Artificial Intelligence Methods in Virus Detection & Recognition

Introduction to heuristic scanning

Wojciech Podgórski

http://podgorski.wordpress.com

October 16, 2008

Presentation outline

Malware

Malware (**mal**icious soft**ware**) is software designed to infiltrate or damage a computer system without the owner's informed consent.

Source: *http://en.wikipedia.org/wiki/Malware*

Malware types

We can distinguish quite few malicious software types. It is important to be aware that nevertheless all of them have similar purpose, each one behave differently.

- Viruses
- Worms
- Wabbits
- Trojan horses
- Exploits/Backdoors
- Spyware/Scumware/Stealware/Parasiteware/Adware
- Rootkits
- Keyloggers/Dialers
- Hoaxes

Malware $\neq$ Viruses

Due to different behaviour, each malware group uses **alternative ways of being undetected**. This forces anti-virus software producers to develop numerous solutions and countermeasures for computer protection.

This presentation focuses on methods used especially for virus detection, **not necessarily effective** against other types of malicious software.

Infection strategies

To better understand how viruses are detected and recognized, it is essential to divide them by their infection ways.

- **Nonresident viruses** The simplest form of viruses which don't stay in memory, but infect founded executable file and search for another to replicate.
- **Resident viruses** More complex and efficient type of viruses which stay in memory and hide their presence from other processes. Kind of TSR apps.
  - *Fast infectors* Type which is designed to infect as many files as possible.
  - *Slow infectors* Using stealth and encryption techniques to stay undetected outlast.

Infection strategies

To better understand how viruses are detected and recognized, it is essential to divide them by their infection ways.

- **Nonresident viruses** The simplest form of viruses which don't stay in memory, but infect founded executable file and search for another to replicate.
- **Resident viruses** More complex and efficient type of viruses which stay in memory and hide their presence from other processes. Kind of TSR apps.
  - *Fast infectors* Type which is designed to infect as many files as possible.
  - *Slow infectors* Using stealth and encryption techniques to stay undetected outlast.

Infection strategies

To better understand how viruses are detected and recognized, it is essential to divide them by their infection ways.

- **Nonresident viruses** The simplest form of viruses which don't stay in memory, but infect founded executable file and search for another to replicate.
- **Resident viruses** More complex and efficient type of viruses which stay in memory and hide their presence from other processes. Kind of TSR apps.
  - *Fast infectors* Type which is designed to infect as many files as possible.
  - *Slow infectors* Using stealth and encryption techniques to stay undetected outlast.

Infection strategies

To better understand how viruses are detected and recognized, it is essential to divide them by their infection ways.

- **Nonresident viruses** The simplest form of viruses which don't stay in memory, but infect founded executable file and search for another to replicate.
- **Resident viruses** More complex and efficient type of viruses which stay in memory and hide their presence from other processes. Kind of TSR apps.
  - *Fast infectors* Type which is designed to infect as many files as possible.
  - *Slow infectors* Using stealth and encryption techniques to stay undetected outlast.

Metaheuristic

Metaheuristic is a heuristic method for solving a **very general class of computational problems** by combining user-given black-box procedures in a **hopefully efficient way**. Metaheuristics are generally applied to problems for which there is **no satisfactory problem-specific algorithm** or heuristic.

Source: *http://en.wikipedia.org/wiki/Metaheuristic*

Heuristic

Heuristic is a method to help solve a problem, commonly an informal method. It is particularly used to **rapidly come to a solution** that is **reasonably close to the best possible answer**, or 'optimal solution'...

Source: *http://en.wikipedia.org/wiki/Heuristic*

General metaheuristics

It is important to remember that metaheuristics are only 'ideas' to solve a problem not a specific way to do that. List below shows main metaheuristics used for virus detection and recognition:

- Pattern matching
- Automatic learning
- Environment emulation
- Neural networks
- Data mining
- Bayes networks
- Hidden Markov models

and other...

General metaheuristics

It is important to remember that metaheuristics are only 'ideas' to solve a problem not a specific way to do that. List below shows main metaheuristics used for virus detection and recognition:

- Pattern matching
- Automatic learning
- Environment emulation
- Neural networks
- Data mining
- Bayes networks
- Hidden Markov models

and other...

General metaheuristics

It is important to remember that metaheuristics are only 'ideas' to solve a problem not a specific way to do that. List below shows main metaheuristics used for virus detection and recognition:

- Pattern matching
- Automatic learning
- Environment emulation
- Neural networks
- Data mining
- Bayes networks
- Hidden Markov models

and other...

General metaheuristics

It is important to remember that metaheuristics are only 'ideas' to solve a problem not a specific way to do that. List below shows main metaheuristics used for virus detection and recognition:

- Pattern matching
- Automatic learning
- Environment emulation
- Neural networks
- Data mining
- Bayes networks
- Hidden Markov models

and other...

General metaheuristics

It is important to remember that metaheuristics are only 'ideas' to solve a problem not a specific way to do that. List below shows main metaheuristics used for virus detection and recognition:

- Pattern matching
- Automatic learning
- Environment emulation
- Neural networks
- Data mining
- Bayes networks
- Hidden Markov models

and other...

General metaheuristics

It is important to remember that metaheuristics are only 'ideas' to solve a problem not a specific way to do that. List below shows main metaheuristics used for virus detection and recognition:

- Pattern matching
- Automatic learning
- Environment emulation
- Neural networks
- Data mining
- Bayes networks
- Hidden Markov models

and other...

General metaheuristics

It is important to remember that metaheuristics are only 'ideas' to solve a problem not a specific way to do that. List below shows main metaheuristics used for virus detection and recognition:

- Pattern matching
- Automatic learning
- Environment emulation
- Neural networks
- Data mining
- Bayes networks
- Hidden Markov models

and other...

General metaheuristics

It is important to remember that metaheuristics are only 'ideas' to solve a problem not a specific way to do that. List below shows main metaheuristics used for virus detection and recognition:

- Pattern matching
- Automatic learning
- Environment emulation
- Neural networks
- Data mining
- Bayes networks
- Hidden Markov models

and other...

Concrete heuristics

Specific heuristics practically used in virus detection and recognition, are naturally inherited from metaheuristics.
And so, for example concrete method for virus detection using neural networks can be implementation of SOM (Self Organizing Map).

Neural Networks (metaheuristic) $\rightarrow$ **SOM** (heuristic)

The most popular, and one of most efficient heuristic used by anti-virus software is technique called **Heuristic Scanning**.

Lacks in specific detection I

Great deal of modern viruses are only slightly changed versions of few conceptions developed years ago. Specific detection methods like signature scanning became very efficient ways of detecting known threats. Finding specific signature in code allows scanner to recognize every virus which signature has been stored in built-in database.

BB ?2 B9 10 01 81 37 ?2 81 77 02 ?2 83 C3 04 E2 F2

FireFly virus signature(hexadecimal)

Lacks in specific detection II

Problem occurs when virus source is changed by a programmer or mutation engine. Signature is being malformed due to even minor changes. Virus may behave in an exactly same way but is undetectable due to new, unique signature.

BB ?2 B9 10 01 81 37 ?2 81 A1 D3 ?2 01 C3 04 E2 F2

Malformed signature(hexadecimal)

Heuristic scanning conception I

Q: How to recognize a virus without any knowledge about its internal structure?

Heuristic scanning conception I

Q: How to recognize a virus without any knowledge about its internal structure?

A: By examining its behaviour and characteristics.

Heuristic scanning conception II

Heuristic scanning in its basic form is implementation of three metaheuristics:

1. Pattern matching
2. Automatic learning
3. Environment emulation

Of course modern solutions provide more functionalities but principle stays the same.

Heuristic scanning conception II

Heuristic scanning in its basic form is implementation of three metaheuristics:

1. Pattern matching
2. Automatic learning
3. Environment emulation

Of course modern solutions provide more functionalities but principle stays the same.

Heuristic scanning conception II

Heuristic scanning in its basic form is implementation of three metaheuristics:

1. Pattern matching
2. Automatic learning
3. Environment emulation

Of course modern solutions provide more functionalities but principle stays the same.

Heuristic scanning conception II

Heuristic scanning in its basic form is implementation of three metaheuristics:

1. Pattern matching
2. Automatic learning
3. Environment emulation

Of course modern solutions provide more functionalities but principle stays the same.

Heuristic scanning conception III

The basic idea of heuristic scanning is to examine assembly language instruction sequences(step-by-step) and qualify them by their potential harmfulness. If there are sequences behaving suspiciously, program can be qualified as a virus. The phenomenon of this method is that it actually **detects threats that aren't yet known!**

Recognizing potential threat I

In real anti-virus software, heuristic scanning is implemented to **recognize threats by following built-in rules**, e.g. if program tries to format hard drive its behaviour is highly suspicious but it can be only simple disk utility. **Singular suspicion is never a reason to trigger the alarm**. But if the same program also tries to stay resident and contains routine to search for executables, it is highly probable that it's a real virus. AV software very often classifies sequences by their behaviour granting them a flag. **Every flag has its weight, if total values for one program exceeds a predefined threshold, scanner regards it as virus.**

Heuristic Scanning as artificial neuron

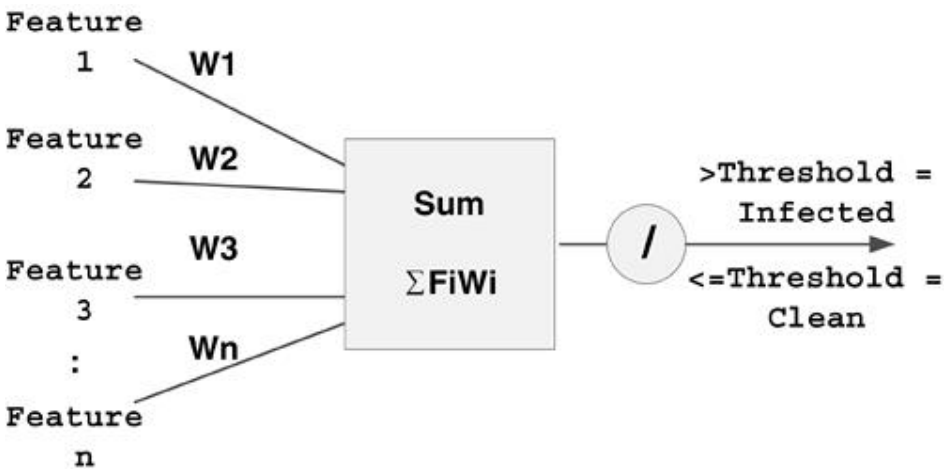


Figure: Single-layer classifier with threshold From [1]

Recognizing potential threat II

```
F = Suspicious file access. Might be able to infect a file.
R = Relocator. Program code will be relocated in a suspicious way.
A = Suspicious Memory Allocation. The program uses a non-standard way to search for, and/or allocate memory.
N = Wrong name extension. Extension conflicts with program structure.
S = Contains a routine to search for executable (.COM or .EXE) files.
# = Found an instruction decryption routine. This is common for viruses but also for some protected software.
E = Flexible Entry-point. The code seems to be designed to be linked on any location within an executable file. Common for viruses.
L = The program traps the loading of software. Might be a virus that intercepts program load to infect the software.
D = Disk write access. The program writes to disk without using DOS.
M = Memory resident code. This program is designed to stay in memory.
! = Invalid opcode (non-8088 instructions) or out-of-range branch.
T = Incorrect timestamp. Some viruses use this to mark infected files.
J = Suspicious jump construct. Entry point via chained or indirect jumps. This is unusual for normal software but common for viruses.
? = Inconsistent exe-header. Might be a virus but can also be a bug.
G = Garbage instructions. Contains code that seems to have no purpose other than encryption or avoiding recognition by virus scanners.
U = Undocumented interrupt/DOS call. The program might be just tricky but can also be a virus using a non-standard way to detect itself.
Z = EXE/COM determination. The program tries to check whether a file is a COM or EXE file. Viruses need to do this to infect a program.
O = Found code that can be used to overwrite/move a program in memory.
B = Back to entry point. Contains code to re-start the program after modifications at the entry-point are made. Very usual for viruses.
K = Unusual stack. The program has a suspicious stack or an odd stack.
```

Figure: TbScan 6.02 heuristic flags From [3]

For instance Jerusalem/PLO virus would raise *FRLMUZ* flags.

Malware evolves

After presenting specific scanning to AV software, malware authors were obligated to introduce new techniques of being undetected. Beside of polymorphism and mutation engines viruses started to use various **stealth techniques** which basically hooked interrupts and took control over them. This allowed them to be invisible for traditional scanner. Moreover, most of them started using **real-time encryption** which made them look like totally harmless program.

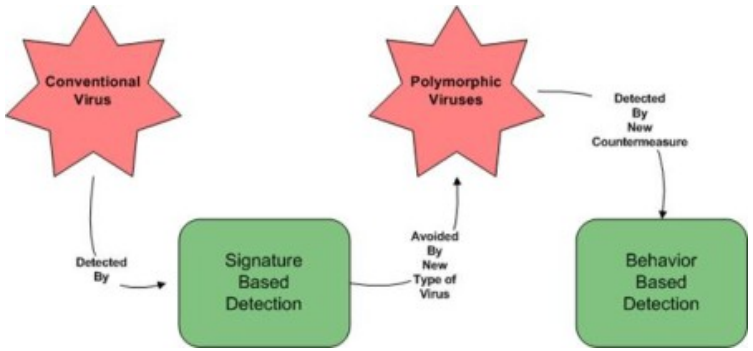
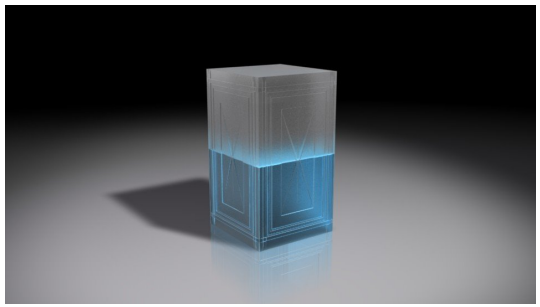


Figure: Virus evolution chain [Source http://searchsecurity.techtarget.com]

Pattern matching is not enough

Mixing stealth techniques with encryption and anti-heuristic sequences (code obfuscated by meaningless instructions) allowed viruses to be unseen even by signature and heuristic scanning combined together.
It was obvious that new solution was needed. The idea came from VM conceptions. Why not to create **artificial runtime environment** to let the virus do its job?
Such approach found implementation in **environment emulation engines**, which became standard AV software weapon.



Virtual reality

The idea of environment emulation is simple. Anti-virus program provides a **virtual machine with independent operating system** and allows virus to perform its routines. Behaviour and characteristics are being **continuously examined**, while **virus is not aware** that is working on a fake system.
This leads to decryption routines and **revealment of its true nature**. Also stealth techniques are useless because whole VM is monitored by AV software.

False positives & automatic learning

HS as a heuristic method is only *reasonably close to the best possible answer*. In this case we can imagine that heuristic scanning will blame innocent programs for being potential threats. Such behaviour is called **false positive**

We must be aware that program is right when rising alarm, because scanned app posses suspicious sequences, we can't blame scanner for failure. So what can be done to avoid false positives?

False positives & automatic learning

HS as a heuristic method is only *reasonably close to the best possible answer*. In this case we can imagine that heuristic scanning will blame innocent programs for being potential threats. Such behaviour is called **false positive**
We must be aware that program is right when rising alarm, because scanned app posses suspicious sequences, we can't blame scanner for failure. So what can be done to avoid false positives?

A: Automatic learning!

Avoiding false positives & improving accuracy I

Some applications, especially non-commercial ones, can raise false positive, because of their suspicious routines. e.g. *UnHash v1.0* through its encryption functionalities (used for finding hash collisions) almost every time is qualified as virus. What we can do to prevent it, is to:

1. **Let Monitor learn**
   Teach AV monitor to recognize programs causing false positives. (requires advanced user)

Avoiding false positives & improving accuracy I

Some applications, especially non-commercial ones, can raise false positive, because of their suspicious routines. e.g. *UnHash v1.0* through its encryption functionalities (used for finding hash collisions) almost every time is qualified as virus. What we can do to prevent it, is to:

1. **Let Monitor learn**
   Teach AV monitor to recognize programs causing false positives. (requires advanced user)

Avoiding false positives & improving accuracy II

2. **Set proper scanning depth**
   Configure Monitor with suitable heuristic scanning depth by manipulating threshold computed from flag weights.
3. **Assume that machine is not infected**
   Some AV software can scan through computer knowing it's clean and learn which programs are false positives.
4. **Combine scanning techniques**
   Combine multiple scanning techniques to exclude potential false positives.
5. **Perform scan as often as possible**
   Knowing what's going on is essential...

Avoiding false positives & improving accuracy II

2. **Set proper scanning depth**
   Configure Monitor with suitable heuristic scanning depth by manipulating threshold computed from flag weights.
3. **Assume that machine is not infected**
   Some AV software can scan through computer knowing it's clean and learn which programs are false positives.
4. **Combine scanning techniques**
   Combine multiple scanning techniques to exclude potential false positives.
5. **Perform scan as often as possible**
   Knowing what's going on is essential...

Avoiding false positives & improving accuracy II

2. **Set proper scanning depth**
   Configure Monitor with suitable heuristic scanning depth by manipulating threshold computed from flag weights.
3. **Assume that machine is not infected**
   Some AV software can scan through computer knowing it's clean and learn which programs are false positives.
4. **Combine scanning techniques**
   Combine multiple scanning techniques to exclude potential false positives.
5. **Perform scan as often as possible**
   Knowing what's going on is essential...

Avoiding false positives & improving accuracy II

2. **Set proper scanning depth**
   Configure Monitor with suitable heuristic scanning depth by manipulating threshold computed from flag weights.
3. **Assume that machine is not infected**
   Some AV software can scan through computer knowing it's clean and learn which programs are false positives.
4. **Combine scanning techniques**
   Combine multiple scanning techniques to exclude potential false positives.
5. **Perform scan as often as possible**
   Knowing what's going on is essential...

Avoiding false positives & improving accuracy II

2. **Set proper scanning depth**
   Configure Monitor with suitable heuristic scanning depth by manipulating threshold computed from flag weights.
3. **Assume that machine is not infected**
   Some AV software can scan through computer knowing it's clean and learn which programs are false positives.
4. **Combine scanning techniques**
   Combine multiple scanning techniques to exclude potential false positives.
5. **Perform scan as often as possible**
   Knowing what's going on is essential...

Presenting **Panda Security** solutions

Modern Panda Security solutions belong to the third generation AV software.

- **First Generation: Antivirus**
  From 1990's, signature scanning including polymorphic virus recognition. Primitive heuristics.
- **Second Generation: Antimalware**
  From 2000, integrated firewall, anti-malware engine.
- **Third Generation: Proactive Technologies**
  From 2004, TruPrevent®, genetic and rootkit heuristics, behavioral analysis and blocking, uncloaking techniques, generic unpacking

Modern Panda Security solutions belong to the third generation AV software.

- **First Generation: Antivirus**
  From 1990's, signature scanning including polymorphic virus recognition. Primitive heuristics.
- **Second Generation: Antimalware**
  From 2000, integrated firewall, anti-malware engine.
- **Third Generation: Proactive Technologies**
  From 2004, TruPrevent®, genetic and rootkit heuristics, behavioral analysis and blocking, uncloaking techniques, generic unpacking

Modern Panda Security solutions belong to the third generation AV software.

- **First Generation: Antivirus**
  From 1990's, signature scanning including polymorphic virus recognition. Primitive heuristics.
- **Second Generation: Antimalware**
  From 2000, integrated firewall, anti-malware engine.
- **Third Generation: Proactive Technologies**
  From 2004, TruPrevent®, genetic and rootkit heuristics, behavioral analysis and blocking, uncloaking techniques, generic unpacking

Modern Panda Security solutions belong to the third generation AV software.

- **First Generation: Antivirus**
  From 1990's, signature scanning including polymorphic virus recognition. Primitive heuristics.
- **Second Generation: Antimalware**
  From 2000, integrated firewall, anti-malware engine.
- **Third Generation: Proactive Technologies**
  From 2004, TruPrevent®, genetic and rootkit heuristics, behavioral analysis and blocking, uncloaking techniques, generic unpacking

Modern Panda Security solutions belong to the third generation AV software.

- **First Generation: Antivirus**
  From 1990's, signature scanning including polymorphic virus recognition. Primitive heuristics.
- **Second Generation: Antimalware**
  From 2000, integrated firewall, anti-malware engine.
- **Third Generation: Proactive Technologies**
  From 2004, TruPrevent®, genetic and rootkit heuristics, behavioral analysis and blocking, uncloaking techniques, generic unpacking

Genetic Heuristic Engine, codename: Nereus was initially released in 2005. The innovation connected with Nereus rely on idea inspired by the field of genetics and its usefulness to understand how organisms are individually identified and associated to other organisms. Features:

- More than **few hundred characteristics** of each file that is scanned
- Complex malware recognition (**type determination**)
- **Rootkit heuristics** (time based analysis)
- Generic packer detectors and generic unpacking algorithms
- New threat automatic notification
- Automatic **creation of detection and disinfection signatures** for samples previously analyzed by processing and classification module

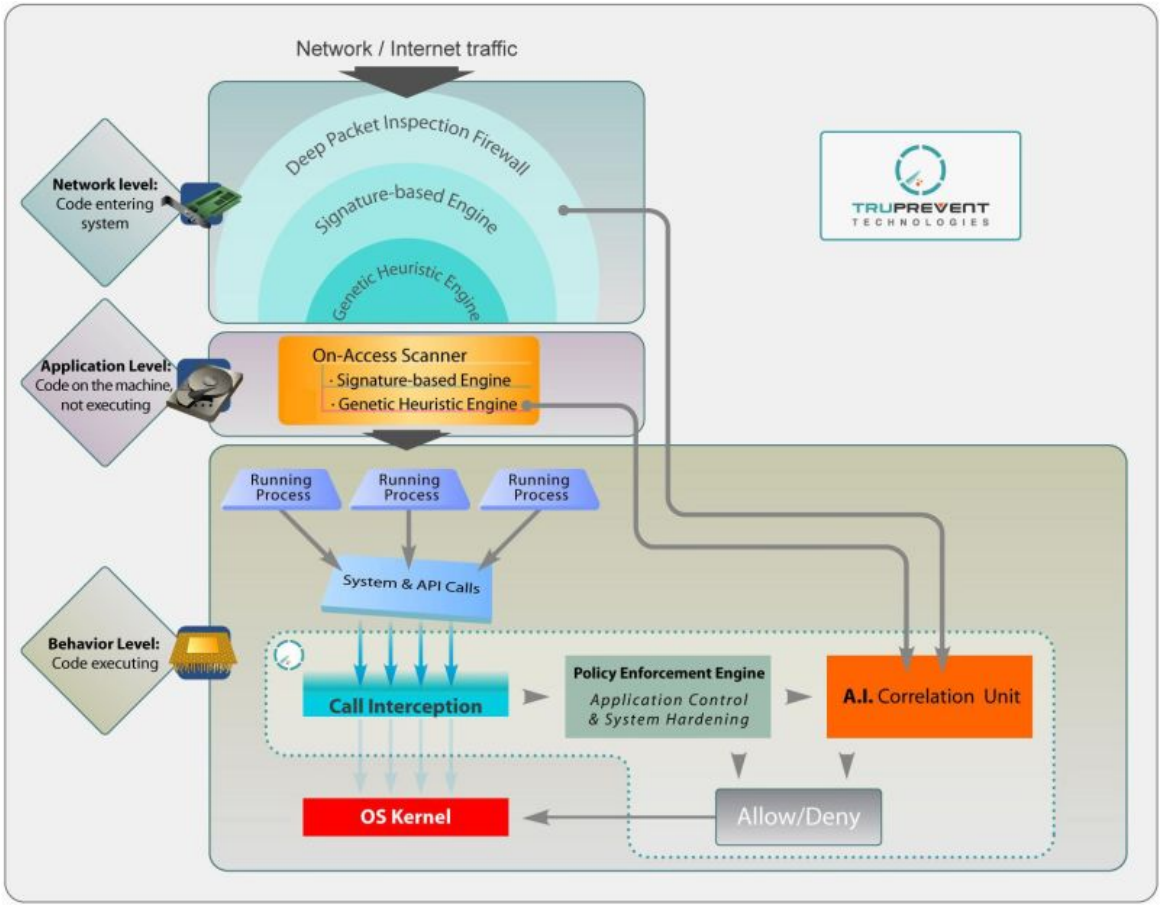


Figure: Panda's integrated endpoint security From [6]

Heuristic scanning is the most popular heuristic method for virus detection and recognition. Basically it is inherited from combination of pattern matching, automatic learning and environment emulation metaheuristics. As a heuristic method it's not 100% effective. So why do we apply HS?

Pros

- Can detect 'future' threats
- User is less dependent on product update
- Improves conventional scanning results

Cons

- False positives
- Making decision after alarm requires knowledge


Peter Szor
The Art of Computer Virus Research and Defense
Addison-Wesley Professional, 1st edition, February 2005


Tomasz Andel, Krzysztof Zawadzki
Techniki pisania wirusów oraz antywirusów
Inynieria bezpieczeństwa systemów sieciowych i internetowych, PWr Wroclaw 2008


Frans Veldman
Heuristic Anti-Virus Technology
http://mirror.sweon.net/madchat/vxdevl/vdat/epheurs1.htm


Richard Zwienenberg
Heuristic Scanners: Artificial Intelligence?
http://mirror.sweon.net/madchat/vxdevl/vdat/epheurs2.htm


edited by Éric Filiol
Journal in Computer Virology
Springer Paris, Volume 1-4
http://www.springerlink.com/content/119769


Panda Research
From Traditional Antivirus to Collective Intelligence
August 2007
http://research.pandasecurity.com/

Various online knowledge repositories

For starters it's good to search wikipedia...

VX Heavens http://vx.netlux.org/lib

Breaking Business and Technology News https://silicon.com/

IEEE http://ieeexplore.ieee.org/

Zines: Phalcon/Skism: 40HEX, VLAD, VBB: Viruses Bits & Bytes, Immortal Riot: Insane Reality, NuKe: NuKe IntoJournal, Dark Angel VirGuide

AND OTHER...

Why?...

Questions ?

What if?...

THANK YOU