清华大学出版社
TSINGHUA UNIVERSITY PRESS

重点大学信息安全专业规划系列教材

恶意代码与计算机病毒
——原理、技术和实践

第2章 计算机病毒理论模型

刘功申
上海交通大学网络空间安全学院

本章学习目标

- 掌握计算机病毒的抽象描述
- 掌握基于图灵机的计算机病毒模型
- 掌握基于递归函数的计算机病毒模型
- 掌握网络蠕虫传播模型
- 掌握计算机病毒预防理论模型

虚拟案例

- 一个文本编辑程序被病毒感染了。每当使用文本编辑程序时，它总是先进行感染工作并执行编辑任务，其间，它将搜索合适文件以进行感染。每一个新被感染的程序都将执行原有的任务，并且也搜索合适的程序进行感染。这种过程反复进行。当这些被感染的程序跨系统传播，被销售，或者送给其他人时，将产生病毒扩散的新机会。最终，在1990年1月1日以后，被感染的程序终止了先前的活动。现在，每当这样的一个程序执行时，它将删除所有文件。

计算机病毒伪代码

```
{main:=
                Call injure;
                …
                Call submain;
                …
                Call infect;
}
{injure:=
        If condition then whatever damage is to be done and halt;
}
{infect:=
        If condition then infect files;
}
```

案例病毒的伪代码

```
{main:=
        Call injure;
        Call submain;
        Call infect;
}
{injure:=
        If date>= Jan. 1,1990 then
                While file != 0
                        File = get-random-file;
                        Delete file;
                Halt;
}
```

- {infect:=
- If true then
- File = get-random-executable-file;
- Rename main routine submain;
- Prepend self to file;
- }

精简后的伪代码(压缩或变型)

- {main:=
- Call injure;
- Decompress compressed part of program;
- Call submain;
- Call infect;
- }
- {injure:=
- If false then halt;
- }

- {infect:=
- If executable != 0 then
- File = get-random-executable-file;
- Rename main routine submain;
- Compress file;
- Prepend self to file;
  }

病毒伪代码的共同性质

- 1．对于每个程序，都存在该程序相应的感染形式。也就是，可以把病毒看作是一个程序到一个被感染程序的映射。
- 2．每一个被感染程序在每个输入（输入是指可访问信息，例如，用户输入，系统时钟，数据或程序文件等）上形成如下3个选择：

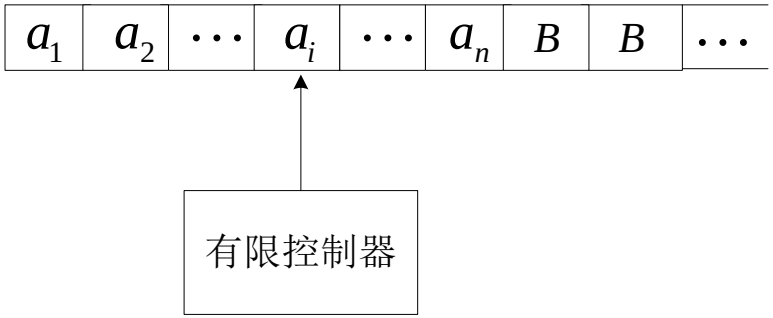
- 破坏(Injure)：不执行原先的功能，而去完成其它功能。何种输入导致破坏以及破坏的形式都与被感染的程序无关，而只与病毒本身有关。
- 传染(Infect)：执行原先的功能，并且，如果程序能终止，则传染程序。对于除程序以外的其它可访问信息（如时钟、用户/程序间的通信）的处理，同感染前的原程序一样。另外，不管被感染的程序其原先功能如何（文本编辑或编译器等），它传染其它程序时，其结果是一样的。也就是说，一个程序被感染的形式与感染它的程序无关。
- 模仿(Imitate)：既不破坏也不传染，不加修改地执行原先的功能。这也可看作是传染的一个特例，其中被传染的程序的个数为零。

清华大学出版社
TSINGHUA UNIVERSITY PRESS

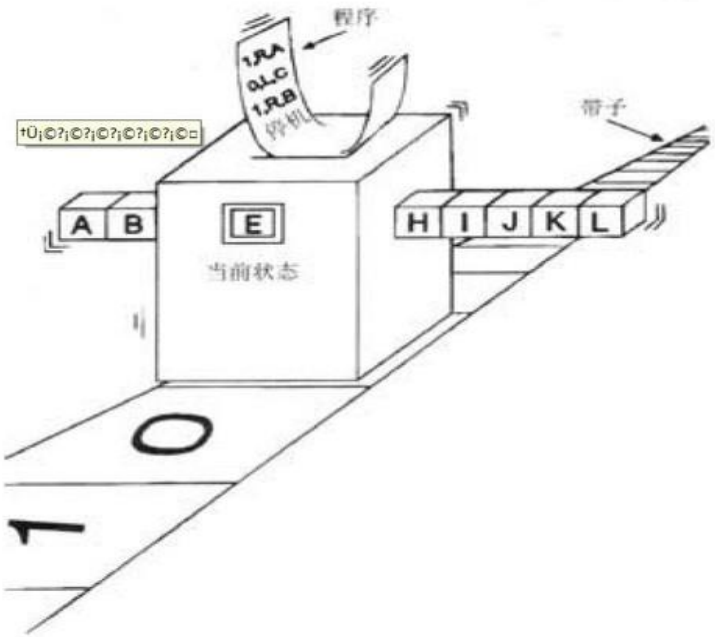
重点大学信息安全专业规划系列教材

基于图灵机的计算机病毒的计算模型

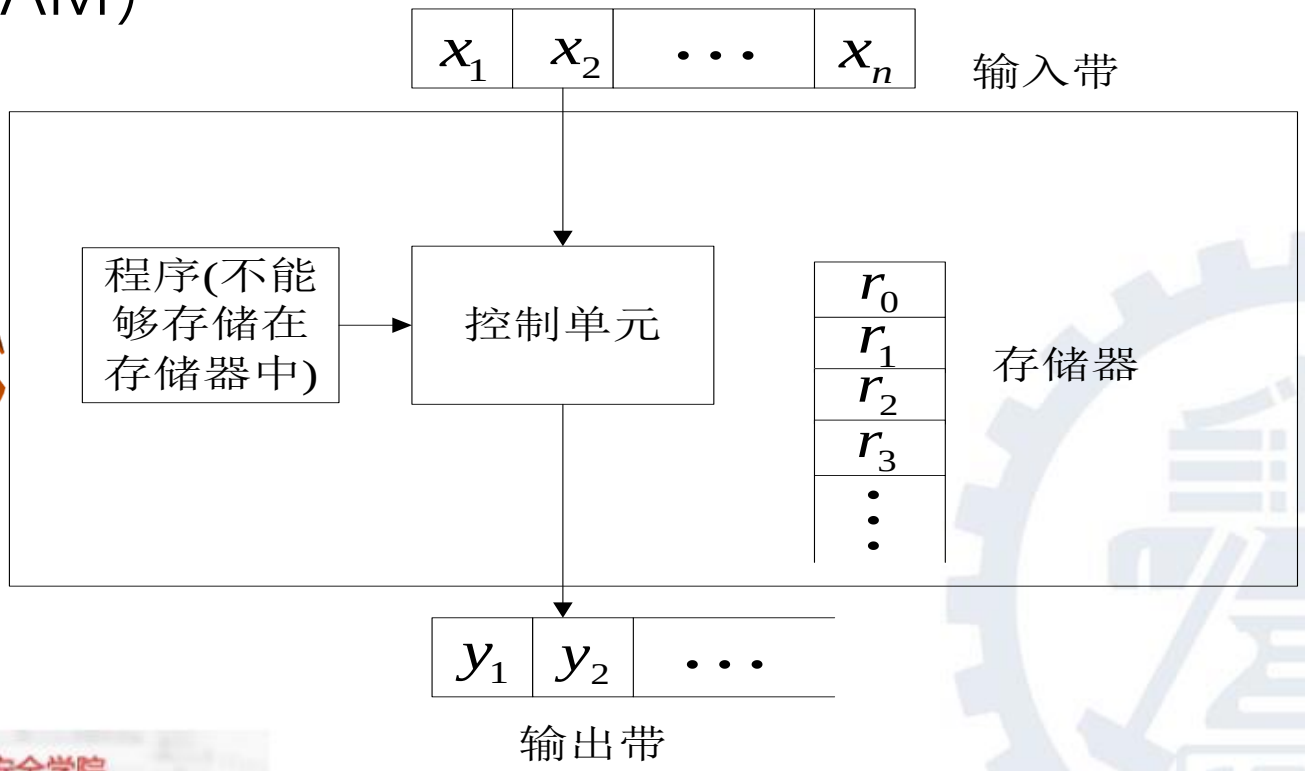
恶意代码与计算机病毒
——原理、技术和实践



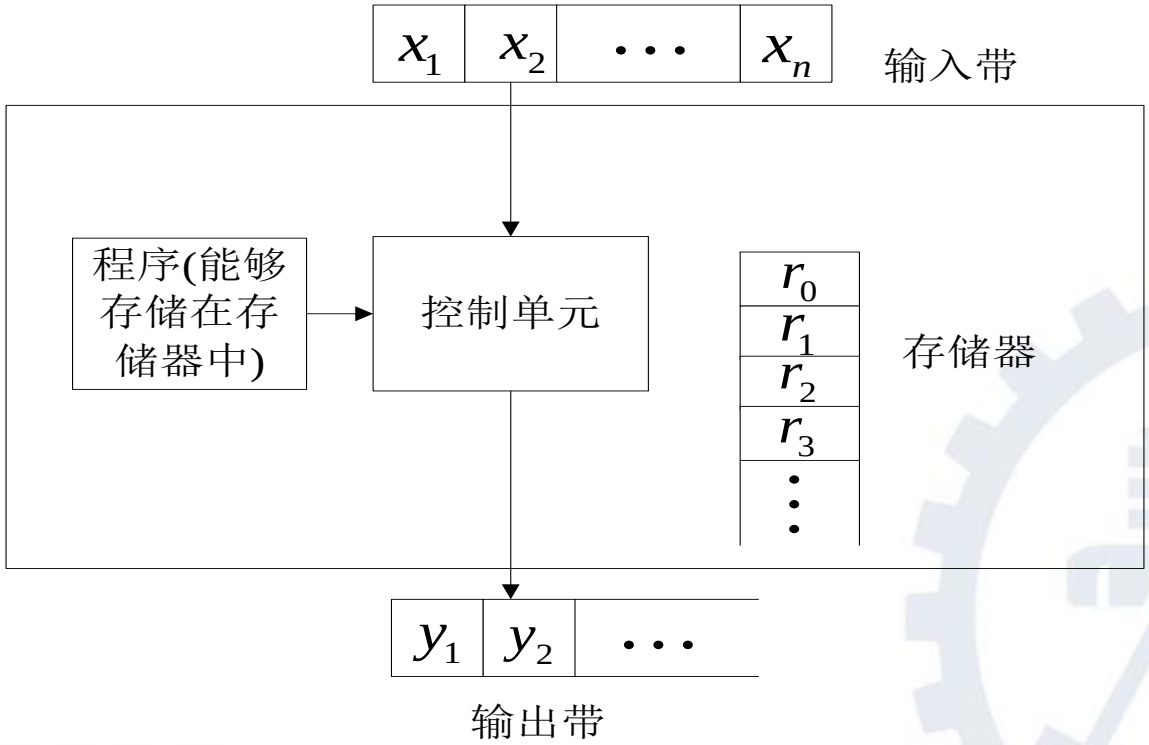
- 基本图灵机(TM)
- 图灵机的经典问题：
  - 图灵机停机问题
  - 图灵机存在不可计算数



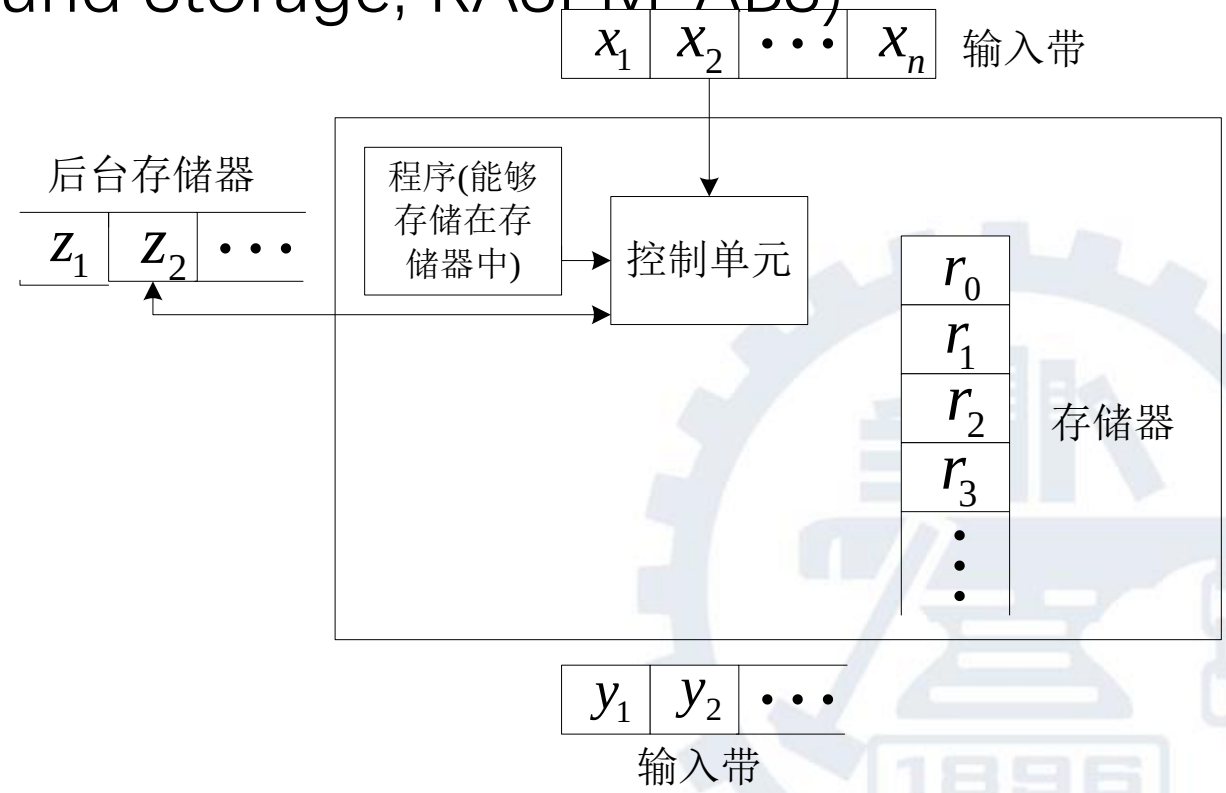
- 随机访问计算机（Random Access Machine —— RAM）



- 随机访问存储程序计算机（Ramdom Access Stored Program Machine,RASPM）



- 附带后台存储带的随机访问存储程序计算机(The Random Access Stored Program Machine with Attached Background Storage, RASPM_ABS)



现在的计算机

- **基于RASPM_ABS的病毒**
  - 计算机病毒被定义成程序的一部分，该程序附着在某个程序上并能将自身链接到其他程序上。当病毒所附着的程序被执行时，计算机病毒的代码也跟着被执行。

- 1.病毒的传播模型
  - 如果病毒利用了计算机的一些典型特征或服务，那么病毒的这种传播方式被称作专用计算机的传播方式。如果病毒在传播时没有利用计算机的服务，那么此传播方式被称为独立于计算机的传播方式。
  - PC中，引导型病毒就具有专用计算机的传播方式
  - 感染C源文件的病毒就是具有独立计算机的传播方式

- 2.少态型病毒和多态型病毒
  - 当有两个程序被同样的病毒以指定传播方式感染，并且病毒程序的代码顺序相同时，这种传播方式称为少形态的。
  - 当有两个程序被同样的病毒以指定传播方式感染，并且病毒程序的代码顺不同时，这种传播方式称为多形态的。病毒代码的全部或部分被使用不同的密钥加密是多态的一种特殊形式。

- 多态型病毒的实现要比少态型病毒的实现复杂得多，它们能改变自身的译码部分。两种实现方式：
- 1）通过从准备好的集合中任意选取译码程序。
- 2）通过在传播期间随机产生程序指令来完成。例如，可以通过如下的方法来实现：
  - 改变译码程序的顺序；
  - 处理器能够通过一个以上的指令（序列）来执行同样的操作；
  - 向译码程序中随机地放入哑命令（Dummy Command）。

- 3.病毒检测的一般问题
  - 如果存在着某一能够解决病毒检测问题的算法，那么就能通过建立图灵机来执行相应的算法。不幸的是，即使在最简单的情况下，我们也不可能制造出这样的图灵机。
  - 定理：不可能制造出一个图灵机，利用该计算机，我们能够判断RASPM_ABS中的可执行文件是否含有病毒。

- 4.病毒检测方法
  - 如果我们只涉及一些已知病毒的问题，那么就可能简化病毒检测问题。在此情况下，可以将已知病毒用在检测算法上。
  - 我们从每个已知病毒提取一系列代码，当病毒进行传播时，它们就会在每个被感染了的文件中显示出来。我们将这一系列代码成为序列。病毒检测程序的任务就是在程序中搜寻这些序列。

- 检测多态型病毒的难点
  - 不能确定多态型病毒是否含有某些序列，能够通过这些序列可以检测病毒的所有变异。
  - 当发现序列是随机的时，不知道发生错误报警的概率。发现任意序列的概率：

$$P \approx L \cdot M \cdot \frac{1}{n^N}$$

  - N表示一个序列的长度；
  - M表示序列的总个数；用L(L>>N)表示被检测文件的总长度；
  - n是字符集大小（对应二进制代码为16）
  - 该采用什么样的费用标准来衡量序列搜寻算法的实现。

论文查重复？？？

基于递归函数的计算机病毒的数学模型

- Adlemen给出的计算机病毒形式定义：
- （1）S表示所有自然数有穷序列的集合。
- （2）e表示一个从S╳S到N的可计算的入射函数，它具有可计算的逆函数。
- （3）对所有的s,t∈S,用<s,t>表示e（s,t）。
- （4）对所有部分函数f：N→N及所有s,t∈S,用f(s,t)表示f(<s,t>)。
- （5）e′表示一个从N╳N到N的可计算的入射函数，它具有可计算的逆函数，并且对所有i,j∈N，e′(i,j)≥i。

- （6）对所有i,j∈N，<i,j>表示e′(i,j)。
- （7）对所有部分函数f：N→N及所有i,j∈N,f(i,j)表示f(<i,j>)。
- （8）对所有部分函数f：N→N及所有n∈N,f(n)↓表示f(n)是有定义的。
- （9）对所有部分函数f：N→N及所有n∈N,f(n)↑表示f(n)是未定义的。

Adlemen病毒模型有如下缺陷：

- (1)计算机病毒的面太广。 不具传染性的也当作病毒
- (2)定义并没有反映出病毒的传染特性。
- (3)定义不能体现出病毒传染的传递特性。
- (4)“破坏”的定义不合适。 原程序功能保留不明确

Internet蠕虫传播模型

- SI(Susceptible[易受感染的]-Infected)
- SIS(Susceptible-Infected-Susceptible)
- SIR(Susceptible-Infected-Removed)

SIS模型和SI模型

- 某种群中不存在流行病时，其种群（N）的生长服从微分系统。

$$N' = be^{-\alpha N} N - dN$$

- 其中 $N = N(t)$ 表示t时刻该环境中总种群的个体数量，
- $be^{-\alpha N}$ 表示种群中单位个体的生育率，
- d表示单位个体的自然死亡率。

- 有疾病传播时的模型

- S，I分别表示易感者类和染病者类
- β表示一个染病者所具有的最大传染力
- r表示疾病的恢复力
- d表示自然死亡率
- a表示额外死亡率

- 流行病的传播服从双线形传染率的SIS模型

$$\begin{cases} S' = be^{-\alpha N} N - \beta SI - dS + rI \\ I' = \beta SI - (d + a + r)I \end{cases}$$

- 总种群的生长为：

$$N(t) = S(t) + I(t)$$

$$N' = be^{-\alpha N} N - dN - \alpha I$$

在SIS模型中，当a=0时，该模型变为SI模型。

- SIR模型
  - 两个假设：
    - 已被病毒感染的文件（档）具有免疫力。
    - 病毒的潜伏期很短，近似地认为等于零。
  - 把系统中可执行程序分为三种：
    - 被传播对象，即尚未感染病毒的可执行程序，用S(t)表示其数目。
    - 带菌者，即已感染病毒的可执行程序，用p(t)表示其数目。
    - 被感染后具有免疫力的可执行程序，也包括被传播后在一定时间内不会运行的可执行程序（相当患病者死去），用R(t)表示其数目。

$$\begin{cases} \dfrac{dS}{dt} = -\lambda \bar{k} pS \\ \dfrac{dR}{dt} = \mu p \end{cases}$$

- λ 表示传播（感染）速度；$\bar{k}$ 表示每个时间段接触次数；u 表示第 II 类程序变成第 III 类程序的速度；
- 公式的解释：
  - (1) S(t)的变化率即经第 I 类程序变成第 II 类程序的变化率，它与传染者和被传染者之间的接触次数有关，并且正比于这两类文件的乘积。
  - (2) R(t)的变化率即第 II 类程序变成第 III 类程序的变化率，与当时第 II 类的可执行程序数目成正比。
  - (3) 在考虑的时间间隔内，系统内可执行程序的总数变化不大，并且假设它恒等于常数（即没有文件被撤消，也没有外面的新文件进来），从而可执行程序总数的变化率为零。

$$S(t)+\rho(t)+R(t)=N$$

$$\frac{d\rho}{dt}=-\frac{dS}{dt}-\frac{dR}{dt}$$

$$\longrightarrow \begin{cases} \dfrac{dS}{dt}=-\lambda\bar{k}\rho S \\ \dfrac{d\rho}{dt}=-\mu\rho+\lambda\bar{k}\rho S \\ \dfrac{dR}{dt}=\mu\rho \end{cases}$$

预防理论模型

- Fred.Cohen"四模型"理论
  - (1) 基本隔离模型
    - 该模型的主要思想是取消信息共享，将系统隔离开来，使得计算机病毒既不能从外部入侵进来，也不可能把系统内部的病毒扩散出去。
  - (2) 分隔模型
    - 将用户群分割为不可能互相传递信息的若干封闭子集。由于信息处理流的控制，使得这些子集可被看作是系统被分割成的相互独立的子系统，使得计算机病毒只能感染整个系统中的某个子系统，而不会在子系统之间进行相互传播。

- (3) 流模型
  - 对共享的信息流通过的距离设定一个阀值，使得一定量的信息处理只能在一定的区域内流动，若该信息的使用超过设定的阀值，则可能存在某种危险。
- (4) 限制解释模型
  - 即限制兼容，采用固定的解释模式，就有可能不被计算机病毒感染。

类IPM模型

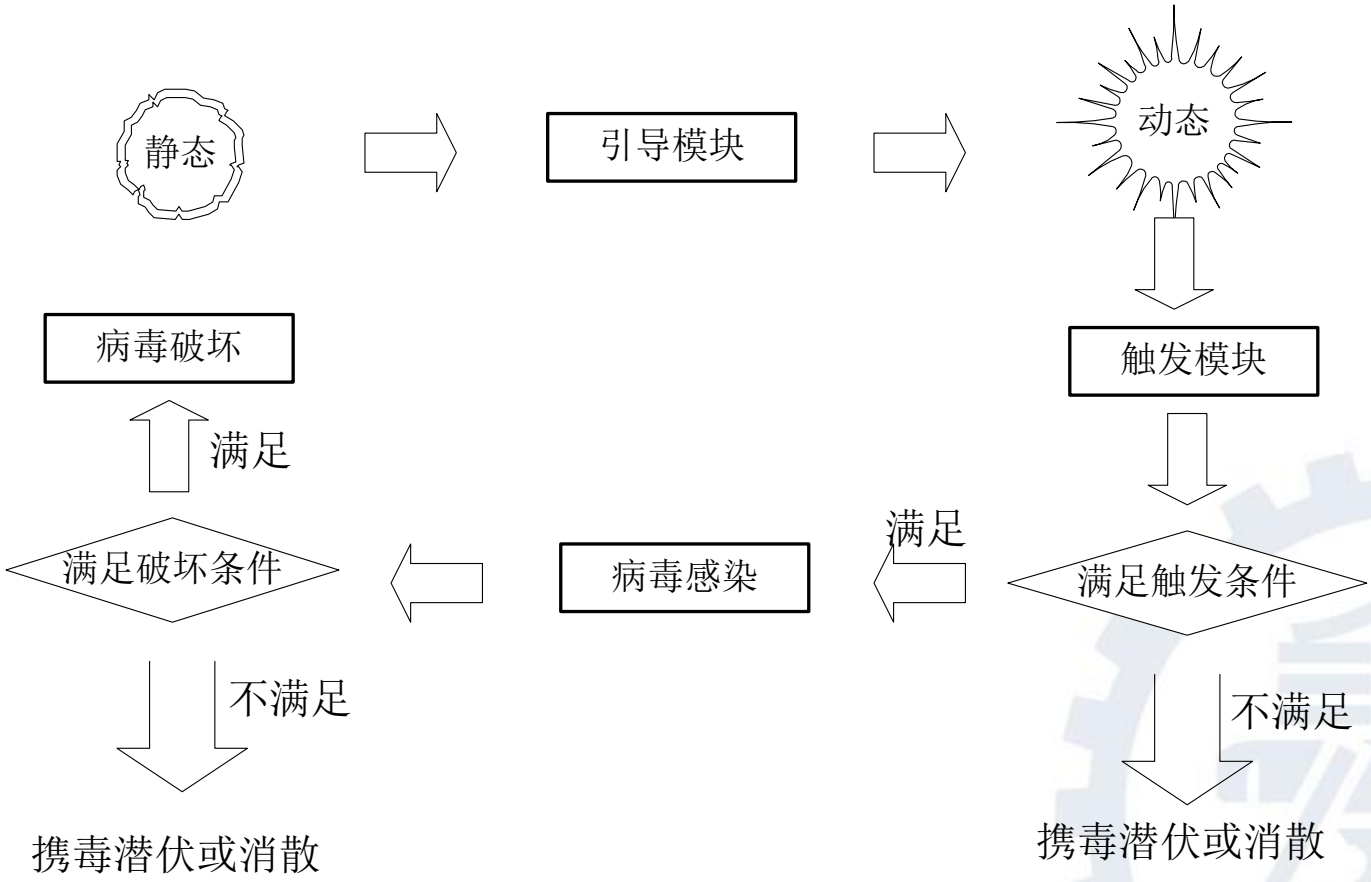
- 把计算机程序或磁盘文件类比为不断生长变化的植物。
- 把计算机系统比作一个由许多植物组成的田园。
- 把计算机病毒看成是侵害植物的害虫。
- 把计算机信息系统周围的环境看作农业事物处理机构。

- 农业上的IPM(Integrated Pest Management)模型实质上是一种综合管理方法。它的基本思想是：一个害虫管理系统是与周围坏境和害虫种类的动态变化有关的。它以尽可能温和的方式利用所有适用技术和措施治理害虫，使它们的种类维持在不足以引起经济损失的水平之下。

计算机病毒的结构和工作机制

- 四大模块：
  - 感染模块
  - 触发模块
  - 破坏模块（表现模块）
  - 引导模块（主控模块）
- 两个状态：
  - 静态
  - 动态

工作机制

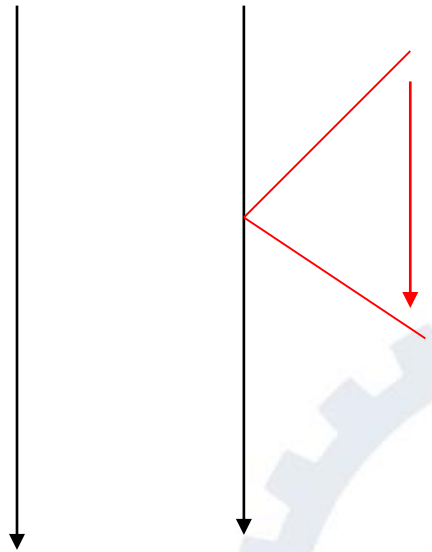

静态
引导模块
动态
触发模块
满足
满足触发条件
不满足
携毒潜伏或消散
病毒感染
满足
满足破坏条件
不满足
携毒潜伏或消散
病毒破坏
满足

引导模块

- 引导前——寄生
  - 寄生位置：
    - 引导区
    - 可执行文件
  - 寄生手段：
    - 替代法（寄生在引导区中的病毒常用该法）
    - 链接法（寄生在文件中的病毒常用该法）

- 引导过程
  - 驻留内存
  - 窃取系统控制权
  - 恢复系统功能
- 引导区病毒引导过程
  - 搬迁系统引导程序-〉替代为病毒引导程序
  - 启动时-〉病毒引导模块-〉加载传染、破坏和触发模块到内存-〉使用常驻技术
  - 最后，转向系统引导程序-〉引导系统

- 文件型病毒引导过程
  - 修改入口指令->替代为跳转到病毒模块的指令
  - 执行时->跳转到病毒引导模块->病毒引导模块->加载传染、破坏和触发模块到内存->使用常驻技术
  - 最后，转向程序的正常执行指令->执行程序



感染模块

- 病毒传染的条件
  - 被动传染（静态时）
    - 用户在进行拷贝磁盘或文件时，把一个病毒由一个载体复制到另一个载体上。或者是通过网络上的信息传递，把一个病毒程序从一方传递到另一方。这种传染方式叫做计算机病毒的被动传染。
  - 主动传染（动态时）
    - 以计算机系统的运行以及病毒程序处于激活状态为先决条件。在病毒处于激活的状态下，只要传染条件满足，病毒程序能主动地把病毒自身传染给另一个载体或另一个系统。这种传染方式叫做计算机病毒的主动传染。

- 传染过程
  - 系统（程序）运行-〉各种模块进入内存-〉按多种传染方式传染
- 传染方式
  - 立即传染，即病毒在被执行的瞬间，抢在宿主程序开始执行前，立即感染磁盘上的其他程序，然后再执行宿主程序。
  - 驻留内存并伺机传染，内存中的病毒检查当前系统环境，在执行一个程序、浏览一个网页时传染磁盘上的程序，驻留在系统内存中的病毒程序在宿主程序运行结束后，仍可活动，直至关闭计算机。

- 文件型病毒传染机理
  - 首先根据病毒自己的特定标识来判断该文件是否已感染了该病毒；
  - 当条件满足时，将病毒链接到文件的特定部位，并存入磁盘中；
  - 完成传染后，继续监视系统的运行，试图寻找新的攻击目标。
- 文件型病毒传染途径
  - 加载执行文件
  - 浏览目录过程
  - 创建文件过程

破坏模块

- 破坏是Vxer的追求，病毒魅力的体现
- 破坏模块的功能
  - 破坏、破坏、还是破坏……
- 破坏对象
  - 系统数据区、文件、内存、系统运行速度、磁盘、CMOS、主板和网络等。
- 破坏的程度

触发模块

- 触发条件
  - 计算机病毒在传染和发作之前，往往要判断某些特定条件是否满足，满足则传染或发作，否则不传染或不发作或只传染不发作，这个条件就是计算机病毒的触发条件。
- 触发模块的目的是调节病毒的攻击性和潜伏性之间的平衡
  - 大范围的感染行为、频繁的破坏行为可能给用户以重创，但是，它们总是使系统或多或少地出现异常，容易使病毒暴露。
  - 而不破坏、不感染又会使病毒失去其特性。
  - 可触发性是病毒的攻击性和潜伏性之间的调整杠杆，可以控制病毒感染和破坏的频度，兼顾杀伤力和潜伏性。

病毒常用的触发条件

- 日期触发
- 时间触发
- 键盘触发
- 感染触发
  - 例如，运行感染文件个数触发、感染序数触发、感染磁盘数触发、感染失败触发等。
- 启动触发
- 访问磁盘次数触发
- CPU型号/主板型号触发

清华大学出版社
TSINGHUA UNIVERSITY PRESS

常见计算机病毒的技术特征

- 驻留内存
- 病毒变种
- EPO（Entry Point Obscuring）技术
- 抗分析技术（加密、反跟踪）
- 隐蔽性病毒技术
- 多态性病毒技术
- 插入型病毒技术
- 超级病毒技术
- 破坏性感染技术
- 网络病毒技术

如果离开这些技术，就只能是实验教学病毒，不具有实战能力

清华大学出版社
TSINGHUA UNIVERSITY PRESS

重点大学信息安全专业规划系列教材

恶意代码与计算机病毒
——原理、技术和实践

谢 谢