Offensive AI

A fun and not (so) explored field mixing **offensive cybersecurity** and **machine learning**.

Dorian BACHELOT

About Me!

Dorian Bachelot

Research Engineer

@LHS/CentraleSupélec/INRIA

#Malware Developer/Detection

https://dorianb.net

CentraleSupélec LHS LABORATOIRE HAUTE SÉCURITÉ Inria UMR IRISA

Introduction

AI, ML, DL, OAI...

Field Overview

Field Overview

Definition

Offensive artificial intelligence (OAI) can be defined as the field in which artificial intelligence algorithms or models are used to **automate one or multiple parts of an offensive security task**.

Field Overview

Definition

Offensive artificial intelligence (OAI) can be defined as the field in which artificial intelligence algorithms or models are used to **automate one or multiple parts of an offensive security task**.

This cover:

- Vulnerability scanning
- Exploit automation
- Attack path identification
- ...

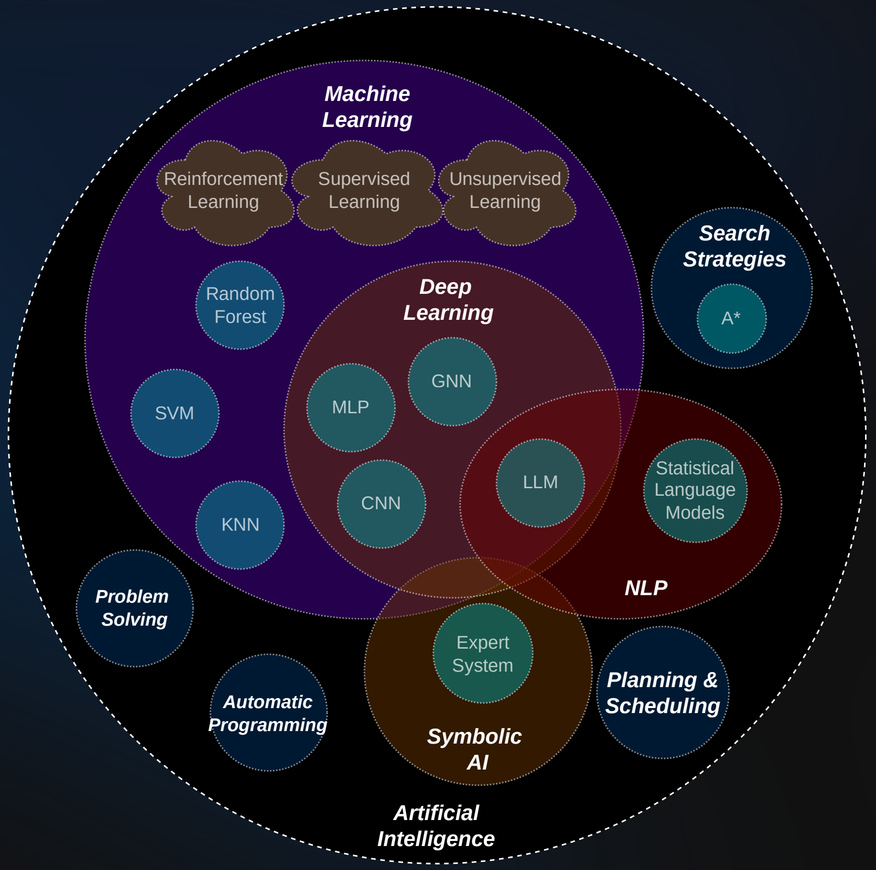
Reminder on AI

Reminder on AI

We can classify as AI **any system that solves a problem** or **achieve a goal** and have some degree of **adaptation and autonomy.**

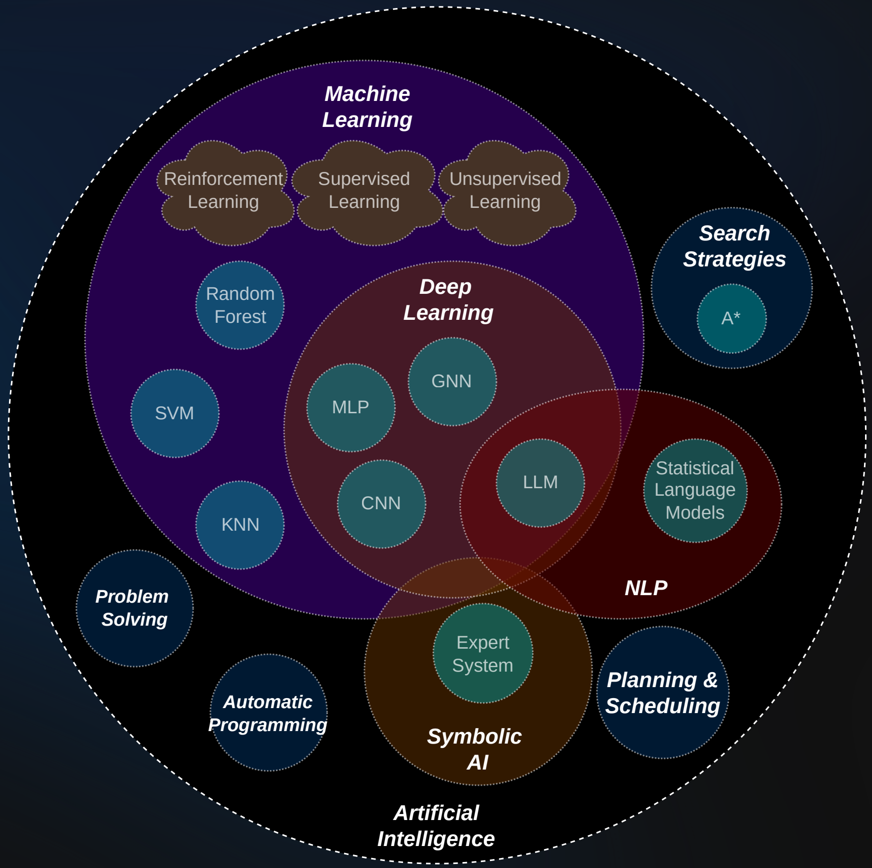
Reminder on AI

We can classify as AI **any system that solves a problem** or **achieve a goal** and have some degree of **adaptation and autonomy.**



Reminder on AI

We can classify as AI **any system that solves a problem** or **achieve a goal** and have some degree of **adaptation and autonomy.**



Machine Learning ≠ AI ≠ Deep Learning ≠ LLMs

OAI

Global overview

Overview

Overview

OAI can also be cut in **multiple subfields**, which correlate closely to some **use cases like pentest or red team automation.**

Overview

OAI can also be cut in **multiple subfields**, which correlate closely to some **use cases like pentest or red team automation.**

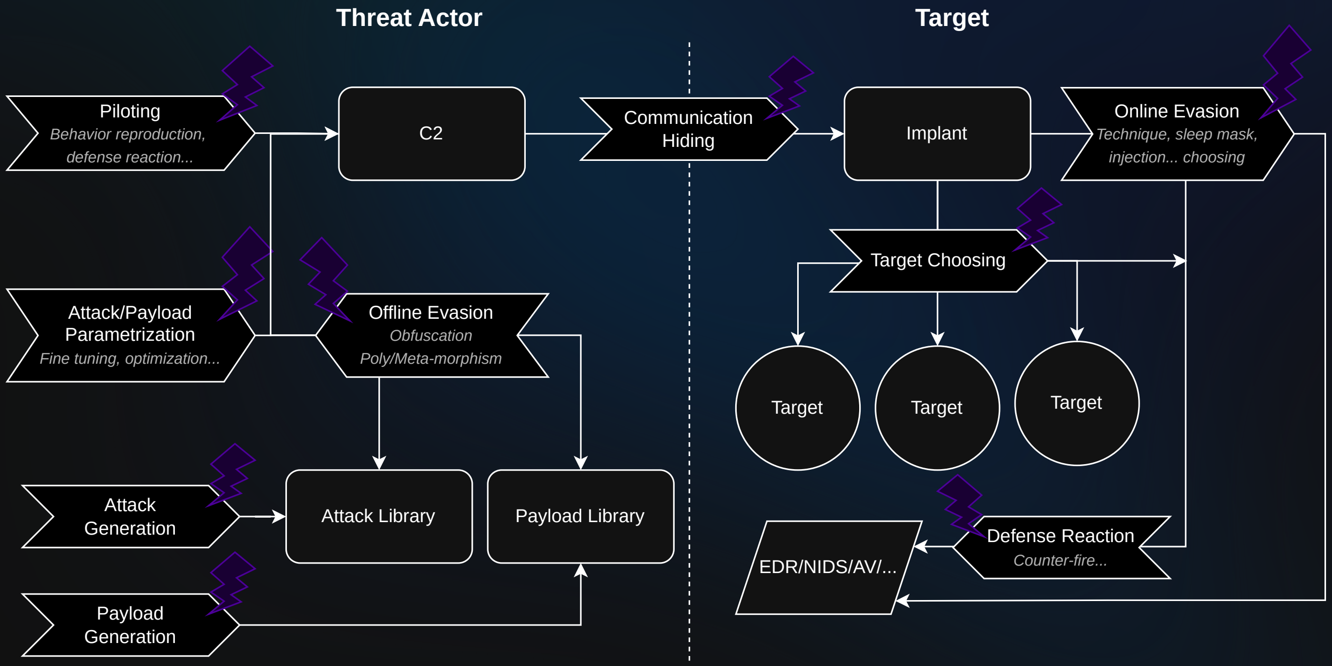
Overview

OAI can also be cut in **multiple subfields**, which correlate closely to some **use cases like pentest or red team automation.**

Automation ≠ Better than Humans

The different subfields

The different subfields



OAI for C2 Piloting

Target choosing, attack choosing...

C2 Piloting

C2 Piloting

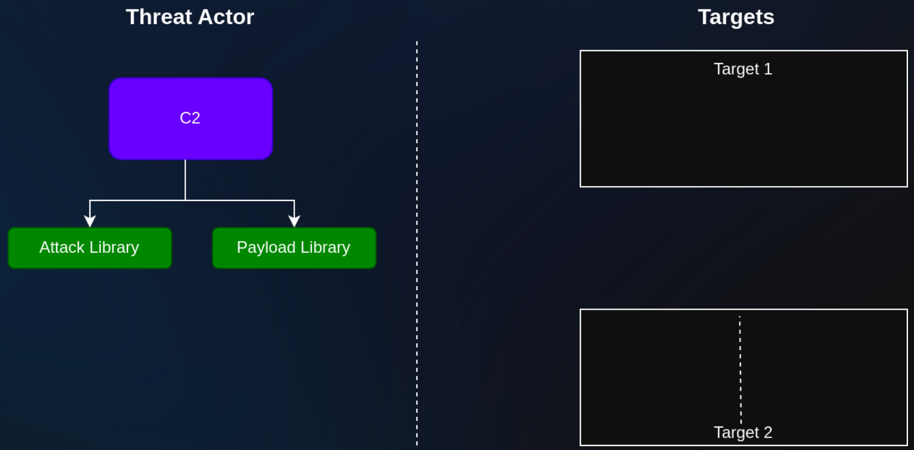
What is a C2?

- **Pilot implants**, that have infected some hosts.
- Implants can communicate directly with the C2, or can do it **through another implant**.
- Can issue commands, and **send attacks** (modules).
- Focus on **evasion**, hiding communication and using advanced implants.
- Used to **pivot** and **deeply infect** a network.
- Controlled by an operator (or **AI!**).

C2 Piloting

What is a C2?

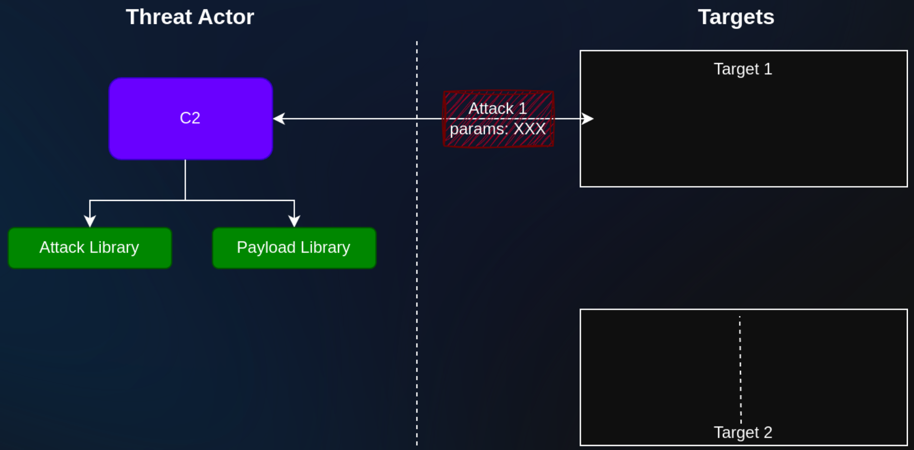
- **Pilot implants**, that have infected some hosts.
- Implants can communicate directly with the C2, or can do it **through another implant**.
- Can issue commands, and **send attacks** (modules).
- Focus on **evasion**, hiding communication and using advanced implants.
- Used to **pivot** and **deeply infect** a network.
- Controlled by an operator (or **AI!**).



C2 Piloting

What is a C2?

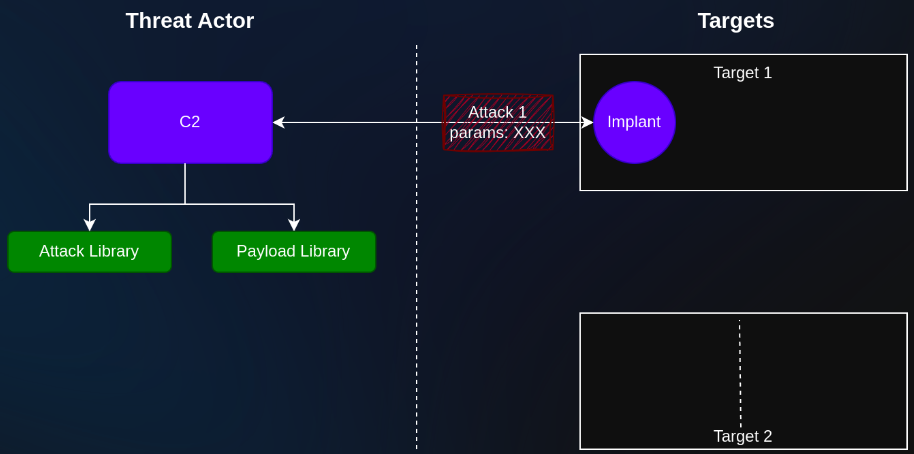
- **Pilot implants**, that have infected some hosts.
- Implants can communicate directly with the C2, or can do it **through another implant**.
- Can issue commands, and **send attacks** (modules).
- Focus on **evasion**, hiding communication and using advanced implants.
- Used to **pivot** and **deeply infect** a network.
- Controlled by an operator (or **AI!**).



C2 Piloting

What is a C2?

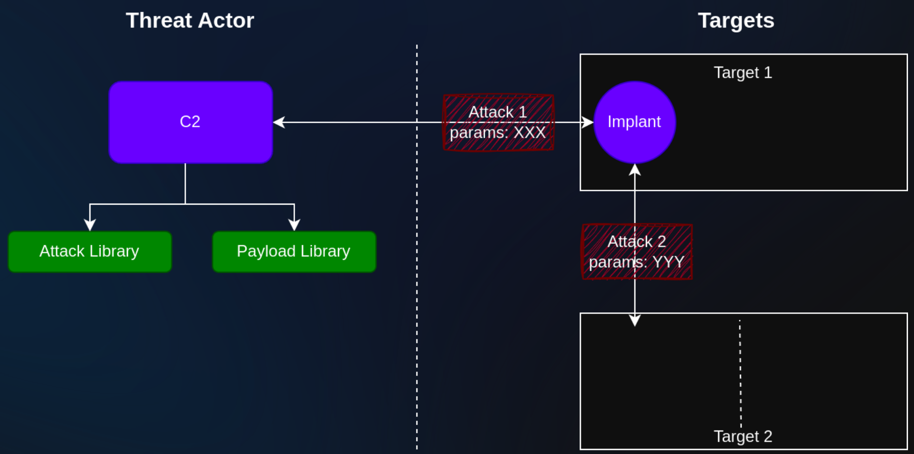
- **Pilot implants**, that have infected some hosts.
- Implants can communicate directly with the C2, or can do it **through another implant**.
- Can issue commands, and **send attacks** (modules).
- Focus on **evasion**, hiding communication and using advanced implants.
- Used to **pivot** and **deeply infect** a network.
- Controlled by an operator (or **AI!**).



C2 Piloting

What is a C2?

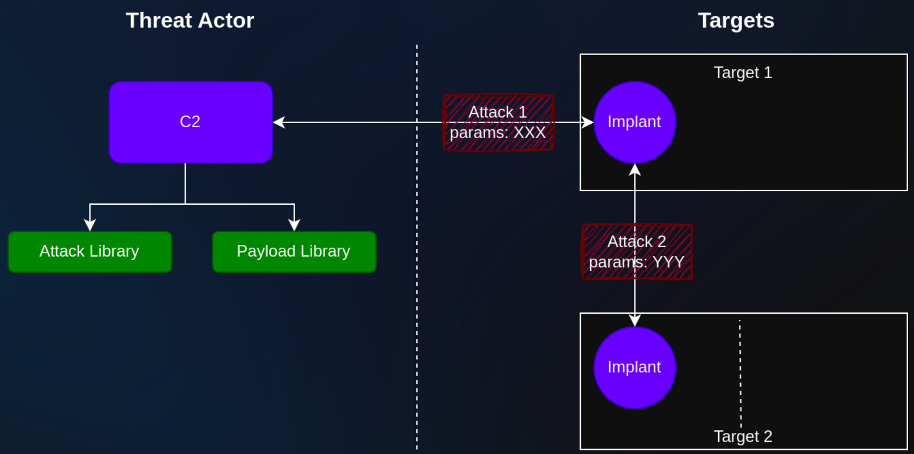
- **Pilot implants**, that have infected some hosts.
- Implants can communicate directly with the C2, or can do it **through another implant**.
- Can issue commands, and **send attacks** (modules).
- Focus on **evasion**, hiding communication and using advanced implants.
- Used to **pivot** and **deeply infect** a network.
- Controlled by an operator (or **AI!**).



C2 Piloting

What is a C2?

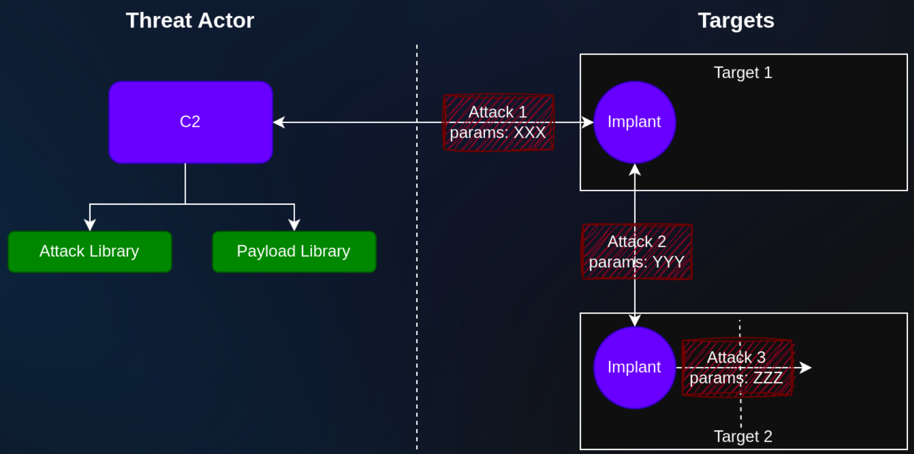
- **Pilot implants**, that have infected some hosts.
- Implants can communicate directly with the C2, or can do it **through another implant**.
- Can issue commands, and **send attacks** (modules).
- Focus on **evasion**, hiding communication and using advanced implants.
- Used to **pivot** and **deeply infect** a network.
- Controlled by an operator (or **AI!**).



C2 Piloting

What is a C2?

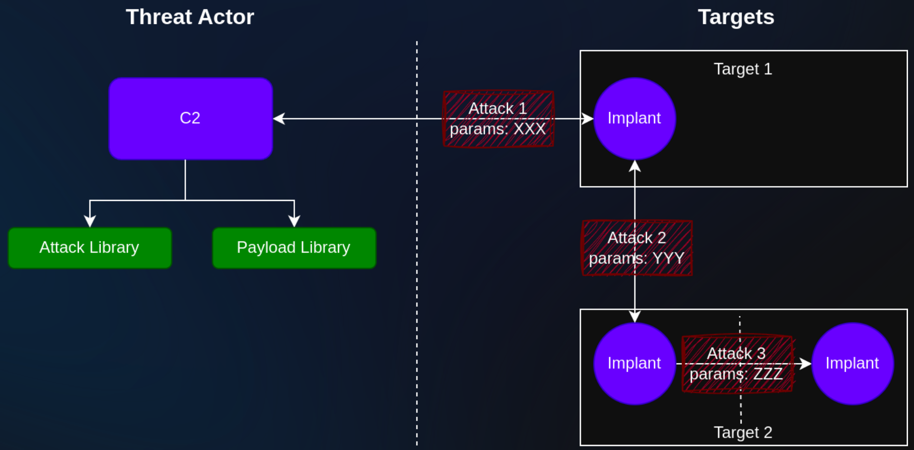
- **Pilot implants**, that have infected some hosts.
- Implants can communicate directly with the C2, or can do it **through another implant**.
- Can issue commands, and **send attacks** (modules).
- Focus on **evasion**, hiding communication and using advanced implants.
- Used to **pivot** and **deeply infect** a network.
- Controlled by an operator (or **AI!**).



C2 Piloting

What is a C2?

- **Pilot implants**, that have infected some hosts.
- Implants can communicate directly with the C2, or can do it **through another implant**.
- Can issue commands, and **send attacks** (modules).
- Focus on **evasion**, hiding communication and using advanced implants.
- Used to **pivot** and **deeply infect** a network.
- Controlled by an operator (or **AI!**).



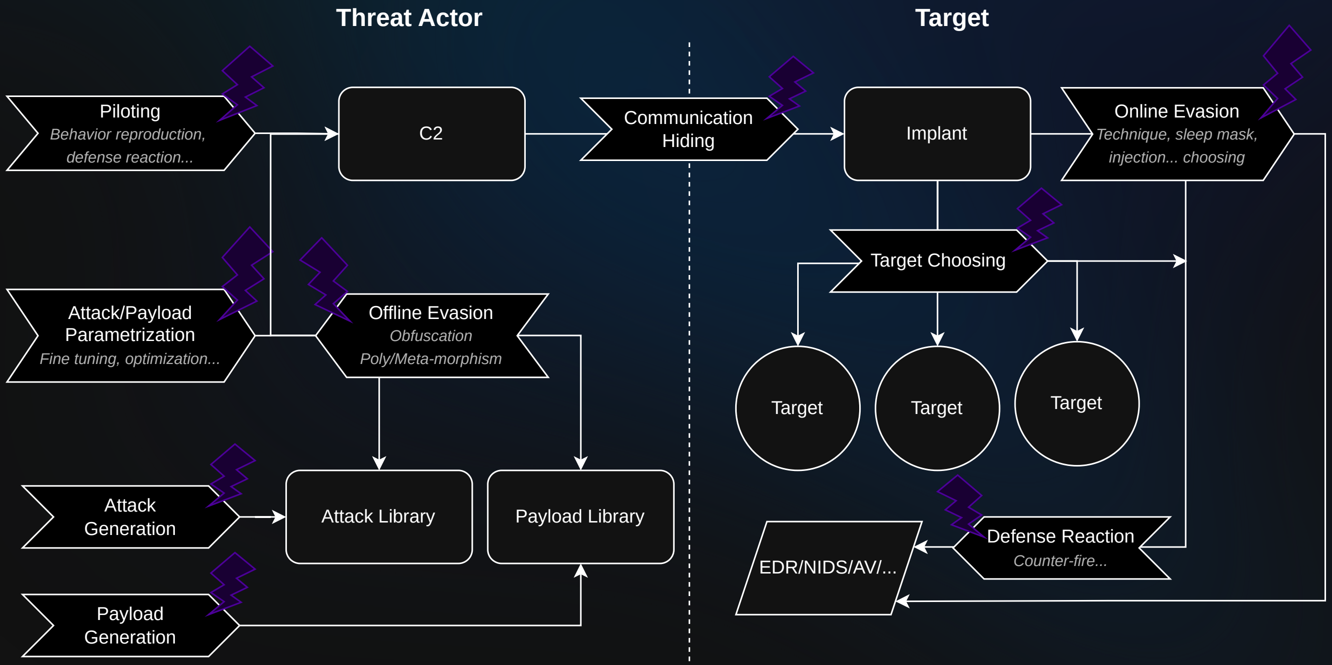
C2 Piloting / The Idea

C2 Piloting / The Idea

Consists of selecting a **target**, an **attack** and **parameters** using a model/algorithm.

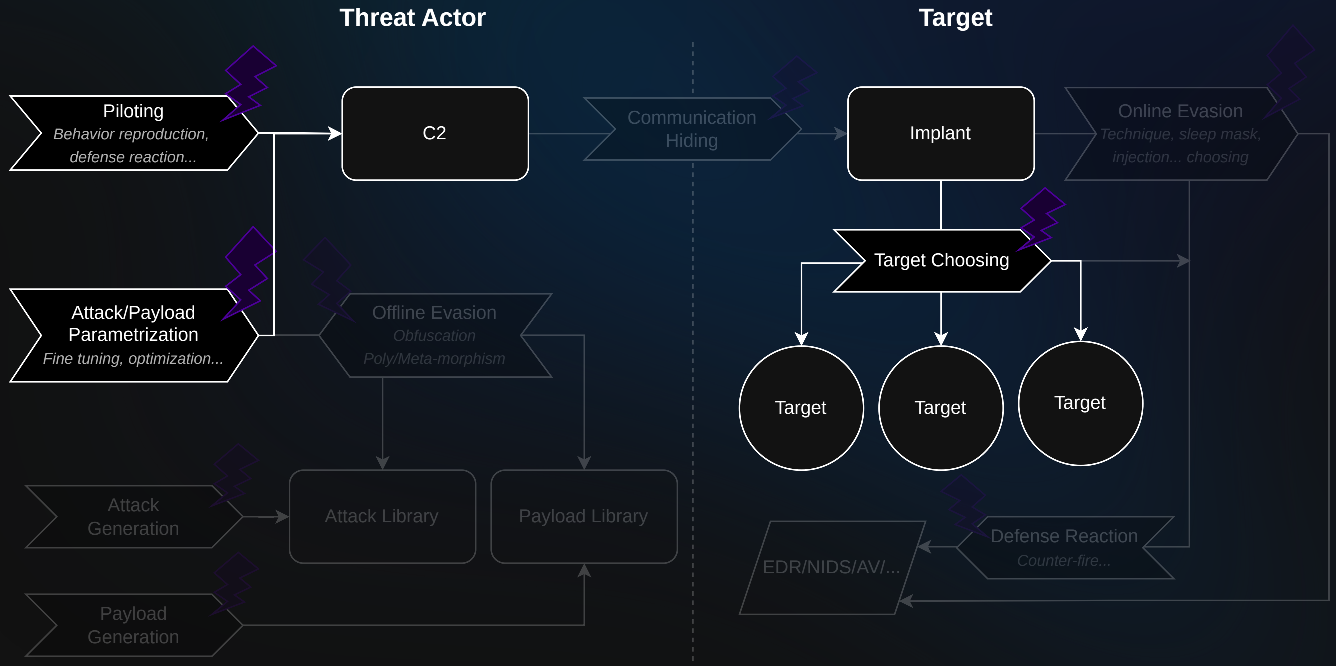
C2 Piloting / The Idea

Consists of selecting a **target**, an **attack** and **parameters** using a model/algorithm.



C2 Piloting / The Idea

Consists of selecting a **target**, an **attack** and **parameters** using a model/algorithm.



C2 Piloting / What's Needed?

C2 Piloting / What's Needed?

Recipe:

- Take a **C2 with an API** (giving automation power on ALL operations).
- Prepare an **attack library** (modules), with exposed parameters.
- Add some model/algorithm.
- Shake. Done!

C2 Piloting / What's Needed?

Recipe:

- Take a **C2 with an API** (giving automation power on ALL operations).
- Prepare an **attack library** (modules), with exposed parameters.
- Add some model/algorithm.
- Shake. Done!

(@chrisrohlf)

C2 Piloting / What's Needed?

Recipe:

- Take a **C2 with an API** (giving automation power on ALL operations).
- Prepare an **attack library** (modules), with exposed parameters.
- Add some model/algorithm.
- Shake. Done!

(@chrisrohlf)

Not that simple in reality... A lot of AI-related issues.

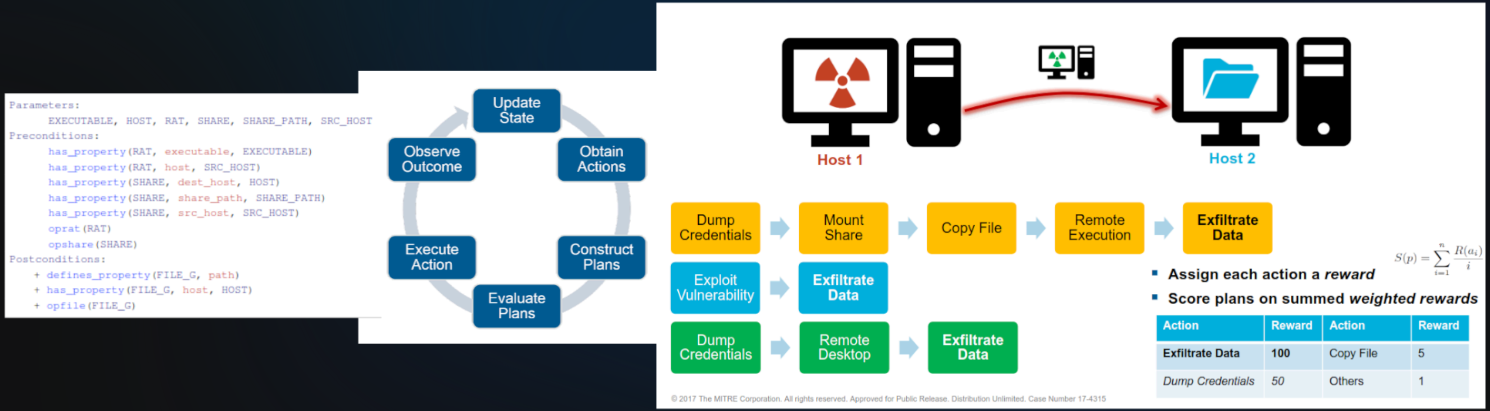
C2 Piloting / Heuristics

C2 Piloting / Heuristics

Caldera[1], an automated framework for adversary emulation (by Mitre), rely on a **heuristic** to select attacks & targets.

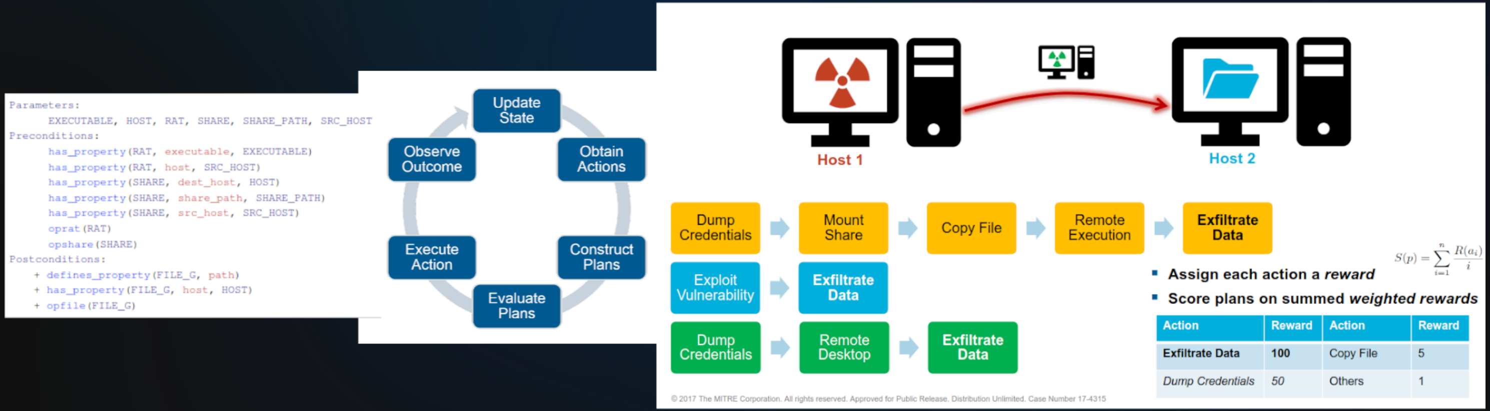
C2 Piloting / Heuristics

Caldera[1], an automated framework for adversary emulation (by Mitre), rely on a **heuristic** to select attacks & targets.



C2 Piloting / Heuristics

Caldera[1], an automated framework for adversary emulation (by Mitre), rely on a **heuristic** to select attacks & targets.



Based on **pre & post conditions**, really basic and not very "smart" system.

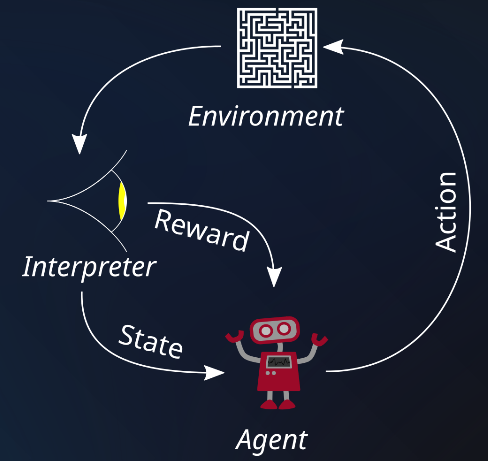
C2 Piloting / RL

C2 Piloting / RL

Another approach used by the literature is Reinforcement Learning (ML).

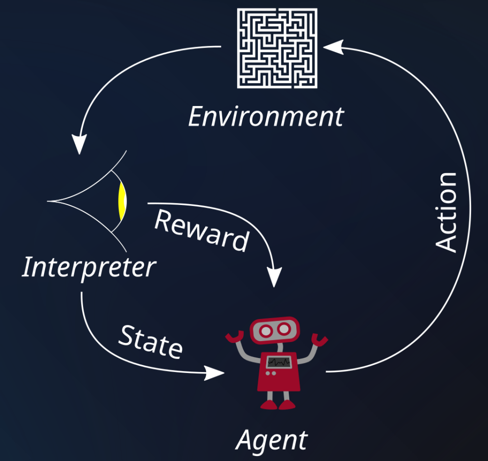
C2 Piloting / RL

Another approach used by the literature is Reinforcement Learning (ML).



C2 Piloting / RL

Another approach used by the literature is Reinforcement Learning (ML).



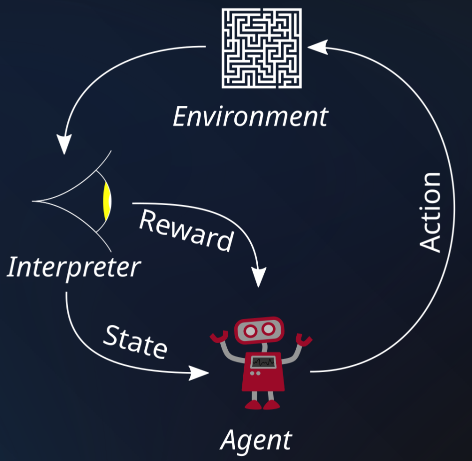
- S is the set of all states,
- A is the set of all actions,
- $R : S \times A \times S \to \mathbb{R}$ is the reward function,
- For a step k, $r_k = R(s_k, a_k, s_{k+1})$ is the associated reward function,

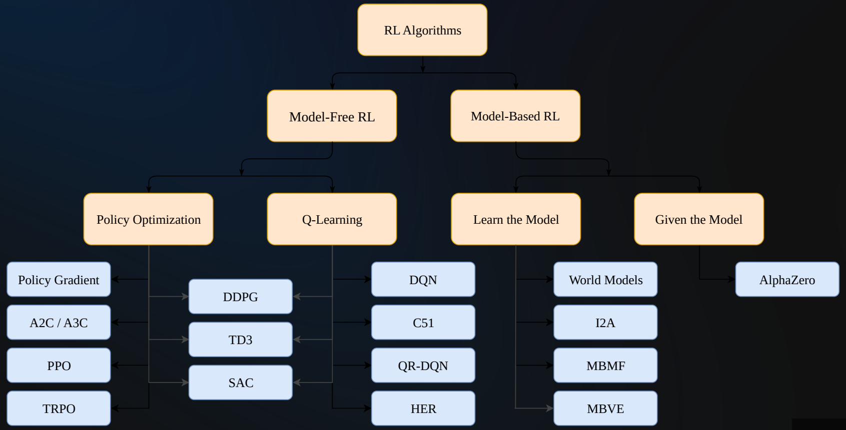
C2 Piloting / RL

Another approach used by the literature is Reinforcement Learning (ML).

- S is the set of all states,
- A is the set of all actions,
- $R : S \times A \times S \to \mathbb{R}$ is the reward function,
- For a step k, $r_k = R(s_k, a_k, s_{k+1})$ is the associated reward function,

Reinforcement learning is vast:





(OpenAI)

C2 Piloting / RL for Piloting

C2 Piloting / RL for Piloting

RL fit well for C2 piloting:

- S contains all network states (like machines, OSes, open ports...),
- A contains all the modules in the attack library (**times possible parameters?!**),
- R is possible to construct, has demonstrated with Caldera's heuristic (**not that easy...**).

C2 Piloting / RL for Piloting

RL fit well for C2 piloting:

- S contains all network states (like machines, OSes, open ports...),
- A contains all the modules in the attack library (**times possible parameters?!**),
- R is possible to construct, has demonstrated with Caldera's heuristic (**not that easy...**).

Playing with R make it possible to complexify model behavior (e.g., valorize exploration vs exploitation, take defense into consideration...)

C2 Piloting / RL Environment

C2 Piloting / RL Environment

First issue, to use RL you need a training environment that'll simulate (mock) inputs/outputs [2][3]. This is called a simulation environment (or Gym).

Multiple environments are available, but are globally oriented toward Deep Learning, making them **very** basic:

C2 Piloting / RL Environment

First issue, to use RL you need a training environment that'll simulate (mock) inputs/outputs [2][3]. This is called a simulation environment (or Gym).

Multiple environments are available, but are globally oriented toward Deep Learning, making them **very** basic:

Address	Compr.	Reach.	Disc.	Value	Access	linux	windows	proftpd	drupal	phpwiki	e_search	wp_ninja	mysql
(1, 0)	True	True	True	0.0	1.0	False	False	False	False	False	True	True	False
(3, 1)	False	True	True	0.0	0.0	False	False	False	False	False	False	False	False
(3, 0)	True	True	True	100.0	2.0	False	False	True	True	True	False	False	True
(4, 1)	False	True	True	0.0	0.0	False	False	False	False	False	False	False	False
(4, 0)	True	True	True	0.0	1.0	False	False	True	False	True	False	False	False

(3)

C2 Piloting / RL Environment

First issue, to use RL you need a training environment that'll simulate (mock) inputs/outputs [2][3]. This is called a simulation environment (or Gym).

Multiple environments are available, but are globally oriented toward Deep Learning, making them **very** basic:

Address	Compr.	Reach.	Disc.	Value	Access	linux	windows	proftpd	drupal	phpwiki	e_search	wp_ninja	mysql
(1, 0)	True	True	True	0.0	1.0	False	False	False	False	False	True	True	False
(3, 1)	False	True	True	0.0	0.0	False	False	False	False	False	False	False	False
(3, 0)	True	True	True	100.0	2.0	False	False	True	True	True	False	False	True
(4, 1)	False	True	True	0.0	0.0	False	False	False	False	False	False	False	False
(4, 0)	True	True	True	0.0	1.0	False	False	True	False	True	False	False	False

(3)

Most, if not all, are **oversimplifying** the problem, and some doesn't even provide realistic evaluation (emulation, or running in a real IS). For example, none allow **attack parametrisation**. And most are using a probability of failure per action (artificial realism).

C2 Piloting / Some RL Environments

C2 Piloting / Some RL Environments

Name	Release Date	Features
NASim	2019	Simulation only
CyberBattleSim	2021	Simulation only, Microsoft
CybORG (CAGE)	2022	Simulation & some level of emulation (~Metasploit C2)
NASimEmu	2023	Simulation & Emulation (Metasploit C2)

C2 Piloting / Some RL Environments

Name	Release Date	Features
NASim	2019	Simulation only
CyberBattleSim	2021	Simulation only, Microsoft
CybORG (CAGE)	2022	Simulation & some level of emulation (~Metasploit C2)
NASimEmu	2023	Simulation & Emulation (Metasploit C2)

Writing such environment takes time since the **level of abstraction is very high**. There is also a **scaling issue**.

C2 Piloting / Some RL Environments

Name	Release Date	Features
NASim	2019	Simulation only
CyberBattleSim	2021	Simulation only, Microsoft
CybORG (CAGE)	2022	Simulation & some level of emulation (~Metasploit C2)
NASimEmu	2023	Simulation & Emulation (Metasploit C2)

Writing such environment takes time since the **level of abstraction is very high**. There is also a **scaling issue**.

Metasploit is the **ONLY** used C2 across the literature.

C2 Piloting / More RL Agents

C2 Piloting / More RL Agents

Model	Technique	Complex Behavior	Parametrisation	Generalization	Year
POMDP [4]	POMDP	No	No	No	2013
LD-PenTesting [5]	POMDP	~Defender Behaviour	No	No	2020
EPPTA [6]	POMDP + PPO	No	No	No	2023
AutoRed [7]	POMDP	No	No	GNN	2024

C2 Piloting / More RL Agents

Model	Technique	Complex Behavior	Parametrisation	Generalization	Year
POMDP [4]	POMDP	No	No	No	2013
LD-PenTesting [5]	POMDP	~Defender Behaviour	No	No	2020
EPPTA [6]	POMDP + PPO	No	No	No	2023
AutoRed [7]	POMDP	No	No	GNN	2024

Multiple issues with these approaches :

- Questionable generalization (except AutoRed),
- Imperfect reward function R/policy π (complexity),
- No attack parametrisation,
- Questionable performances,
- Low quality/high abstraction environments.

C2 Piloting / A Complex Problem

C2 Piloting / A Complex Problem

Multiple questions:

- **One or multiple** models to select the target, the attack and the parameters?
- Does the target selection model **see the whole network**, or just select the target with the best attack?
- Since a network can have a (virtually) **infinite number of machines** (targets), how to construct the input/context for the target selection model?
- How to create an adequate **reward function/policy**?
- How to capture the **complexity** of the problem (like taking into account defense reaction)?
- C2 Piloting is a **planning problem** (with goals & subgoals)?
- On a big network, with a lot of possible actions, each with many parameters, how **not to collapse**?

C2 Piloting / New Approach

C2 Piloting / New Approach

Proposed approach

Issue	Solution	Description
Generalization	GNN / Carrousel	Ingest the network context independently from its size. Learn to retain only interesting information. Context-based ingestion.
Partially Observable	POMDP + Dreamer (*JEPA?*)	Internal environment representation, latent space reasoning.
Complex Behavior	Complex R & Goals	Complexify the reward function. Add goals & subgoals to complexify policies (planning).
C2 Automation	New Specific C2	Create a C2 with automation in mind.
RL Environment	New Emulation First RL Environment	Create a new RL environment centered around emulation (maximum realism).

C2 Piloting / Other Approaches

C2 Piloting / Other Approaches

Other approaches exist:

- Some people are trying to use **LLMs for C2 piloting** (me! [8]), but also end-to-end OAI. Results are not great, OAI is not really **hallucination-compatible** (e.g., commands needs to be exact, IPs also, impact if the wrong target...).
- RL is not the only learning method: **Self-Supervised Learning** is trending.
- **Heuristics** are not dead.
- **Expert systems** (analogy to the defensive side) should also be explored.

C2 Piloting / Other Approaches

Other approaches exist:

- Some people are trying to use **LLMs for C2 piloting** (me! [8]), but also end-to-end OAI. Results are not great, OAI is not really **hallucination-compatible** (e.g., commands needs to be exact, IPs also, impact if the wrong target...).
- RL is not the only learning method: **Self-Supervised Learning** is trending.
- **Heuristics** are not dead.
- **Expert systems** (analogy to the defensive side) should also be explored.

The dream is to have **end-to-end OAI** (target, attack, parameters choosing with attack generation).

C2 Piloting / Conclusion

C2 Piloting / Conclusion

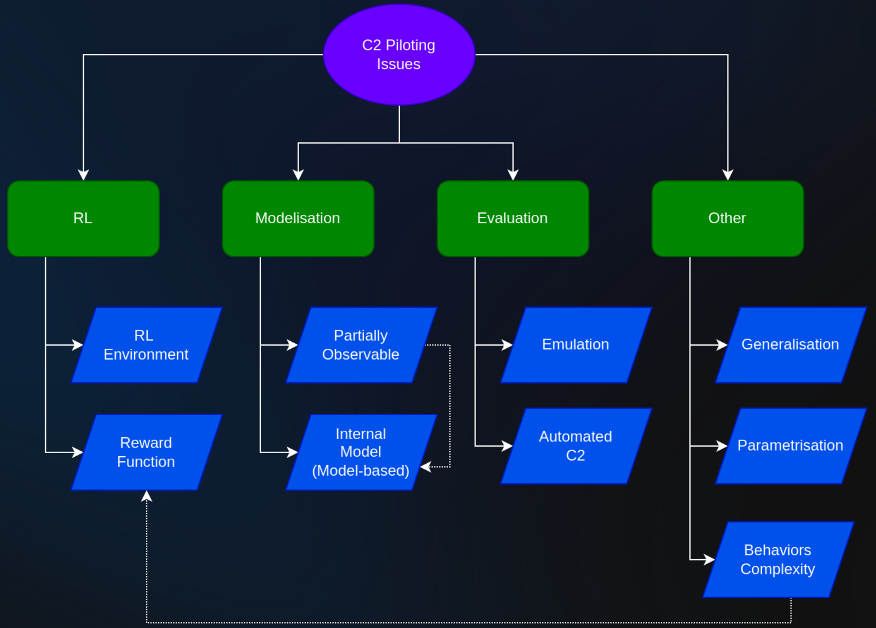
Some "Hot Takes":

- C2 Piloting is **hard**.
- This is not just a **model choice issue**.
- Past literature (minus some exception) are **focusing on the wrong problematic**.
- Linked **use cases** are huge.
- Ask many **fundamental** questions.
- When it will work, the **impact might be important** for defenses.
- (*LLMs are not the solution*)

C2 Piloting / Conclusion

Some "Hot Takes":

- C2 Piloting is **hard**.
- This is not just a **model choice issue**.
- Past literature (minus some exception) are **focusing on the wrong problematic**.
- Linked **use cases** are huge.
- Ask many **fundamental** questions.
- When it will work, the **impact might be important** for defenses.
- (*LLMs are not the solution*)



OAI for Evasion

Online & Offline evasion...

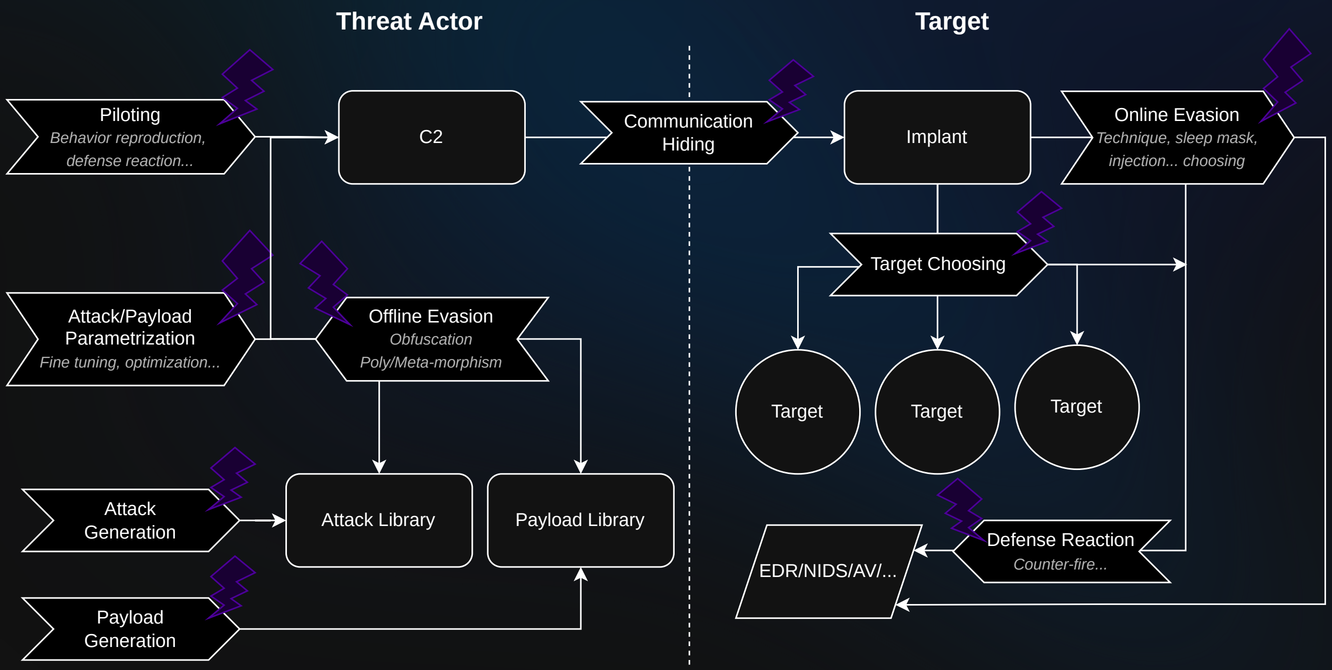
Evasion

Evasion

Consists of **optimizing** (e.g., by selecting parameters) an evasion method.

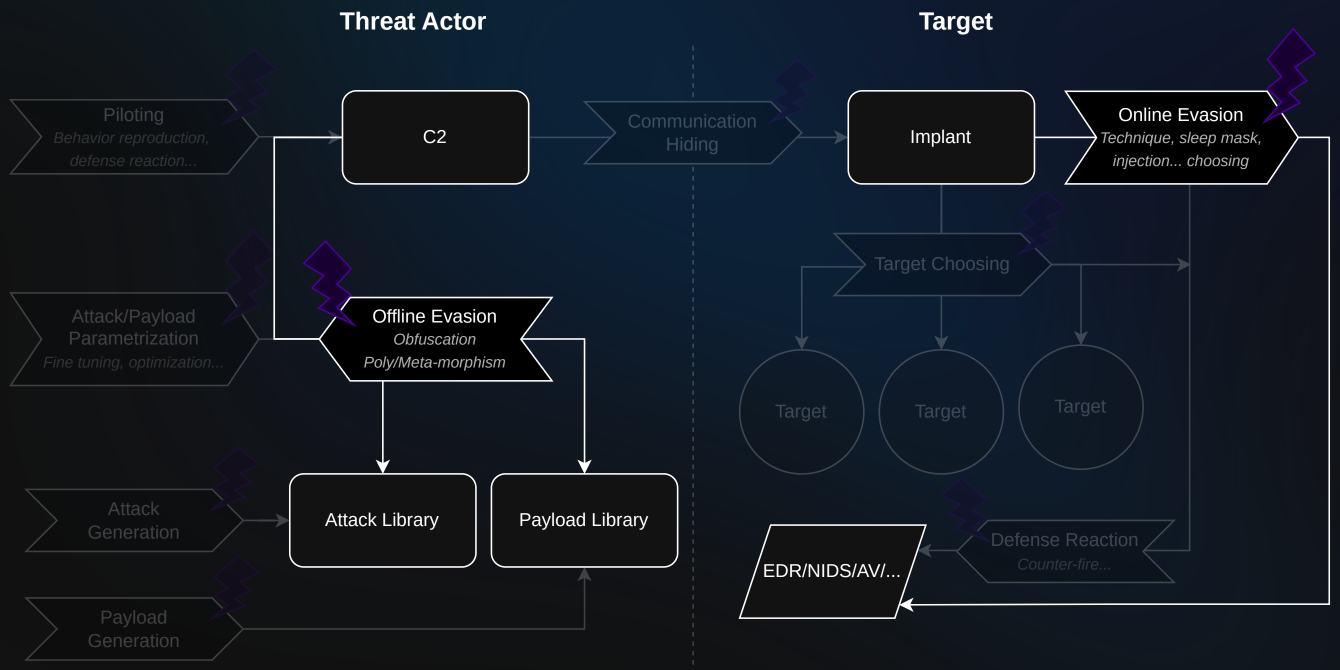
Evasion

Consists of **optimizing** (e.g., by selecting parameters) an evasion method.



Evasion

Consists of **optimizing** (e.g., by selecting parameters) an evasion method.



Evasion / The Idea

Evasion / The Idea

You want to **bypass defenses** (and here, specifically host based sensors, like EDRs).

Many evasion methods are available, but they generally have **parameters**.

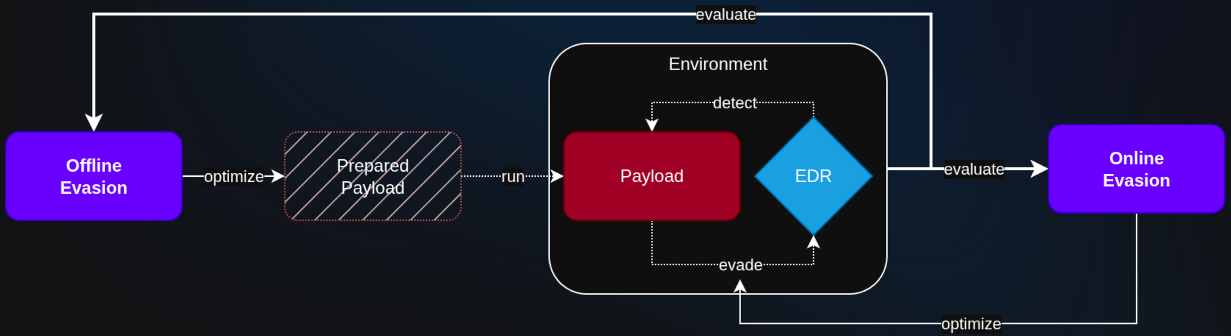
Each sensor is generally **more vulnerable** to specific combination of evasion techniques, associated with specific parameters.

Evasion / The Idea

You want to **bypass defenses** (and here, specifically host based sensors, like EDRs).

Many evasion methods are available, but they generally have **parameters**.

Each sensor is generally **more vulnerable** to specific combination of evasion techniques, associated with specific parameters.

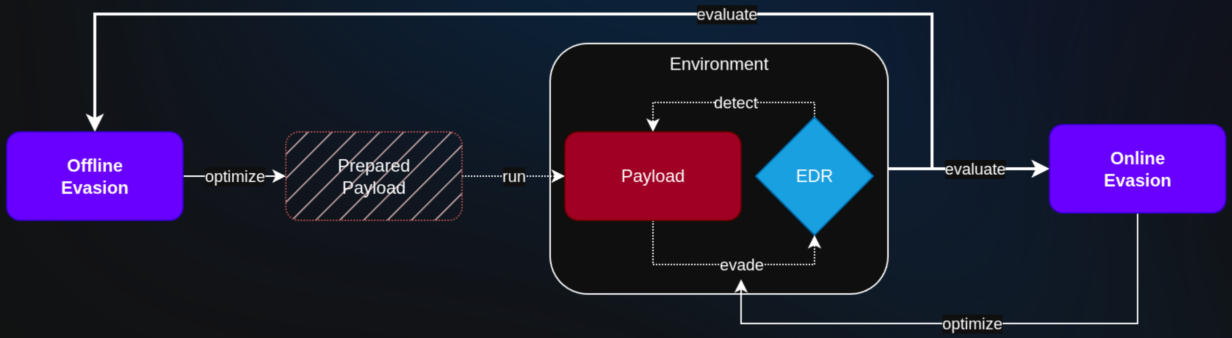


Evasion / The Idea

You want to **bypass defenses** (and here, specifically host based sensors, like EDRs).

Many evasion methods are available, but they generally have **parameters**.

Each sensor is generally **more vulnerable** to specific combination of evasion techniques, associated with specific parameters.



AI can be used to find these combinations and associated parameters, **optimizing evasion**.

Offline Evasion

Offline Evasion

Offline evasion includes any method that reduces detection & evade defenses **statically** (e.g., modifying a payload before sending it). We want to **obfuscate a payload** not to be detected by static analysis.

Offline Evasion

Offline evasion includes any method that reduces detection & evade defenses **statically** (e.g., modifying a payload before sending it). We want to **obfuscate a payload** not to be detected by static analysis.

This cover:

- Not having the same hash,
- Obfuscating code,
- Hidding strings,
- Packing,
- ...

Offline Evasion

Offline evasion includes any method that reduces detection & evade defenses **statically** (e.g., modifying a payload before sending it). We want to **obfuscate a payload** not to be detected by static analysis.

This cover:

- Not having the same hash,
- Obfuscating code,
- Hidding strings,
- Packing,
- ...

Some techniques:

- Polymorphism
- Metamorphism
- Obfuscation (excluding *morphism)
- ...

Offline Evasion

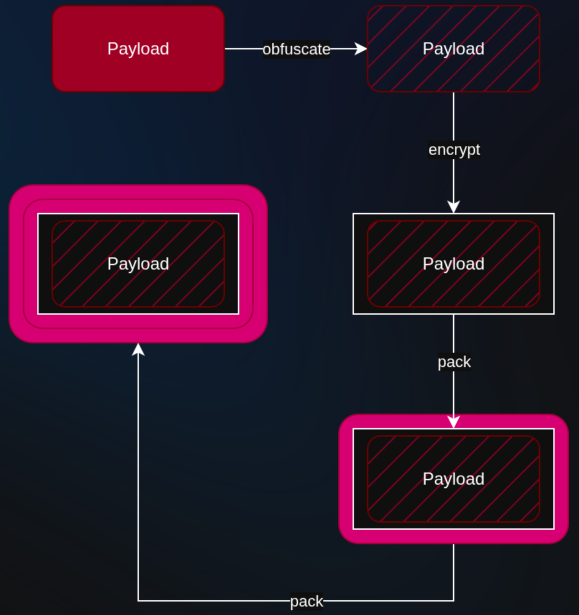
Offline evasion includes any method that reduces detection & evade defenses **statically** (e.g., modifying a payload before sending it). We want to **obfuscate a payload** not to be detected by static analysis.

This cover:

- Not having the same hash,
- Obfuscating code,
- Hidding strings,
- Packing,
- ...

Some techniques:

- Polymorphism
- Metamorphism
- Obfuscation (excluding *morphism)
- ...



Offline Evasion / *morphism

Offline Evasion / *morphism

Objective: Complexify part of a binary to make detection and/or reverse engineering **harder**. This maintains same functionalities, but make **analysis more difficult**.

Offline Evasion / *morphism

Objective: Complexify part of a binary to make detection and/or reverse engineering **harder**. This maintains same functionalities, but make **analysis more difficult**.

Automation Surface:

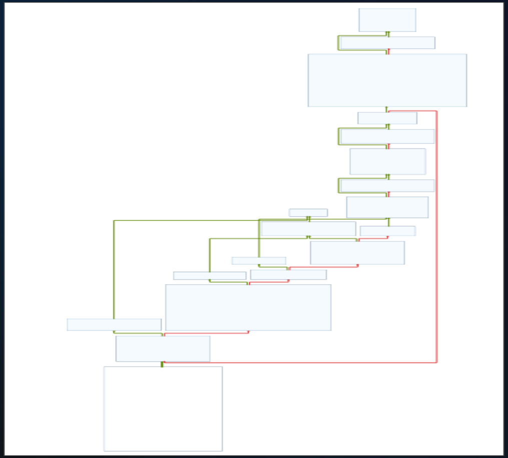
- Call graph obfuscation

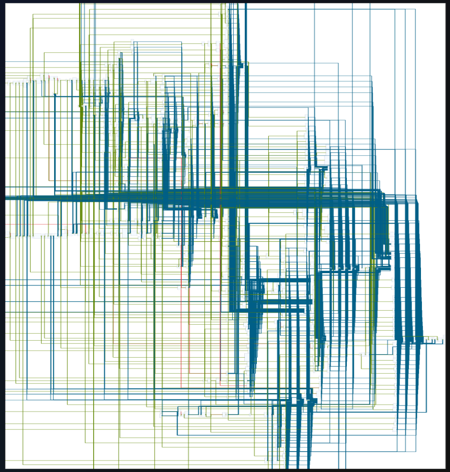
Offline Evasion / *morphism

<u>Objective:</u> Complexify part of a binary to make detection and/or reverse engineering **harder**. This maintains same functionalities, but make **analysis more difficult**.

Automation Surface:

- Call graph obfuscation





Offline Evasion / *morphism

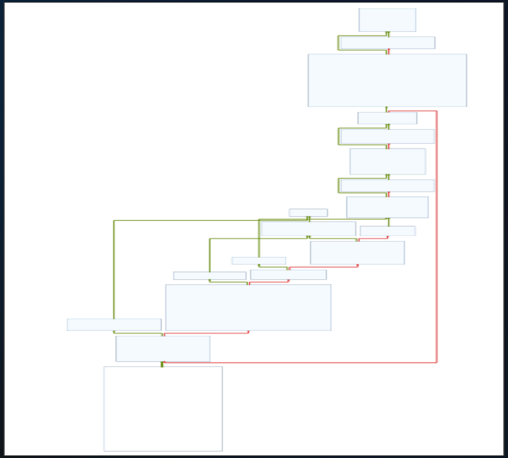
Objective: Complexify part of a binary to make detection and/or reverse engineering **harder**. This maintains same functionalities, but make **analysis more difficult**.

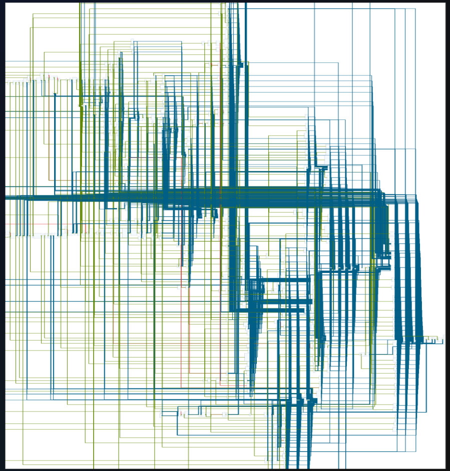
Automation Surface:

- Call graph obfuscation

Examples:

- 1 Maximize binary diff
- 2 Maximize number of subroutines
- 3 Maximize number of calls between each subroutines





Offline Evasion / *morphism

Objective: Complexify part of a binary to make detection and/or reverse engineering **harder**. This maintains same functionalities, but make **analysis more difficult**.

Automation Surface:

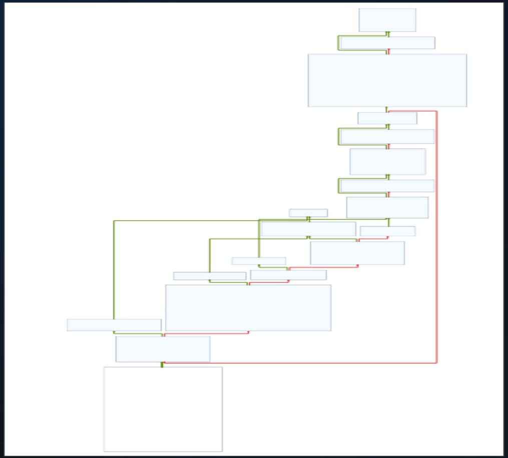
- Call graph obfuscation

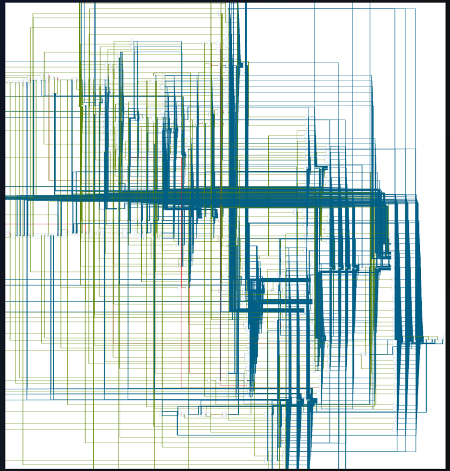
Examples:

- 1 Maximize binary diff
- 2 Maximize number of subroutines
- 3 Maximize number of calls between each subroutines

Automation Possiblities:

- Meta-heuristic (e.g., Genetic Algorithm): 1 2 3
- Reinforcement Learning: 1 2 3
- Heuristic: 2 3





Offline Evasion / Packing

Offline Evasion / Packing

Objective: Hide a payload by including it **into another**. At execution, the latter will **unpack** the payload and **pass execution** to it. Can include payload encryption & compression.

Offline Evasion / Packing

Objective: Hide a payload by including it **into another**. At execution, the latter will **unpack** the payload and **pass execution** to it. Can include payload encryption & compression.

Automation Surface:

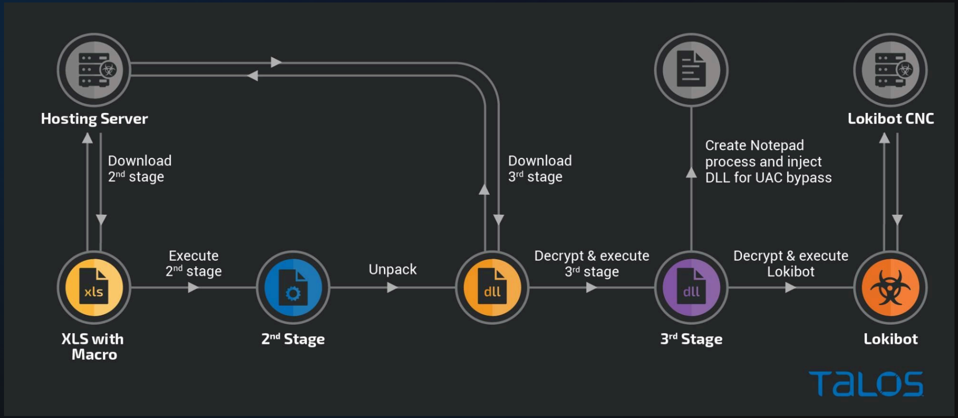
- Packer nesting
- Packer optimization

Offline Evasion / Packing

Objective: Hide a payload by including it **into another**. At execution, the latter will **unpack** the payload and **pass execution** to it. Can include payload encryption & compression.

Automation Surface:

- Packer nesting
- Packer optimization



Offline Evasion / Packing

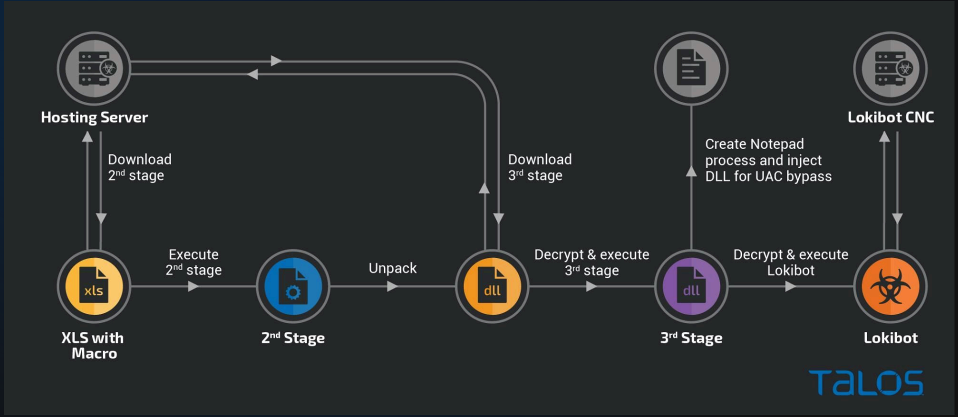
Objective: Hide a payload by including it **into another**. At execution, the latter will **unpack** the payload and **pass execution** to it. Can include payload encryption & compression.

Automation Surface:

- Packer nesting
- Packer optimization

Examples:

- 1 Minimize packer entropy
- 2 Maximize waves diversity



Offline Evasion / Packing

Objective: Hide a payload by including it **into another**. At execution, the latter will **unpack** the payload and **pass execution** to it. Can include payload encryption & compression.

Automation Surface:

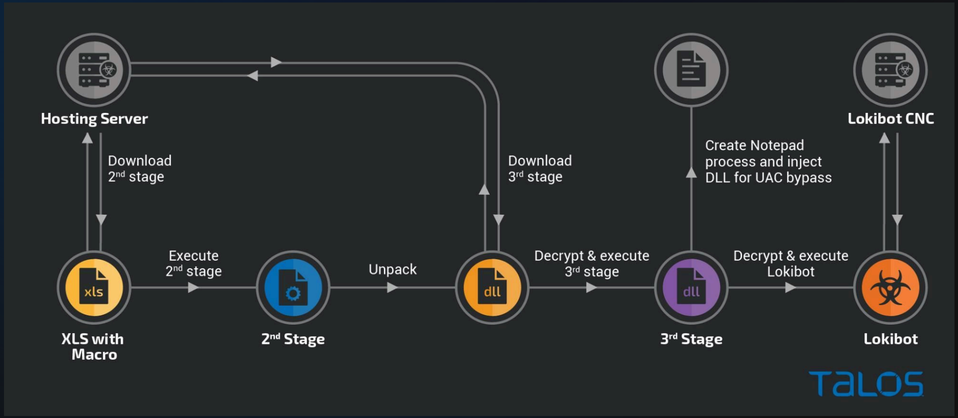
- Packer nesting
- Packer optimization

Examples:

- 1 Minimize packer entropy
- 2 Maximize waves diversity

Automation Possibilities:

- Meta-heuristic (e.g., Genetic Algorithm) 1 2
- Reinforcement Learning 1 2
- Heuristic: 1



Online Evasion

Online Evasion

Definition

Online evasion includes any method that **reduce detection** & **evade** defenses **dynamically** (at runtime). We want to **hide traces** not to be detected by dynamic analysis.

Online Evasion

Definition

Online evasion includes any method that **reduce detection** & **evade** defenses **dynamically** (at runtime). We want to **hide traces** not to be detected by dynamic analysis.

This cover:

- Hide API calls,
- Hide in another process,
- Mask/Encrypt part of the payload in memory,
- Reproduce legitimate behaviors,
- ...

Online Evasion

Definition

Online evasion includes any method that **reduce detection** & **evade** defenses **dynamically** (at runtime). We want to **hide traces** not to be detected by dynamic analysis.

This cover:

- Hide API calls,
- Hide in another process,
- Mask/Encrypt part of the payload in memory,
- Reproduce legitimate behaviors,
- ...

Some techniques:

- Sleep Obfuscation
- Memory Masking
- Process Injection
- Call Stack Spoofing
- ...

Online Evasion

Definition

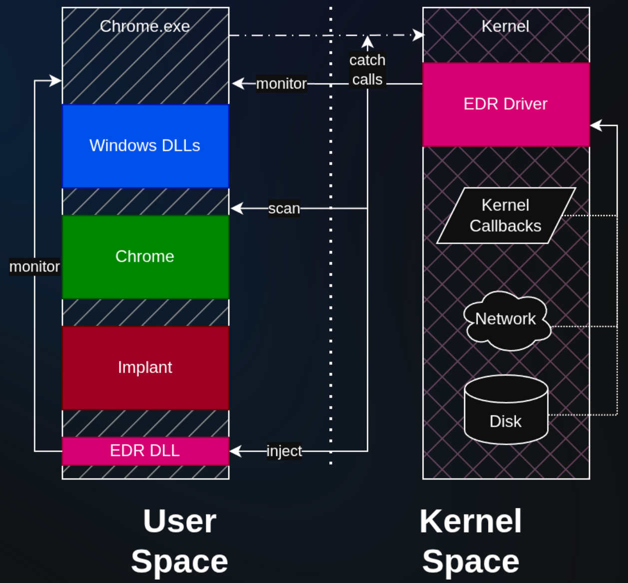
Online evasion includes any method that **reduce detection** & **evade** defenses **dynamically** (at runtime). We want to **hide traces** not to be detected by dynamic analysis.

This cover:

- Hide API calls,
- Hide in another process,
- Mask/Encrypt part of the payload in memory,
- Reproduce legitimate behaviors,
- ...

Some techniques:

- Sleep Obfuscation
- Memory Masking
- Process Injection
- Call Stack Spoofing
- ...



Online Evasion / Process Injection

Online Evasion / Process Injection

Objective: Execute **malicous code** in the context of a **target process**.

This can turn any legitimate process into a malicious one.

Online Evasion / Process Injection

Objective: Execute **malicous code** in the context of a **target process**.
This can turn any legitimate process into a malicious one.

Automation Surface:

- Find API combination
- Choose target process

Online Evasion / Process Injection

Objective: Execute **malicous code** in the context of a **target process**.

This can turn any legitimate process into a malicious one.

Automation Surface:

- Find API combination
- Choose target process

Online Evasion / Process Injection

Objective: Execute **malicous code** in the context of a **target process**.

This can turn any legitimate process into a malicious one.

Automation Surface:

- Find API combination
- Choose target process

Examples:

- 1 Maximize behavior similarity of target process
- 2 Select best injection method (against specific defense)

Online Evasion / Process Injection

Objective: Execute **malicous code** in the context of a **target process**.

This can turn any legitimate process into a malicious one.

Automation Surface:

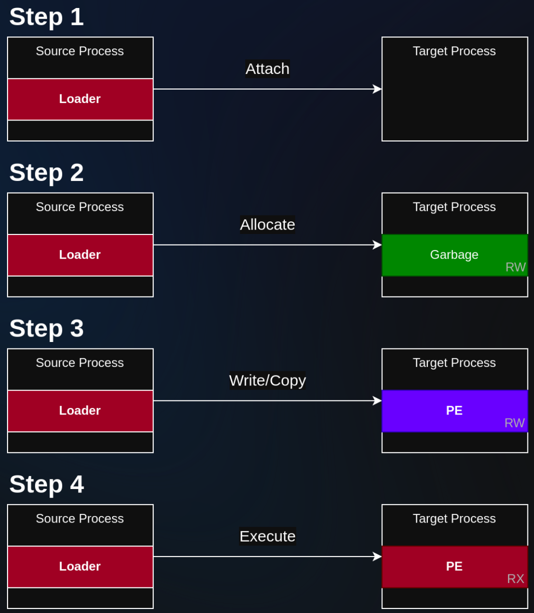
- Find API combination
- Choose target process

Examples:

- 1 Maximize behavior similarity of target process
- 2 Select best injection method (against specific defense)

Automation Possibilities:

- Meta-heuristic (e.g., Genetic Algorithm): 1 2
- Reinforcement Learning: 1 2
- Heuristic: 1



Online Evasion / Memory Masking

Online Evasion / Memory Masking

<u>Objective:</u> Try to **hide** part, or the entirety, of a payload **in memory**. This is useful against **memory scanners**, like YARA. **Sleep obfuscation** is a variant, where a payload will sleep & be encrypted in memory during that time.

Online Evasion / Memory Masking

Objective: Try to **hide** part, or the entirety, of a payload **in memory**. This is useful against **memory scanners**, like YARA. **Sleep obfuscation** is a variant, where a payload will sleep & be encrypted in memory during that time.

Automation Surface:

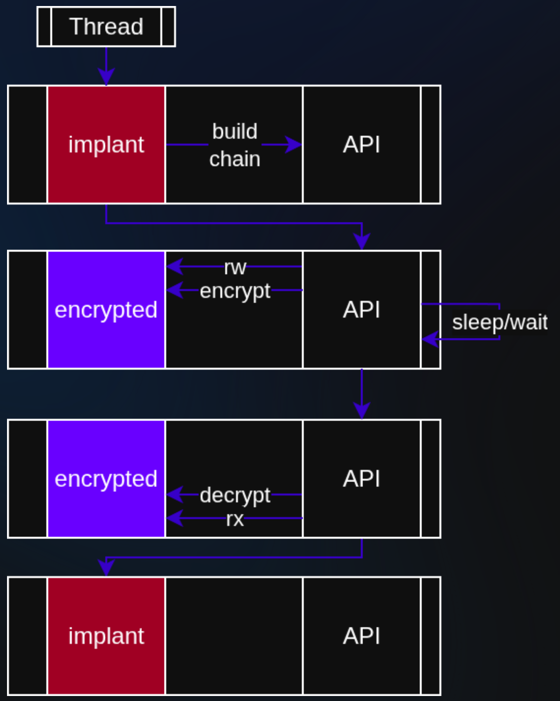
- Technique parameters
- API used

Online Evasion / Memory Masking

Objective: Try to **hide** part, or the entirety, of a payload **in memory**. This is useful against **memory scanners**, like YARA. **Sleep obfuscation** is a variant, where a payload will sleep & be encrypted in memory during that time.

Automation Surface:

- Technique parameters
- API used



Online Evasion / Memory Masking

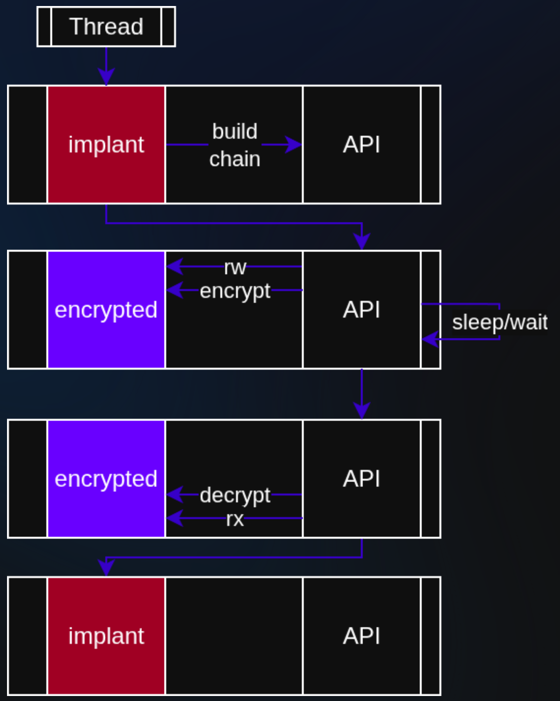
<u>Objective:</u> Try to **hide** part, or the entirety, of a payload **in memory**. This is useful against **memory scanners**, like YARA. **Sleep obfuscation** is a variant, where a payload will sleep & be encrypted in memory during that time.

Automation Surface:

- Technique parameters
- API used

Examples:

- 1 Find "optimal" sleep/jitter value
- 2 Find "optimal" delay before remasking



Online Evasion / Memory Masking

Objective: Try to **hide** part, or the entirety, of a payload **in memory**. This is useful against **memory scanners**, like YARA. **Sleep obfuscation** is a variant, where a payload will sleep & be encrypted in memory during that time.

Automation Surface:

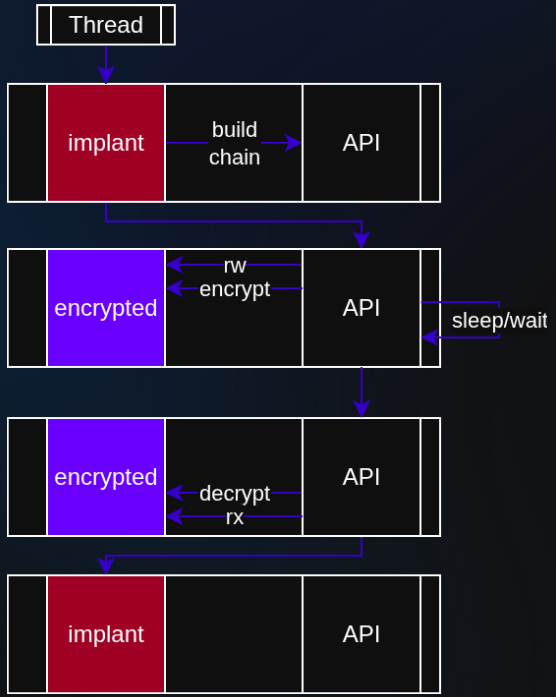
- Technique parameters
- API used

Examples:

- 1 Find "optimal" sleep/jitter value
- 2 Find "optimal" delay before remasking

Automation Possibilities:

- Meta-heuristic (e.g., Genetic Algorithm): 1 2
- Heuristic: 1 2



Online Evasion / Call Stack

Online Evasion / Call Stack

<u>Objective:</u> Spoof or construct a **fake call stack**, to **hide original callers** and evade detection.

Online Evasion / Call Stack

Objective: Spoof or construct a **fake call stack**, to **hide original callers** and evade detection.

Automation Surface:

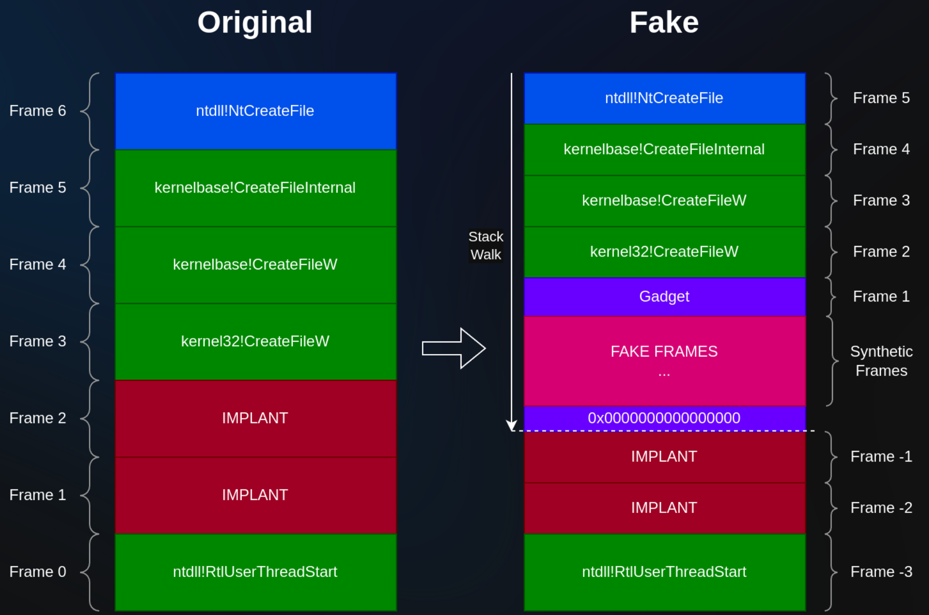
- Fake API choice

Online Evasion / Call Stack

Objective: Spoof or construct a **fake call stack**, to **hide original callers** and evade detection.

Automation Surface:

- Fake API choice



Online Evasion / Call Stack

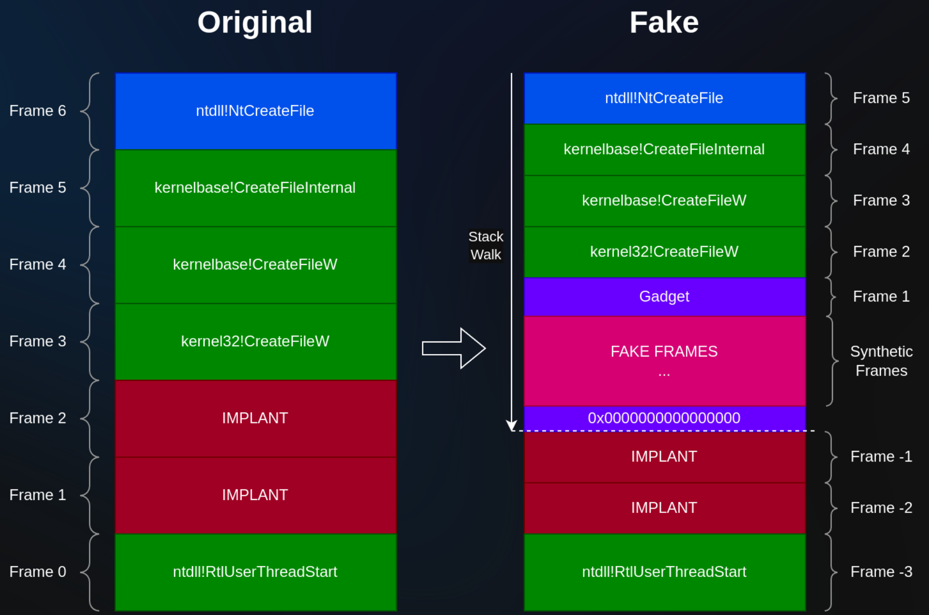
Objective: Spoof or construct a **fake call stack**, to **hide original callers** and evade detection.

Automation Surface:

- Fake API choice

Examples:

- 1 Maximize similarity to injected process call stack
- 2 Find suspicious call stacks



Online Evasion / Call Stack

Objective: Spoof or construct a **fake call stack**, to **hide original callers** and evade detection.

Automation Surface:

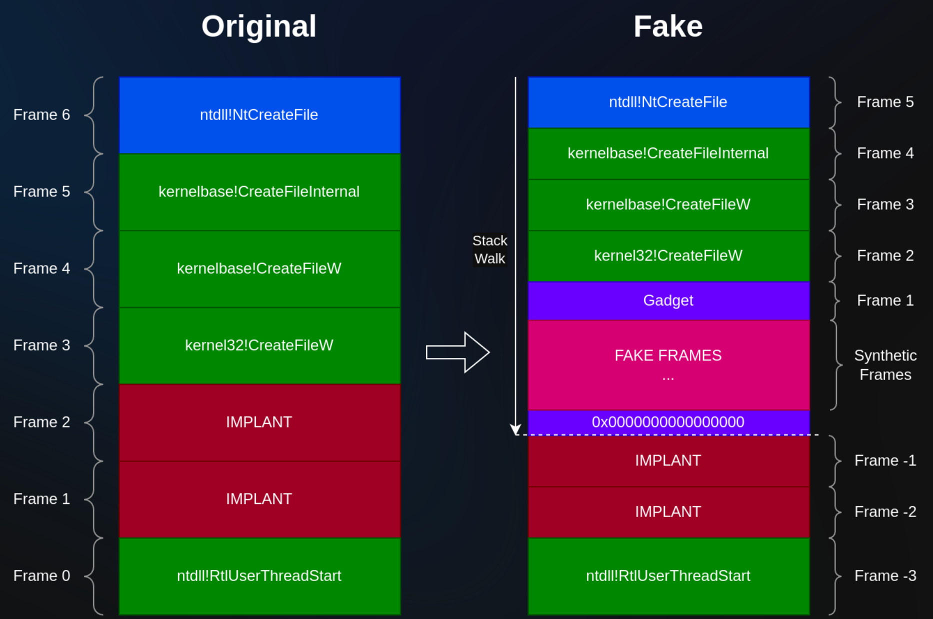
- Fake API choice

Examples:

- 1 Maximize similarity to injected process call stack
- 2 Find suspicious call stacks

Automation Possibilities:

- Meta-heuristic (e.g., Genetic Algorithm): 1 2
- Reinforcement Learning: 1 2
- Heuristic: 1



Meta-optimization

Meta-optimization

Objective: Each evasion technique has some variation available. Considering all variations of all techniques, this creates a lot of combination possibilities.

Meta-optimization

Objective: Each evasion technique has some variation available. Considering all variations of all techniques, this creates a lot of combination possibilities.

Automation Surface:

- Techniques combinations
- Techniques variations

Meta-optimization

Objective: Each evasion technique has some variation available. Considering all variations of all techniques, this creates a lot of combination possibilities.

Automation Surface:

- Techniques combinations
- Techniques variations

Examples:

- 1 Minimize number of techniques
- 2 Find all combinations leading to bypass

Meta-optimization

Objective: Each evasion technique has some variation available. Considering all variations of all techniques, this creates a lot of combination possibilities.

Automation Surface:

- Techniques combinations
- Techniques variations

Examples:

- 1 Minimize number of techniques
- 2 Find all combinations leading to bypass

Automation Possibilities:

- Meta-heuristic (e.g., Genetic Algorithm): 1 2
- Reinforcement Learning: 1 2
- Surrogate Model: 2

Meta-optimization

<u>Objective:</u> Each evasion technique has some variation available. Considering all variations of all techniques, this creates a lot of combination possibilities.

Automation Surface:

- Techniques combinations
- Techniques variations

Examples:

- 1 Minimize number of techniques
- 2 Find all combinations leading to bypass

Automation Possibilities:

- Meta-heuristic (e.g., Genetic Algorithm): 1 2
- Reinforcement Learning: 1 2
- Surrogate Model: 2

Product	Call Stack	Process Injection	Metamorphism	Memory Masking
EDR1	Variant 2	"Explorer"	Variant 1	X
EDR2	Variant 2	X	Variant 1	X
EDR2 + Win11	Variant 3	"Google Chrome"	X	Variant 2

Surrogate Model

Surrogate Model

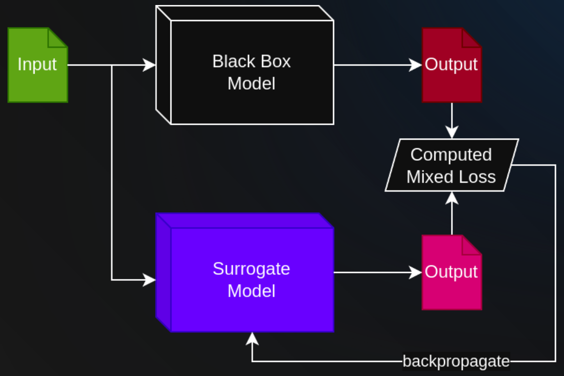
Definition

A surrogate model approximates a black box model by learning pairs of input/output.

Surrogate Model

Definition

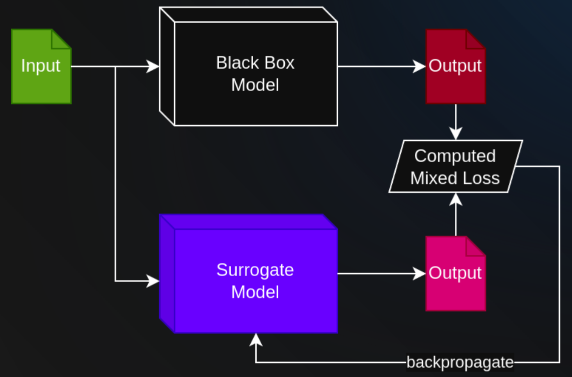
A surrogate model approximates a black box model by learning pairs of input/output.



Surrogate Model

Definition

A surrogate model approximates a black box model by learning pairs of input/output.



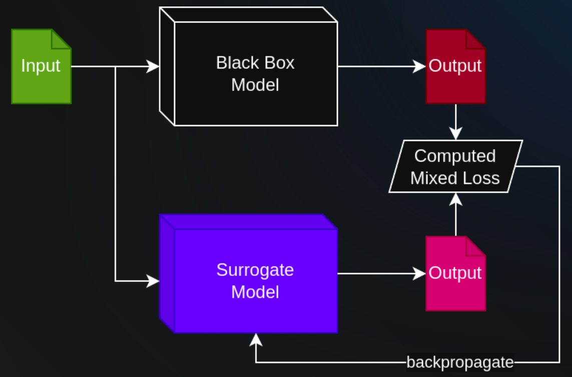
Objective

We can then use the surrogate model to build adversarial inputs to force misclassification by the black box model. For that, we can use the surrogate model internals (like embedding or gradients).

Surrogate Model

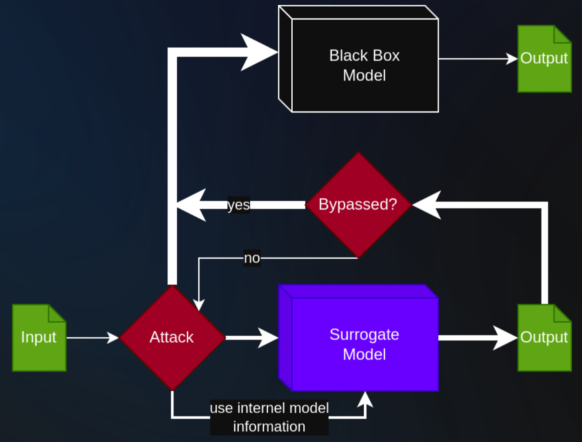
Definition

A surrogate model approximates a black box model by learning pairs of input/output.



Objective

We can then use the surrogate model to build adversarial inputs to force misclassification by the black box model. For that, we can use the surrogate model internals (like embedding or gradients).



Surrogate Model: Evasion

Surrogate Model: Evasion

Usage for evasion

The black box model can be the classification algorithm of an EDR, the input a binary (offline evasion) or dynamic traces generated by it (online evasion) and the output the benign/malicious classification.

Surrogate Model: Evasion

Usage for evasion

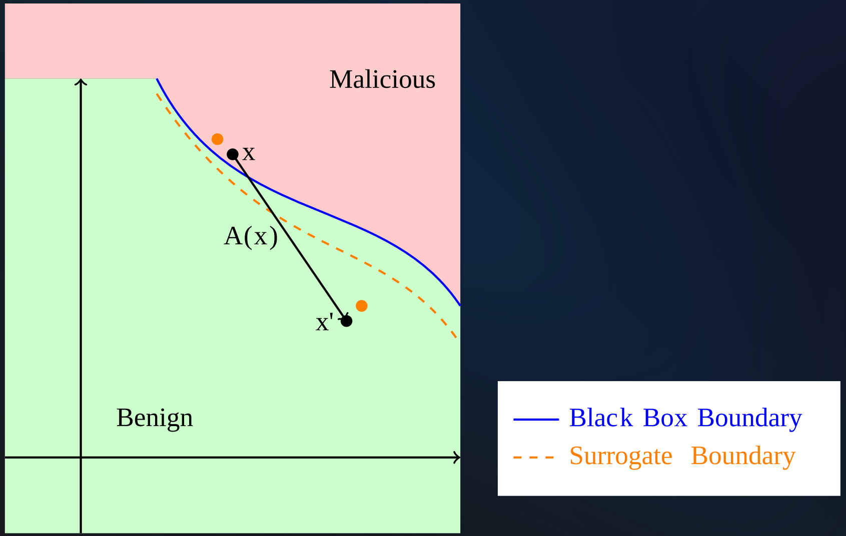
The black box model can be the classification algorithm of an EDR, the input a binary (offline evasion) or dynamic traces generated by it (online evasion) and the output the benign/malicious classification.

Possible attacks

- Add obfuscation, packing...
- Use process injection, call stack spoofing...
- Mask/Obfuscate memory...

All of these techniques are primitives that we can leverage to build adversarial binaries, using our surrogate model.

Surrogate Model: Visualization



Where $A(x) = \epsilon_{\Omega}(x)$.

Conclusion

Conclusion

Some "Hot Takes":

- Automation for evasion is **easier** than C2 piloting.
- Not much literature on the subject.
- Might not be explored by anyone?
- Multimodal $>$ Unimodal
- Linked **use cases** are important.
- When it will work, the **impact might be important** for defenses.
- (*LLMs are not the solution*)

Conclusion

OAI is larger!

OAI is larger!

Not discussed, but:

- OAI for **attack generation.**
- OAI for **phishing.**
- OAI for **defense reaction.**
- OAI for **payload generation.**
- OAI for **communication hiding.**
- ...

OAI is larger!

Not discussed, but:

- OAI for **attack generation**.
- OAI for **phishing**.
- OAI for **defense reaction**.
- OAI for **payload generation**.
- OAI for **communication hiding**.
- ...

The subject is **vast**!

OAI is larger!

Not discussed, but:

- OAI for attack generation.
- OAI for phishing.
- OAI for defense reaction.
- OAI for payload generation.
- OAI for communication hiding.
- ...

The subject is vast!

But not very explored (in the literature).

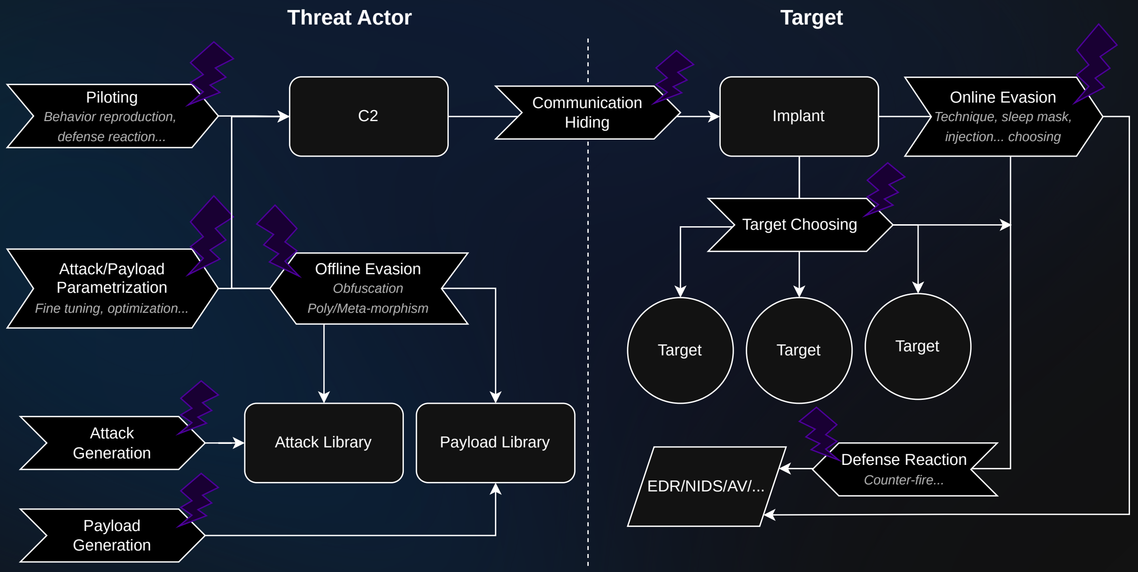
OAI is larger!

Not discussed, but:

- OAI for attack generation.
- OAI for phishing.
- OAI for defense reaction.
- OAI for payload generation.
- OAI for communication hiding.
- ...

The subject is vast!

But not very explored (in the literature).



Thanks!

A complete version of this conference is available here https://dorianb.net/Talks/OAI2026/Full

For more https://dorianb.net/

Bibliography

- 1. Miller D, Alford R, Applebaum A, Foster H, Little C, Strom BE. Automated Adversary Emulation: A Case for Planning and Acting with Unknowns. In 2018. Available from: https://api.semanticscholar.org/CorpusID:145814722 (13)
- 2. Schwartz J, Kurniawati H. Autonomous Penetration Testing using Reinforcement Learning. ArXiv [Internet]. 2019;abs/1905.05965. Available from: https://api.semanticscholar.org/CorpusID:155090057 (16)
- 3. Janisch J, Pevn’y T, Lis’y V. NASimEmu: Network Attack Simulator & Emulator for Training Agents Generalizing to Novel Scenarios. ArXiv [Internet]. 2023;abs/2305.17246. Available from: https://api.semanticscholar.org/CorpusID:258959386 (16)

page 1 / 5

Bibliography

- 4. Sarraute C, Buffet O, Hoffmann J. POMDPs Make Better Hackers: Accounting for Uncertainty in Penetration Testing [Internet]. 2013. Available from: https://arxiv.org/abs/1307.8182 (18)
- 5. Schwartz J, Kurniawati H, El-Mahassni E. POMDP + Information-Decay: Incorporating Defender's Behaviour in Autonomous Penetration Testing. Proceedings of the International Conference on Automated Planning and Scheduling [Internet]. 2020;30(1):235–43. Available from: https://ojs.aaai.org/index.php/ICAPS/article/view/6666 (18)
- 6. Li Z, Zhang Q, Yang G. EPPTA: Efficient partially observable reinforcement learning agent for penetration testing applications. Engineering Reports. 2023; (18)

page 2 / 5

Bibliography

- 7. Hasegawa K, Hidano S, Fukushima K. AutoRed: Automating Red Team Assessment via Strategic Thinking Using Reinforcement Learning. In: Proceedings of the Fourteenth ACM Conference on Data and Application Security and Privacy [Internet]. Porto, Portugal: Association for Computing Machinery; 2024. p. 325–36. (CODASPY '24). Available from: https://doi.org/10.1145/3626232.3653252 (18)
- 8. Genevey-Metat C, Bachelot D, Gourmelen T, Quemat A, Satre PM, Scotto L, et al. Red Team LLM: towards an adaptive and robust automation solution. In: Conference on Artificial Intelligence for Defense [Internet]. Rennes, France: DGA Maîtrise de l'Information; 2023. Available from: https://hal.science/hal-04328468 (21)
- 9. Zhou S, Liu J, Hou D, Zhong X, Zhang Y. Autonomous Penetration Testing Based on Improved Deep Q-Network. Applied Sciences [Internet]. 2021;11(19). Available from: https://www.mdpi.com/2076-3417/11/19/8823

page 3 / 5

Bibliography

- 10. Yang Y, Liu X. Behaviour-Diverse Automatic Penetration Testing: A Curiosity-Driven Multi-Objective Deep Reinforcement Learning Approach [Internet]. 2022. Available from: https://arxiv.org/abs/2202.10630
- 11. Blumenthal O, Shani G. Domain Independent Heuristics for Online Stochastic Contingent Planning. 2023.
- 12. Tran K, Akella A, Standen M, Kim J, Bowman D, Richer T, et al. Deep hierarchical reinforcement agents for automated penetration testing [Internet]. 2021. Available from: https://arxiv.org/abs/2109.06449

page 4 / 5

Bibliography

- 13. Chen Z, Kang F, Xiong X, Shu H. A Survey on Penetration Path Planning in Automated Penetration Testing. Applied Sciences [Internet]. 2024;14(18). Available from: https://www.mdpi.com/2076-3417/14/18/8355

page 5 / 5