielt zur Verbreitung aus.

Der ILY-Wurm, der sich im Mai 2000 weltweit ausbreitete, wird in der Literatur häufig als einer der erfolgreichsten Social-Engineering-Angriffe der IT-Geschichte beschrieben [19], [30]. ILY verschickte sich dabei als E-Mail und tarnte sich als harmloses Liebesgeständnis mit dem Betreff „ILOVEYOU“ und dem Dateianhang „LOVE-LETTER-FOR-YOU.TXT.vbs“ [Abbildung 3].

Abbildung 3: „ILOVEYOU“ E-Mail mit Wurmanhang
Quelle: [17, S. 81]

Technisch basierte die Infektion darauf, dass die angehängte Datei ein ausführbares Visual-Basic-Skript (vbs) enthielt. Die Verwendung einer doppelten Dateiendung verschleierte diesen Umstand zusätzlich (siehe Abbildung 3). Begünstigt wurde dies außerdem durch die damalige Standardkonfiguration des Windows Explorers, in der bekannte Dateiendungen wie *.vbs* standardmäßig ausgeblendet wurden [17].

Die Verbreitung von ILY basierte vollständig auf menschlicher Interaktion, da der Schadcode erst durch das Öffnen des E-Mail-Anhangs aktiviert wurde [17]. Nach der Ausführung versandte der Wurm eigene Kopien an sämtliche im Adressbuch gespeicherten Kontakte. Durch die damit vorgetäuschte Vertrauenswürdigkeit bestehender Kontakte wurde die Wahrscheinlichkeit der Skriptausführung erhöht. Der in Abbildung 3 dargestellte E-Mail-Inhalt verstärkte diesen Prozess zusätzlich, indem er gezielt emotionale Reize wie Hoffnung und Neugier ansprach.

Über die reine Ausbreitung hinaus verursachte der Wurm erhebliche Schäden, indem er zahlreiche Dateien überschrieb und durch Kopien seines eigenen Codes ersetzte [17]. Eine Vielzahl gängiger Dateiformate (unter anderem *.vbs, .js, .jpg, .doc, .xls und .mp3*) war davon betroffen, was zu einem umfangreichen Datenverlust führte. Ergänzend installierte ILY auf betroffenen Systemen ein Programm, das Nutzerdaten und Passwörter ausspähte [31].

Die Ausbreitung von *ILY* beschränkte sich nicht nur auf private Haushalte und Unternehmen, sondern betraf auch zahlreiche staatliche Einrichtungen, darunter die NASA, das amerikanische Verteidigungsministerium sowie das deutsche Bundeswirtschaftsministerium [31], [32]. Innerhalb weniger Stunden infizierte der Wurm somit Millionen von Systemen weltweit und verursachte wirtschaftliche Schäden in Höhe von etwa zehn Milliarden US-Dollar [33].

Im Gegensatz zu netzwerkbasierten Würmern wie dem Morris-Wurm zeigte ILY, dass Social Engineering eine ebenso wirksame Verbreitungsstrategie darstellen kann. Während Programme wie *Creeper* experimentelle Ursprünge hatten, verdeutlichte ILY, wie menschliches Verhalten gezielt für erfolgreiche Infektionen und deren Weiterverbreitung ausgenutzt werden kann.

Eine Malware-Art, die diesen Aspekt besonders deutlich widerspiegelt und häufig mit anderer Schadsoftware kombiniert wird, ist der Trojaner. Die enge Verbindung zu SE macht ihn zu einem wichtigen Vertreter moderner Schadsoftware, dessen Funktionsweise und historische Bedeutung im folgenden Kapitel näher untersucht werden.

2.3 Trojaner

Der Begriff „Trojaner“ leitet sich vom trojanischen Pferd der griechischen Mythologie ab. Während eines langen, erfolglosen Krieges griffen die Griechen gegen die Trojaner zu einer List

[22], [34]. Anstatt die Stadt weiter direkt anzugreifen, errichteten die Griechen ein großes hölzernes Pferd und präsentierten es den Trojanern als vermeintliches Friedensgeschenk. Im Inneren des hohlen Pferdes verbargen sich jedoch Soldaten, die nach dem Einlass in die Stadt hervorkamen und so den Fall Trojas ermöglichten.

Diese Geschichte verdeutlicht die wichtigsten Merkmale der Trojaner-Schadsoftware: Ihr Schadenspotenzial wird erst nach Download und Ausführung sichtbar, da sie im Hintergrund agiert [22]. Sie kann unbemerkt Daten weiterleiten oder ein System für weitere Angriffe öffnen.

Im Gegensatz zu Viren und Würmern replizieren Trojaner ihren Schadcode nicht selbstständig. Stattdessen handelt es sich um eigenständige Programme, die entweder durch andere Malware eingeschleust werden oder von unwissenden Nutzern installiert werden [22]. Bereits im Jahr 2006 waren Trojaner für rund 80 % aller registrierten Malware-Infektionen verantwortlich und überholten damit Würmer sowie andere Schadsoftwarearten [22].

Wie Viren lassen sich Trojaner in verschiedene Kategorien unterteilen:

Remote-Access-Trojaner (RAT) ermöglichen Angreifern, umfassende Administrationsrechte über das infizierte System zu erhalten [22]. Damit lassen sich nahezu alle Prozesse und Daten des Opfers einsehen oder manipulieren. In der Praxis besteht ein *RAT* meist aus zwei Teilen: Auf dem infizierten Rechner läuft eine Komponente, die auf Befehle wartet, während der Angreifer über ein eigenes Steuerprogramm die Verbindung aufbaut und das System kontrolliert.

Backdoor Trojaner hingegen geben dem Angreifer keine gesonderten Rechte auf den Opfer-Systemen, sondern konfigurieren neue Netzwerkverbindungen [22]. Die auf diese Weise eingerichteten Hintertüren ermöglichen es, zusätzliche Malware nachzuladen oder Dateien zu verändern. Solche Zugänge sind ohne gezielte Netzwerk- oder Systemanalysen nur schwer erkennbar, sodass Infektionen häufig erst spät festgestellt werden.

Eine weitere Art stellen die *Network Redirection Tojaner* dar. Sie nutzen infizierte Systeme als Zwischenstation, über die der Datenverkehr gezielt an andere Ziele weitergeleitet werden kann [22]. Durch diese Umleitung erscheinen Angriffe als legitimer Datenverkehr, wodurch Sicherheitsmechanismen wie Firewall-Filterregeln umgangen werden können. Dies erschwert eine Rückverfolgung der Angreifer.

Distributed Attacks Trojaner, die für verteilte Attacken eingesetzt werden, infizieren eine große Zahl von Systemen, um diese anschließend zentral steuern zu können [22]. Infizierte

Systeme bleiben meist unauffällig und führen erst auf Anweisung Angriffe wie DDoS aus oder sammeln Informationen. Die einzelnen Systeme werden als Zombies bezeichnet und das Gesamtsystem als Botnetz.

Banking Trojaner sind darauf ausgelegt, vertrauliche Finanzdaten wie Kontoinformationen, Zugangsdaten oder Sitzungskennungen auszulesen und zu speichern [6]. Sie verfügen häufig über umfangreiche Funktionen zur unbemerkten Datenerfassung und Manipulation laufender Transaktionen. Aufgrund ihrer Anpassungsfähigkeit können sie Sicherheitsmechanismen wie die Zwei-Faktor-Authentifizierung umgehen. Zudem sind sie schwer zu erkennen, da neue Banking-Trojaner-Varianten an ständig verändernde Schutzmaßnahmen angepasst werden.

Password-Stealing Trojaner konzentrieren sich auf das Ausspähen sensibler Zugangsdaten, indem sie Benutzereingaben überwachen oder gespeicherte Zugangsdaten entnehmen [17]. Häufig arbeiten sie in Kombination mit Programmen, die jeden Tastenschlag abspeichern (Keylogger), oder anderen Modulen, die die gesammelten Daten automatisch an die Angreifer übermitteln. Ein gestohlenes Passwort kann dadurch den Zugriff auf mehrere Dienste und weitergehende Berechtigungen ermöglichen. Dies gilt insbesondere, wenn Zugangsdaten für mehrere Anwendungen wiederverwendet werden.

Entscheidend für den Erfolg einer Trojaner-Infektion ist vor allem ihr Weg ins System. Angreifer nutzen gezielte Täuschung, um Schadprogramme zu verbreiten oder Nutzer zur Ausführung manipulierter Software zu bewegen. Social Engineering stellt damit einen zentralen Erfolgsfaktor der beschriebenen Trojaner-Typen dar.

Im Folgenden werden diese Zusammenhänge anhand historisch relevanter Beispiele weiter vertieft.

Der *AIDS Trojan Disk* (ATD), aus dem Jahr 1989, gilt als einer der frühesten Trojaner [17]. Er wurde gezielt an über 7.000 Organisationen versendet, darunter sich zahlreiche medizinische und forschungsnahe Einrichtungen im Gesundheitssektor befanden [17]. Die Verbreitung erfolgte über Disketten, die mit der Aufschrift *„AIDS Information - Introductory Diskette“* (siehe Abbildung 4) versehen waren. Zusätzlich wurden sie an Teilnehmende der *World Health Organization AIDS Conference* verteilt [35].

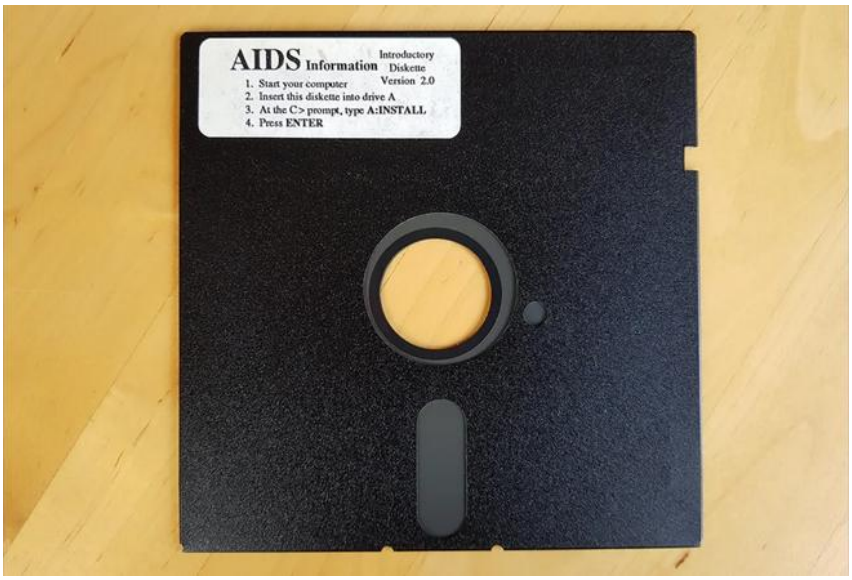


Abbildung 4: ATD mit Beschriftung
Quelle: [36]

Nach der Ausführung manipulierte der Trojaner das Dateisystem des betroffenen Rechners, indem er Dateinamen vertauschte, Inhalte verschleierte und freien Speicher mit bedeutungslosen Daten füllte [17]. Nach mehreren Systemstarts blockierte das Programm den Zugriff auf vorhandene Daten, indem es sie verschlüsselte, und forderte ein Lösegeld. Der Trojaner forderte hierzu Zahlungen von mehreren hundert Dollar an ein Postfach in Panama, um die Daten wieder freizugeben [37]. Bei Nichtzahlung wurde mit rechtlichen Konsequenzen gedroht.
Insgesamt waren schätzungsweise bis zu 20.000 Systeme weltweit betroffen [35], [37]. Obwohl die eingesetzte Verschlüsselung aus heutiger Sicht sehr einfach war, verloren viele Betroffene zunächst den Zugriff auf ihre Daten. In einigen Fällen führte dies dazu, dass Festplatten vollständig gelöscht wurden, wodurch einige Forschungsdaten unwiederbringlich verloren gingen.
Als einer der ersten bekannten Trojaner zeigte der AIDS Trojan Disk eindrücklich die zentrale Rolle von SE als Einfallstor für Schadsoftware. Durch die Nutzung eines thematisch vertrauten Kontexts sowie glaubwürdiger Bezeichnungen wurde die Hemmschwelle, fremde Disketten in eigene Systeme einzuführen, erheblich gesenkt. Erst diese psychologische Komponente ermöglichte ADTs weitreichende Verbreitung und machte den Angriff in seinem Ausmaß überhaupt möglich.

Der Banking Trojaner *Zeus* stellt ein weiteres Beispiel für langfristig schädliche Schadsoftware dar. Diese Trojaner-Art ermöglicht das Auslesen sensibler personenbezogener Informationen

sowie die Manipulation der Rechnerkommunikation. Neuere Varianten von *Zeus* [38] nutzen zudem polymorphe Verschlüsselungstechniken, wodurch sich ihr Schadcode bei jeder Ausführung verändert. Dies erschwert die Erkennung erheblich und schränkt die Wirksamkeit klassischer Schutzmechanismen deutlich ein.

Zeus verbreitete sich unter anderem über Phishing-E-Mails, die Nutzer auf gefälschte Webseiten bekannter Dienste wie Facebook lenkten, um sie zur unbewussten Preisgabe ihrer Zugangsdaten zu verleiten [39]. Dieser Trojaner infiziert vor allem Windows-Systeme, lädt weitere Schadsoftware nach, stiehlt Login-Daten und kann Bank- und Finanzdaten abfangen [6], [40].

Erste Berichte über Zeus erschienen bereits 2007. Mit der Veröffentlichung des Quellcodes im Jahr 2011 wurde die Entwicklung zahlreicher Varianten begünstigt. Insbesondere Ableger wie *GameOver Zeus* erweiterten den Funktionsumfang deutlich, etwa durch verbesserte Tarnmechanismen und zusätzliche Erpressungsfunktionen [40].

Aufgrund seiner langjährigen Existenz und der Kommerzialisierung des Quellcodes, der als Toolkit für bis zu 4.000 US-Dollar verkauft wird [38], weist die Schadsoftware eine hohe Verbreitung auf. Schätzungen gehen davon aus, dass *Zeus* und seine Weiterentwicklungen finanzielle Schäden von über 100 Millionen US-Dollar verursacht haben [41]. Allein in Deutschland wurden im Jahr 2020 rund 500.000 Fälle von Zeus-Infektionen registriert [12].

Ein weiterer berühmter Trojaner, der stark auf SE setzt, ist *Emotet*.

Diese Malware ist seit 2014 aktiv und entwickelte sich im Laufe der Zeit von einem klassischen Banking Trojaner zu einem mehrschichtig aufgebauten Schadprogramm. Infizierte Systeme werden dauerhaft infiltriert und als Backdoor (S. 16) für weitere Angriffe genutzt. Der Schwerpunkt von Emotet liegt jedoch weniger auf eigenen Schadfunktionen als auf dem Nachladen zusätzlicher Schadsoftware [42].

Charakteristisch für *Emotet* ist sein Geschäftsmodell als *Malware-as-a-Service* [7]. Die Betreiber stellen infiltrierte Systeme sowie verschiedene Module gegen Bezahlung anderen Kriminellen zur Verfügung. Auf diese Weise wird der Erstzugang zu Zielsystemen zentral bereitgestellt und für weitere Angriffe nutzbar gemacht. Europol bezeichnet dieses Modell als *Access-as-a-Service* [43] und beschreibt es als Ausdruck der zunehmenden Professionalisierung und Arbeitsteilung im Cybercrime-Umfeld. Dies verdeutlicht, dass Trojaner wie *Emotet* dazu genutzt werden, Einstiegshürden für weitere Malware-Attacken zu senken und das Schadenspotenzial dadurch erheblich zu erhöhen.

Die Auswirkungen dieser Entwicklung spiegeln sich zudem in den Schadenszahlen wider: Schätzungen zufolge infizierte *Emotet* im Jahr 2021 weltweit über 1,5 Millionen Systeme und verursachte wirtschaftliche Schäden in Milliardenhöhe [44]. Bis 2020 existierten zudem über 200.000 unterschiedliche Versionen der Schadsoftware [43].
Die Verbreitung erfolgt über verschiedene Methoden, wobei E-Mail-Thread-Hijacking besonders effektiv ist [45]: Hierbei greift der Angreifer auf reale, zuvor aus übernommenen E-Mail-Postfächern abgeflossene Nachrichtenverläufe zurück. Er nutzt diese, um täuschend echt wirkende Nachrichten mit manipulierten Inhalten an bereits vorhandene Kontakte zu versenden. Da schädliche Anhänge oder Links in einem vertrauten Kontext erscheinen und von bekannten Absendern stammen, ist die Wahrscheinlichkeit der Ausführung deutlich erhöht.
Diese Vorgehensweise ähnelt der des *ILY-Wurms* und stellt ein Beispiel für SE dar. Durch das Ausnutzen bestehender Vertrauensverhältnisse zwischen Absender und Empfänger kann die Verbreitung gefährlicher Inhalte gefördert werden.

Zusammenfassend zeigt dieses Kapitel, dass sich Trojaner trotz ihrer unterschiedlichen technischen Ausprägungen durch ein gemeinsames Merkmal auszeichnen: Ihren Erfolg verdanken sie in hohem Maße der gezielten Ausnutzung menschlicher Verhaltensweisen und Emotionen. Die Beispiele *ADT*, *Zeus* und *Emotet* verdeutlichen, dass technische Schadfunktionen allein keine großflächigen Infektionen ermöglichen, sondern SE maßgeblich zum Erfolg solcher Schadsoftware-Kampagnen beiträgt.
Unabhängig von der jeweiligen Malware-Kategorie stellt SE damit einen Hauptbestandteil moderner Schadsoftware dar. Das folgende Kapitel widmet sich daher der detaillierten Betrachtung sozialer Manipulation und zeigt anhand eines praktischen Versuchs, wie menschliche Faktoren gezielt für Cyberangriffe ausgenutzt werden können.

3 Social Engineering

Die Entwicklung schädlicher Programme steht in engem Zusammenhang mit veränderten Angriffsmethoden der Cyberkriminalität. Während frühe Malware-Varianten wie *Creeper* oder *Morris* technische Schwachstellen ausnutzten, sind moderne IT-Systeme in vielen Bereichen durch umfangreiche Sicherheitsmechanismen abgesichert. Direkte technische Angriffe werden dadurch zunehmend erschwert. Infolge dieser Entwicklung verlagern sich Cyberangriffe

zunehmend auf nicht-technische Angriffspunkte. Dabei rückt insbesondere der Mensch als Angriffsziel in den Mittelpunkt. Viele Angriffe setzen gezielt an der Nutzerinteraktion an, noch bevor technische Schutzmechanismen wirksam greifen können. Bedrohungsanalysen aus dem Jahr 2024 bestätigen, dass viele erfolgreiche Angriffe ihren Ursprung nicht in der Überwindung technischer Sicherheitsvorkehrungen haben, sondern den Prinzipien des Social Engineerings entstammen [46], [47].

Berichte europäischer Sicherheitsbehörden zeigen zudem, dass Phishing-E-Mails in vielen dokumentierten SE-Vorfällen eine zentrale Rolle spielen [46]. Dabei werden häufig Inhalte wie Lohnabrechnungen oder Terminänderungen vorgetäuscht, um Empfänger auf Webseiten zu leiten, auf denen sie ihre Zugangsdaten eingeben sollen. Die auf diese Weise erlangten Anmeldedaten ermöglichen Angreifern anschließend unautorisierten Zugriff auf interne Systeme.

Weitere Ereignisse unterstreichen die Bedeutung dieses Vorgehens: Gezielte Kampagnen gegen staatliche Einrichtungen und Unternehmen, wie der Angriff auf das staatliche polnische Mail-Netzwerk im Frühjahr 2024 [46], zeigen, dass der Erstzugang häufig über manipulierte interne Kommunikationsinhalte erfolgt.

Zudem ist in den vergangenen Jahren ein Anstieg KI-gestützter SE-Angriffskampagnen zu beobachten, bei denen automatisiert erzeugte Inhalte zur Täuschung eingesetzt werden, etwa in Form von Voice-Cloning-Anrufen, die die Stimmen bekannter Personen imitieren [46], [48]. Diese Entwicklungen verdeutlichen die Vielschichtigkeit moderner Malwareangriffe und zeigen, dass die gezielte Ausnutzung menschlicher Emotionen einen zentralen Einstiegspunkt darstellt. Die folgenden Abschnitte legen daher die begrifflichen und theoretischen Grundlagen von SE dar.

3.1 Begriffserklärung und theoretische Grundlagen

Der Begriff *Social Engineering* bezeichnet die gezielte Manipulation von Menschen mit dem Ziel, bestimmte Handlungen auszulösen [49]. Dazu zählen etwa die Preisgabe vertraulicher Informationen oder die Ausführung sicherheitsrelevanter Aktionen, die Angreifern den Zugriff auf interne Systeme ermöglichen. Häufig beruht diese Beeinflussung auf dem Vortäuschen vertrauenswürdiger Identitäten sowie dem gezielten Einsatz von Autorität oder zeitlichem Druck über verbreitete Kommunikationswege. Auf diese Weise werden Betroffene zu Interak-

tionen verleitet, die aus technischer Sicht vermeidbar gewesen wären. Im Gegensatz zu klassischen systemseitigen Angriffsmethoden basiert dieser Ansatz primär auf der Ausnutzung menschlicher Interaktion und psychologischer Mechanismen, weshalb SE auch als Form des „human hacking“ beschrieben wird [50].

Der Erfolg von Social Engineering basiert auf der gezielten Ausnutzung psychologischer Einflussfaktoren, die das menschliche Entscheidungsverhalten prägen. Aufbauend auf den Arbeiten von Robert Cialdini lassen sich im Folgenden zentrale Prinzipien festhalten, die in SE-Angriffen häufig auch in Kombination eingesetzt werden [49], [51]:

Das *Reziprozitätsprinzip* beschreibt die Neigung von Menschen, empfangene Gefälligkeiten erwidern zu wollen. Dadurch steigt die Bereitschaft, späteren Aufforderungen nachzukommen, etwa nach vorgespielten Hilfsangeboten oder dem Erhalt scheinbar nützlicher Informationen.

Das *Knappheitsprinzip (Scarcity)* nutzt zeitliche oder anderweitige wichtige Begrenzungen, um Entscheidungsdruck zu erzeugen und impulsives Handeln zu begünstigen, wie durch Deadlines oder begrenzt verfügbare Angebote.

Das *Autoritätsprinzip* basiert auf der Neigung, Anweisungen vermeintlich übergeordneter oder fachlich legitimierter Personen zu folgen, etwa wenn Entscheidungen aufgrund formaler Titel, Uniformen oder offizieller Rollen getroffen werden.

Das *Sympathieprinzip (Liking)* beschreibt die Tendenz, eher den Bitten von Personen nachzukommen, zu denen eine positive Beziehung besteht. Sympathie kann dabei durch wahrgenommene Ähnlichkeiten, Komplimente sowie kooperatives Verhalten entstehen. Typischerweise geschieht dies, wenn gemeinsame Interessen betont oder eine persönliche Nähe vermittelt wird.

Das *Prinzip der Verpflichtung und Konsistenz* beschreibt den Wunsch, einmal getroffene Entscheidungen oder eingegangene soziale Bindungen beizubehalten, was die Wahrscheinlichkeit weiterer Interaktionen erhöht. Dies zeigt sich beispielsweise darin, dass Personen nach einer ersten, unverbindlichen Zustimmung eher bereit sind, weiteren darauf aufbauenden Aufforderungen nachzukommen.

Sozialer Beweis (Consensus) beschreibt das Phänomen, dass Personen ihre Entscheidungen insbesondere in unsicheren Situationen am Verhalten anderer ausrichten, sofern dieses als mehrheitlich oder gesellschaftlich akzeptiert wahrgenommen wird.

Ergänzend beschreibt das *Unity-Prinzip* die verstärkte Beeinflussbarkeit innerhalb sozialer Identitäten oder Gruppenzugehörigkeiten. Personen reagieren dabei eher auf Bitten oder Vorschläge, die von Mitgliedern derselben sozialen oder beruflichen Gruppe stammen.

In der Praxis werden diese Prinzipien selten allein eingesetzt. Vielmehr kombinieren SE-Angriffe mehrere dieser Einflüsse, um Opfer schrittweise zu einer sicherheitsgefährdenden Handlung zu bewegen. Es wird davon ausgegangen, dass diese Prinzipien grundsätzlich auf alle Menschen anwendbar sind, da sie grundlegende Emotionen und Bedürfnisse wie Empathie, Zugehörigkeitsgefühl, Neugier und Angst für ihren Vorteil ausnutzen [51]. Die in Tabelle 1 dargestellten SE-Methoden veranschaulichen, wie diese Prinzipien in der Praxis von Angreifern angewandt werden können:

Tabelle 1: SE-Methoden (in Anlehnung an [48, S. 5])

Methode	*Erklärung*	*Mögliche Einflussprinzipien nach Cialdini*
Pretexting	Ein vorgefertigtes Szenario wird genutzt, um an Informationen oder Zugänge des Opfers zu kommen	Sympathie, Verpflichtung & Konsistenz, Autorität, Unity
Baiting	Das Opfer wird durch ein Lockangebot dazu gebracht Sicherheitsmechanismen zu umgehen, u.a. durch Herunterladen von kostenlosen Programmen	Knappheit, Reziprozität
Tailgaiting	Sicherheitspersonal/Mitarbeiter werden getäuscht, um physischen Zugang zu gesicherten Bereichen zu erlangen	Unity, Sozialer Beweis, Sympathie
Quizzing	Das Opfer wird durch ein vermeintliches Quiz oder eine Befragung dazu verleitet Informationen preiszugeben	Reziprozität, Verpflichtung & Konsistenz

Scareware	Durch das Vortäuschen eines angeblichen Virenbefalls wird das Opfer zum Kauf einer gefälschten Sicherheitssoftware verleitet	Autorität, Knappheit
Phishing	Das Opfer erhält absichtlich täuschende Emails, die es dazu auffordert, Login-Dateien oder persönliche Informationen preiszugeben	Autorität, Knappheit, Unity, Sozialer Beweis

Die in Tabelle 1 dargestellten Methoden beruhen auf einem oder mehreren Einflussprinzipien nach Cialdini. Dabei werden gezielt psychologische Mechanismen wie Autorität, Knappheit oder sozialer Beweis eingesetzt, um Vertrauen zu erzeugen und die Entscheidungsfindung der Betroffenen zu beeinflussen. Obwohl sich die einzelnen Methoden voneinander unterscheiden, verfolgen sie ein gemeinsames Ziel: das Umgehen technischer Schutzmaßnahmen, indem menschliches Verhalten ausgenutzt wird.

Die gezielte Beeinflussung von Menschen ist kein ausschließlich modernes Phänomen, sondern findet sich bereits in historischen Kontexten. Ein häufig herangezogenes Beispiel ist das trojanische Pferd, bei dem Täuschung gezielt genutzt wurde, um Zugang zu einer geschützten Umgebung zu erlangen (siehe S. 15). Diese Strategie lässt sich dem Reziprozitätsprinzip zuordnen. Indem die Griechen den Trojanern das hölzerne Pferd als Friedensgeschenk überreichten, erzeugten sie eine soziale Verpflichtung zur Annahme. Die Trojaner brachten das Geschenk innerhalb ihrer Stadtmauern unter, was zur Einnahme Trojas führte.

Im frühen IT-Kontext wurde der Begriff insbesondere durch Kevin Mitnick geprägt. In den 1990er- und 2000er-Jahren gelang es ihm, durch gezielte soziale Beeinflussung von Mitarbeitenden sicherheitskritische Informationen zu erlangen, ohne technische Schutzmaßnahmen direkt überwinden zu müssen [49], [52], [53]. Dabei nutzte er unter anderem das Autoritäts- und Sympathieprinzip aus. In einem Fall gab er sich dabei als Mitarbeiter eines Forschungszentrums aus und erhielt auf diesem Weg den Quellcode eines Motorola-Mobiltelefons von der zuständigen Assistenz der Projektmanagerin.

Die dargestellten theoretischen Grundlagen zeigen, dass SE einen zentralen Angriffsansatz darstellt, der weitgehend unabhängig von der eingesetzten Technologie wirksam ist. Moderne

Malware nutzt diesen Ansatz gezielt, um initialen Zugriff auf abgesicherte IT-Systeme zu erlangen.

Vor diesem Hintergrund wird im folgenden Kapitel ein praktisches Szenario konzipiert und umgesetzt, in dem soziale Manipulation in Form einer E-Mail zur Einschleusung selbstreplizierender Schadsoftware in ein Netzwerk eingesetzt wird.

3.2 Entwurf eines Social-Engineering-Szenarios

Für das in dieser Arbeit untersuchte Szenario habe ich die E-Mail bewusst als zentrales Kommunikations- und Verbreitungsmedium gewählt. Sie ist nach wie vor ein fester Bestandteil der digitalen Kommunikation in Unternehmensnetzwerken und zugleich ein häufig genutzter Ansatzpunkt für SE-basierte Angriffe [1], [2]. Im Gegensatz zu vielen historischen Beispielen von manipulativen E-Mail-Angriffen ist der Inhalt der umgesetzten E-Mail nicht im privaten, sondern gezielt im beruflichen Kontext angesiedelt.

Die Entscheidung, ein selbstreplizierendes Programm als Anhang mitzuschicken, orientiert sich an historischen Beispielen von SE-E-Mail-Angriffen wie dem ILY-Wurm. Dieser zeigte, dass bereits einfach gestaltete, als legitim getarnte E-Mails ausreichen können, um durch Nutzerinteraktion eine weitreichende Verbreitung von Malware auszulösen.

Auf Grundlage dieser Überlegungen habe ich ein vereinfachtes Verbreitungsszenario modelliert. Der Fokus meiner Modellierung liegt bewusst auf dem E-Mail-Inhalt und der Nutzerinteraktion und nicht auf der detaillierten, technischen Abbildung eines Schadprogramms.

```
From: sop@attacker.local
To: vboxuser@opfer1.local
Subject: Abrechnung WICHTIG!!
MIME-Version: 1.0
Content-Type: multipart/mixed; boundary="================4478692476507156585=="

--================4478692476507156585==
Content-Type: text/plain; charset="utf-8"
Content-Transfer-Encoding: quoted-printable

Guten Tag,
Sie erhalten hiermit die aktualisierte Abrechnung fuer das letzte Quartal (4/25).
Bitte pruefen Sie die Angaben umgehend und geben Sie uns bis spaetestens Mittwoch (04.02.26) eine
verbindliche Rueckmeldung ab.
Vielen Dank.

Mit freundlichen Gruessen
Finanzberatung GmbH
```

Abbildung 5: Mail-Inhalt

Quelle: eigene Darstellung

Der von mir konzipierte Mail-Inhalt (siehe Abbildung 5) thematisiert eine vermeintliche Abrechnung und ist bewusst sachlich sowie professionell formuliert. Die E-Mail fordert den Empfänger dazu auf, ein beigefügtes Dokument zu überprüfen und bis zu einem festgelegten Zeitpunkt eine Rückmeldung abzugeben. Der Absender gibt sich dabei als Mitarbeiter einer Finanzberatung aus. Als inhaltliche Inspiration für diesen Entwurf dient Anhang C Abbildung 17.

Einordnung der eingesetzten Social-Engineering-Prinzipien

Der entworfene E-Mail-Inhalt basiert auf der gezielten Kombination mehrerer etablierter SE-Prinzipien, die sich in ihrer Wirkung gegenseitig verstärken:

Autoritätsprinzip:
Der E-Mail-Absender gibt sich als Vertreter einer übergeordneten Instanz (Finanzberatung) aus und fordert zur Rückmeldung zu einem wichtigen Dokument auf. Die vermeintliche fachliche Kompetenz und die formelle Kommunikation verleihen der Nachricht Legitimität. Die Wahrscheinlichkeit, dass die Aufforderung als verbindlich wahrgenommen wird, wird erhöht.

Scarcity-Prinzip:
Die gesetzte Deadline (04.02.2026) erzeugt zeitlichen Druck. Dieser reduziert die Wahrscheinlichkeit einer kritischen Reflexion mit dem Inhalt und begünstigt impulsives Handeln.

Prinzip der Verpflichtung und Konsistenz:
Im Arbeitskontext wird von Empfängern erwartet, geschäftlichen und organisatorischen Aufforderungen nachzukommen. Die indirekte Erwartung einer Zustimmung oder Rückmeldung verstärkt den Handlungsdruck zusätzlich. Durch die sachliche Gestaltung erscheint die Nachricht plausibel und unauffällig, was die Ausführung des Anhangs erhöht.

Abgrenzung des Szenarios (SE)

Das entwickelte Szenario ist als vereinfachtes Modell zu verstehen und bildet die tatsächliche Komplexität interner Unternehmenskommunikation nur reduziert ab. Aspekte wie individuelles Nutzerverhalten oder organisatorische Rahmenbedingungen werden nicht im Detail berücksichtigt und können innerhalb der Simulation nur vereinfacht dargestellt werden.

Für die Simulation nehme ich an, dass die versendeten E-Mails geöffnet und die enthaltenen Anhänge ausgeführt werden.
Ziel der Untersuchung ist daher nicht, die Wahrscheinlichkeit einer solchen Nutzerinteraktion zu untersuchen, sondern zu zeigen, dass Social Engineering grundsätzlich als Einstiegspunkt für eine Infektion genutzt werden kann und welche Auswirkungen dies auf die weitere Verbreitung im Netzwerk hat.
Der Schwerpunkt liegt bewusst auf der Nutzerinteraktion als Auslöser der Verbreitung sowie auf der Wirkung der eingesetzten SE-Methode. Der Mensch wird dabei gezielt als mögliche Schwachstelle betrachtet. Die Simulation dient somit als Verbindung zwischen den theoretischen Grundlagen und der praktischen Umsetzung im weiteren Verlauf der Arbeit.

Bezug zu verwandten Arbeiten

Bei der Gestaltung des Szenarios habe ich mich an bestehenden wissenschaftlichen Arbeiten zur Simulation selbstverbreitender Schadsoftware orientiert. In der Arbeit *„Experiences with Worm Propagation Simulations“* [54] wird die Ausbreitung von Netzwerk-Würmern anhand eines vereinfachten Modells beschrieben, das den Verbreitungsprozess in mehrere aufeinanderfolgende Schritte unterteilt. Dazu zählen unter anderem die Auswahl eines Zielsystems, dessen Infektion sowie die anschließende Weitergabe der Schadsoftware. Ein vergleichbarer mehrstufiger Ablauf bildet die Grundlage der vorliegenden Simulation.
Im Unterschied dazu steht in dieser Arbeit die detaillierte technische Nachbildung eines autonomen Wurms nicht im Fokus. Stattdessen wird die Ausbreitung gezielt durch SE ausgelöst. Die technische Umsetzung der Verbreitung habe ich bewusst vereinfacht, um den Einfluss der Nutzerinteraktion als zentralen Auslöser der Weiterverbreitung deutlich hervorzuheben.

Ergänzend liefern Studien wie „*Emotional Manipulation in Phishing Emails: Experimental Study of Affective Responses and Human Classification Errors in a Simulated Email Environment*“ [55] zentrale Erkenntnisse zur Wirkung emotionaler und personalisierter E-Mails. Die Ergebnisse zeigen, dass vor allem persönliche sowie arbeitsbezogene Inhalte die Wahrscheinlichkeit erhöhen, dass Empfänger mit einer E-Mail interagieren. Obwohl sich die Untersuchung hauptsächlich mit Phishing-Szenarien befasst, lassen sich die gewonnenen Erkenntnisse auf vergleichbare E-Mail-basierte SE-Angriffe übertragen, da diese ebenfalls auf gezielter sozialer Manipulation beruhen.

Zusammenfassend lässt sich festhalten, dass Social Engineering als Faktor für das Auslösen E-Mail-basierter Angriffe eine wichtige Rolle spielt. Auf Grundlage dieser theoretischen Überlegungen und des entwickelten SE-Konzepts werden zwei Hypothesen für den praktischen Teil der Arbeit formuliert und untersucht:

H1: Die initiale Verbreitung selbstreplizierender Schadsoftware innerhalb eines Netzwerks kann durch Social Engineering ausgelöst werden, ohne dass technische Schwachstellen ausgenutzt werden müssen.

H2: Die Ausführung des manipulierten E-Mail-Anhangs und damit die Erstinfektion sind unmittelbar von der Nutzerinteraktion abhängig.

3.2.1 Technologien, Programm-Architektur und Versuchsaufbau

Testumgebung

Zur Umsetzung des entworfenen Szenarios habe ich eine virtualisierte Testumgebung mit VirtualBox und dem Linux-Betriebssystem Ubuntu eingerichtet. Abbildung 6 zeigt die vereinfachte Umgebung, bestehend aus drei virtuellen Rechnern: einem Angreifer-System sowie zwei Opfer-Systemen. Dieses simple Rollenmodell ermöglicht eine übersichtliche und gut nachvollziehbare Analyse der E-Mail-basierten Verbreitung. Die Nutzung virtueller Maschinen stellt sicher, dass der simulierte Angriff reproduzierbar bleibt und keine Auswirkungen auf reale Systeme hat. Gleichzeitig ermöglicht sie die modellhafte Nachbildung eines internen Unternehmensnetzwerks.

Ubuntu[1] wurde als Betriebssystem gewählt, da es eine stabile Linux-Distribution ist. Aufgrund seiner umfangreichen Paketverwaltung sowie der Unterstützung von Netzwerk- und Mailserver-Diensten eignet es sich besonders für experimentelle Netzwerkumgebungen. VirtualBox[2] ermöglicht als Virtualisierungsumgebung die flexible Einrichtung isolierter Testsysteme und die Reproduzierbarkeit der Versuchsbedingungen.

[1] Ubuntu Desktop 24.04 LTS: https://ubuntu.com/desktop
[2] VirtualBox 7.2.6: https://www.virtualbox.org/

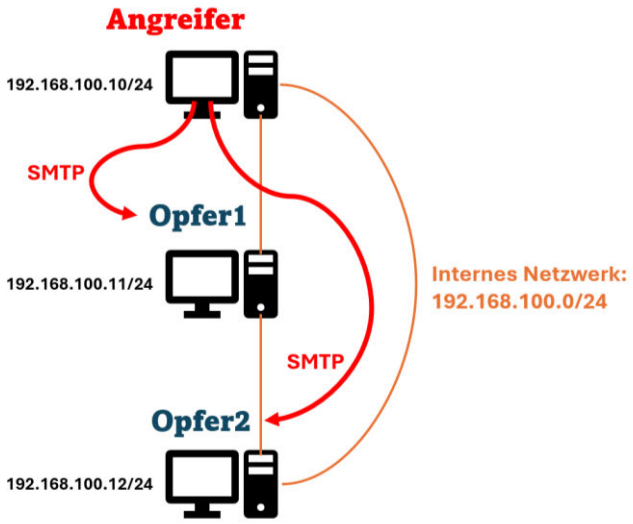


Abbildung 6: Netzwerkkonfiguration und Rollenverteilung
Quelle: eigene Darstellung

Netzwerk- & Programmebenen

Die Kommunikation zwischen den Systemen erfolgt über ein internes Netzwerk und das *Simple Mail Transfer Protocol* (SMTP), das als Standardprotokoll für den Versand von E-Mails dient. Die Wahl von SMTP erlaubt eine Auswertung mit Hilfe von Wireshark[3] hinsichtlich Versandstatus, Zieladressen und Zustellvorgängen.

Das Protokoll bildet die Grundlage, um die Übertragbarkeit der selbstreplizierenden Software über den E-Mail-Verkehr sowie deren Ausbreitungsgeschwindigkeit zu untersuchen. Durch die direkte Nutzung von SMTP wird der Fokus bewusst auf den eigentlichen Übertragungsmechanismus gelegt, ohne zusätzliche Darstellungsebenen durch grafische Mailclients einzubeziehen.

Der E-Mail-Versand soll nicht nur der initialen Infektion dienen, sondern zugleich die Grundlage für die weitere Verbreitung innerhalb des Netzwerks bilden. Nach einer simulierten Nutzerinteraktion sollen infizierte Systeme automatisiert weitere Nachrichten an gespeicherte Kontaktadressen senden.

Zur Umsetzung dieses Verhaltens wurde die folgende Programmstruktur mithilfe eines UML-Diagramms dargestellt (siehe Abbildung 7).

[3] Wireshark 4.6.3: https://www.wireshark.org/

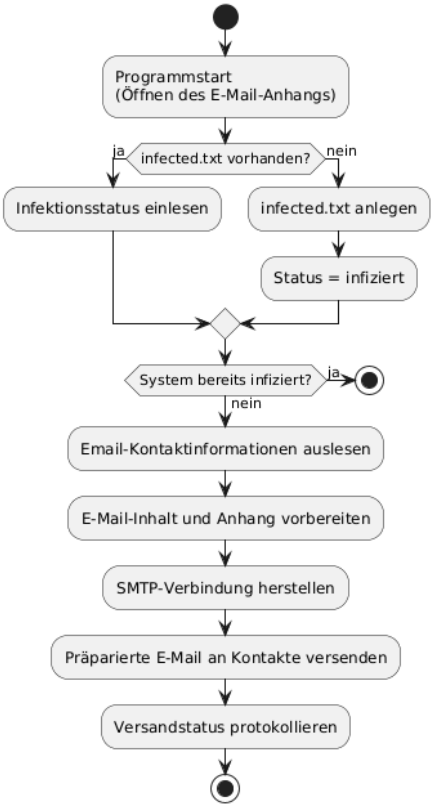


Abbildung 7: UML-Diagramm des selbstreplizierenden Programms im E-Mail-Anhang
Quelle: eigene Darstellung

Der Versuchs- und Programmablauf lässt sich in mehrere Schritte gliedern:
Das Angreifer-System wird zunächst als regulärer Teilnehmer in das Netzwerk integriert. Danach werden E-Mails mit einem ausführbaren Anhang aus diesem netzwerkinternen Benutzerkonto an die Opfer-Systeme gesendet. Die Verbreitung erfolgt dabei nicht eigenständig, sondern setzt eine simulierte Nutzerinteraktion voraus. Erst durch das Öffnen des Anhangs auf den zu manipulierenden Rechnern wird die weitere Replikation ausgelöst. Dadurch lassen sich die Auswirkungen der eingesetzten SE-E-Mail untersuchen.

Abbildung 7 zeigt den Ablauf des selbstreplizierenden Programms nach dem Öffnen des E-Mail-Anhangs. Zu Beginn prüft das Programm anhand der Datei *infected.txt*, ob das System bereits als infiziert markiert ist. Abhängig von der Existenz dieser Datei wird der erneute Versand weiterer E-Mails gestattet oder unterbunden. Sollte der jeweilige Rechner nicht infiziert sein, werden hinterlegte Kontaktadressen ausgelesen und der E-Mail-Inhalt vorbereitet. Anschließend werden die E-Mails über eine SMTP-Verbindung verschickt. Zuletzt wird der Versandstatus protokolliert und im Terminal ausgegeben. Auf diese Weise wird der grundlegende

Ablauf eines E-Mail-basierten, interaktionsabhängigen selbstreplizierenden Programms vereinfacht nachgebildet.

Um den automatisierten Mailversand zu ermöglichen, wurden auf den beteiligten Systemen Kontaktinformationen in einer Textdatei (*contacts.txt*) hinterlegt (siehe Anhang C, Abbildung 14). Diese enthält E-Mail- und IP-Adressen der jeweiligen Systeme im Netzwerk.

Der Angreifer nutzt für den Versand das Skript *send_malicious_mail.py*, während die Opfer den Anhang *Abrechnung4.Quartal25.py* öffnen. Beide Programme besitzen die gleichen Funktionalitäten. Durch den als Anhang getarnten Dateinamen wird jedoch die Weiterverbreitung der E-Mail ausgelöst (siehe Anhang A, Codeblock 1). Auf diese Weise verbreitet sich der Anhang weiter.

Die Anwendung nutzt außerdem Python-Module wie *smtplib*[4] und *email.message*[5], zur programmgesteuerten Übertragung der Nachrichten sowie zum Versand des ausführbaren Anhangs (siehe Anhang A, Codeblock 1).

Einschränkungen und praktische Herausforderungen

Während der Einrichtung der Testumgebung traten technische Probleme bei der Netzwerkeinrichtung auf, sodass ich die Netzwerkparameter und verwendeten Module mehrfach anpassen musste.

Während der praktischen Umsetzung traten weitere technische Einschränkungen auf, die den Umfang des Versuchsaufbaus beeinflussten. So konnte ich die Einbindung eines grafischen Mailclients wie etwa *Thunderbird*[6] oder *KMail*[7] nicht umsetzen. Da der Schwerpunkt der Arbeit jedoch auf dem Verbreitungsmechanismus und der Nutzerinteraktion lag, habe ich daher bewusst auf eine grafische Oberfläche verzichtet. Diese Entscheidung stellt eine methodische Vereinfachung dar und schränkt die Übertragbarkeit der Ergebnisse auf vergleichbare reale Angriffsszenarien entsprechend ein.

[4] smtplib Python 3.14.2: https://docs.python.org/3.14/library/smtplib.html
[5] email.message Python 3.14.2: https://docs.python.org/3/library/email.message.html
[6] Thunderbird 147.0: https://www.thunderbird.net/en-US/
[7] KMail 6.6.1: https://apps.kde.org/kmail2/

Abgrenzung des Szenarios (Technik)

Der Versuchsaufbau basiert auf einer begrenzten Anzahl an Systemen, wodurch eine direkte Übertragung der Ergebnisse auf große Unternehmensnetzwerke nur eingeschränkt möglich ist. In realen Umgebungen sind in der Regel deutlich mehr Systeme und Nutzer beteiligt.
Zudem enthält das entwickelte Programm keinen Schadcode und ist bewusst einfach gehalten. Es ist nicht darauf ausgelegt, mit dem Schadenspotenzial klassischer Schadprogramme wie Würmern vergleichbar zu sein, und unterscheidet sich deutlich von professionell entwickelten Malware-Systemen.
Darüber hinaus handelt es sich nicht um ein vollständig autonomes selbstreplizierendes Programm im Sinne klassischer Netzwerk-Würmer. Die Weiterverbreitung erfolgt ausschließlich nach einer simulierten Nutzerinteraktion, vergleichbar mit dem *ILY-Wurm* oder anderen massenhaft versendeten Spam-E-Mails (*Mass-Mailer*).
Für den Versuchsaufbau nehme ich an, dass der Angreifer bereits Zugriff auf das interne Netzwerk besitzt. Sicherheitsmechanismen wie Firewalls oder Intrusion-Detection-Systeme werden daher nicht berücksichtigt.
Ziel meiner Simulation ist es, grundlegende Verbreitungsmechanismen eines selbstreplizierenden Programms in vereinfachter Form darzustellen. Der Fokus liegt dabei auf der methodischen Analyse von Social Engineering als Ausgangspunkt des Angriffs.

3.2.2 E-Mail-basierte Verbreitung im Netzwerk

Die Simulation bildet die Verbreitung innerhalb des virtuellen Netzwerks in mehreren Phasen ab. Hierbei gilt die Ausführung des manipulierten E-Mail-Anhangs als zentraler Auslöser für den weiteren Ablauf.
In der ersten Phase initiiert das Angreifer-System den Versand der präparierten E-Mail mithilfe des Skripts *send_malicious_mail.py*. Abbildung 8 zeigt den zugehörigen SMTP-Kommunikationsverlauf und belegt die erfolgreiche Übertragung der E-Mail inklusive des manipulierten Anhangs an das erste Opfer-System.
Die anschließende Nutzerinteraktion wird durch das Öffnen des als Abrechnungsdokument getarnten Programms auf dem System des Angreifers modelliert. *Abbildung 9* zeigt exemplarisch die Ausführung des Anhangs, durch die der weitere Verbreitungsmechanismus aktiviert wird. Aufgrund fehlender Komponenten auf den Opfersystemen, insbesondere eines E-Mail-

Clients, erfolgt die Ausführung des Anhangs in der Simulation auf der Angreifer-VM. Dadurch lässt sich der nutzerinitiierte Weiterverbreitungsmechanismus trotz der abstrahierten Ausführung nachvollziehbar darstellen.

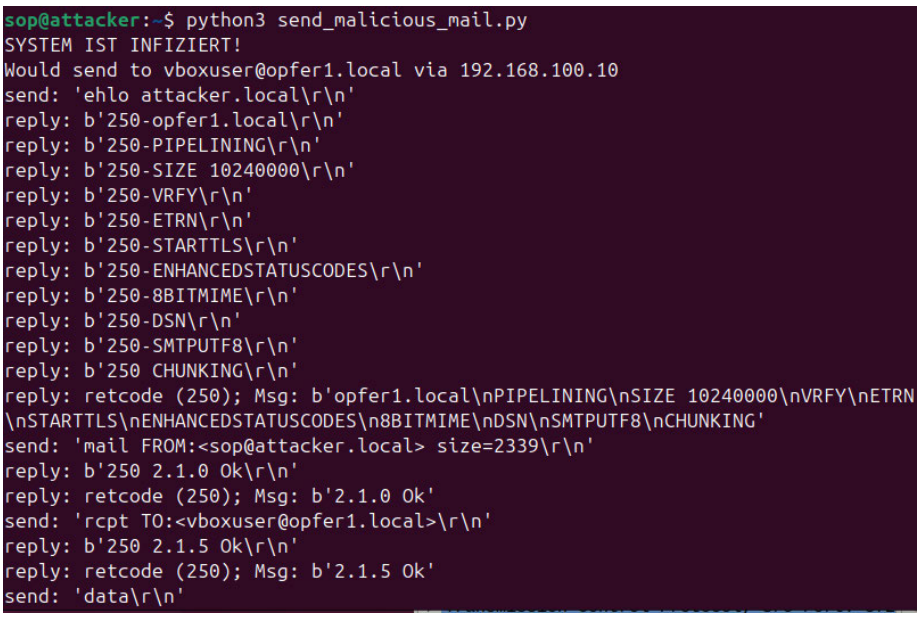


Abbildung 8: Öffnen des Mailskripts im Terminal von Attacker-VM
Quelle: eigene Darstellung

Abbildung 9: Öffnen der Abrechnungsdatei im Terminal von Attacker-VM
Quelle: eigene Darstellung

Nach der erstmaligen Ausführung der Skripte wurde die Datei *infected.txt* lokal im Home-Verzeichnis des Systems angelegt und der Infektionsstatus im Terminal ausgegeben: „SYSTEM IST

INFIZIERT“. Unmittelbar im Anschluss erfolgte der automatisierte Versand von E-Mails an alle in der Kontaktliste hinterlegten Systeme.
Ein Mailversand wird dabei nur auf Systemen ohne vorhandene *infected.txt-Datei* ausgelöst; auf bereits infizierten Systemen bleibt er aus (siehe Abbildung 10).

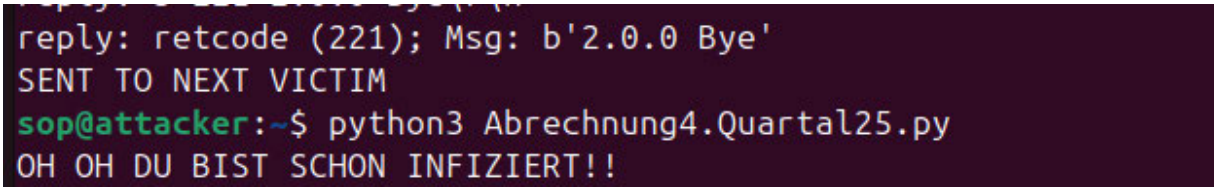


Abbildung 10: Abbruch der Mail-Replikation bei infizierter VM
Quelle: eigene Darstellung

Die Analyse der Netzwerkkommunikation zeigt, dass der Versand der E-Mails unmittelbar nach dem Aufbau der TCP-Verbindung erfolgt und innerhalb sehr kurzer Zeit abgeschlossen ist (siehe Anhang

B SMTP Analyse des SE Szenarios). Die in Wireshark sichtbaren SMTP- und IMF-Daten bestätigen, dass vollständige E-Mail-Inhalte, inklusive Absenderinformationen und Anhang, übertragen werden (siehe Anhang C Sonstige Dateien SE Szenario Abbildungen 15 und 16).

Die bisherigen Kapitel haben gezeigt, wie moderne Schadsoftware mit menschlicher Manipulation zusammenwirken kann. Die Simulation eines beispielhaften SE-Angriffs veranschaulichte diese Wirkung.
Darauf aufbauend werden im nächsten Kapitel Sicherheitsmaßnahmen vorgestellt, die darauf abzielen, netzwerkweite SE-basierte Angriffe zu verhindern und ihre Folgen zu begrenzen.

4 IT-Sicherheitsstrategien

Seit der Erschaffung von Schadprogrammen existieren Ansätze zu deren Abwehr. Das *Reaper* -Programm stellte eine der ersten solchen Verteidigungslinien dar. Als eine Art Antivirusprogramm erkannte es schadhafte Programminstanzen und löschte sie.
Mit der zunehmenden Vielfalt technischer Merkmale der verschiedenen Malware-Arten (siehe Kapitel 2) mussten Sicherheitsmechanismen kontinuierlich an sie angepasst werden.

Ziel war es, die Sicherheit von IT-Infrastrukturen durch die Absicherung technischer Schwachstellen zu gewährleisten.
Social Engineering stellt in diesem Zusammenhang eine veränderte Angriffsstrategie von Schadprogramm-Angriffen dar. Durch die Beeinflussung menschlicher Gefühle können schadhafte Inhalte in abgesicherte Infrastrukturen gelangen, ohne Sicherheitslücken dafür ausnutzen zu müssen.
Die im Rahmen dieser Arbeit entwickelte Simulation verdeutlicht diesen Zusammenhang anhand der Kombination eines E-Mail-basierten SE-Szenarios mit einer selbstreplizierenden Schadsoftware. Da solche Programme gezielt darauf ausgelegt sind, mehrere Systeme automatisiert zu erreichen und zu schädigen, ergeben sich daraus erhöhte Anforderungen an geeignete Schutz- und Abwehrmaßnahmen.

Im folgenden Kapitel werden Schutzmechanismen vorgestellt, die auf dem in der Simulation eingesetzten Szenario basieren. Hierzu werden technische, organisatorische sowie menschliche Faktoren gezielt einbezogen.

4.1 System- und Plattformschutz

Selbstreplizierende Programme wie Würmer nutzen häufig Schwachstellen auf Betriebssystem- und Anwendungsebene, um sich automatisiert zu verbreiten und weiteren Schadcode ausführen zu können. Entsprechend spielen systemnahe Schutzmechanismen eine zentrale Rolle bei der Prävention und Eindämmung solcher Angriffe.

Moderne Betriebssysteme und Compiler verfügen über eine Vielzahl integrierter Schutzmechanismen. Diese zielen darauf ab, klassische speicherbasierte Angriffe frühzeitig zu erkennen oder vollständig zu verhindern. Diese Maßnahmen sollen insbesondere die Ausführung eingeschleuster Schadcodes erschweren.
Ein zentraler Mechanismus hierfür ist der Einsatz sogenannter Stack-Schutzverfahren. Dabei werden vor sicherheitskritischen Speicherbereichen spezielle Prüfwerte angelegt, die beim Verlassen der Funktion überprüft werden [17]. Wird ein solcher Wert verändert, etwa durch das Überschreiben von Speicherbereichen außerhalb der vorgesehenen Grenzen *(Buffer-Overflow-Angriff)*, erkennt das System diese Manipulation. Es beendet dann den betroffenen Prozess, bevor eingeschleuster Code ausgeführt werden kann. Auf diese Weise lassen sich

viele klassische Formen der Schwachstellenausnutzung bereits in einer frühen Phase unterbinden.

Ergänzend dazu kommt die Adressraum-Randomisierung zum Einsatz [17]. Hierbei werden Speicheradressen von Prozessen, Dateien und Systemkomponenten bei jedem Programmstart zufällig neu angeordnet. Schadprogramme, insbesondere Würmer, waren häufig auf fest kodierte Speicherbereiche ausgelegt, um Funktionen oder Bibliotheken aufzurufen. Durch die zufällige Anordnung der Komponenten wurde dieser Angriffsweg ineffektiv und verlor damit an Wirksamkeit.

Zero-Trust

Über diese klassischen Betriebssystemmechanismen hinaus gewinnt die Einbindung einer *Zero-Trust-Architektur* zunehmend an Bedeutung. *Zero-Trust* basiert auf der Grundannahme, dass keinem Benutzer, Gerät oder Dienst grundsätzlich vertraut werden darf, unabhängig davon, ob sich dieser innerhalb oder außerhalb des internen Netzwerks befindet [56], [57]. Stattdessen wird jeder Zugriff kontinuierlich überprüft und kontextabhängig bewertet.

Für Würmer ist dies besonders relevant, da deren Verbreitung meist automatisiert erfolgt und sich durch auffällige Zugriffsmuster äußern kann [17]. Erkennt das System ein erhöhtes Risiko, kann die weitere Ausbreitung unmittelbar unterbunden werden.

Ein zentraler Aspekt dieser Architektur ist die konsequente Authentifizierung aller beteiligten Entitäten [56]. Selbstreplizierende Programme nutzen häufig manipulierte Prozesse oder Dienste mit erweiterten Rechten, um ihre eigene Ausführung zu ermöglichen [17]. *Zero-Trust* verlangt daher nicht nur die Authentifizierung des Nutzers, sondern auch die des ausführenden Prozesses sowie des verwendeten Endgeräts. Durch regelmäßige Re-Authentifizierungen wird sichergestellt, dass ein Prozess nicht dauerhaft privilegiert handeln kann, selbst nachdem er kompromittiert wurde.

Darüber hinaus wird der Sicherheitszustand der beteiligten Systeme kontinuierlich überprüft [56]. Dazu zählen unter anderem die Aktualität der verwendeten Sicherheitssoftware, die Konfiguration des Systems und mögliche Malware-Indikatoren. Wird dabei festgestellt, dass eine selbstreplizierende Schadsoftware das System angreift, kann der Zugriff auf weitere Ressourcen blockiert oder eine laufende Sitzung beendet werden. Dies kann die netzwerkinterne Verbreitung eines Wurms stark einschränken.

Zugriffe erfolgen außerdem standardmäßig nach dem Prinzip der minimalen Rechtevergabe [56]. Selbst wenn sich ein Wurm auf einem System einnistet, schränkt eine Vielzahl von Berechtigungen seinen Zugriff auf weitere Dienste und Dateien ein. Diese ressourcenbezogene Zugriffskontrolle reduziert die Möglichkeit, dass das Programm sich innerhalb eines Systems oder Netzwerks weiter ausbreiten könnte.
Zusätzlich werden alle Zugriffe und sicherheitsrelevanten Ereignisse kontinuierlich überwacht und protokolliert [56]. Auffällige Muster wie automatisierte Prozessaufrufe können frühzeitig erkannt und in die laufende Risikobewertung einbezogen werden. Auf diese Weise lassen sich selbstreplizierende Aktivitäten auch dann identifizieren, wenn keine bekannte Signatur vorliegt.

Anwendungs- und Ausführungsrestriktionen

Außerhalb von *Zero-Trust-Architekturen* stellt die Einschränkung der Programmausführung einen zentralen Schutzmechanismus gegen selbstreplizierende Schadsoftware dar. Dies wird durch Features wie die hardwareseitige Implementierung des NX-Bits[8] (No-Execute) ermöglicht. Es verhindert die Ausführung von *Buffer-Overflow-Angriffen,* beispielsweise auf Windows-Systemen [17]. Dieser Aspekt ist besonders relevant, da Würmer wie der Morris-Wurm entsprechende Techniken einsetzen, um sich auf andere Systeme zu verbreiten [23].
Ein weiterer Schutzansatz stellt das *Application Whitelisting* dar, bei dem ausschließlich zuvor freigegebene Programme ausgeführt werden dürfen [58], [59]. Nicht autorisierte Anwendungen werden standardmäßig blockiert, wodurch neuartige oder modifizierte Programme, die Würmer enthalten können, an der Ausführung gehindert werden.
Der *Blacklisting*-Ansatz gilt demgegenüber als weniger wirksam, da selbstreplizierende Schadsoftware durch Veränderungen ihres Auftretens statische Sperrlisten umgehen kann [60].
Der richtige Umgang mit Dateitypen und Ausführungskennzeichnungen spielt ebenfalls eine wichtige Rolle. Schadsoftware kann irreführende Dateiendungen nutzen, um ihre potenzielle Ausführbarkeit zu verschleiern. Die Verbreitung des *ILY-Wurms* wurde durch dieses Vorgehen begünstigt [17]. Betriebssysteme, die die Ausführung von Programmen an explizite Kennzeichnungen oder Berechtigungen knüpfen, können die unbeabsichtigte Ausführung eingeschleuster Codes reduzieren.

[8] Scribd NX bit: https://www.scribd.com/document/60416747/NX-bit

Frühzeitiges Schwachstellen- und Integritätsmanagement

Neben den bereits genannten Maßnahmen ist ein proaktives Vulnerabilitätsmanagement ein wesentlicher Bestandteil des System- und Betriebssystemschutzes. Dazu zählen regelmäßige Sicherheits- und Softwareupdates, die Durchführung regelmäßiger System-Backups sowie die Begrenzung von Benutzerrechten [61], [62]. Auch die konsequente Durchsetzung angemessener Passwortkomplexität trägt maßgeblich zur Reduktion der Angriffsfläche bei.

Die Notwendigkeit dieser Maßnahmen zeigt sich exemplarisch am Wurm *NotPetya*, der gezielt veraltete Systeme ausnutzte, um sich innerhalb von Netzwerken rasant zu verbreiten [61].

Ergänzend dazu ist ein konsequentes Integritätsmanagement erforderlich, das Manipulationen an Systemdateien, Konfigurationen oder sicherheitsrelevanten Komponenten erkennt, die auf eine aktive Infektion hinweisen [15].

Darüber hinaus sollten alle nicht benötigten Netzwerkports deaktiviert oder unzugänglich gemacht werden, da diese potenzielle Eintrittspunkte für netzwerkbasierte Schadsoftware darstellen [17].

Zusätzlich werden Blockchain-Technologien als ergänzende Maßnahme zur digitalen Identitätsauthentifizierung erforscht [7].

4.2 Netzwerkschutz

Da sich selbstreplizierende Schadsoftware nach der Erstinfektion häufig über Netzwerkschnittstellen weiterverbreitet, ist die Absicherung verteilter Umgebungen besonders wichtig. Netzwerkbasierte Schutzmaßnahmen zielen darauf ab, die Verbreitung schädlicher Inhalte bereits auf der Übertragungsebene zu verhindern.

Würmer sind insbesondere auf offene Netzwerkdienste und automatisierte Kommunikationsprozesse angewiesen, um neue Zielsysteme zu erreichen. Durch die folgenden Strategien soll diese Ausbreitung frühzeitig eingeschränkt oder vollständig unterbunden werden.

Eine erste Schutzinstanz stellen Router dar, die mithilfe von Zugriffskontrolllisten wie einer *Access List* den Netzwerkverkehr filtern. Diese beurteilen anhand von Quell- und Zieladressen, Ports sowie verwendeten Protokollen, ob eine Verbindung zugelassen werden soll [17].

Würmer nutzen zudem häufig Mechanismen wie ICMP-Echo-Anfragen, um erreichbare Zielsysteme zu identifizieren [17]. Werden diese blockiert, kann die automatische Suche nach Zielen eingeschränkt werden. Gleichzeitig kann diese vollständige Blockierung den regulären

Netzwerkbetrieb beeinträchtigen und sollte daher nur auf Systemen mit hohem Schutzbedarf eingesetzt werden.

Ebenso kann das Unterbinden von Paketfragmentierung dazu beitragen, Angriffe zu verhindern, bei denen Inhalte fragmentiert verschickt werden, um *Access Listen* zu umgehen [17]. Wie bereits zuvor dargestellt, können auch hierbei Einschränkungen der Netzwerkfunktionen auftreten.

Ein zentraler Punkt der Netzwerksicherheit sind *Firewalls*, die den ein- und ausgehenden Netzwerkverkehr kontrollieren und nicht autorisierte Verbindungen blockieren [17], [63]. Firewalls unterscheiden sich jedoch hinsichtlich ihrer Funktionsweise.

Während einfache Paketfilter einzelne Netzwerkpakete betrachten, analysieren zustandsbehaftete Firewalls den gesamten Verbindungsaufbau. Proxybasierte Firewalls agieren zusätzlich als Vermittler zwischen Client und Zielsystem und prüfen den Datenverkehr auf Anwendungsebene. Das bedeutet, dass sie nicht nur technische Verbindungsinformationen wie IP-Adressen oder Ports betrachten, sondern auch den eigentlichen Inhalt der Kommunikation, etwa E-Mail-Nachrichten oder Webanfragen.

Richtig konfiguriert können Firewalls die Auswirkungen eines Schadprogrammbefalls erheblich reduzieren, indem schädliche Anfragen frühzeitig erkannt und abgefangen werden.

Neben der Kontrolle eingehender Verbindungen ist auch die Überwachung ausgehenden Datenverkehrs relevant, da kompromittierte Systeme häufig selbst neue Verbindungen initiieren, um weitere Zielsysteme ausfindig zu machen [17].

Ergänzend zu Firewalls kommen *Intrusion-Detection/Prevention-Systeme* (IDS/IPS) zum Einsatz, die den Netzwerkverkehr kontinuierlich analysieren und verdächtige Aktivitäten identifizieren [17]. Ein IPS kann zusätzlich das Eindringen von Schadsoftware verhindern [63].

Im Gegensatz zu Firewalls analysieren IDS/IPS den Netzwerkverkehr detailliert und berücksichtigen dabei auch zeitliche Zusammenhänge. Solche Systeme können entweder protokollierend arbeiten oder aktiv in den Datenverkehr eingreifen. Die Möglichkeit des signaturbasierten Erfassens von Bytefolgen oder Mustern innerhalb von Netzwerkpaketen eignet sich insbesondere zur Erkennung bereits bekannter Schadsoftware. Ergänzend dazu ist eine Analyse auf Basis vordefinierter Kriterien sowie des Rufs von IP-Adressen möglich [63].

Da selbstreplizierende Schadsoftware häufig in modifizierter Form auftritt, spielen anomaliebasierte Erkennungsverfahren eine immer größere Rolle [17], [63]. Diese analysieren das nor-

male Netzwerkverhalten und identifizieren Abweichungen wie ungewöhnlich lange Protokollfelder oder atypische Kommunikationsmuster. Diese Unregelmäßigkeiten können Hinweise auf Angriffsversuche sein. Eine besondere Herausforderung stellt dabei die gezielte Fragmentierung von Paketen dar, mit der Schadprogramme versuchen, Erkennungsmechanismen zu umgehen [17]. Effektive IDS/IPS müssen daher in der Lage sein, fragmentierte Pakete korrekt wieder zusammenzusetzen, bevor eine Analyse der Netzwerkkommunikation erfolgt.

Neben typischen netzwerkbasierten Schutzmechanismen rücken neuartige datengetriebene Ansätze zur Angriffserkennung in den Fokus. KI-basierte Verfahren, insbesondere solche des Machine Learning, ermöglichen die Echtzeit-Erkennung von Anomalien im Netzwerkverkehr und im Nutzerverhalten [7]. Basierend auf den verwendeten Trainingsdaten können sie vergangene Malware-Angriffe analysieren und dadurch Rückschlüsse auf neue Malware-Varianten ziehen. Somit können sie diese erkennen, bevor eine Infektion stattfinden kann. Diese Angriffserkennung ist damit insbesondere gegen neuartige oder schwer erkennbare selbstreplizierende Schadsoftware wirksam.

Unabhängig davon bleiben strukturelle Schwachstellen auch in cloudbasierten Umgebungen ein zentrales Sicherheitsproblem. Fehlkonfigurationen stellen einen der häufigsten Angriffsvektoren dar [64]. Offen erreichbare Speicherressourcen, überprivilegierte Benutzerkonten sowie öffentlich erreichbare, schlecht gesicherte Anwendungen begünstigen automatisierte Angriffe. Durch diese Faktoren kann eine schnelle Ausbreitung selbstreplizierender Schadsoftware zusätzlich erleichtert werden.

Aufbauend auf dem bereits angesprochenen *Zero-Trust*-Ansatz (S.36) können die dort lokal verwendeten Sicherheitsrichtlinien standortunabhängig durch cloudbasierte Sicherheitsdienste durchgesetzt werden [56]. Dies unterstützt die Eindämmung einer unkontrollierten Ausbreitung selbstreplizierender Schadsoftware insbesondere in verteilten Infrastrukturen.

Auch die Absicherung von Programmierschnittstellen (API) gewinnt zunehmend an Bedeutung, da diese häufig direkt aus dem Internet erreichbar sind und als zentraler Zugriffspunkt zwischen Anwendungen dienen. Unzureichend geschützte APIs können gezielt von Schadprogrammen angesprochen werden, etwa zur automatisierten Ausnutzung fehlerhafter Authentifizierungs- oder Zugriffskontrollen [64].

Zusätzlich kann in sicherheitskritischen Umgebungen die vollständige Trennung vom Netzwerk (*Air-Gap)* eine Ausbreitung von Schadsoftware verhindern [9]. Eine solche Isolation

bringt jedoch deutliche Einschränkungen im Betrieb und bei Wartungsarbeiten mit sich, weshalb sie nur in Einzelfällen durchgesetzt werden sollte.

4.3 E-Mail-Sicherheit

Da die E-Mail im untersuchten Szenario (S. 25) den primären Verbreitungskanal der Schadsoftware darstellt, gilt sie zugleich als Beispiel für einen besonders kritischen Angriffsvektor. Neben technischen Schwachstellen wird dabei gezielt das Verhalten der Nutzer instrumentalisiert, um Malware in Systeme oder Netzwerke einzuschleusen.

Insbesondere gegen E-Mail-basierte Schadprogramme kommen spezialisierte Erkennungs- und Blockierungsmechanismen zum Einsatz. Netzwerkbasierte Systeme wie Firewalls und IDS/IPS können unter anderem ungewöhnlich hohe Verbindungsraten und fehlerhafte Zustellversuche erkennen, die auf eine automatisierte Verbreitung hindeuten [17].
Darüber hinaus existieren hostbasierte Schutzmechanismen, die speziell auf das selbstständige Versenden von E-Mails reagieren. SMTP-Proxys können beispielsweise erkennen, ob der Vorgang, der eine E-Mail versendet, gleichzeitig Bestandteil des Anhangs ist und den Versand entsprechend unterbinden [17].
Solche Verfahren haben sich insbesondere gegen *Mass-E-Mails* als äußerst wirksam erwiesen und können auffällige Versandmuster frühzeitig erkennen und unterbinden [17]. Die schädliche "*Mydoom*"-Variante konnte durch sie in vielen Fällen vor der Weiterverbreitung gestoppt werden [17].
Ergänzend zu diesen Mechanismen kommen präventive Maßnahmen zum Einsatz, die den E-Mail-Versand technisch einschränken sollen. Durch die Begrenzung mailfähiger Rechner und das Blockieren von Port 25 lassen sich Angriffe von außen oder durch unbekannte Absender erschweren [65]. Dies bietet jedoch keinen zuverlässigen Schutz, wenn sich der Angreifer durch Adressenmanipulation *(Spoofing)* als interner Absender ausgibt oder sich bereits innerhalb des Netzwerks befindet. In Kombination mit weiteren SE-Methoden erschweren diese Faktoren eine wirksame Abwehr erheblich.
Ein weiterer wichtiger Schritt zur Verbesserung der E-Mail-Sicherheit ist die Absicherung des Mailverkehrs auf Domainebene. Die Domainebene beschreibt dabei die Ebene der absendenden *E-Mail-Domain* (z. B. *@beispiel.com*) und ermöglicht es, bereits während der Übertragung

zu überprüfen, ob eine E-Mail von einer autorisierten Quelle stammt. Sie trägt dazu bei, dass keine manipulierten E-Mails im Postfach des Empfängers landen.

Verfahren, die speziell auf dieser Ebene ansetzen, sind SPF, DKIM und DMARC**.** *Sender Policy Framework* (SPF) [65], [66] ist ein Verfahren, mit dem eine Domain festlegt, welche Mailserver berechtigt sind, E-Mails in ihrem Namen zu versenden. Diese Information wird als Eintrag im *Domain Name System* (DNS) hinterlegt. Der empfangende Mailserver überprüft bei E-Mail-Eingang die IP-Adresse des sendenden Servers. Diese wird mit den für die Domain hinterlegten erlaubten Absendern abgeglichen. Ist die Absenderadresse nicht hinterlegt, schlägt die SPF-Prüfung fehl.

SPF trägt dazu bei, unerlaubte SMTP-Verbindungen von nicht autorisierten Mailservern zu erkennen. Es kann jedoch keine *Spoofing*-Angriffe verhindern, die von einem kompromittierten System innerhalb des gleichen Netzwerks ausgehen.

DomainKeys Identified Mail (DKIM) [65], [66] ist ein Verfahren, mit dem eine Domain die Echtheit des Inhalts einer E-Mail feststellt. Dazu wird die Nachricht vom sendenden Mailserver mit einer kryptografischen Signatur, beispielsweise RSA[9], versehen. Beim Empfang der E-Mail prüft der empfangende Mailserver diese Signatur, indem er den zugehörigen öffentlichen Schlüssel über einen DNS-Eintrag der signierenden Domain abruft. Ist die Signatur gültig, dann wurde die E-Mail während der Übertragung nicht verändert. DKIM schützt daher primär die Authentizität einer E-Mail. Das Verfahren überprüft jedoch nicht den konkreten Inhalt auf schädliche sowie manipulative Nachrichten, die trotz gültiger Verschlüsselung Nutzer beeinflussen können.

Während SPF- und DKIM-empfangende Mailserver lediglich Hinweise zur Herkunft und Authentizität einer E-Mail liefern, ermöglicht DMARC das konkrete Aussortieren von risikoreichen E-Mails.

Domain-based Message Authentication, Reporting and Conformance (DMARC) [65], [66] ergänzt SPF und DKIM um verbindliche Richtlinien, indem Domaininhaber festlegen können, wie E-Mails zu behandeln sind, die die Prüfungen der Absenderauthentizität nicht bestehen. DMARC verbindet die Ergebnisse von SPF und DKIM und prüft, ob diese zur Absenderdomain passen. Dadurch lassen sich gefälschte oder manipulierte E-Mails besser erkennen.

[9] RSA Erklärung: http://www.nord-com.net/h-g.mekelburg/krypto/mod-asym.htm#rsa

Darüber hinaus stellt DMARC ein Berichtssystem bereit, das Domaininhabern Informationen zu E-Mails bereitstellt, die vorgeben, von ihrer Domain zu stammen. Auf diese Weise kann der mögliche Missbrauch einer Domain transparenter gemacht und besser bewertet werden. Um Angriffe mit gespooften Absenderadressen zu erschweren, sollten SPF, DKIM und DMARC gemeinsam eingesetzt werden. Diese Verfahren setzen bereits zwischen Sender- und Empfängerdomain an und können die Zustellung schadhafter E-Mails einschränken.

Weitere Möglichkeiten zur Absicherung der Mail-Infrastruktur sind der Einsatz von *Black*- und *Whitelists* für Absenderadressen [65]. Dabei können unerwünschte Absender gezielt blockiert oder ausschließlich bestimmte Absender zugelassen werden. Diese Listen sollten jedoch dynamisch gepflegt werden, da Angreifer die Absenderadressen häufig verändern. Statische Listen bieten deshalb nur einen eingeschränkten Schutz.
Zusätzlich können ausgehende E-Mails überprüft werden, indem Absender- und Empfängeradressen mit bestehenden *Blacklists* abgeglichen werden. Dies soll mögliche Infektionen frühzeitig erkennbar machen [65].

Content-Filtering ist eine zusätzliche Maßnahme, die den Mail-Verkehr sicherer machen soll. Sie eignet sich insbesondere dafür, E-Mail-basierte Angriffe wie Phishing abzufangen [65]. Der Inhalt einer E-Mail wird gezielt auf typische SE-Formulierungen und verdächtige Begriffe analysiert. Zusätzlich können Anhänge wie Dateien, Bilder oder Links überprüft werden, indem sie in einer isolierten Umgebung (*Detonation Chamber*) geöffnet oder ausgeführt werden. Wird dabei schadhafter Inhalt erkannt, wird die E-Mail nicht an den Empfänger weitergeleitet. Auch ausgehende E-Mails können auf Schadsoftware oder Phishing-Inhalte überprüft werden, um deren interne Verbreitung zu verhindern.

Aufbauend auf diesen inhaltsbasierten Prüfmechanismen können zusätzliche Filter eingesetzt werden. Sie sollen dem massenhaften automatisierten Versand oder Empfang von E-Mails, etwa im Rahmen einer DDoS-Attacke, entgegenwirken [65].
Filter erkennen, wenn eine hohe Anzahl identischer E-Mail-Inhalte innerhalb kurzer Zeit eintrifft, und blockieren entsprechende Nachrichten. Ferner können übergroße E-Mails oder bestimmte Inhalte bereits im Vorfeld anhand von *Blacklists* abgewiesen werden [65].

4.4 Organisatorische Maßnahmen

Social Engineering wird bei Cyberangriffen zunehmend eingesetzt, um Schadsoftware wie selbstreplizierende Programme trotz technischer Schutzmaßnahmen in IT-Umgebungen einzuschleusen. Der alleinige Einsatz technikbezogener Sicherheitsvorkehrungen stößt dabei an seine Grenzen. Vor diesem Hintergrund wird es notwendig, auch die Nutzenden gezielt im Umgang mit Manipulationstechniken zu schulen.
Bereits die Infektion einzelner Systeme kann ausreichen, um die Sicherheit eines ganzen Netzwerks zu gefährden. Aus diesem Grund sind klare organisatorische Regelungen sowie verbindliche Verhaltensrichtlinien im Umgang mit E-Mails und digitalen Inhalten erforderlich.

Ein möglicher Ansatz zur Stärkung der organisatorischen Sicherheit stellt das *Continuous Threat Exposure Management* dar. Dieses Modell setzt auf Vorbeugung, indem Angriffsflächen überwacht, Angriffe simuliert sowie Erkennungs- und Abwehrreaktionen automatisiert werden [64]. Ziel ist es, potenzielle Risiken frühzeitig zu identifizieren und Sicherheitsmaßnahmen laufend anzupassen.
Die Vorbereitung auf Sicherheitsvorfälle spielt daher eine zentrale Rolle für unternehmensweites Sicherheitsbewusstsein. Die Bewältigung solcher Vorfälle sollte regelmäßig geübt werden, um Geschäftsprozesse im Ernstfall aufrechterhalten oder schnell wiederherstellen zu können [67]. Dazu zählen klar definierte Abläufe zur Eindämmung von Schäden sowie regelmäßig erstellte und überprüfte Datensicherungen, die im Notfall zuverlässig wiederhergestellt werden können.
Zudem können reale Angriffsszenarien nachgestellt werden, um Nutzer auf typische Angriffsmuster hinzuweisen. Hierfür eignen sich verschiedene Planspiele, unter anderem zu Phishing-Attacken oder als Dokumente getarnten Schadprogrammen [68].
Eine weitere Maßnahme stellt die gezielte Schulung von Nutzern im Umgang mit Phishing-E-Mails und anderen SE-Techniken dar [65], [68]. Nutzer sollten darüber aufgeklärt werden, wie typische Merkmale verdächtiger E-Mails zu erkennen sind. Dazu zählen unter anderem standardisierte Anreden wie „Sehr geehrter Kunde“ oder „Guten Tag“, inhaltlich unspezifische Texte, die Druck erzeugen oder Neugier wecken, sowie Aufforderungen zur Preisgabe sensibler Informationen.

Auch das Überprüfen von Links und auffälligen Datei-Endungen kann dazu beitragen, Phishing-Versuche frühzeitig zu erkennen. Werden auf verlinkten Webseiten persönliche oder finanzielle Daten abgefragt, sollte grundsätzlich keine Eingabe erfolgen.

Zudem kann bei ausgehenden E-Mails aufmerksames Nutzerverhalten dabei helfen, ungewöhnliches Verhalten der eigenen Systeme zu erkennen und frühzeitig zu reagieren. Geschulte Mitarbeiter stellen dabei häufig die letzte Verteidigungslinie dar, insbesondere wenn automatisierte Schutzmechanismen schadhafte Inhalte nicht erkennen konnten [65]. Ein erhöhtes Sicherheitsbewusstsein kann deshalb dazu beitragen, verdächtige E-Mails zu identifizieren, um schädliche Interaktionen zu vermeiden.

Innerhalb des Unternehmensumfelds sollte darauf aufbauend eine offene Sicherheitskultur gefördert werden, in der das Stellen von Fragen als normal gilt [68]. Dies unterstützt, dass auffällige Ereignisse wie die massenhafte Verbreitung ungewöhnlicher Dateien frühzeitig gemeldet und intern ausgewertet werden können.

Basierend auf dieser internen Meldekultur ist auch die Meldung von Sicherheitsvorfällen an externe Stellen ein wichtiger organisatorischer Aspekt. Neben der Möglichkeit einer Strafanzeige können Vorfälle auch an staatliche Institutionen wie das BSI gemeldet werden [67]. Dies trägt dazu bei, einen besseren Überblick über aktuelle Bedrohungen zu gewinnen und wiederkehrende Angriffsmuster erkennen zu können.

Da unternehmenseigene Maßnahmen jedoch nicht alle Bedrohungen vollständig abdecken können, lässt sich ein weitreichender Schutz in vielen Fällen nur durch die Einbindung externer Dienstleister ermöglichen [67]. Dazu zählen unter anderem unabhängige Schwachstellenanalysen sowie die Unterstützung durch externe *Computer Emergency Response Teams* bei der Analyse und Bewältigung von Sicherheitsvorfällen.

Ergänzend zu den oben genannten organisatorischen, technischen und menschlichen Strategien können Konzepte wie ein *Social Engineering Defensive Framework* eingesetzt werden, um diese Ansätze zu bündeln und gezielt gegen SE-Angriffe einzusetzen [68].

Viele der beschriebenen Herangehensweisen basieren letztlich auf regulatorischen Vorgaben wie der europäischen *NIS2-Richtlinie* [69] oder dem deutschen *BSI-Gesetz* [70].

Sie verdeutlichen zusätzlich, dass organisatorische Maßnahmen, Meldeprozesse und Verantwortlichkeiten im Bereich der IT-Sicherheit zunehmend verbindlich vorgegeben sind. Eine Nichteinhaltung erhöht nicht nur das Sicherheitsrisiko für Unternehmen, sondern kann auch Sanktionen und Bußgelder nach sich ziehen.

Die bisherigen Erkenntnisse zeigen, dass bereits eine Vielzahl von Sicherheitsmaßnahmen gegen Malware und SE-basierte Angriffe angewandt wird. Ihre Wirksamkeit hängt jedoch maßgeblich davon ab, wie gut diese Maßnahmen aufeinander abgestimmt sind und in welchem Umfang sie an neue Bedrohungen angepasst werden. Da sich Angriffsformen kontinuierlich weiterentwickeln, ist die IT-Sicherheit Teil eines fortlaufenden Prozesses, der auch künftig weitere Anpassungen erfordert.

4.5 Zukunftsaussichten

Zum Abschluss des Sicherheitskapitels wird ein Ausblick auf zukünftige Lösungsansätze auf verschiedenen Sicherheitsebenen gegeben, um auch künftig Schadsoftware-Angriffe frühzeitig erkennen, verhindern und ihre Auswirkungen reduzieren zu können.

System- und Plattformsicherheit

Es ist davon auszugehen, dass Schadsoftware zukünftig vermehrt KI nutzen wird, um ihren Code dynamisch anzupassen und so signaturbasierte Schutzmechanismen zu umgehen [6]. Um dem entgegenzuwirken, gewinnen vorbeugende Sicherheitsansätze mehr an Bedeutung. Dazu zählen insbesondere KI-gestützte Methoden zur frühzeitigen Erkennung von Software-Schwachstellen wie automatische Codeanalysen zur Abwehr von *Buffer-Overflow-Angriffen* [71].

Ein weiterer wichtiger Zukunftsaspekt ist die langfristige Sicherheit sensibler Daten. Im Rahmen der sogenannten „*Harvest-now-decrypt-later*"-Strategie sammeln Angreifer verschlüsselte Informationen, um diese zu einem späteren Zeitpunkt mit leistungsfähigeren Technologien wie Quantencomputern zu entschlüsseln [72]. Vor diesem Hintergrund wird der frühzeitige Einsatz quantensicherer Verschlüsselungsverfahren zunehmend als notwendig erachtet. Diese gelten gegenüber zukünftigen Quantencomputern als widerstandsfähig.

Ergänzend dazu rückt der Ausbau von *Zero-Trust-Architekturen* weiter in den Fokus.
In Kombination mit KI-gestützten Echtzeitanalysen von Nutzerverhalten und Systemaktivitäten sollen so Angriffe frühzeitig erkannt und unautorisierte Zugriffe verhindert werden [73].

Die genannten Aspekte hebt die Präsidentin von Cloudflare zusätzlich hervor: *„AI-driven threats require AI-powered defenses. Zero Trust must be the standard. Post-quantum readiness is not a tomorrow problem — it needs to happen today.“* [72, S. 2].

Gleichzeitig wird der menschenzentrierte Systementwurf besonders hervorgehoben, da er menschliches Fehlverhalten nicht als Ausnahme, sondern als erwartbaren und realistischen Bestandteil moderner IT-Systeme betrachtet [74].

Netzwerksicherheit

Im Bereich der Netzwerksicherheit sind KI-basierte Verfahren, darunter Machine Learning, weiterhin Teil der zukünftigen Entwicklung [7]. Anomaliebasierte Erkennungsalgorithmen könnten dabei weiter verbessert werden, um neuartige Schadprogramme zuverlässiger zu identifizieren.

Auf dieser Grundlage könnten IDS zunehmend durch große Sprachmodelle ergänzt werden, die Netzwerk- und Log-Daten situationsabhängig auswerten [75]. Dadurch wären sie in der Lage, bislang unbekannte Angriffsmuster oder Angriffe, die nicht dokumentierte Schwachstellen nutzen, zu identifizieren.

Ergänzend dazu eröffnen *Generative Adversarial Networks* (GANs) neue Möglichkeiten, bestehende Schutzmechanismen weiterzuentwickeln. Sie erlauben die Erzeugung simulierter Netzwerkdaten, mit denen Sicherheitssysteme realitätsnah getestet und gezielt auf Schwachstellen überprüft werden könnten [76].

E-Mail-Sicherheit

Im Bereich der E-Mail-Sicherheit ist mit einer zunehmenden Professionalisierung von Angriffen zu rechnen. Angreifer nutzen KI-basierte Werkzeuge zur automatisierten Erstellung täuschend echter Phishing-Nachrichten sowie schädlicher Inhalte, wodurch SE-Angriffe anpassungsfähiger, stärker automatisierbar und schwerer erkennbar werden [64].

Trotz dieser technischen Weiterentwicklung zeigen Studien, dass ein Großteil erfolgreicher Angriffe weiterhin auf menschliche Faktoren zurückzuführen ist, etwa durch Täuschung oder Fehlverhalten des Nutzers [72].

Zukünftige Schutzansätze setzen daher verstärkt auf flexible Erkennungsverfahren, die nicht nur bekannte Angriffsmuster analysieren, sondern auch psychologische Merkmale wie Über-

zeugungskraft oder Täuschungsabsicht berücksichtigen [74]. Entsprechende Systeme auf Basis großer Sprachmodelle und neuronaler Netze erreichen bereits eine Erkennungsgenauigkeit von über 90 % bei Spam- und Phishing-E-Mails [77].

Organisatorische Maßnahmen und Regulierung

Neben technischen Entwicklungen nimmt auch der Fortschritt organisatorischer Maßnahmen eine wichtigere Rolle ein.

Schulungs- und Aufklärungskonzepte sollten künftig stärker auf KI-gestützte SE-Angriffe eingehen und weniger auf einzelne Angriffstypen [74]. Zudem sollte das Verhalten der Nutzer gegenüber diesen Angriffen stärker in den Fokus rücken als der Angriff an sich. Ziel ist es, sichere Verhaltensmuster zu fördern, die auch unter hoher Belastung eingesetzt werden können.

Ergänzend dazu werden anpassungsfähige Trainingskonzepte erforscht, die Nutzende regelmäßig mit realitätsnahen, jedoch ungefährlichen Angriffsszenarien konfrontieren, um Sicherheitsbewusstsein und Handlungskompetenz nachhaltig zu stärken [74].

Darüber hinaus beeinflussen rechtliche Rahmenbedingungen zunehmend die zukünftige Ausgestaltung der IT-Sicherheit. Mit dem europäischen *AI-Act* wird ein verbindlicher Rechtsrahmen geschaffen, der den Einsatz von KI-Systemen nach ihrem Risiko kategorisiert und insbesondere ihre missbräuchliche oder sicherheitskritische Anwendung begrenzt [78].

Diese regulatorischen Vorgaben wirken sich unmittelbar auf den Einsatz von KI in der Cybersecurity aus. KI-gestützte Sicherheitslösungen zur Bekämpfung von Social Engineering und selbstreplizierender Malware müssen künftig nicht nur funktional, sondern auch transparent, nachvollziehbar und rechtskonform gestaltet werden.

Zudem schafft der *AI-Act* erstmals einen übergeordneten gesetzlichen Rahmen für den Einsatz von KI. Auf dieser Grundlage könnten künftig weitere Regelungen entstehen, die sich gezielt mit dem missbräuchlichen Einsatz von KI, etwa zur Erstellung von Schadsoftware oder manipulativen Inhalten, befassen.

Zusammenfassend zeigt sich, dass eine zuverlässige IT-Sicherheit nicht durch einzelne Maßnahmen erreicht werden kann, sondern ein Zusammenspiel technischer, organisatorischer und menschlicher Schutzmechanismen erforderlich ist. Die vorgestellten Strategien verdeutlichen, dass insbesondere SE-basierte Angriffe nur durch einen mehrstufigen, weiterentwickelbaren Sicherheitsansatz effektiv abgewehrt werden können.

Im nächsten Abschnitt werden die Ergebnisse der Simulation analysiert und in den Kontext bestehender Forschung zu SE, Malware und IT-Sicherheit eingeordnet. Das Fazit reflektiert anschließend die gewonnenen Erkenntnisse kritisch und stellt mögliche weiterführende Forschungsansätze dar.

5 Diskussion, Fazit und Ausblick

5.1 Diskussion der Ergebnisse

Im Rahmen dieser Arbeit wurde untersucht, welche Rolle SE innerhalb moderner Schadprogramm-Attacken spielt. Dabei wurde analysiert, inwiefern manipulative Methoden als Ausgangspunkt für Malware-Infektionen dienen können, insbesondere in Kombination mit selbstreplizierenden Programmen (Würmern).

Konkret wurde betrachtet, inwiefern SE als zentraler Einflussfaktor zur Verbreitung selbstreplizierender Schadsoftware beiträgt, ohne Systemschwächen auszunutzen (H1:). Zudem wurde evaluiert, in welchem Maße ein erster Malware-Befall vom Nutzer abhängt (H2).
Zur Erforschung von H1 und H2 wurde eine Simulation mit drei Rechnern innerhalb einer virtualisierten Netzwerkumgebung erstellt. Sie sollte veranschaulichen, wie sich ein selbstreplizierendes Programm mithilfe einer manipulativen E-Mail über das SMTP-Protokoll verbreiten kann.

Ergebnisse der Simulation

Die im Zuge dieser Arbeit untersuchten Hypothesen **H1**: und **H2** konnten durch die entwickelte Simulation ***bestätigt*** werden.
Die Simulation zeigt, dass die initiale Verbreitung der selbstreplizierenden Software innerhalb der Rechnerumgebung ausschließlich durch eine Social-Engineering-E-Mail ausgelöst wurde (siehe Anhang

B SMTP Analyse des SE Szenarios). Der Programmcode wurde nach erfolgreicher Manipulation zur Dateiausführung in weniger als einer Sekunde an jedes Zielsystem im Netzwerk verschickt. Eine Ausnutzung systemeigener Schwachstellen wie unsicherer Netzwerkverbindungen war hierfür nicht erforderlich.

Darüber hinaus wurde deutlich, dass eine erste Infektion direkt von der Nutzerinteraktion abhing. Erst durch das Öffnen des manipulierten E-Mail-Anhangs wurde die Schadsoftware ausgeführt und der weitere Verbreitungsprozess in Gang gesetzt (siehe Abbildung 9). Nach dem Versand der E-Mails kam es zu keiner direkten automatischen Ausführung des Programms, sodass keine autonome Infektion stattfand.

Interpretation der Ergebnisse

Die Resultate zur Hypothese H1: zeigen, dass menschliches Verhalten eine zentrale Rolle in modernen Malware-Praktiken spielen kann. Die Simulation verdeutlichte, dass selbst Schadprogramme mit wenigen Funktionen sich weitreichend innerhalb eines Netzwerks verbreiten können. Diese Verbreitung war dadurch möglich, dass Nutzer durch SE zur Ausführung eines schädlichen Anhangs verleitet wurden.

Damit stehen die Ergebnisse im Einklang mit bestehenden Forschungserkenntnissen zum Menschen als Schwachstelle, da sie das Ziel von SE-Angriffen darstellen. Veröffentlichungen von SE-Experten wie Kevin Mitnick „*The Art of Deception*“ [53] belegen ebenfalls, dass solche Angriffe primär durch psychologische Manipulation und nicht nur durch das Ausnutzen technischer Schwachstellen erfolgreich sind.

Zusätzlich machen die Ergebnisse bezüglich der Hypothese H2 deutlich, dass die bloße Zustellung einer präparierten E-Mail nicht ausreicht, um eine Infektion auszulösen. Erst durch bewusste Nutzerinteraktion, konkret das Öffnen des Anhangs, wird der Verbreitungsprozess gestartet. Diese Erkenntnis deckt sich mit der bestehenden Forschung zu E-Mail-basierten Angriffen, die den Erfolg solcher Angriffe stark an das Verhalten der Nutzer koppelt [55]. Die Simulation verdeutlicht damit anschaulich, dass Nutzer nicht nur Angriffsziel, sondern ein aktiver Bestandteil der Infektionskette sind.

Grundsätzlich lässt sich festhalten, dass die Entwicklung eines funktionsbegrenzten selbstreplizierenden Programms bereits mit grundlegenden Kenntnissen in den Bereichen Computer-Netzwerke, Programmierung und logischer Abläufe möglich ist. Der Programmcode verdeutlicht, dass hierfür keine hochspezialisierten technischen Fähigkeiten erforderlich sind (siehe Anhang A, Codeblock 1).

Daraus lässt sich ableiten, dass auch Personen mit begrenzter Erfahrung in der Lage sein könnten, funktionale selbstreplizierende Programme zu erstellen. Darüber hinaus ist zu erwarten,

dass Entwicklungen wie KI-gestützte Tools zur Malwareerstellung die Vielfalt schädlicher Programme weiter vergrößern könnten.

Insgesamt zeigt die Simulation, dass Social Engineering einen effektiven Ansatz für die Infizierung von Netzwerken mit Schadsoftware und deren Verbreitung darstellt.
Die technische Einfachheit des entwickelten Programms zeigt, dass nicht primär technische Schwachstellen, sondern die menschliche Entscheidungsfindung maßgeblich zum Erfolg solcher Angriffe beitragen. Die Ergebnisse unterstreichen damit die zentrale Rolle des menschlichen Faktors, dem innerhalb von defensiven Sicherheitsstrategien besondere Bedeutung zukommen sollte.
Vor dem Hintergrund zukünftiger Einbindungen KI-erschaffener gefährlicher Inhalte oder automatisierter Malware-Kampagnen ist davon auszugehen, dass sich der Implementierungsaufwand für solche Angriffe weiter verringert. Dies könnte dafür sorgen, dass das potenzielle Risiko einer erfolgreichen Schadprogramminfektion innerhalb von gefährdeten Netzwerken entsprechend zunimmt.

Limitationen

Die Ergebnisse dieser Arbeit sind im Kontext der in Kapitel 3 dargestellten methodischen Einschränkungen zu interpretieren. Insbesondere die vereinfachte, virtualisierte Umgebung sowie die grobe Darstellung der Nutzerinteraktion beeinflussen die Übertragbarkeit der Ergebnisse auf reale Netzwerke.
Außerdem wurden die in Kapitel 4 dargestellten Sicherheitsmaßnahmen nicht in den Versuchsaufbau integriert. Entsprechend lassen sich aus der Simulation keine Aussagen über deren konkrete Wirksamkeit ableiten.
Es ist nicht auszuschließen, dass bereits einfache technische Maßnahmen wie ein IDS/IPS oder restriktive Content-Filter die nutzerausgelöste Verbreitung frühzeitig hätten unterbinden können. Sie hätten die entsprechende E-Mail eventuell vor dem Empfang auf den Opfer-Systemen blockiert.
Ergänzend könnten die Folgen organisatorischer Maßnahmen wie gezielter Schulungen von Mitarbeitern zusätzlich zum Fehlschlag eines solchen Angriffs beitragen. Durch die Sensibilisierung der Nutzer hinsichtlich verdächtiger E-Mail-Inhalte wäre der schädliche Anhang wahrscheinlich nicht geöffnet worden.

Unabhängig von den genannten Einschränkungen bleibt offen, inwieweit vergleichbare Ergebnisse erzielt worden wären, wenn sich der Angreifer-Rechner nicht im selben Netzwerk wie die Zielsysteme befunden hätte.

5.2 Fazit

Diese Arbeit untersucht die Rolle von Social Engineering im Zusammenhang mit moderner Schadsoftware. Ziel der Arbeit ist es, zu verdeutlichen, dass moderne Schadsoftware ihren Erfolg nicht mehr ausschließlich aus der Ausnutzung technischer Schwachstellen bezieht, sondern zunehmend auf den Menschen als Angriffsfaktor zurückgreift.

Zur Untersuchung dieses Zusammenhangs wurde ein simulationsbasiertes Szenario entwickelt, in dem ein Angreifer E-Mails mit SE-Inhalten versendete. Diese E-Mails enthielten ein als Dokument getarntes selbstreplizierendes Programm. Nach dem Öffnen des Anhangs wurde das Programm ausgeführt und versendete anschließend automatisch eine eigene Kopie per E-Mail im Netzwerk.

Die Ergebnisse der Simulation unterstreichen, dass SE eine zentrale Rolle bei der initialen Infektion moderner Schadsoftware spielt. Bereits eine einfache Nutzerinteraktion reichte aus, um sicherheitskritische Prozesse auszulösen. Dadurch wurden die Ausführung eines als Schadsoftware modellierten Programms auf den Zielsystemen, der Zugriff auf lokale Kontaktlisten sowie die Initiierung weiterer Netzwerkkommunikation ermöglicht. In Kombination mit der selbstreplizierenden Programmlogik führte dies zu einer schnellen Ausbreitung innerhalb des Netzwerks.

Dies zeigt auf, dass erfolgreiche Cyberangriffe nicht zwingend auf der Ausnutzung komplexer technischer Schwachstellen basieren, sondern auf vorhersehbarem menschlichem Verhalten beruhen können. Dies gilt insbesondere, wenn technische Schutzmechanismen fehlen oder nicht wirksam greifen.
Die beobachteten Ergebnisse stehen zudem im Einklang mit den von Robert Cialdini beschriebenen Prinzipien der sozialen Beeinflussung. Im Hinblick auf die Gestaltung und Wirkung der verwendeten SE-E-Mail dienen diese Prinzipien als möglicher Erklärungsrahmen für das in der Simulation abgebildete Nutzerverhalten.

Die Arbeit hebt somit die anhaltende Relevanz von SE in Angriffsszenarien in Kombination mit Schadsoftware, insbesondere selbstreplizierenden Programmen, hervor. Die entwickelte Simulation zeigt exemplarisch, wie ein solcher realitätsnaher Angriff aufgebaut werden kann.
Abstrahiert zeigen die daraus gewonnenen Erkenntnisse, dass die Auswirkungen von SE innerhalb vernetzter IT-Infrastrukturen erhebliche Dimensionen annehmen können. Damit einhergehend sind weitreichende Folgen wie erhebliche finanzielle Schäden, der Verlust sensibler Daten sowie des Vertrauens in die eigene IT-Sicherheit zu erwarten.
Damit bestätigt die Arbeit die in Kapitel 4 dargestellte Notwendigkeit eines mehrschichtigen Sicherheitsansatzes, der technische Maßnahmen mit organisatorischer Vorsorge und Nutzeraufklärung kombiniert.

5.3 Ausblick

Die entwickelte Simulation verwendet eine vereinfachte, kontrollierte Netzwerkumgebung und bildet reale Netzwerke daher nur eingeschränkt ab. Aufbauend auf den gewonnenen Erkenntnissen ergeben sich jedoch verschiedene Ansätze für weiterführende Untersuchungen.

Zukünftige Arbeiten könnten den Versuchsaufbau um eingesetzte Sicherheitsmechanismen wie *Firewalls* oder *Blacklisting* erweitern, um ihre tatsächliche Wirksamkeit gegen SE-basierte Netzwerkangriffe zu untersuchen.
Ebenso wäre es sinnvoll, unterschiedliche Nutzerentscheidungsmodelle in die Simulation zu integrieren, um die Wirkung von Sensibilisierungsmaßnahmen gegenüber SE-Angriffen zu untersuchen.
Darüber hinaus könnten zukünftige Untersuchungen auf der hier entwickelten selbstreplizierenden Programmlogik aufbauen und diese in Netzwerksimulatoren wie *ns-3*[10] integrieren. Dadurch könnte erforscht werden, wie sich gleichartige Verbreitungsmechanismen unter variierenden Netzwerkbedingungen verhalten.

[10] ns-3 3.46.1: https://www.nsnam.org/

Literaturverzeichnis

[1] C. D. Hylender, P. Langlois, A. Pinto, und S. Widup, „2025 Data Breach Investigations Report“, Verizon Business, 18, 2025. Zugegriffen: 8. Januar 2026. [Online]. Verfügbar unter: https://www.verizon.com/business/resources/Tcf3/reports/2025-dbir-data-breach-investigations-report.pdf

[2] S. Abraham und I. Chengalur-Smith, „An overview of social engineering malware: Trends, tactics, and implications“, *Technol. Soc.*, Bd. 32, Nr. 3, S. 183–196, Aug. 2010, doi: 10.1016/j.techsoc.2010.07.001.

[3] T. F. Stafford und R. Poston, „Online Security Threats and Computer User Intentions“, *Computer*, Bd. 43, Nr. 1, S. 58–64, Jan. 2010, doi: 10.1109/MC.2010.18.

[4] S. Cordey, „Cyber Influence Operations: An Overview and Comparative Analysis“, Center for Security Studies (CSS), ETH Zürich, Zurich, Okt. 2019. Zugegriffen: 29. November 2025. [Online]. Verfügbar unter: https://css.ethz.ch/content/dam/ethz/special-interest/gess/cis/center-for-securities-studies/pdfs/Cyber-Reports-2019-10-CyberInfluence.pdf

[5] T. Braun, „Geschichte und Entwicklung des Internets“, *Inform.-Spektrum*, Bd. 33, Nr. 2, S. 201–207, Apr. 2010, doi: 10.1007/s00287-010-0423-9.

[6] N. S.A., „Evolution-and-Impact-of-Malware-A-Comprehensive-Analysis-from-the-First-Known-Malware-to-Modern-Day Cyber Threats“, SLIIT, Malabe, Sri Lanka, Aug. 2024. Zugegriffen: 29. November 2025. [Online]. Verfügbar unter: https://www.researchgate.net/profile/Amith-Senewirathna/publication/383177164_Evolution_and_Impact_of_Malware_A_Comprehensive_Analysis_from_the_First_Known_Malware_to_Modern-Day_Cyber_Threats/links/66bf90282ff54d6c9ed752c9/Evolution-and-Impact-of-Malware-A-Comprehensive-Analysis-from-the-First-Known-Malware-to-Modern-Day-Cyber-Threats.pdf

[7] P. Ganguli, „The Rise of Cybercrime-as-a-Service: Implications and Countermeasures“, *Int. J. Multidiscip. Res.*, Bd. 6, Nr. 5, Okt. 2024.

[8] P. K. Kerr, J. Rollins, und C. A. Theohary, „The Stuxnet Computer Worm: Harbinger of an Emerging Warfare Capability“, *CRS Rep. Congr.*, Dez. 2010.

[9] M. Hansel, „Stuxnet und die Sabotage des iranischen Atomprogramms: Ein neuer Kriegsschauplatz im Cyberspace?“, in *Handbuch Kriegstheorien*, T. Jäger und R. Beckmann, Hrsg., Wiesbaden: VS Verlag für Sozialwissenschaften, 2011, S. 564–576. doi: 10.1007/978-3-531-93299-6_47.

[10] R. Thummala und G. Falco, „Hacktivism Goes Orbital: Investigating NB65’s Breach of ROSCOSMOS“, in *AIAA SCITECH 2024 Forum*, Jan. 2024. doi: 10.2514/6.2024-0268.

[11] NB65 [@xxNB65], „We won’t stop until you stop. https://t.co/Cy1kiAN0bc“, Twitter. Zugegriffen: 1. Dezember 2025. [Online]. Verfügbar unter: https://x.com/xxNB65/status/1498563301525102594

[12] „Reports zu Schadprogramm-Infektionen“, Bundesamt für Sicherheit in der Informationstechnik. Zugegriffen: 29. November 2025. [Online]. Verfügbar unter: https://www.bsi.bund.de/DE/Themen/Unternehmen-und-Organisationen/Cyber-Sicherheitslage/Reaktion/CERT-Bund/CERT-Bund-Reports/Schadprogramm-Infektionen/schadprogramm-infektionen.html?nn=129542

[13] „Neue Schadsoftware-Varianten“, BSI Lagebericht 2025. Zugegriffen: 16. Dezember 2025. [Online]. Verfügbar unter: https://medien.bsi.bund.de/lagebericht/de/neue-schadsoftware-varianten/

[14] „Virus ▶ Rechtschreibung, Bedeutung, Definition, Herkunft ▶ Duden“. Zugegriffen: 6. November 2025. [Online]. Verfügbar unter: https://www.duden.de/rechtschreibung/Virus

[15] F. Cohen, „Computer viruses: Theory and experiments“, *Comput. Secur.*, Bd. 6, Nr. 1, S. 22–35, Feb. 1987, doi: 10.1016/0167-4048(87)90122-2.

[16] D. M. Chess und S. R. White, „An Undetectable Computer Virus“. IBM Thomas J. Watson Research Center, New York, USA, 2000. Zugegriffen: 6. November 2025. [Online]. Verfügbar unter: https://web.mit.edu/6.857/OldStuff/Fall03/ref/ChessWhite-AnUndetectableComputerVirus.pdf

[17] P. Szor, *The Art of Computer - Virus Research and Defense*. U.S.: Symantec Press, 2005.

[18] P. Elmer-Dewitt und R. H. Munro/Lahore, „You Must Be Punished“, TIME U.S. Zugegriffen: 7. November 2025. [Online]. Verfügbar unter: https://web.archive.org/web/20081214072232/http://www.time.com/time/magazine/article/0,9171,968490,00.html
[19] N. Milošević, „History of malware“. 16. Januar 2014. Zugegriffen: 29. November 2025. [Online]. Verfügbar unter: https://arxiv.org/pdf/1302.5392
[20] J. Hruska, „Virus detection“, in *European Conference on Security and Detection, 1997. ECOS 97.*, Apr. 1997, S. 128–130. doi: 10.1049/cp:19970437.
[21] A. S. Bist, „Detection of metamorphic viruses: A survey“, in *2014 International Conference on Advances in Computing, Communications and Informatics (ICACCI)*, Sep. 2014, S. 1559–1565. doi: 10.1109/ICACCI.2014.6968246.
[22] E. Konstantinou, „Metamorphic Virus: Analysis and Detection“, Royal Holloway University of London, England, Jan. 2008.
[23] E. H. Spafford, „The internet worm program: an analysis“, *ACM SIGCOMM Comput. Commun. Rev.*, Bd. 19, Nr. 1, S. 17–57, Jan. 1989, doi: 10.1145/66093.66095.
[24] F. B. Cohen, „A formal definition of computer worms and some related results“, *Comput. Secur.*, Bd. 11, Nr. 7, S. 641–652, Nov. 1992, doi: 10.1016/0167-4048(92)90144-G.
[25] D. Ellis, „Worm anatomy and model“, in *Proceedings of the 2003 ACM workshop on Rapid malcode*, Washington DC USA: ACM, Okt. 2003, S. 42–50. doi: 10.1145/948187.948196.
[26] „An Analysis of Conficker C“. Zugegriffen: 5. Dezember 2025. [Online]. Verfügbar unter: https://www.csl.sri.com/users/vinod/papers/Conficker/addendumC/index.html
[27] N. Falliere, L. O. Murchu, und E. Chien, „W32.Stuxnet Dossier“, Symantec, 1.4, Feb. 2011. Zugegriffen: 5. Dezember 2025. [Online]. Verfügbar unter: https://docs.broadcom.com/doc/security-response-w32-stuxnet-dossier-11-en
[28] S. U. M. Kamal, R. J. A. Ali, H. K. Alani, und E. S. Abdulmajed, „SURVEY AND BRIEF HISTORY ON MALWARE IN NETWORK SECURITY CASE STUDY: VIRUSES, WORMS AND BOTS“, Bd. 11, Nr. 1, 2016.
[29] „The Morris Worm“, Federal Bureau of Investigation. Zugegriffen: 7. Dezember 2025. [Online]. Verfügbar unter: https://www.fbi.gov/news/stories/morris-worm-30-years-since-first-major-attack-on-internet-110218
[30] T. Jowitt, „Tales In Tech History: ‚I Love You‘ Virus“, Silicon UK. Zugegriffen: 17. Dezember 2025. [Online]. Verfügbar unter: https://www.silicon.co.uk/security/cyberwar/tales-tech-history-love-bug-213685
[31] K. A. Rhodes, „Information Security: ‚I LOVE YOU‘ Computer Virus Emphasizes Critical Need for Agency and Governmentwide Improvements“. United States General Accounting Office, USA, 10. Mai 2000. Zugegriffen: 7. Dezember 2025. [Online]. Verfügbar unter: https://apps.dtic.mil/sti/citations/ADA376923
[32] D. Reinle (döp), „04. Mai 2010 - Vor 10 Jahren: Computer-Virus ‚I love you‘ verbreitet sich“. Zugegriffen: 7. Dezember 2025. [Online]. Verfügbar unter: https://www1.wdr.de/stichtag/stichtag4586.html
[33] J. Griffiths, „How a badly-coded computer virus caused billions in damage | CNN Business“, CNN. Zugegriffen: 7. Dezember 2025. [Online]. Verfügbar unter: https://www.cnn.com/2020/05/01/tech/iloveyou-virus-computer-security-intl-hnk
[34] K. Schmeh, *Das trojanische Pferd: klassische Mythen erklärt*. Haufe-Lexware, 2007.
[35] A. Security, „AIDS Trojan Analysis: Early Malware Case Study“, Arsen Security. Zugegriffen: 15. Dezember 2025. [Online]. Verfügbar unter: https://arsen.co/en/blog/aids-trojan
[36] S. M. Kelly, „The bizarre story of the inventor of ransomware | CNN Business“, CNN. Zugegriffen: 16. Dezember 2025. [Online]. Verfügbar unter: https://www.cnn.com/2021/05/16/tech/ransomware-joseph-popp
[37] „AIDS Trojan (PC Cyborg Virus)“. Zugegriffen: 15. Dezember 2025. [Online]. Verfügbar unter: https://www.antivirusaz.com/security-center/virus-information/aids-trojan.html
[38] „ZeuS Banking Trojan Report“, Secureworks. Zugegriffen: 15. Dezember 2025. [Online]. Verfügbar unter: https://www.secureworks.com/research/zeus

[39] „New phishing campaign against Facebook led by Zeus“. Zugegriffen: 15. Dezember 2025. [Online]. Verfügbar unter: http://malwareint.blogspot.com/2010/03/new-phishing-campaign-against-facebook.html
[40] „What Is Zeus Trojan? - Zbot Malware Defined | Proofpoint US“, Proofpoint. Zugegriffen: 15. Dezember 2025. [Online]. Verfügbar unter: https://www.proofpoint.com/us/threat-reference/zeus-trojan-zbot
[41] S. Musil, „US disrupts $100M GameOver Zeus malware cybercrime ring“, CNET - Your Guide To A Better Future. Zugegriffen: 15. Dezember 2025. [Online]. Verfügbar unter: https://www.cnet.com/news/privacy/us-disrupts-100m-gameover-zeus-malware-cybercrime-ring/
[42] „The Evolution of Emotet: From Banking Trojan to Threat Distributor“. Zugegriffen: 16. Dezember 2025. [Online]. Verfügbar unter: https://www.security.com/threat-intelligence/evolution-emotet-trojan-distributor
[43] European Union Agency for Law Enforcement Cooperation, „Internet Organised Crime Threat Assessment (IOCTA) 2020“, Europol Europa. Zugegriffen: 16. Dezember 2025. [Online]. Verfügbar unter: https://www.europol.europa.eu/cms/sites/default/files/documents/internet_organised_crime_threat_assessment_iocta_2020.pdf
[44] „Emotet-Malware – Check Point-Software“, Check Point Software. Zugegriffen: 16. Dezember 2025. [Online]. Verfügbar unter: https://www.checkpoint.com/cyber-hub/threat-prevention/what-is-malware/emotet-malware/
[45] C. Patsakis und A. Chrysanthou, „Analysing the fall 2020 Emotet campaign“, 12. November 2020, *arXiv*: arXiv:2011.06479. doi: 10.48550/arXiv.2011.06479.
[46] European Union Agency for Cybersecurity., *ENISA sectorial threat landscape: public administration.* LU: Publications Office, 2025. Zugegriffen: 17. Dezember 2025. [Online]. Verfügbar unter: https://data.europa.eu/doi/10.2824/4606183
[47] ISACA, „State of Cybersecurity 2022“, ISACA, März 2022. Zugegriffen: 17. Dezember 2025. [Online]. Verfügbar unter: https://www.isaca.org/resources/reports/state-of-cybersecurity-2022?tfa_next=%2Fresponses%2Flast_success%3Fjsid%3DeyJ0eXAiOiJKV1QiLCJhbGciOiJIUzI1NiJ9.WyIzMTFhMzcxMjNmNjc5ZmNhYzBhNGMxNTk5MzAwOWYyYiJd.1cb2WzTdyPK32SRxf5xjgJ-g7IKpHYKLH2WUEnSUU6I
[48] M. Schmitt und I. Flechais, „Digital deception: generative artificial intelligence in social engineering and phishing“, *Artif. Intell. Rev.*, Bd. 57, Nr. 12, S. 324, Okt. 2024, doi: 10.1007/s10462-024-10973-2.
[49] C. Hadnagy, *Social Engineering - the Art of Human Hacking*. Indianapolis, Indiana: Wiley Publishing Inc., 2010. Zugegriffen: 17. Dezember 2025. [Online]. Verfügbar unter: https://ia801200.us.archive.org/33/items/TheAgeOfManipulationWilsonBryanKey/Social%20Engineering%20-%20the%20Art%20of%20Human%20Hacking.pdf
[50] „What is Social Engineering? | IBM“. Zugegriffen: 17. Dezember 2025. [Online]. Verfügbar unter: https://www.ibm.com/think/topics/social-engineering
[51] B. Coatesworth, „The psychology of social engineering“, *Cyber Secur. Peer-Rev. J.*, Bd. 6, Nr. 3, S. 261, März 2023, doi: 10.69554/AKTG1392.
[52] „The History of Social Engineering“. Zugegriffen: 21. Dezember 2025. [Online]. Verfügbar unter: https://www.mitnicksecurity.com/the-history-of-social-engineering
[53] K. D. Mitnick und W. L. Simon, *THE ART OF DECEPTION*, 1. Auflage. 2002. Zugegriffen: 21. Dezember 2025. [Online]. Verfügbar unter: https://ia600102.us.archive.org/4/items/pdfsandebooks/UP/Life%20Skills/Teach%20Yourself%20-%20Body%20Language%20%20Ebooks%20-%20Mantesh/The%20Art%20of%20Deception%20-%20Kevin%20Mitnick.pdf
[54] A. Wagner, T. Dübendorfer, B. Plattner, und R. Hiestand, „Experiences with worm propagation simulations“, in *Proceedings of the 2003 ACM workshop on Rapid malcode*, in WORM '03. New York, NY, USA: Association for Computing Machinery, Okt. 2003, S. 34–41. doi: 10.1145/948187.948194.

[55] M. Wiemken, K. Hildebrandt, A. Jeworutzki, und L. Putzar, „Emotional Manipulation in Phishing Emails: Experimental Study of Affective Responses and Human Classification Errors in a Simulated Email Environment“, in *Proceedings of the 18th ACM International Conference on PErvasive Technologies Related to Assistive Environments*, in PETRA '25. New York, NY, USA: Association for Computing Machinery, Juli 2025, S. 583–589. doi: 10.1145/3733155.3736796.
[56] O. Borchert, G. Howell, A. Kerman, S. Rose, und M. Souppaya, „Implementing a zero trust architecture : high-level document“, National Institute of Standards and Technology (U.S.), Gaithersburg, MD, NIST SP 1800-35, Juni 2025. doi: 10.6028/NIST.SP.1800-35.
[57] J. Garbis und J. W. Chapman, *Zero Trust Sicherheit: Ein Leitfaden für Unternehmen*. Berkeley, CA: Apress, 2024. doi: 10.1007/979-8-8688-0105-1.
[58] Homeland Security, „Application Whitelisting (AWL): Strategic Planning Guide“, Federal Network Resilience. Zugegriffen: 20. Januar 2026. [Online]. Verfügbar unter: https://www.cisa.gov/sites/default/files/cdm_files/FNR_NIS_OTH_AWL_Strategic_Planning_Guide.pdf
[59] H. Pareek, „Application Whitelisting: Approaches and Challenges“, *Int. J. Comput. Sci. Eng. Inf. Technol.*, Bd. 2, Nr. 5, S. 13–18, Okt. 2012, doi: 10.5121/ijcseit.2012.2502.
[60] C. Gates, N. Li, J. Chen, und R. Proctor, „CodeShield | Proceedings of the 28th Annual Computer Security Applications Conference“, ACM Other conferences. Zugegriffen: 20. Januar 2026. [Online]. Verfügbar unter: https://dl.acm.org/doi/10.1145/2420950.2420992
[61] LogRythm Labs, „NOTPETYA TECHNICAL ANALYSIS“, LogRythm - The Security Intelligence Company, Juli 2017. Zugegriffen: 14. Januar 2026. [Online]. Verfügbar unter: https://gallery.logrhythm.com/threat-intelligence-reports/notpetya-technical-analysis-logrhythm-labs-threat-intelligence-report.pdf
[62] E. Cole, *Network Security Bible*. John Wiley & Sons, 2011. Zugegriffen: 23. Januar 2026. [Online]. Verfügbar unter: https://books.google.de/books?id=Iq8lPbhGRuYC
[63] A. Sadiqui, „Computer Network Security“, ISTE Ltd, John Wiley & Sons Inc., England, 2020. Zugegriffen: 23. Januar 2026. [Online]. Verfügbar unter: https://edu.anarcho-copy.org/TCP%20IP%20-%20Network/Network%20Security.pdf
[64] FortiGuard Labs, „2025 Fortinet Global Threat Landscape Report“, Fortinet, Mai 2025. Zugegriffen: 15. Januar 2026. [Online]. Verfügbar unter: https://www.fortinet.com/de/resources/reports/threat-landscape-report
[65] S. Rose, J. S. Nightingale, S. Garfinkel, und R. Chandramouli, „Trustworthy email“, National Institute of Standards and Technology, Gaithersburg, MD, NIST SP 800-177r1, Feb. 2019. doi: 10.6028/NIST.SP.800-177r1.
[66] Microsoft, „Email authentication - Microsoft Defender for Office 365“, Microsoft Defender. Zugegriffen: 26. Januar 2026. [Online]. Verfügbar unter: https://learn.microsoft.com/en-us/defender-office-365/email-authentication-about
[67] „Basismaßnahmen der Cyber-Sicherheit“, BSI, Deutschland, Juli 2018. Zugegriffen: 26. Januar 2025. [Online]. Verfügbar unter: https://www.allianz-fuer-cybersicherheit.de/SharedDocs/Downloads/Webs/ACS/DE/BSI-CS/BSI-CS_006.pdf?__blob=publicationFile&v=1
[68] V. Thomas, „Social Engineering“, in *Building an Information Security Awareness Program*, USA, 2014. doi: 10.1016/B978-0-12-419967-5.00007-7.
[69] *Directive (EU) 2022/2555 of the European Parliament and of the Council of 14 December 2022 on measures for a high common level of cybersecurity across the Union, amending Regulation (EU) No 910/2014 and Directive (EU) 2018/1972, and repealing Directive (EU) 2016/1148 (NIS 2 Directive) (Text with EEA relevance)*, Bd. 333. 2022. Zugegriffen: 27. Januar 2026. [Online]. Verfügbar unter: http://data.europa.eu/eli/dir/2022/2555/oj
[70] „BSIG - Gesetz über das Bundesamt für Sicherheit in der Informationstechnik und über die Sicherheit in der Informationstechnik von Einrichtungen“. Zugegriffen: 27. Januar 2026. [Online]. Verfügbar unter: https://www.gesetze-im-internet.de/bsig_2025/BJNR12D0B0025.html

[71] Y. Huang *u. a.*, „Towards combining chain-of-thought and code static analysis for buffer overflow vulnerability detection“, *Softw. Qual. J.*, Bd. 34, Nr. 1, S. 2, Dez. 2025, doi: 10.1007/s11219-025-09733-4.
[72] „2025 Cloudflare Signals Report“, 2025, Zugegriffen: 24. Januar 2026. [Online]. Verfügbar unter: https://www.cloudflare.com/lp/signals-report-2025/
[73] N. Mohamed, „Artificial intelligence and machine learning in cybersecurity: a deep dive into state-of-the-art techniques and future paradigms“, *Knowl. Inf. Syst.*, Bd. 67, Nr. 8, S. 6969–7055, Aug. 2025, doi: 10.1007/s10115-025-02429-y.
[74] R. M. Rodriguez, E. Golob, und S. Xu, „Human Cognition through the Lens of Social Engineering Cyberattacks“, 10. Juli 2020, *arXiv*: arXiv:2007.04932. doi: 10.48550/arXiv.2007.04932.
[75] A. Ali und M. C. Ghanem, „Beyond Detection: Large Language Models and Next-Generation Cybersecurity“, *SHIFRA*, Bd. 2025, S. 81–97, Feb. 2025, doi: 10.70470/SHIFRA/2025/005.
[76] I. K. Dutta, B. Ghosh, A. Carlson, M. Totaro, und M. Bayoumi, „Generative Adversarial Networks in Security: A Survey“, in *2020 11th IEEE Annual Ubiquitous Computing, Electronics & Mobile Communication Conference (UEMCON)*, Okt. 2020, S. 0399–0405. doi: 10.1109/UEMCON51285.2020.9298135.
[77] W. Huang *u. a.*, „EvoMail: Self-Evolving Cognitive Agents for Adaptive Spam and Phishing Email Defense“, 25. September 2025, *arXiv*: arXiv:2509.21129. doi: 10.48550/arXiv.2509.21129.
[78] Official Journal of the European Union, *Regulation (EU) 2024/1689 of the European Parliament and of the Council of 13 June 2024 laying down harmonised rules on artificial intelligence and amending Regulations (EC) No 300/2008, (EU) No 167/2013, (EU) No 168/2013, (EU) 2018/858, (EU) 2018/1139 and (EU) 2019/2144 and Directives 2014/90/EU, (EU) 2016/797 and (EU) 2020/1828 (Artificial Intelligence Act) (Text with EEA relevance)*. 2024. Zugegriffen: 28. Januar 2026. [Online]. Verfügbar unter: http://data.europa.eu/eli/reg/2024/1689/oj

Anhang

A Quellcode SE Szenario

Codeblock 1: „send_malicious_mail.py“ und „Abrechnung4.Quartal25.py“
Quelle: eigene Darstellung

```
1   import smtplib
2   import os
3   from email.message import EmailMessage
4
5   SMTP_PORT = 25
6   MAIL_FROM = "sop@attacker.local"
7   CONTACTS = "contacts.txt"
8   INFECTED = "infected.txt"
9   ATTACHMENT = "Abrechnung4.Quartal25.py"
10  contacts=[]
11
12  #check ob infected.txt schon angelegt wurde
13  if os.path.exists(INFECTED):
14      print("OH OH DU BIST SCHON INFIZIERT!!")
15      exit(0)
16
17  #erstellen der infected.txt
18  with open(INFECTED, "w") as file:
19     file.write("infected")
20     print("SYSTEM IST INFIZIERT!")
21
22  #in contacts.txt nach Email-Addressen und IP-Addressen su-
    chen
23  with open(CONTACTS, "r") as filee:
24     for line in filee:
25         line = line.strip()
26         if not line or line.startswith("?"):
27             continue
28         parts = line.split(",")
29         if len(parts)!= 2:
30            print("Bad format")  #fuer Bugfixing
31         email = parts[0].strip()
32         ip = parts[1].strip()
```

```
33          contacts.append((email,ip))
34  with open(ATTACHMENT, "rb") as file2:
35      attachment = file2.read()
36
37  #Email-Inhalt und Anhang
38  for empfaenger, smtp_server in contacts:
39     msg = EmailMessage()
40     msg["FROM"] = MAIL_FROM
41     msg["To"] = empfaenger
42     msg["Subject"] = "Abrechnung WICHTIG!!"
43     msg.set_content("Guten Tag,"
44     „Sie erhalten hiermit die aktualisierte Abrechnung fuer
    letztes Quartal (4/25)."
45     „Bitte pruefen Sie die Angaben und geben Sie bis Mittwoch
    (04.02.2026) eine verpflichtende Rueckmeldung ab!"
46     „Vielen Dank!"
47     „"
48     „Mit freundlichen Gruessen"
49     „Finanzberatung GmbH"
50     )
51     msg.add_attachment(
52         attachment,
53         maintype = „application"
54         subtype = „octet-stream"
55         filename = ATTACHMENT
56    )
57  #Herstellen der SMTP Verbindung & Statusmeldung
58    try:
59        server    =    smtplib.SMTP(smtp_server,    SMTP_PORT,
    timeout=10)
60        server.set_debuglevel(1)
61        server.ehlo()
62        server.send_message(msg)
63        server.quit()
64        print("SENT TO NEXT VICTIM")
65   except Exception as e:
66        print("FAILED TO INFECT MORE HOSTS: {e}")
```

B SMTP Analyse des SE Szenarios

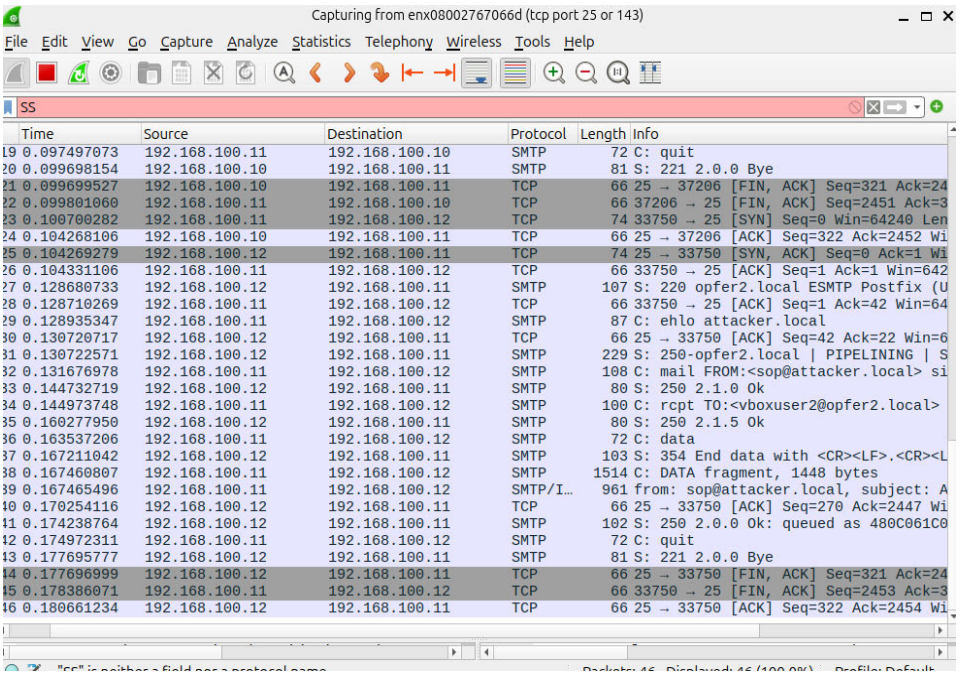


Abbildung 11: Versand der Mails vom Angreifer an die Opfer
Quelle: eigene Darstellung

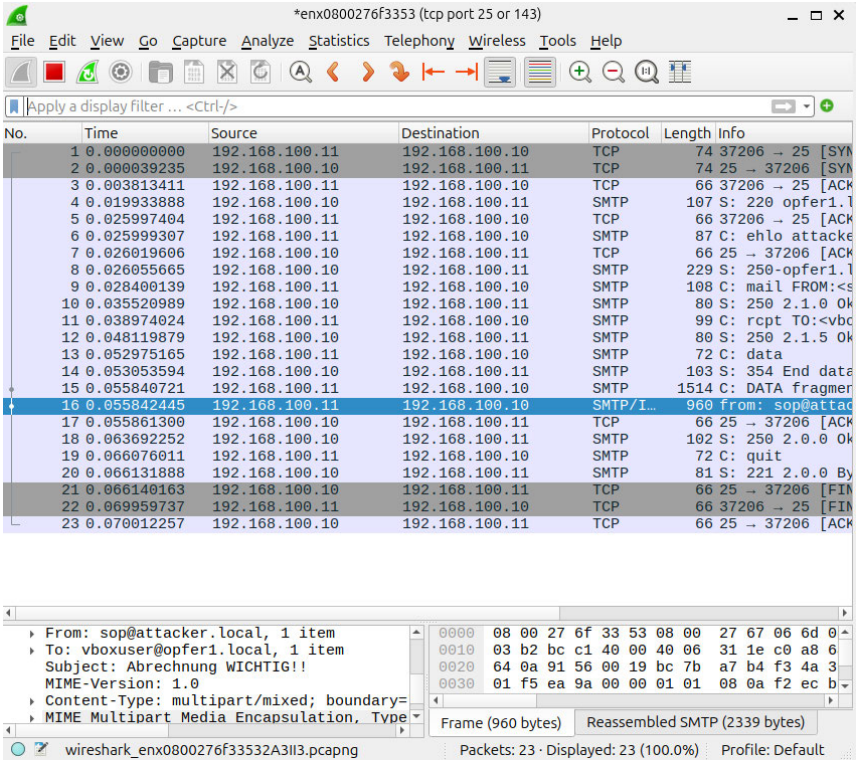


Abbildung 12: Ankunft der Mail mit Anhang bei Opfer 1
Quelle: eigene Darstellung

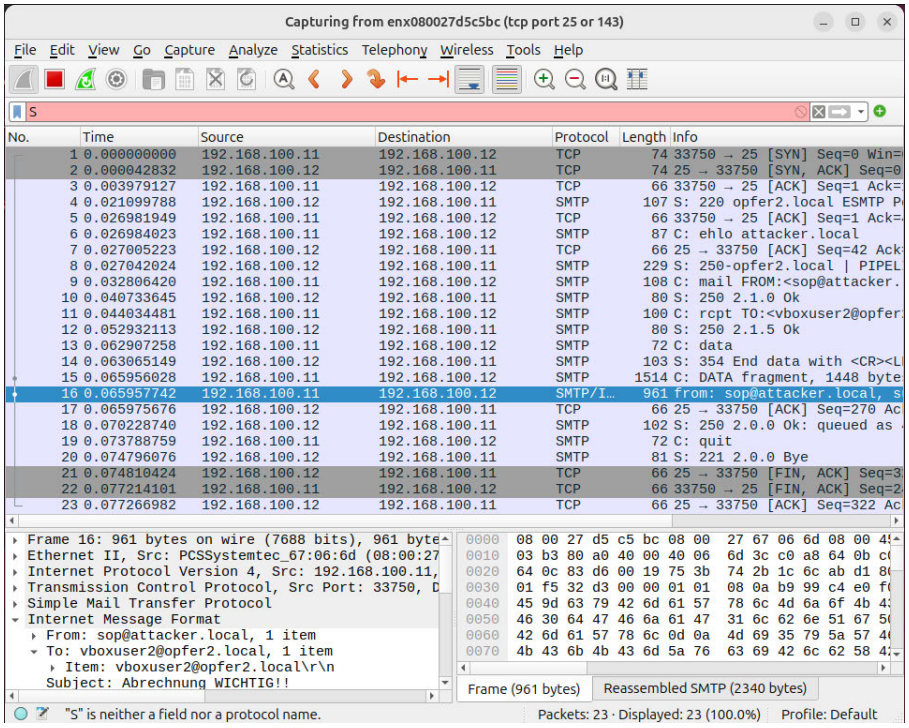


Abbildung 13: Ankunft der Mail mit Anhang bei Opfer 2
Quelle: eigene Darstellung

C Sonstige Dateien SE Szenario

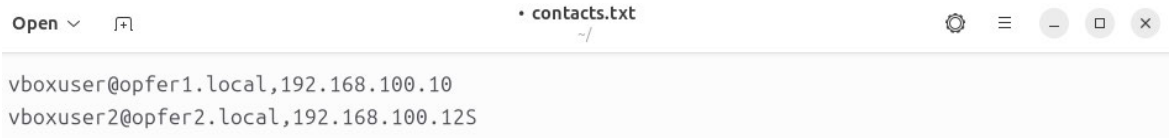


Abbildung 14: Kontaktliste „contacts.txt“
Quelle: eigene Darstellung

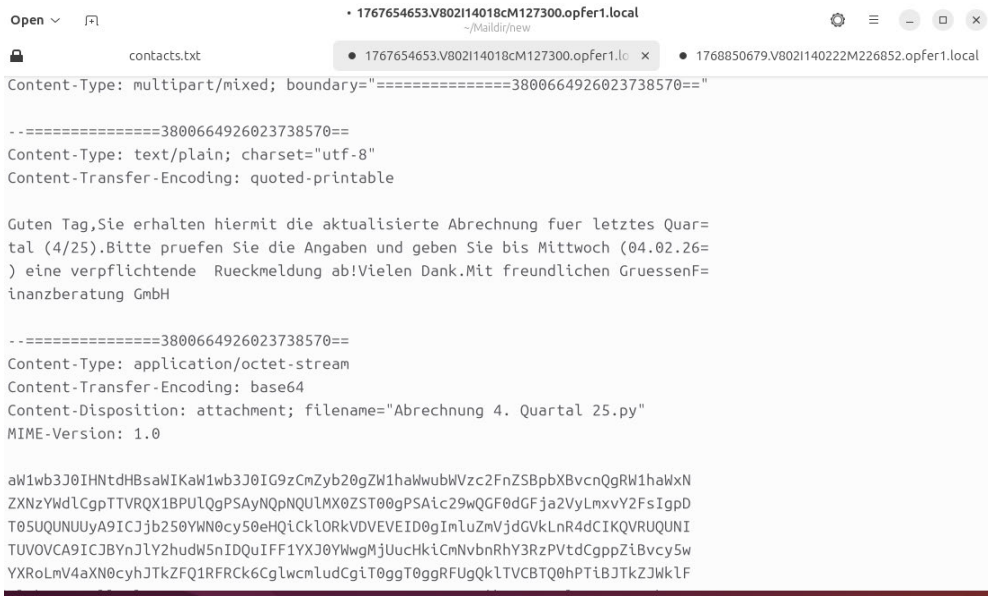


Abbildung 15: Anhang wurde auf Opfer-System 1 übertragen
Quelle: eigene Darstellung

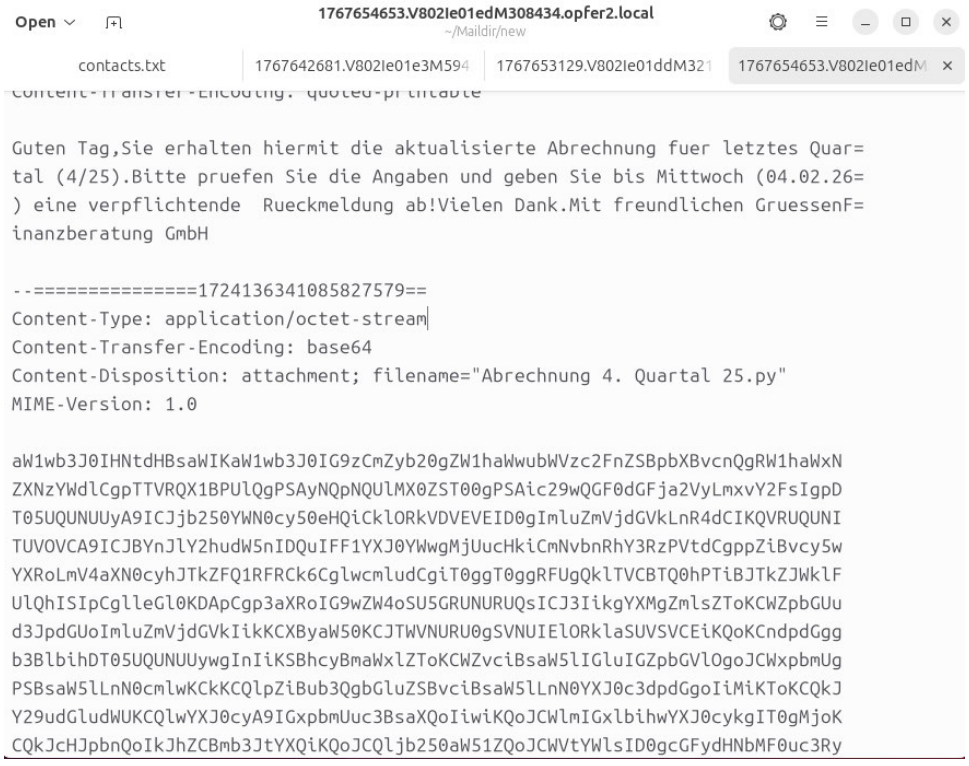
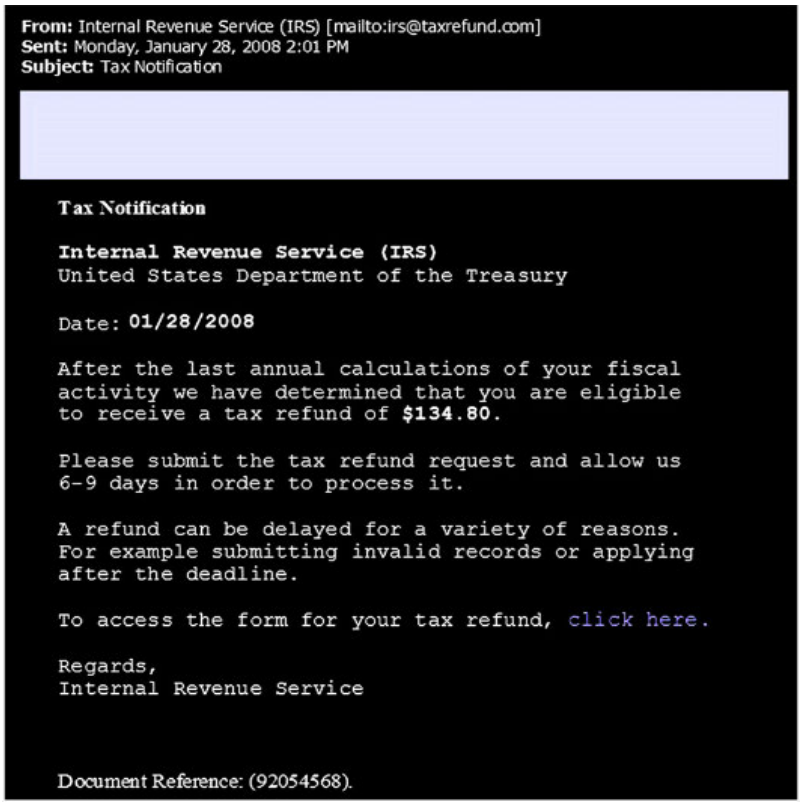


Abbildung 16: Anhang wurde auf Opfer-System 2 übertragen
Quelle: eigene Darstellung



From: Internal Revenue Service (IRS) [mailto:irs@taxrefund.com]
Sent: Monday, January 28, 2008 2:01 PM
Subject: Tax Notification

Tax Notification

Internal Revenue Service (IRS)
United States Department of the Treasury

Date: **01/28/2008**

After the last annual calculations of your fiscal activity we have determined that you are eligible to receive a tax refund of **$134.80**.

Please submit the tax refund request and allow us 6-9 days in order to process it.

A refund can be delayed for a variety of reasons. For example submitting invalid records or applying after the deadline.

To access the form for your tax refund, click here.

Regards,
Internal Revenue Service

Document Reference: (92054568).

Abbildung 17: Beispiel einer Scarcity-Phishing-E-Mail
Quelle: [2]

Eigenständigkeitserklärung

Hiermit versichere ich, dass ich die vorliegende Bachelorarbeit mit dem Titel:

_____

selbständig und nur mit den angegebenen Hilfsmitteln verfasst habe. Alle Passagen, die ich wörtlich aus der Literatur oder aus anderen Quellen wie z. B. Internetseiten übernommen habe, habe ich deutlich als Zitat mit Angabe der Quelle kenntlich gemacht.

Datum	[redacted]
	Unterschrift