


University of Chinese Academy of Sciences

UCAS XGS001CD SPRING 2022 Seminar

Thick Black Theory of Win the interview at the bottom for CS NPEE

计算机考研复试抄底厚黑学

Lecture 4 : Basic of Cyber Security

Jing Li 李敬

2022年3月29日

壬寅二月廿七

北京 · 海淀

中国科学院信息工程研究所

INSTITUTE OF INFORMATION ENGINEERING,CAS

课程介绍 Seminar

- 课程代码：UCAS XGS001CD SPRING 2022 Seminar
- 课程名称：Thick Black Theory of Win the interview at the bottom for CS NPEE，计算机考研复试抄底厚黑学

- As we all know, participating in the NPEE (National Post-graduate Entrance Examination) and getting a good score is only a ticket to the reexamine or interview. There are many cases in which high scorers are eliminated in the retest. It's not that they didn't work hard or are not excellent, but that they didn't find then use the right method. Hence, it is very important to carefully prepare for the interview.
- In this course, we will talk about how to prepare the retest, especially interview. We will explore how to make an effective introduction letter and brand yourself. In particular, we will discuss make an excellent résumé or CV (Curriculum Vitae). We will also discuss how computer scientists and engineers are using machine learning to design state-of-the-art hardware and software security platforms. In particular, we will cover topics such as Artificial Intelligence, Computer System, Network, Open Source and Security. After completing the course, you should be able to appreciate the new trends of using thick black-driven techniques and skills in both interview and reexamine and should be prepared to start your own graduate career.

课程介绍（Cont.）

- 授课团队：
  - Jing Li 李敬 <lixeon.lij@gmail.com>，中科院信工所20级硕士研究生
  - 助教TA：Zhilu Wang 汪芷璐 <wangzhilu20@mails.ucas.edu.cn>，中科院信工所20级直博生
- 课程网站：https://lixeon.com/courses/ucas-xgs001cd-spring2022.html
- 授课方式：线上，腾讯会议
- 授课安排：

Lecture 1 : Introduction

复试面试的重要性, 抄底厚黑学思维, 润色攻心

Lecture 2 : Résumé and Brand Yourself

如何优雅制作有效简历, 社交牛逼症, 套磁的艺术, 恰到好处的恭维, 打造自己的品牌

Lecture 3 : Basic of Artificial Intelligence Security

人工智能基础、对抗机器学习、最佳实践串讲

Lecture 4 : Basic of Cyber Security

计算机科学基础知识、网络空间安全基础、最佳实践串讲

- 课程评估：https://www.wjx.cn/vj/PWcQRxR.aspx

Reference

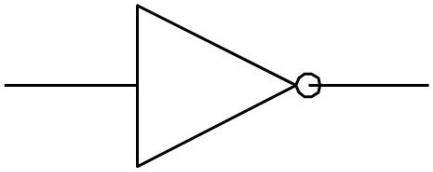
1. 081201M04002H 计算机体系结构, Fall2020, 胡伟武, 中国科学院大学
2. 0839X2M07006H 大数据与人工智能技术, Summer2021, 岳银亮, 中国科学院大学
3. 252-0028-00L Digital Design and Computer Architecture, Spring2022, Onur Mutlu, ETH Zürich
4. Introduction to Computer, Fall2021, Yung-Yu Chuang (莊永裕), 臺灣大學
5. 083900M01002H 网络与系统安全, Fall2020, 荆继武, 中国科学院大学
6. 201M6022H 漏洞利用与攻防实践, Fall2019, 霍玮, 中国科学院大学
7. 0839X6M05002H 高级网络攻防, Spring2021, 龚晓锐, 中国科学院大学
8. 0839X6M04001H 软件安全原理, Fall2020, 邹维, 中国科学院大学
9. 0839X6M05001H 软件安全漏洞分析与发现, Spring2021, 苏璞睿, 中国科学院大学
10. 0839X5M05003H 操作系统安全, Spring2021, 涂碧波, 中国科学院大学
11. 15-213/14-513/15-513 Introduction to Computer Systems, Spring2022, Andersen et al., CMU
12. 083900M01001H 应用密码学, Fall2019, 林东岱, 中国科学院大学

Contents

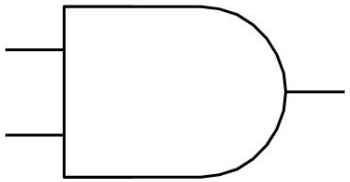
Napoleon Crossing the Alps

https://en.wikipedia.org/wiki/Napoleon_Crossing_the_Alps

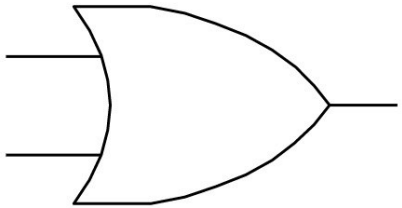
Logic gates



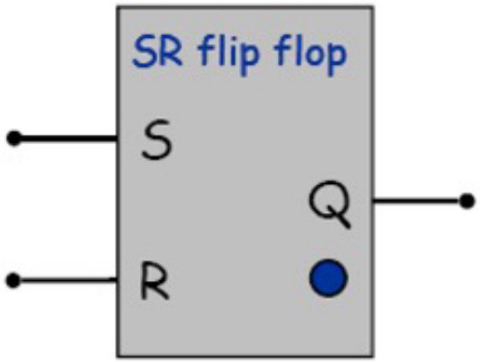
NOT



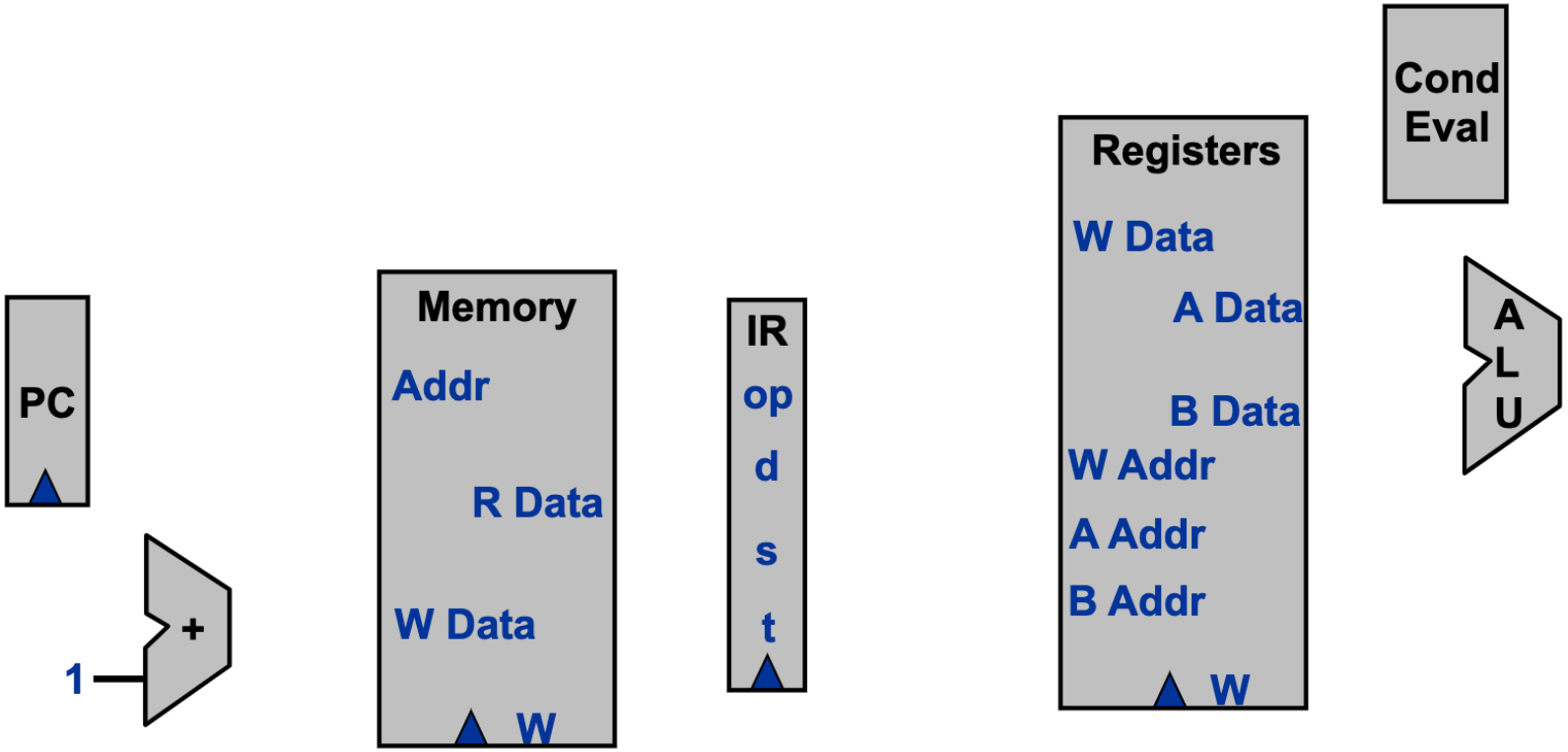
AND



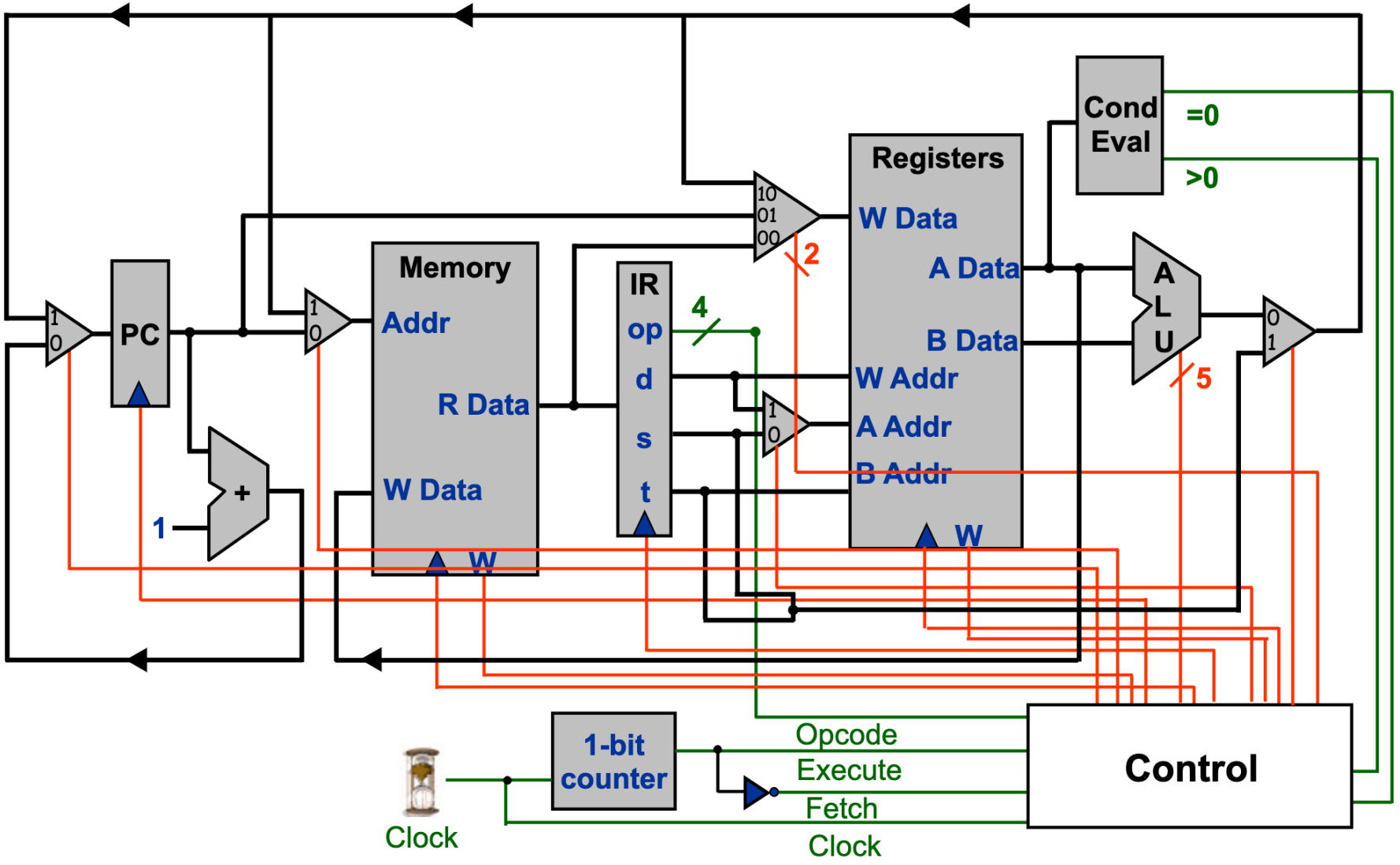
OR



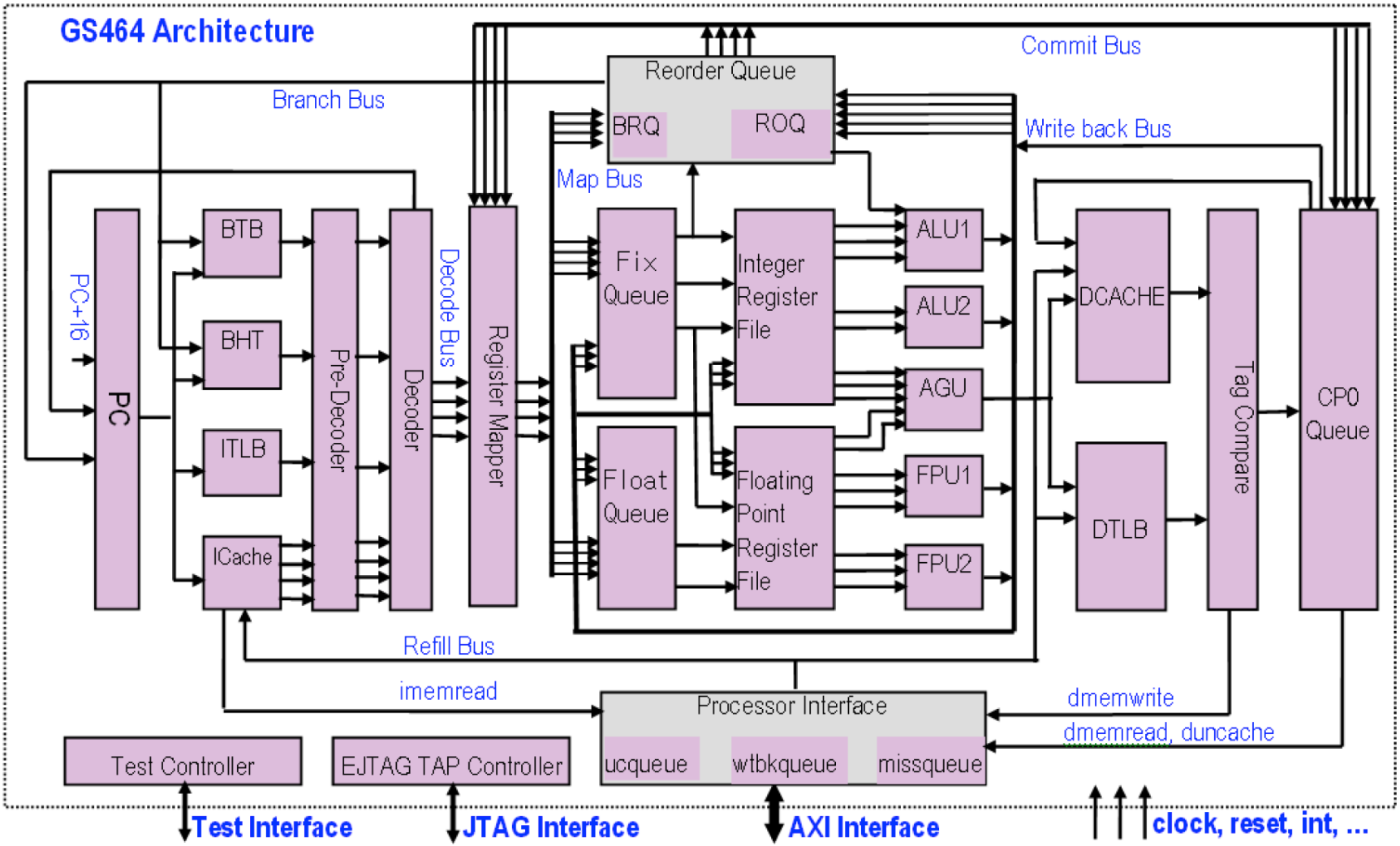
Components


PC
+
1
Memory
Addr
R Data
W Data
W
IR
op
d
s
t
Registers
W Data
A Data
B Data
W Addr
A Addr
B Addr
W
Cond
Eval
A
L
U

Toy machine

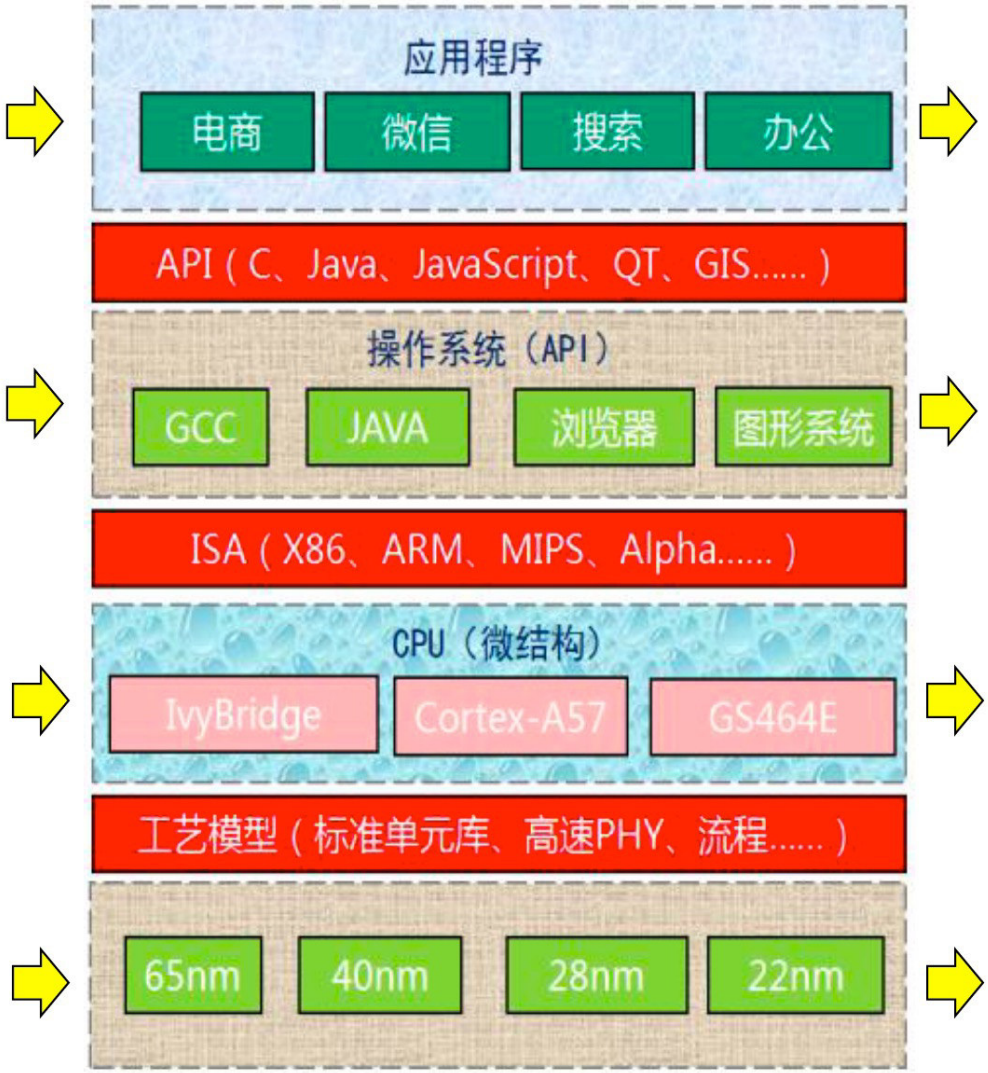

Cond Eval
=0
>0
Registers
W Data
A Data
B Data
W Addr
A Addr
B Addr
W
10
01
00
2
Memory
Addr
R Data
W Data
W
IR
op
d
s
t
4
A
L
U
5
PC
1
0
+
1
1-bit counter
Clock
Opcode
Execute
Fetch
Clock
Control

我的设计


GS464 Architecture
Commit Bus
Reorder Queue
BRQ
ROQ
Branch Bus
Write back Bus
Map Bus
BTB
BHT
ITLB
ICache
PC
PC+16
Pre-Decoder
Decoder
Decode Bus
Register Mapper
Fix Queue
Float Queue
Integer Register File
Floating Point Register File
ALU1
ALU2
AGU
FPU1
FPU2
DCACHE
DTLB
Tag Compare
CP0 Queue
Refill Bus
imemread
Processor Interface
ucqueue
wtbkqueue
missqueue
dmemwrite
dmemread, duncache
Test Controller
EJTAG TAP Controller
Test Interface
JTAG Interface
AXI Interface
clock, reset, int, ...

我国IT产业人才严重失衡

2010年
应用软件移植到自主CPU上
性能有数量级差异



应用发达

主要API性能差3-5倍；国内JS、Java等程序员数以百万计，但没有Java虚拟机、JS引擎等研制队伍

用别人的模块“攒”OS，差距大

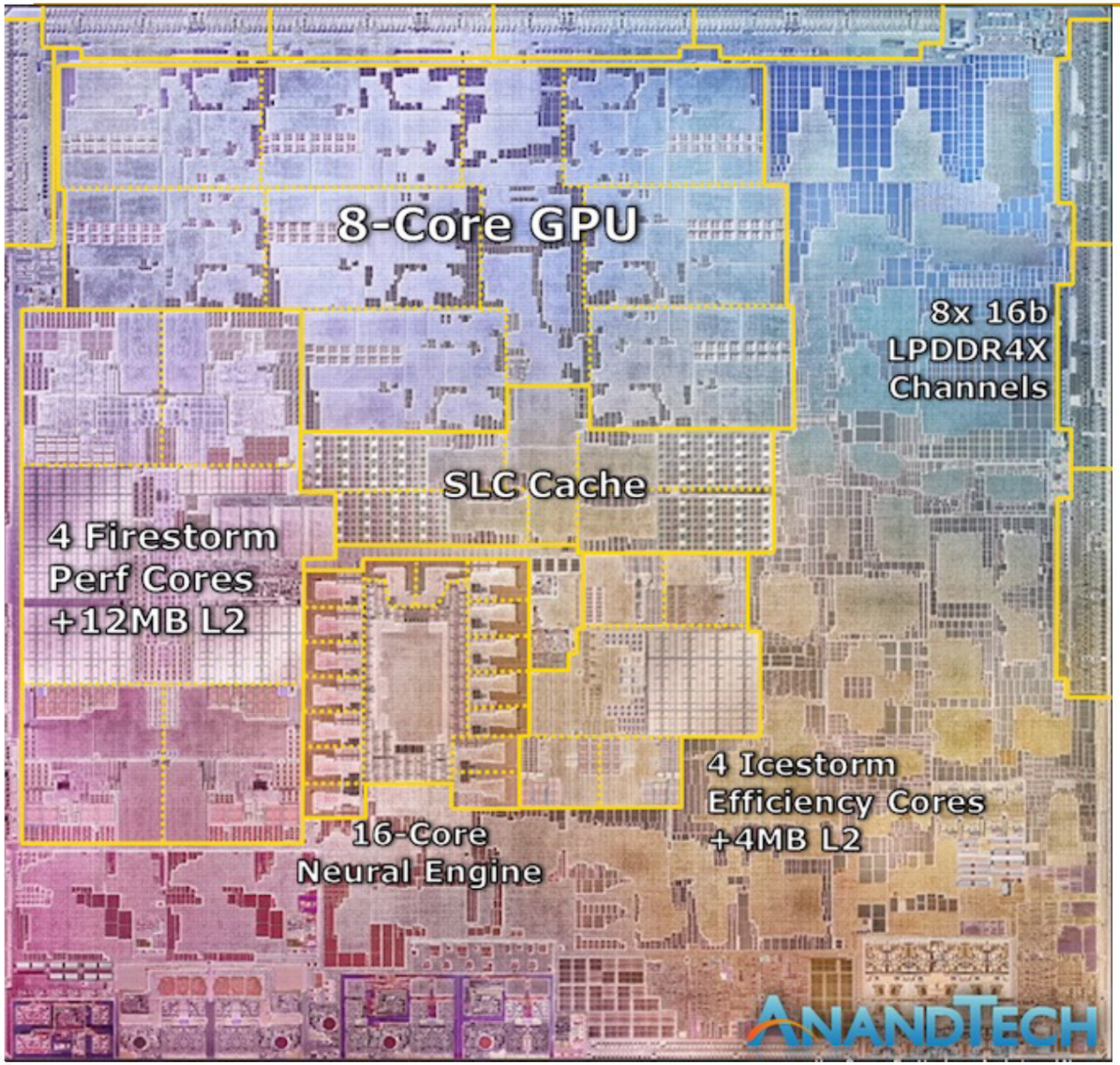
相同主频下比主流X86性能差3-5倍：国内企业主要用ARM CPU“攒”SOC，每年销售几亿片

用别人的模块“攒”芯片，差距大

主频差1-2倍：工艺性能差0.5-1倍，物理设计能力差0.5-1倍

工艺还行

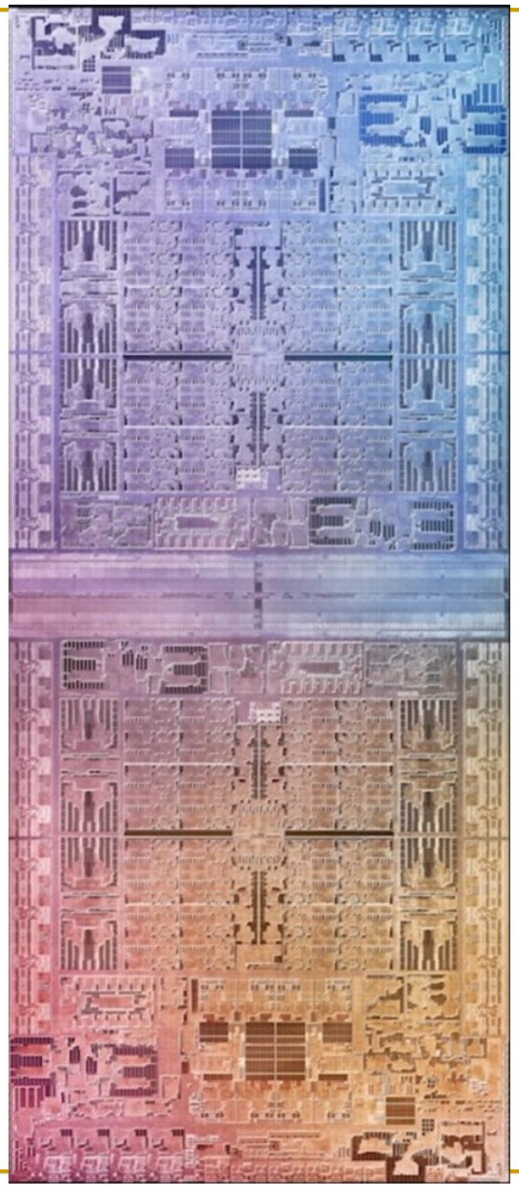
2021: Apple M1



- 4 High-Perf GP Cores
- 4 Efficient GP Cores
- 8-Core GPU
- 16-Core Neural Engine
- Lots of Cache
- Many Caches
- 8x Memory Channels

- 16B transistors

2022: Apple M1 Ultra



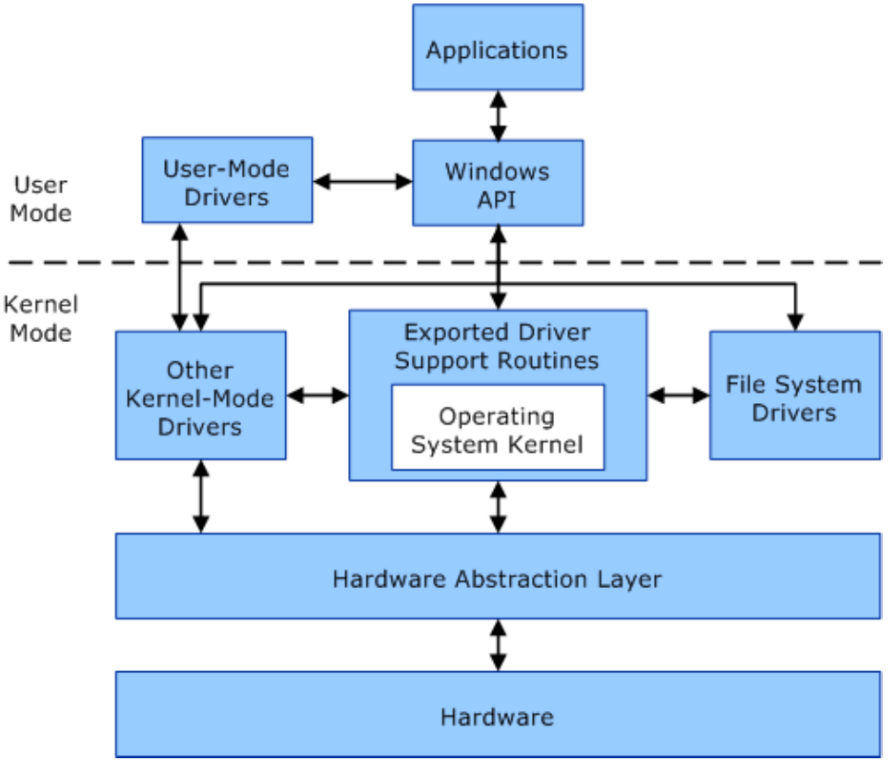
- 16 High-Perf GP Cores
- 4 Efficient GP Cores
- 64-Core GPU
- 32-Core Neural Engine
- Lots of Cache
- Many Caches
- 32x Memory Channels
- 128 GB DRAM

- 114B transistors

用户模式与内核模式-Windows为例

处理器有两个不同模式：用户模式 和内核模式 。 根据处理器上运行的代码的类型，处理器在两个模式之间切换。应用程序在用户模式下运行，核心操作系统组件在内核模式下运行。

启动用户模式应用程序时，Windows 会为该应用程序创建进程 。 进程为应用程序提供专用的“虚拟地址空间” 和专用的“句柄表” 。 由于应用程序的虚拟地址空间为专用空间，因此一个应用程序无法更改属于其他应用程序的数据。 每个应用程序都隔离运行，如果一个应用程序发生故障，则故障仅局限于该应用程序。 其他应用程序和操作系统不会受该故障的影响。



在内核模式下运行的所有代码都共享单个虚拟地址空间。 这意味着内核模式驱动程序不会与其他驱动程序和操作系统本身隔离。 如果内核模式驱动程序意外写入错误的虚拟地址，则属于操作系统或其他驱动程序的数据可能会受到安全威胁。 如果内核模式驱动程序发生故障，整个操作系统就会发生故障。

密码技术与编码技术

- 密码技术

  DES、AES、RSA、ECC...

密文：Buubdl boe Efgfotf
明文：

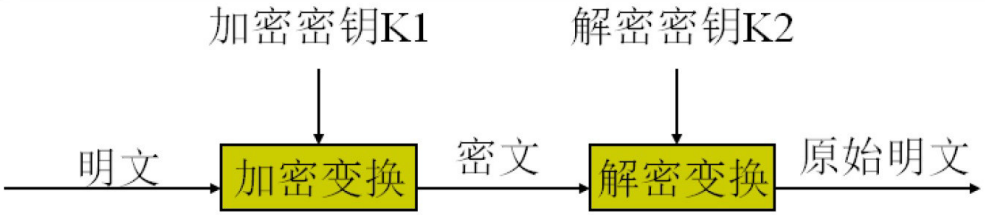
- 编码技术

  Base64、Base32、行程编码、...

  古典密码算法基本上都是编码技术

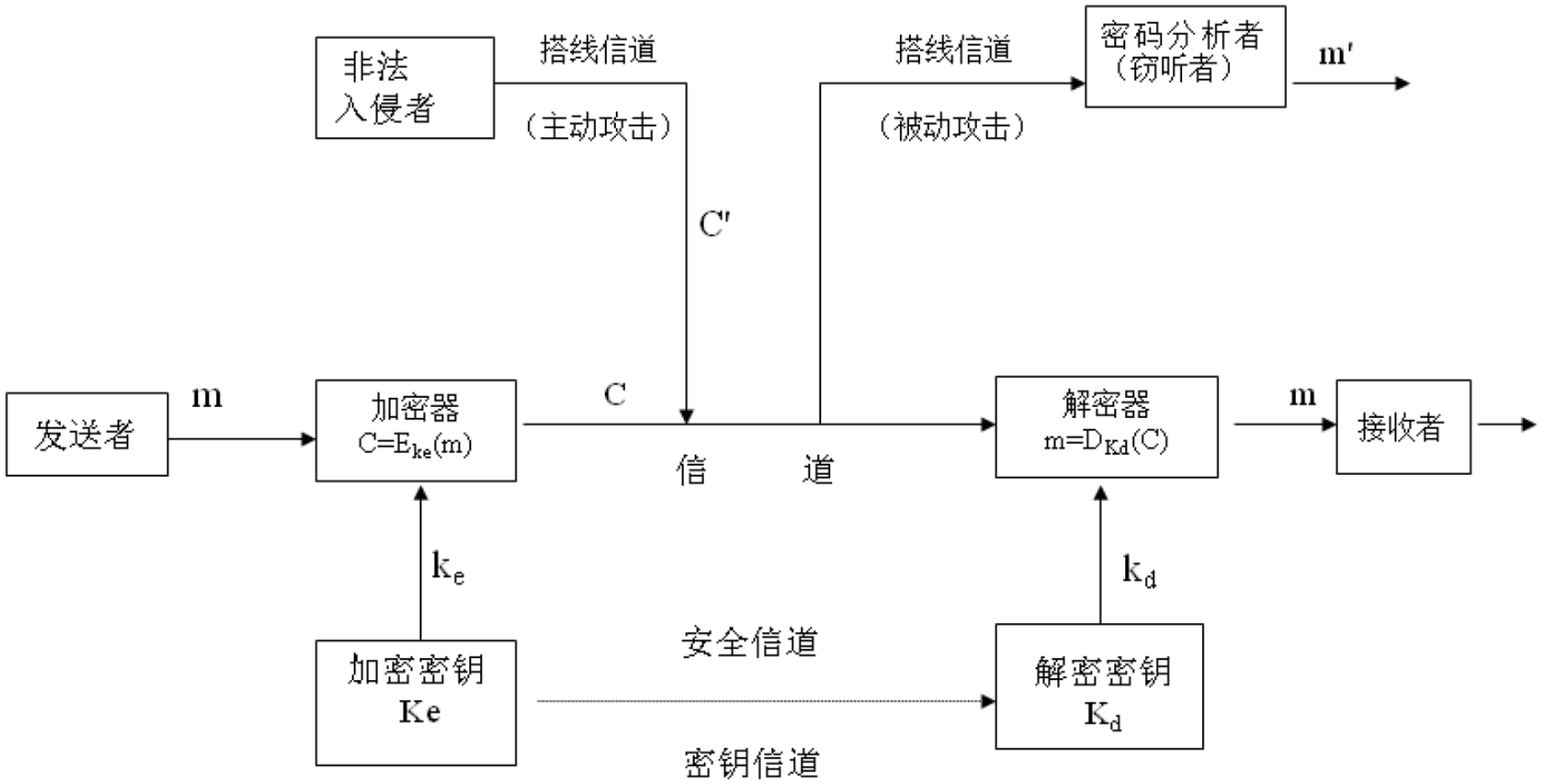
- 单向散列函数/消息摘要函数

  MD5、SHA-1、SHA-256...



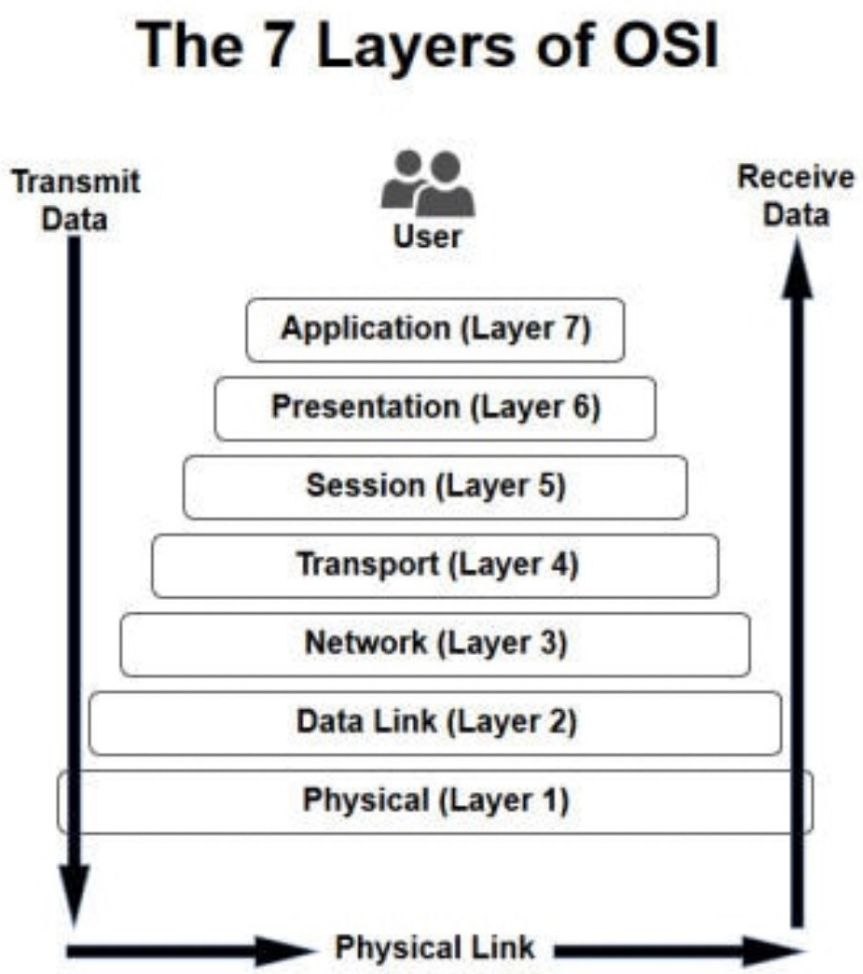
$E_{K1}(M)=C$, $D_{K2}(C)=M$, $D_{K2}(E_{K1}(M))=M$

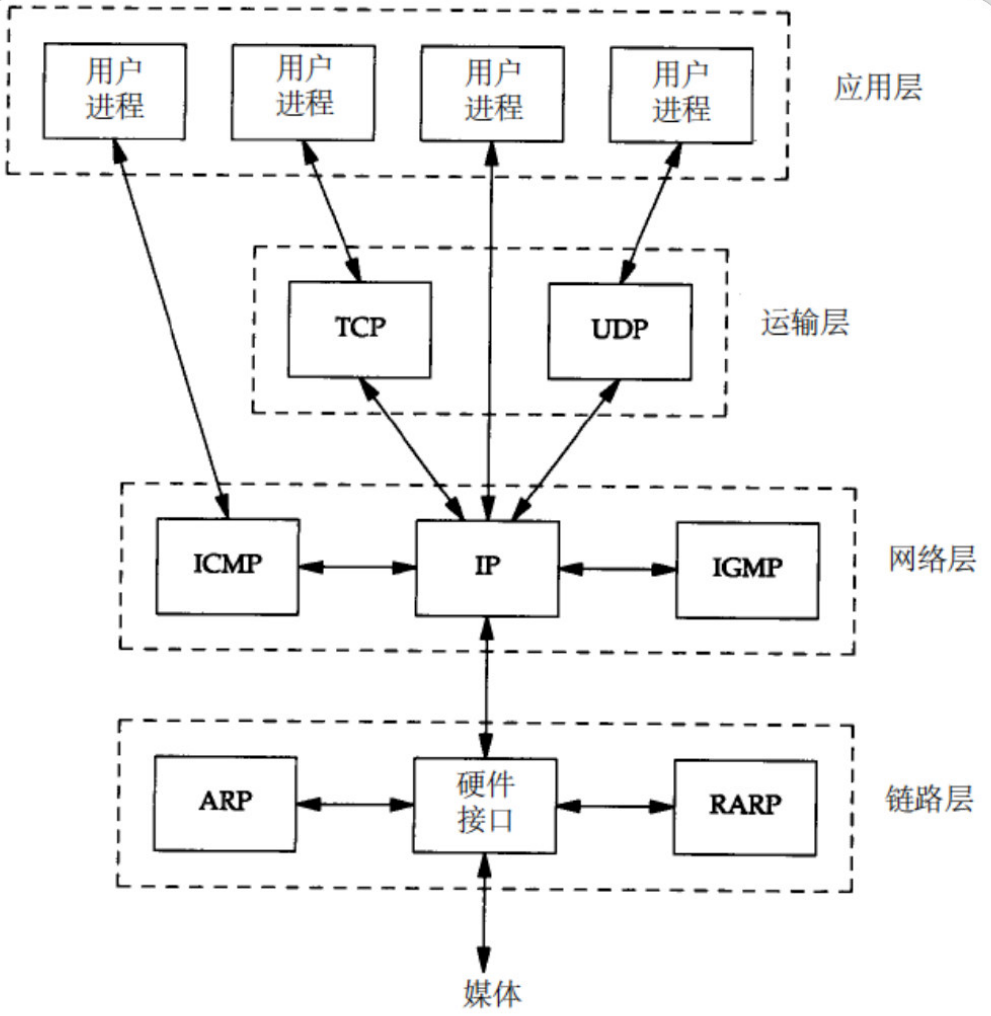
一般保密系统模型


非法
入侵者
搭线信道
（主动攻击）
搭线信道
（被动攻击）
密码分析者
（窃听者）
m′
C′
发送者
m
加密器
C=E_ke(m)
C
信　　道
解密器
m=D_Kd(C)
m
接收者
k_e
k_d
加密密钥
Ke
安全信道
解密密钥
K_d
密钥信道

网络协议分层

- 开放系统互联模型中定义的网络协议分为7层
- 上层与应用衔接、下层与硬件衔接
- 协议从上到下与数据封装过程一致
- 协议从下到上与数据的接收处理过程一致
- 协议分层的目的是为了适应更多的硬件介质和支持更多的应用类型





TCP/IP协议族的分层

- MAC帧的寻址
  解决如何把一个MAC帧发送到目标MAC地址的问题
- IP包寻址
  解决如何把一个IP包发送到目标IP地址的问题
- 域名寻址
  解决如何把一个域名映射到一个IP地址的问题

- 认证与授权的关系
  - 认证只解决他是谁的问题，不解决他有什么权限问题
  - 授权系统根据认证结果赋予其相应的权限
- 系统安全管理中的4A
  - Authentication：认证
  - Authorization：授权
  - Accounting：账号管理
  - Audit：审计

Safety & Security

- **Safety**
  - 自然属性的安全
  - 抵御自然灾害
  - 非人为攻击，不确定性

  波音737-Max

- **Security**
  - 人为属性的安全
  - 抵御故意攻击
  - 人为攻击，强确定性

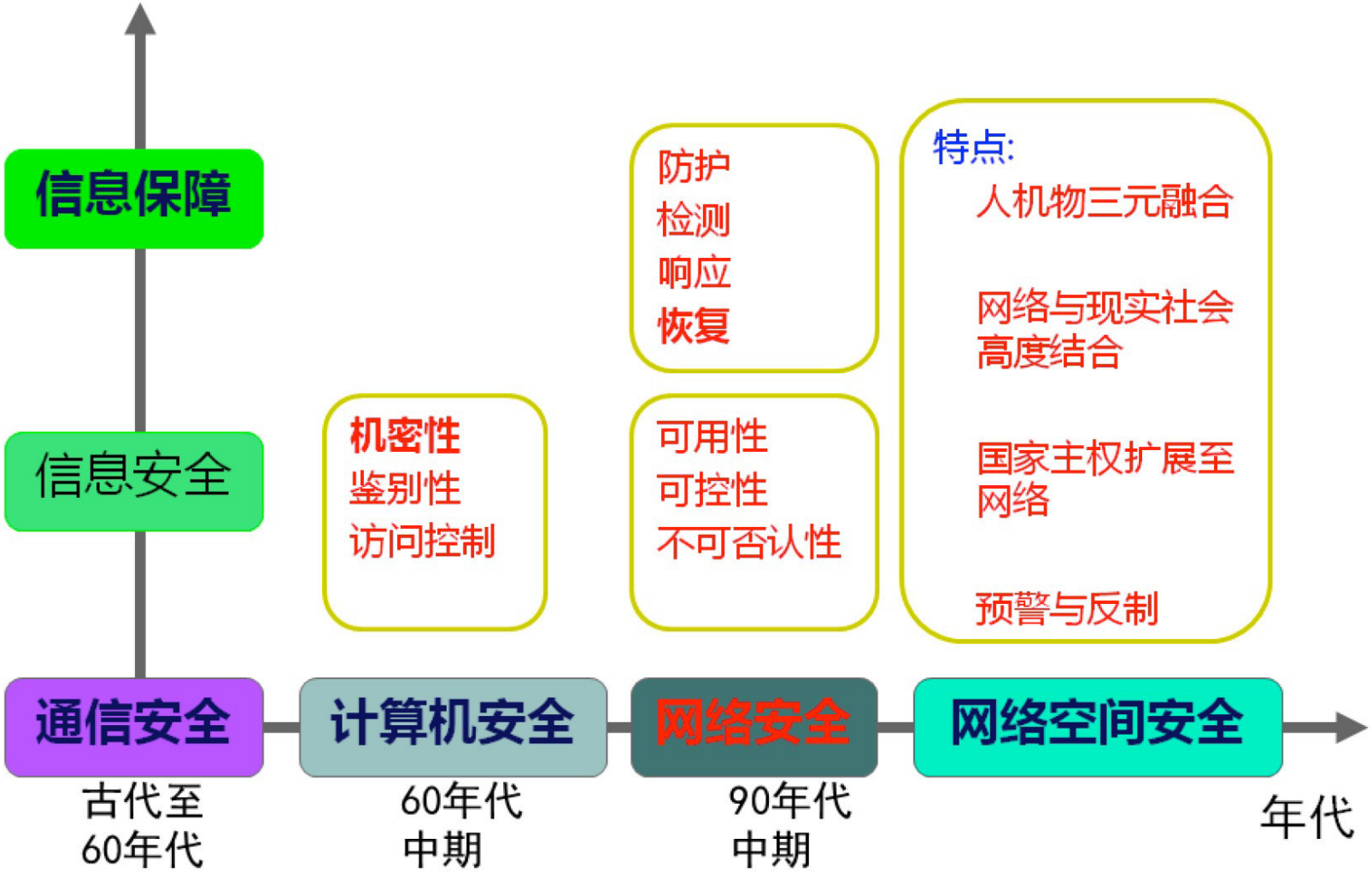
  防火、防盗

  金融：系统性风险、人为操纵

  汽车：ABS、雷达、电子锁

信息安全更多指的是Security

信息安全的历史发展


信息保障
信息安全
防护
检测
响应
恢复
特点:
人机物三元融合
网络与现实社会高度结合
国家主权扩展至网络
预警与反制
机密性
鉴别性
访问控制
可用性
可控性
不可否认性
通信安全
计算机安全
网络安全
网络空间安全
古代至60年代
60年代中期
90年代中期
年代

信息安全三要素

- 概要机密性(Confidentiality)　看不见不该看见的

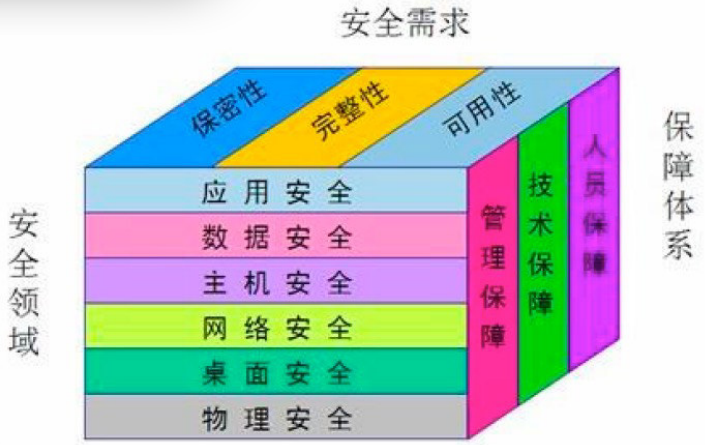
  信息仅被合法的实体访问，不泄漏给未授权的实体。

- 完整性(Integrity)　改不了不该改的

  信息只能由授权实体修改，不被偶然或蓄意地篡改、伪造、丢失等。

- 可用性(Availability)　停不了不该停的

  信息能够随时被授权实体访问并使用。



信息安全需求是什么？

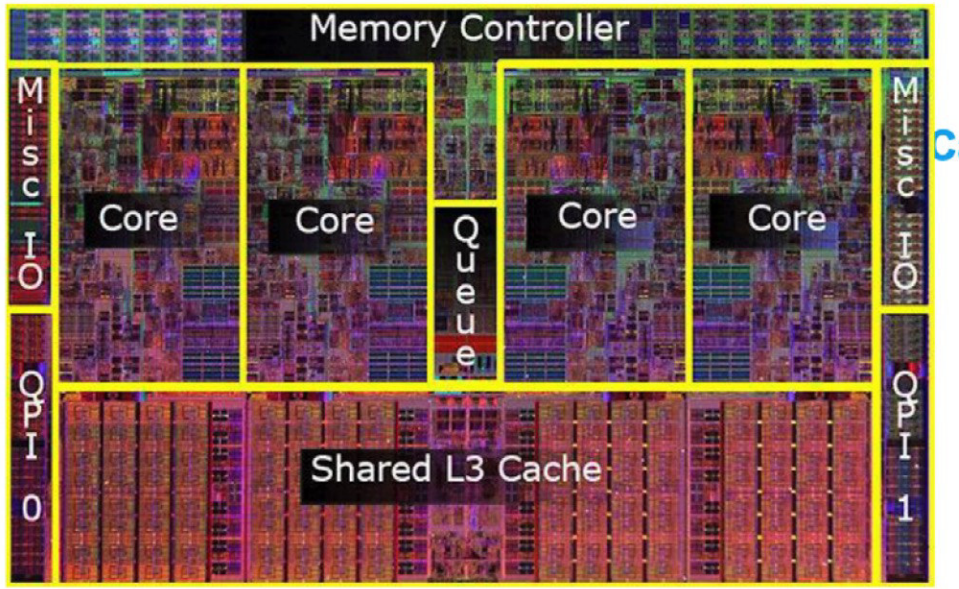
- 我们的建议：
  - 机密性： **Confidentiality**
  - 完整性：**Integrity**
  - 可用性： **Availability**
  - 真实性：**Authenticity （Trustworthy）**

其它特性可以包括在上述概念中，例如非否认性经常被认为是真实和完整性的一部分，同CIA相比主要增加了系统的真实性，

在社会分工明显的未来信息世界，信息及信息服务的真实性显得更为重要。

Microarchitecture Attack

Many potential channels at our disposal



Port contention [CBHGT'18]

Arithmetic timing [AKMJLS'15]

Speculative execution [Spectre'18]

4K aliasing [MES'17]

L1 I Cache

Cache banking [YGH'16]

L2 Cache

Inclusive LLC [LYGHL'15]

Non-inclusive LLC [YSGFCT'19]

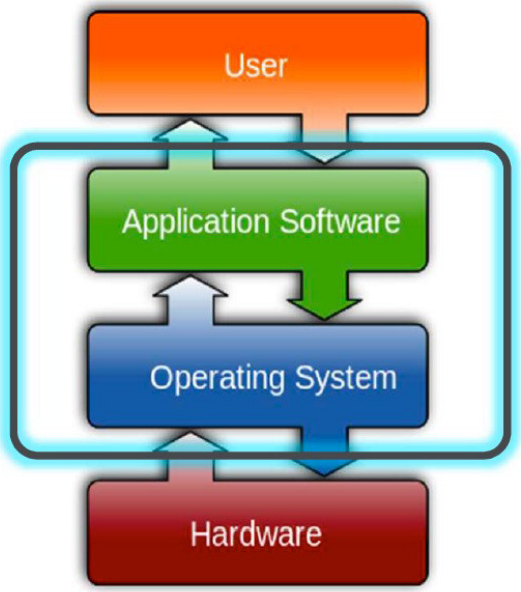
RAND unit [EP'16]

DRAM [PGMSM'16]

DRAM (and/or: stacked DRAM, HMC, NVMs)

软件与程序的定义

- **软件：**
  - 是用户与硬件之间的接口，用户通过软件与计算机交流。
  - 软件包括程序、数据和文档。
- **程序：** 是一组通过计算机执行，以完成特定任务的指令。程序包括以下类型：
  - 源程序(source code)
  - 汇编代码(assembly code)
  - 目标程序(machine code)



软件安全三部曲

软件安全防御

从“白帽”（white hat，安全防御者）视角，研究软件的各类安全防御机制。

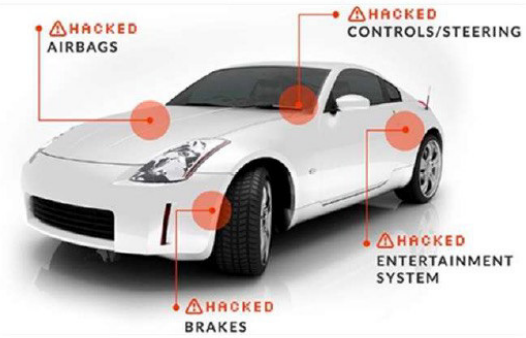
三类重要软件架构

从安全保护者角度研究安全加固方法

软件安全分析与利用

从“黑帽”（black hat，安全破坏者）视角，研究软件存在的各种安全脆弱性问题及其高效发现及利用方法 。

从攻击者角度研究渗透攻击方法



安全的软件开发

确保软件安全的工程化方法，BSI是其核心思想，需要贯彻始终。

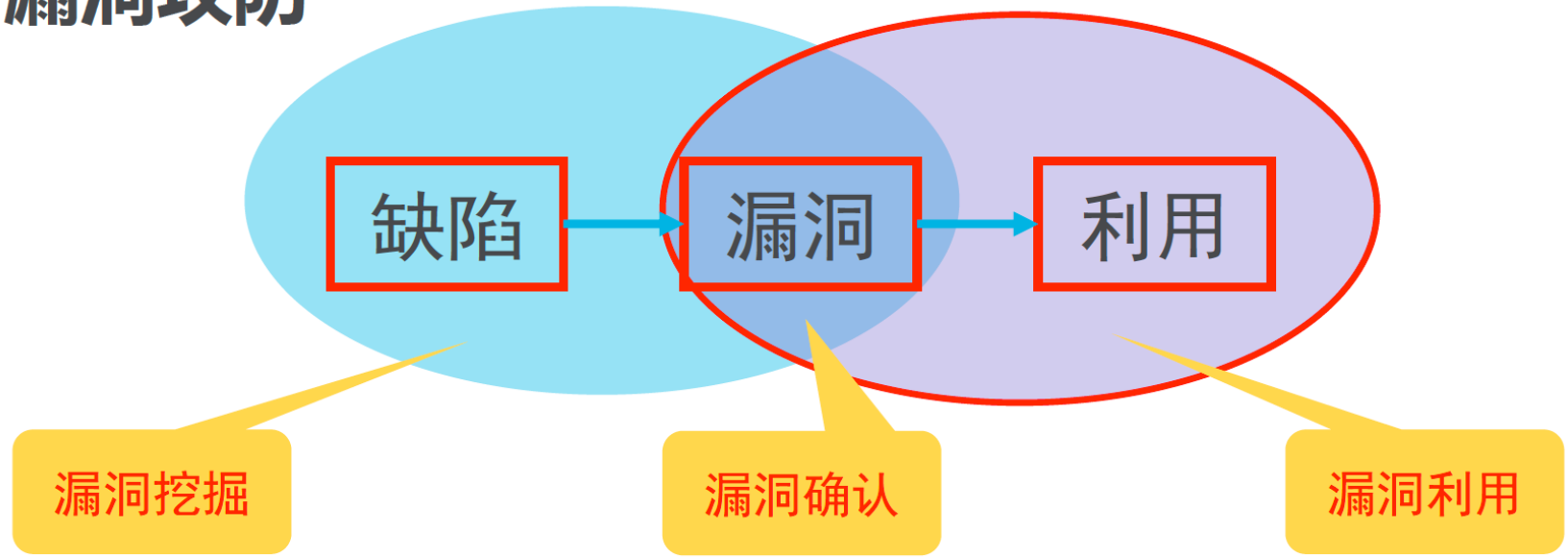
SDLC+BSI=SDL

从生产者角度设计、生产安全的汽车

汽车安全设计与试验、（安全）生产线

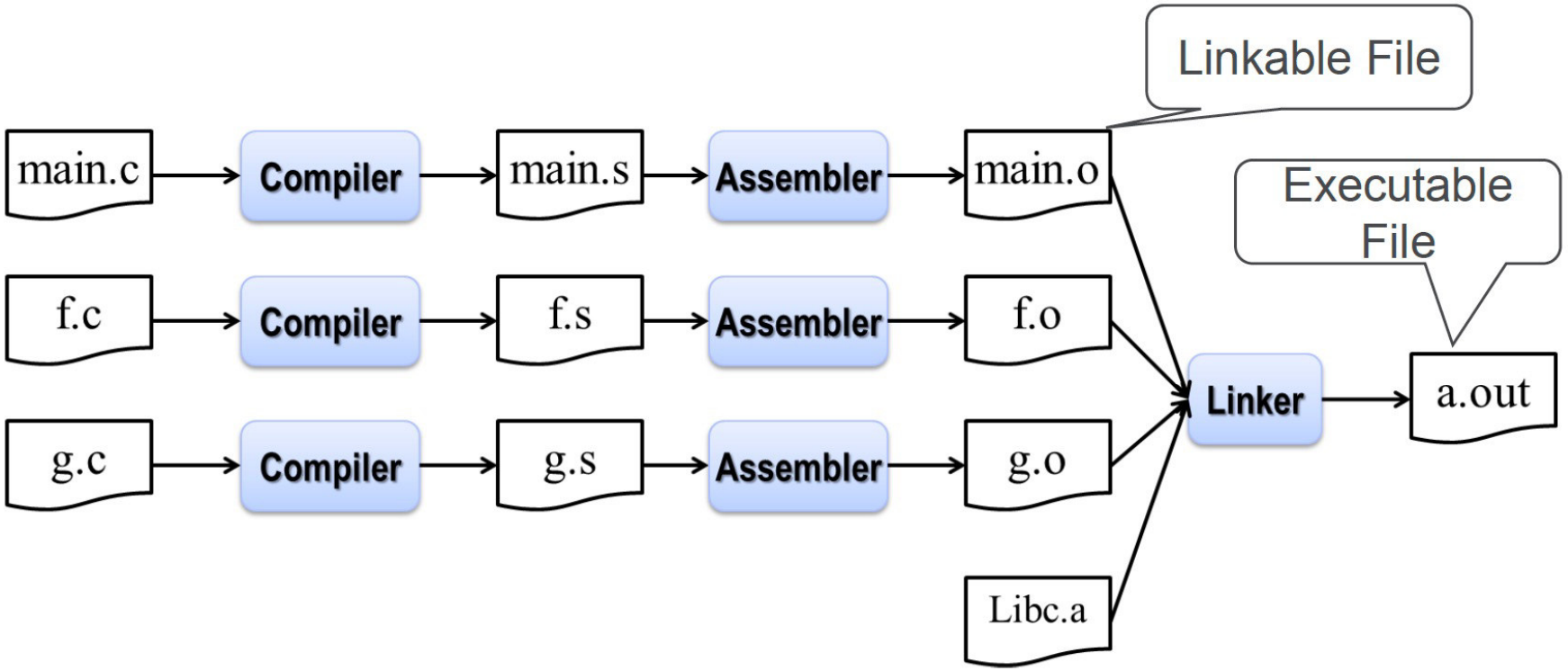
漏洞的核心概念

- 漏洞
  - 一个错误(mistake)如果可以被攻击者用于违反目标系统的一个合理的安全策略，它就是一个漏洞
- 漏洞攻防



二进制文件是编译Compile而来

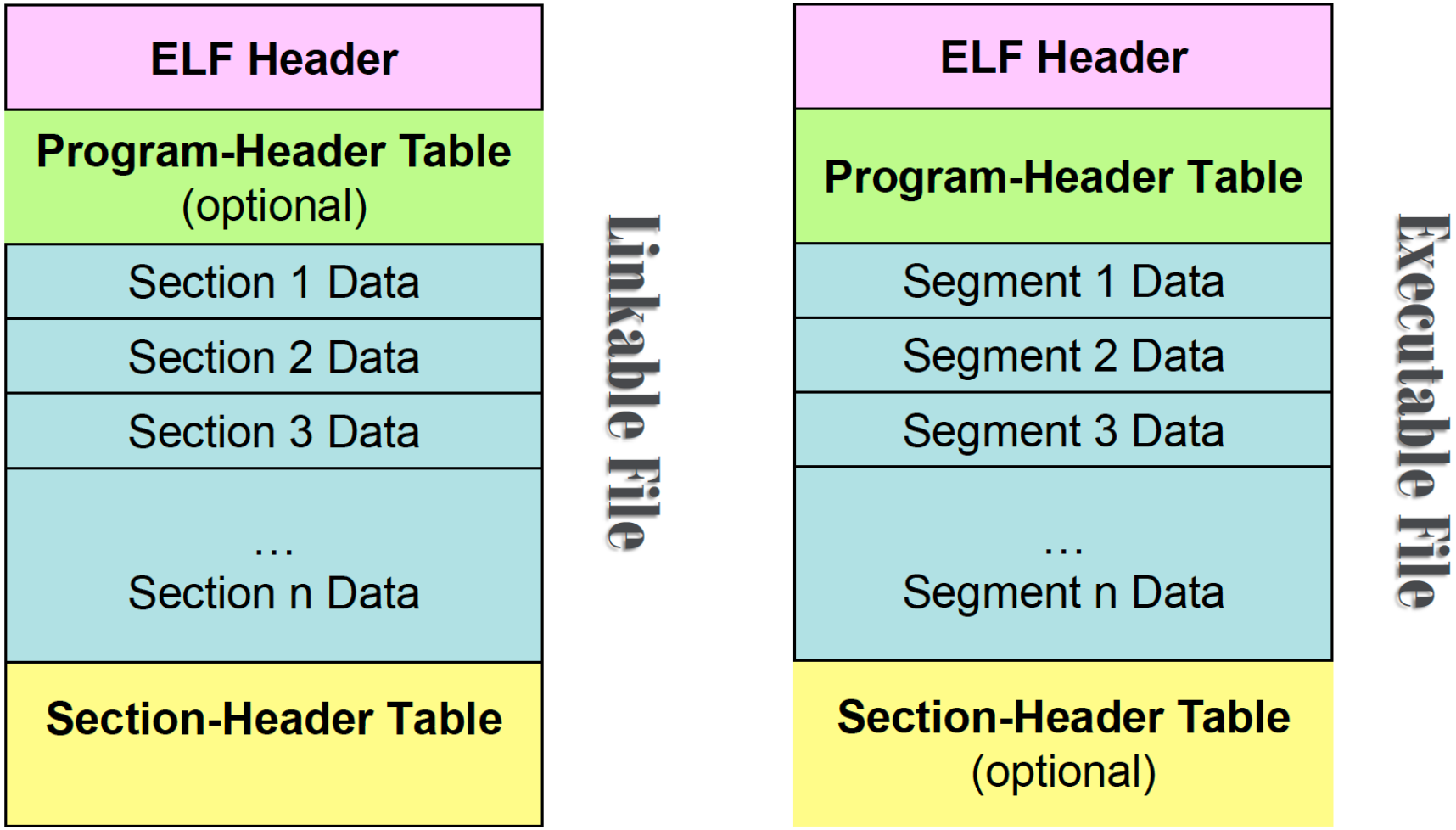
Linux平台使用**ELF**格式，Windows平台常使用**PE COFF**格式(简称PE)



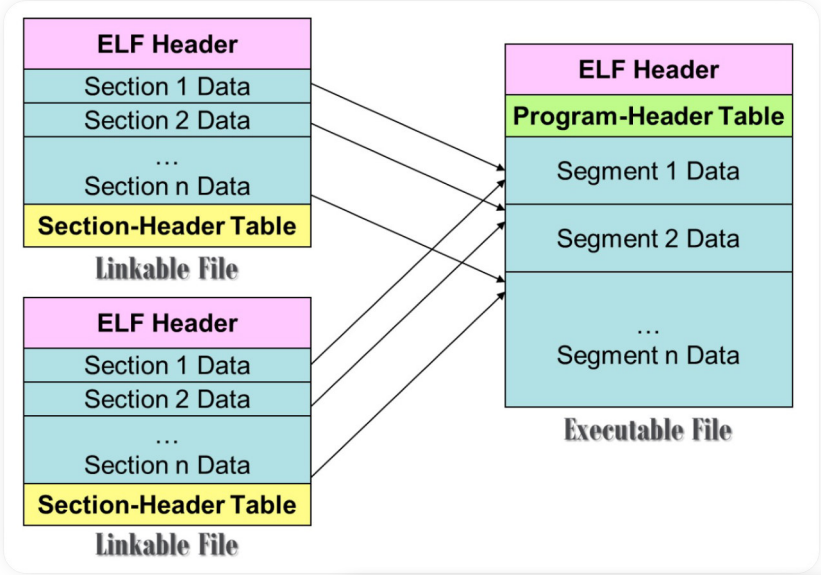
Linux平台为例

典型二进制格式ELF

ELF = Executable and Linking File

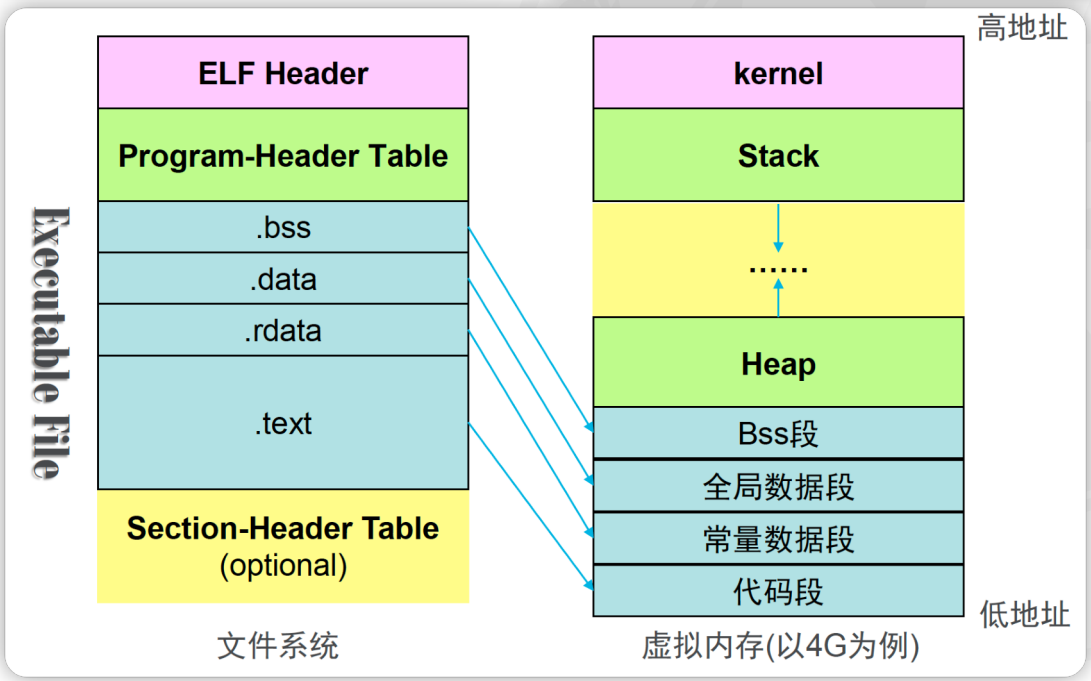


典型二进制格式ELF (Cont.)



Section	sh_type	sh_flags	含义
.bss	SHT_NOBITS	SHF_ALLOC +SHF_WRITE	包含将出现在程序内存映像的未初始化数据 当程序开始时，系统将这些数据初始化为0 不占用文件空间
.data	SHT_PROGBITS	SHF_ALLOC +SHF_WRITE	包含将出现在程序内存映像中已初始化数据 占用文件空间
.init	SHT_PROGBITS	SHF_ALLOC + SHF_EXECINSTR	进程初始化代码 程序在调用main函数前调用这些代码
.plt	SHT_PROGBITS		过程链接表
.rel	SHT_REL	SHF_ALLOC	重定位信息
.text	SHT_PROGBITS	SHF_ALLOC + SHF_EXECINSTR	代码段

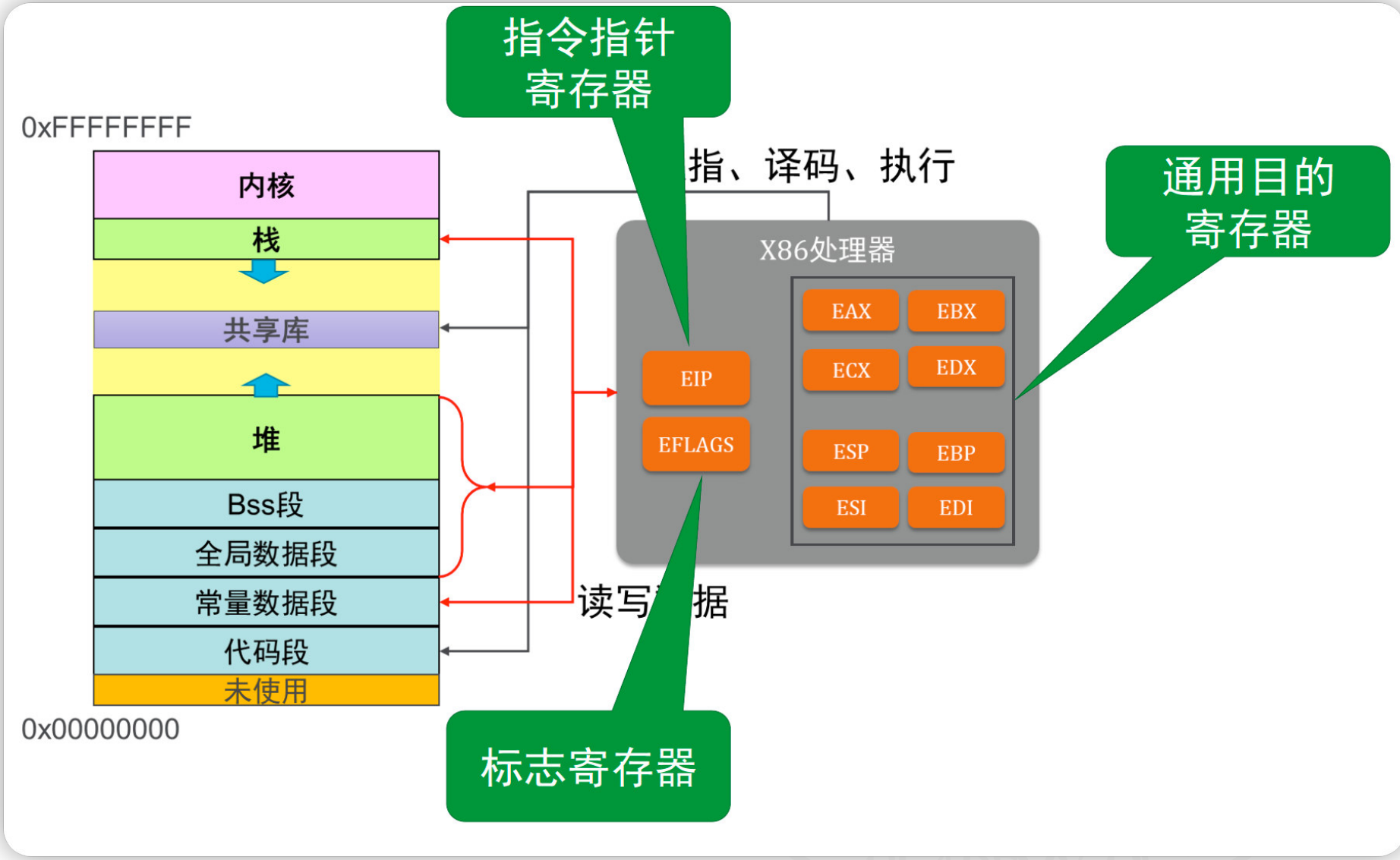
Section Header



典型二进制格式PE

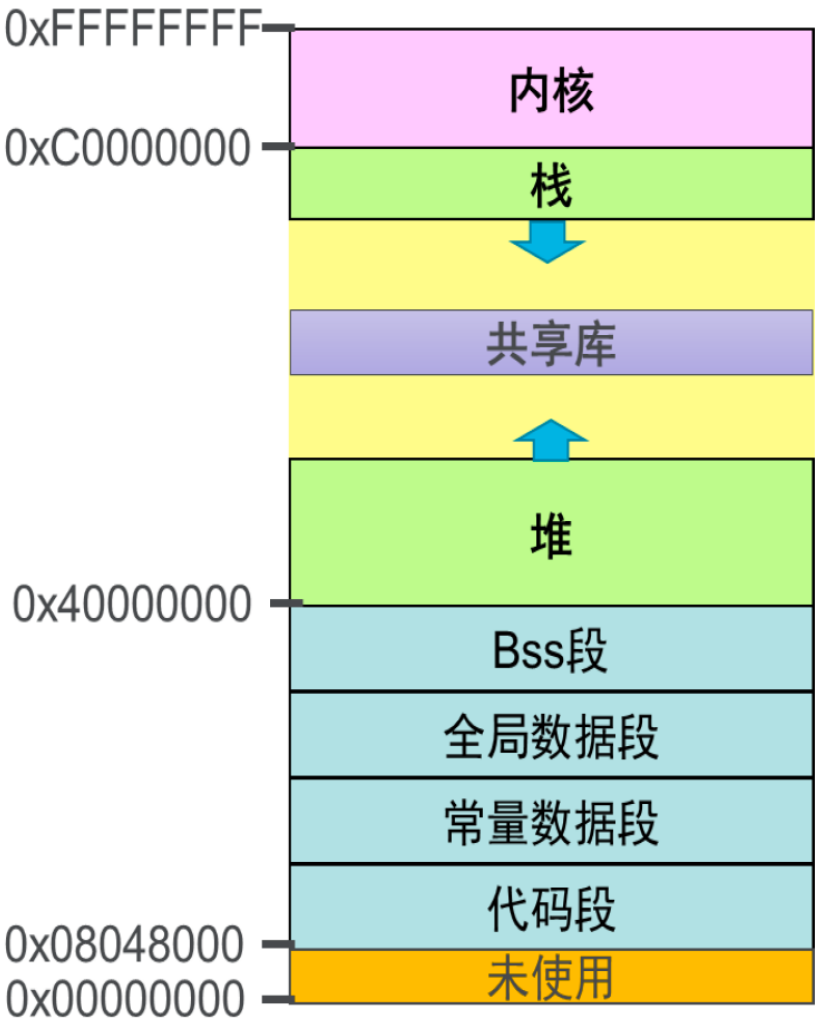

PE头
DOS头
DOS存根
NT头
节区头
节区头
代码节区
.text section
数据节区
.data section
资源节区
.rsrc section
IE11-Windows6.1.exe **OVERWRITE MODE**
Signed Int
little
(select some data)
0x0 out of 0x1FB2D0 bytes

ELF执行模型


指令指针
寄存器
0xFFFFFFFF
内核
栈
共享库
堆
Bss段
全局数据段
常量数据段
代码段
未使用
0x00000000
指、译码、执行
X86处理器
EIP
EFLAGS
EAX
EBX
ECX
EDX
ESP
EBP
ESI
EDI
通用目的
寄存器
读写
据
标志寄存器

内存破坏类漏洞利用

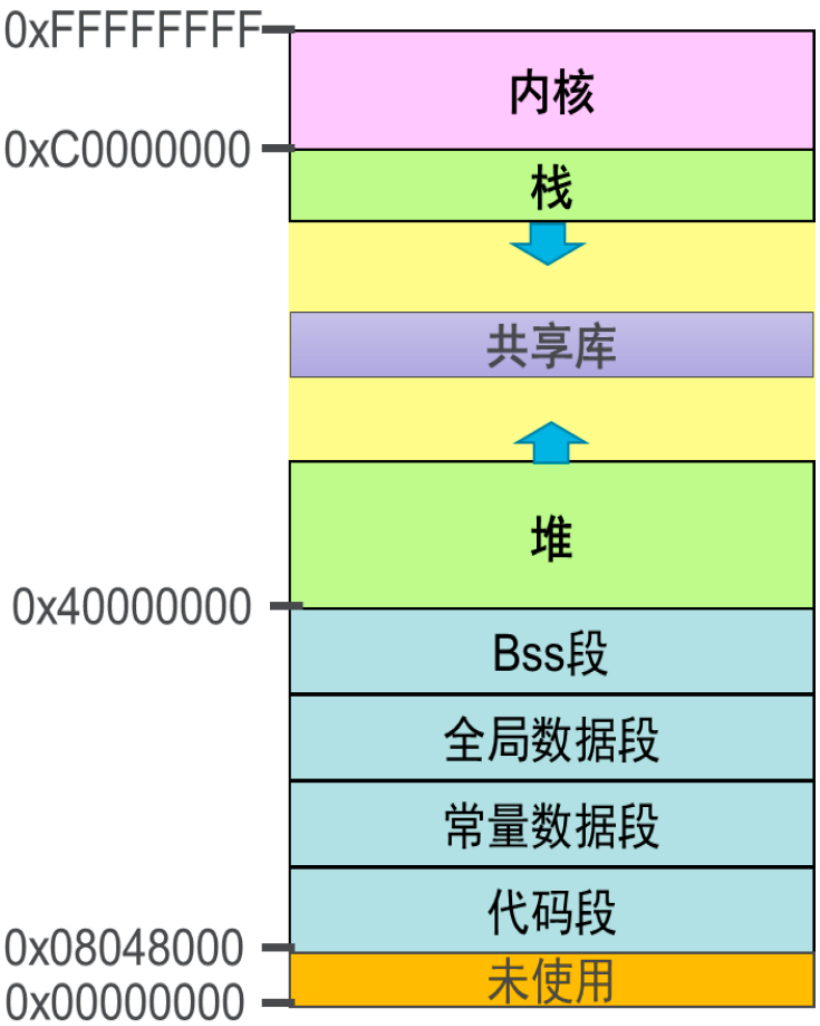
冯 诺依曼体系结构是内存破坏类漏洞的具有利用性的根源



数据和代码以相同的形式存储在内存中

内存破坏类漏洞利用 (Cont.)

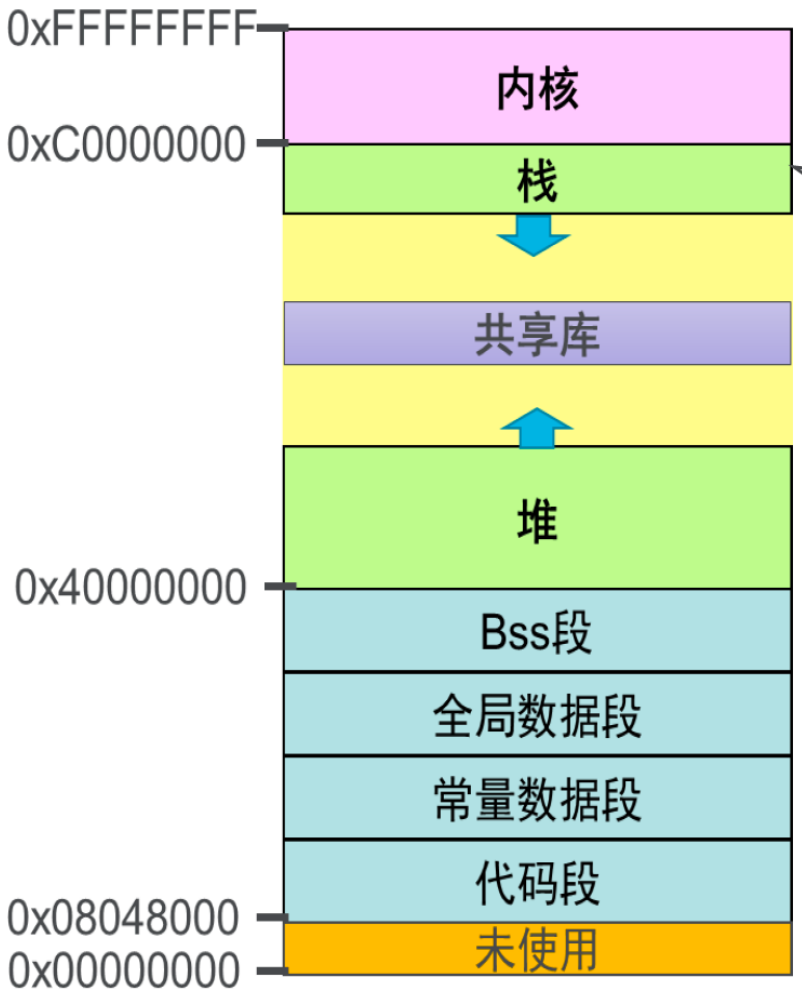
如何劫持控制流



因为需要通过控制输入而控制程序控制流，只能在程序数据区寻找控制程序控制流的渠道

内存破坏类漏洞利用 (Cont.)

如何劫持控制流

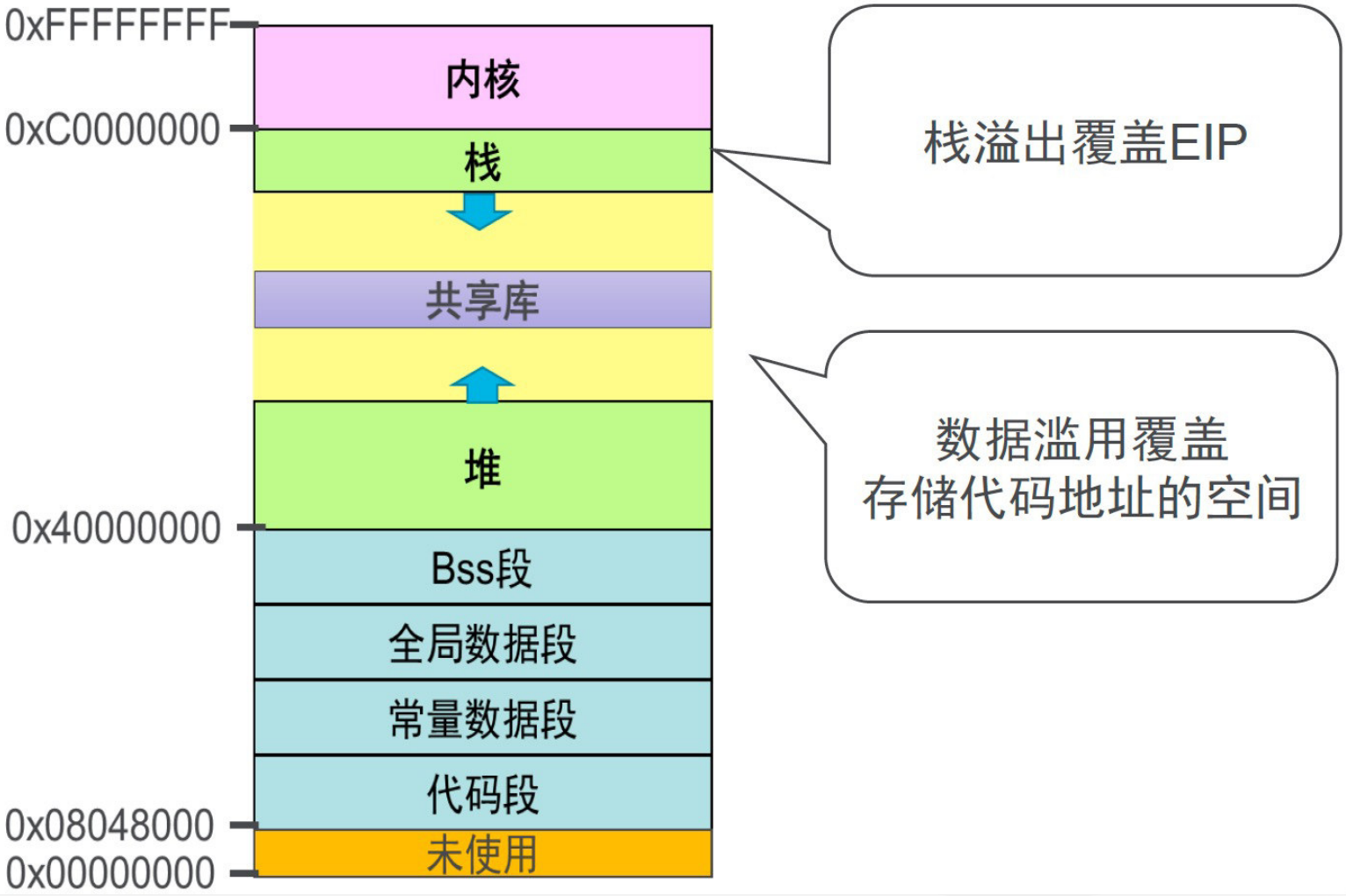


“返回地址”数据借助CPU自动的控制函数返回时将要执行的指令

数据区存储的代码地址，可以借助间接跳转指令实现控制流的改变

内存破坏类漏洞利用 (Cont.)

如何劫持控制流

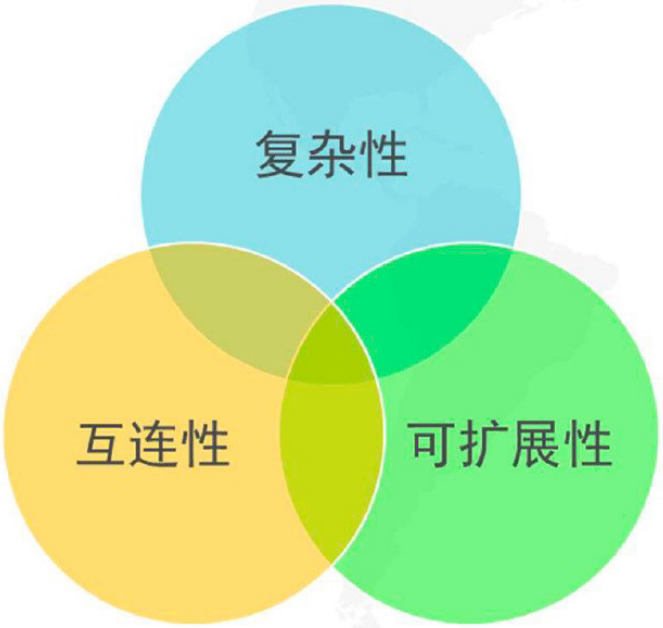


1.4 软件漏洞防护机制

- 地址随机化ASLR(Address space layout randomization)
  - 实现原理
    - ASLR也被直接称作地址随机化，是一种针对缓冲区溢出的安全保护技术，通过对堆、栈、共享库映射等线性区布局的随机化，通过增加攻击者预测目的地址的难度，防止攻击者直接定位攻击代码位置，达到阻止溢出攻击的目的
    - ASLR可以有效的降低缓冲区溢出攻击的成功率，Linux、FreeBSD、Windows等主流操作系统都已采用了该技术
  - 主要方式
    - PE文件加载地址随机化
    - 堆栈随机化
    - PEB、TEB 随机化

软件安全问题的根源

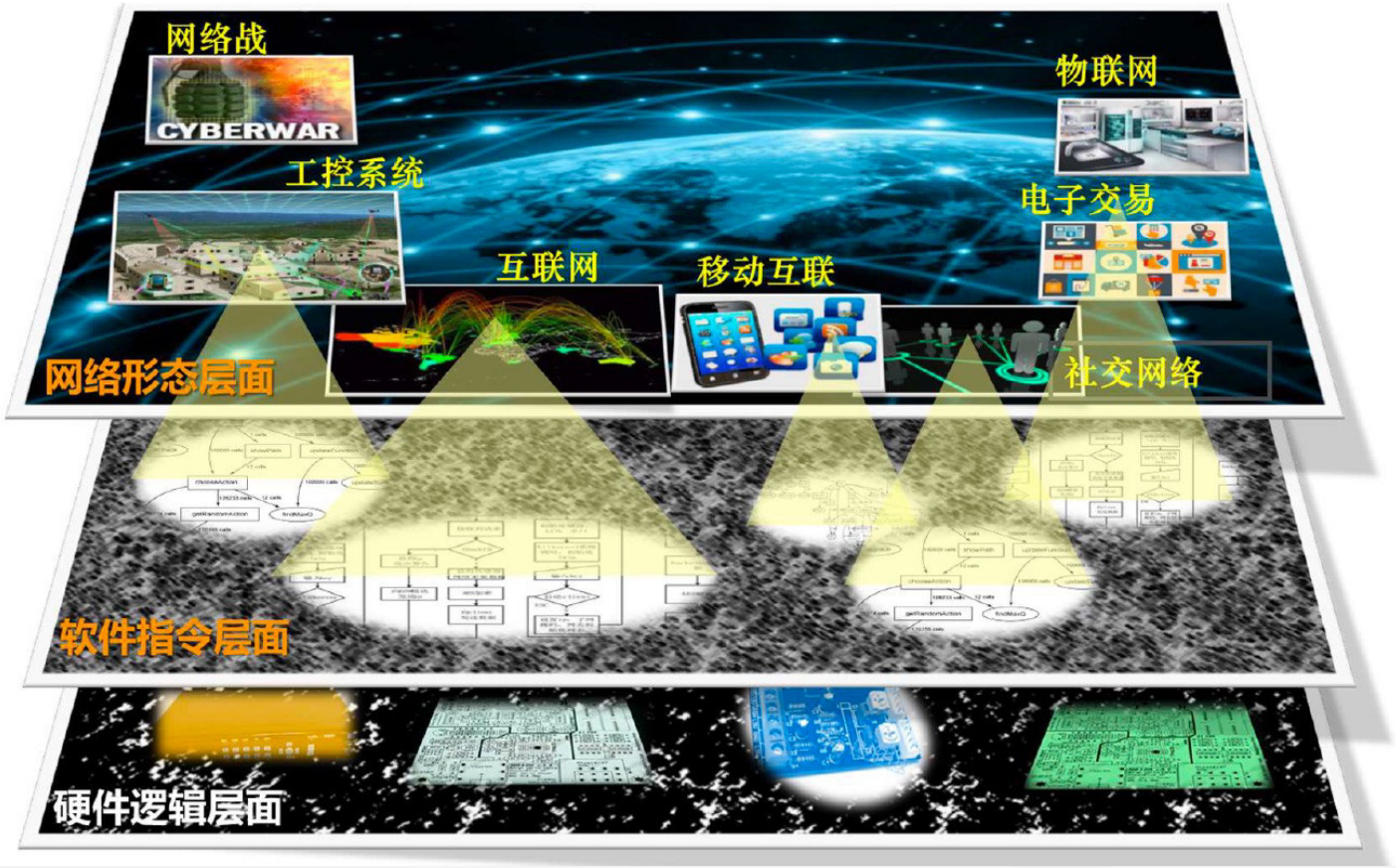
软件的三大问题



软件开发三大问题

软件安全与网络安全的关系

- 网络空间各种各样部件的大多数功能都是通过软件实现的，因此软件漏洞将直接影响网络空间的安全
- 围绕软件漏洞，无论是攻击还是防御，实际上都是在网络层面展开对抗



1.3 典型软件漏洞类型

TCA

- 逻辑漏洞
  - SQL注入漏洞
    - 形成的主要原因是程序员没有对用户输入数据的合法性进行判断，使用户可以提交一段数据库查询代码，根据程序返回的结果，获得某些他想得知的数据
  - 某个网站的登录验证的SQL查询代码为：

```
strSQL = "SELECT * FROM users WHERE (name = '" + userName + "') and (pw = '"+ passWord +"');"
```

  - 恶意填入

```
userName = "1' OR '1'='1"; 与 passWord = "1' OR '1'='1";
```

  - 导致原本的SQL字符串被填为

```
strSQL = "SELECT * FROM users WHERE (name = '1' OR '1'='1') and (pw = '1' OR '1'='1');"
```

  - 也就是实际上运行的SQL命令会变成下面这样的

```
strSQL = "SELECT * FROM users;"
```

  - 无账号密码，也能够登录网站。

1.3 典型软件漏洞类型

TCA

- 逻辑漏洞
  - XSS漏洞

```
<?php
$name = $_GET['name'];
echo "Welcome $name<br>";
echo “<a href="http://www.cnblogs.com/bangerlee/">Click to
Download</a>";
?>
```

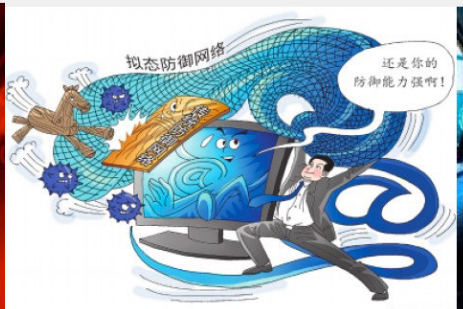
  - 该页面显示两行信息，从URI获取 'name' 参数，并在页面显示，同时显示跳转到一条URL的链接。如果攻击者在代码中嵌入攻击脚本内容，输入时采用如下的参数：

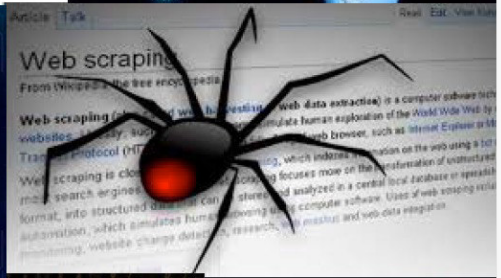
```
index.php?name=guest<script>alert('attacked')</script>
```

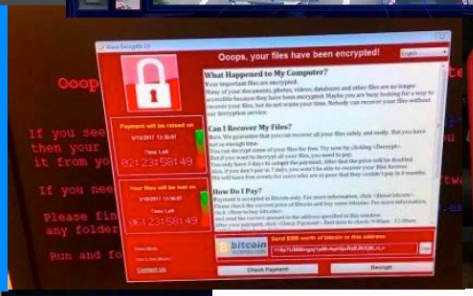
  - 当用户点击该链接时，攻击者提交的脚本内容会被执行，带'attacked'的告警提示框弹出。更进一步，如果攻击者提交一个URL实现修改链接。用户将可能会跳转至攻击者提供的链接。

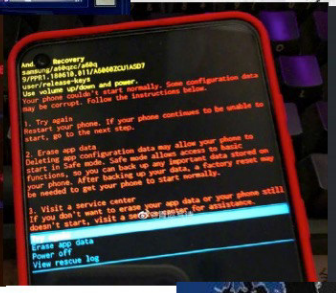
```
index.php?name=<script>window.onload = function() {
var link=document.getElementsByTagName("a");
link[0].href="http://attacker-site.com/";}</script>
```

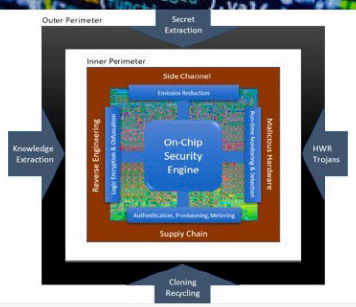
由点及面：开阔信息安全视野

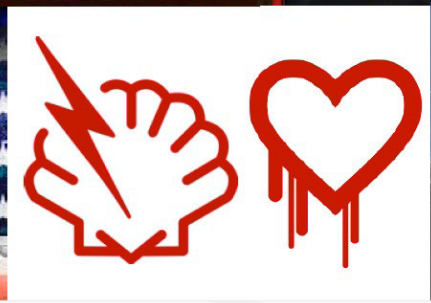












世界一流信息安全机构和组织


DEFENSE ADVANCED RESEARCH PROJECTS AGENCY

公安部第三研究所
The Third Research Institute Of Ministry Of Public Security





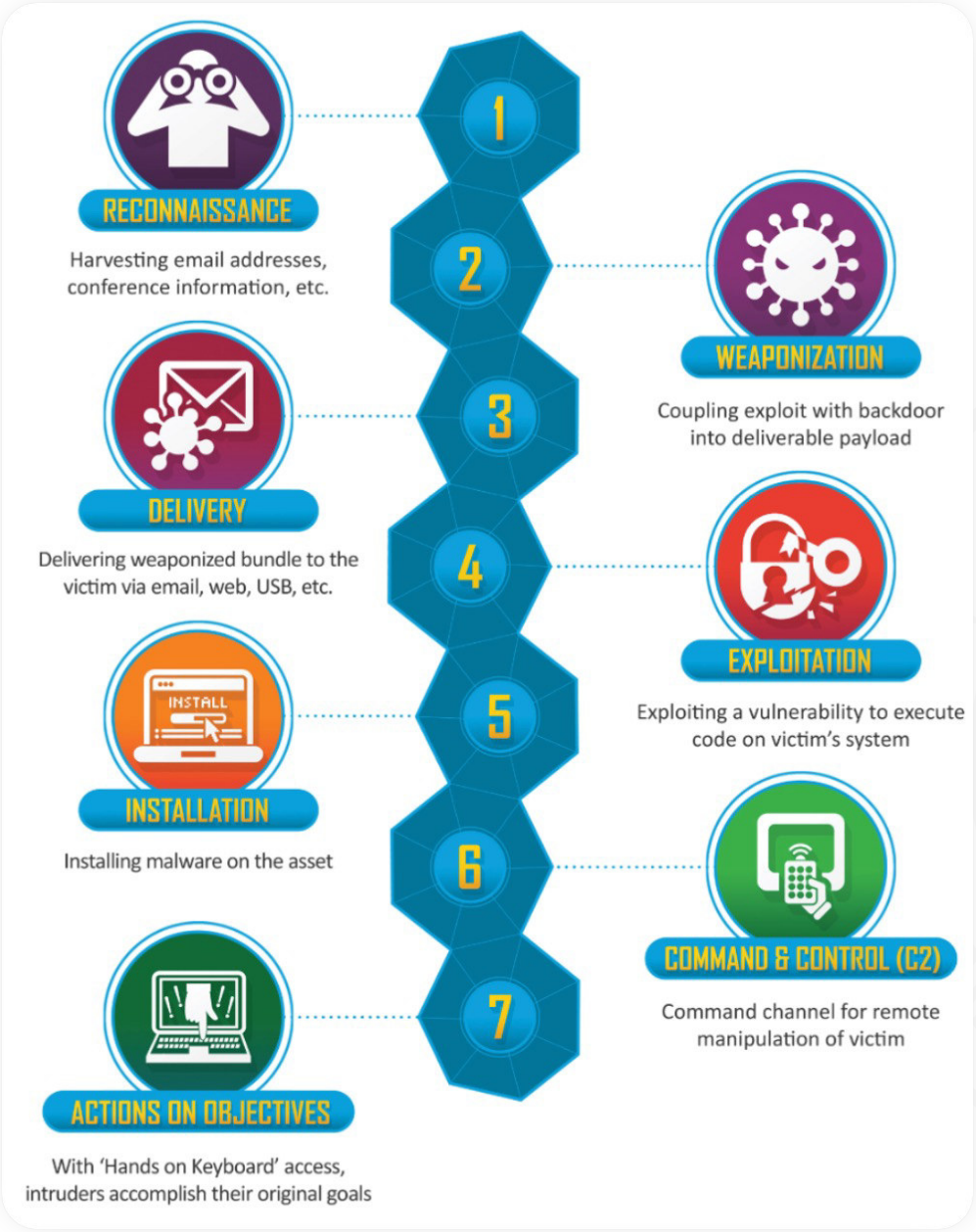

中国科学院信息工程研究所
INSTITUTE OF INFORMATION ENGINEERING,CAS

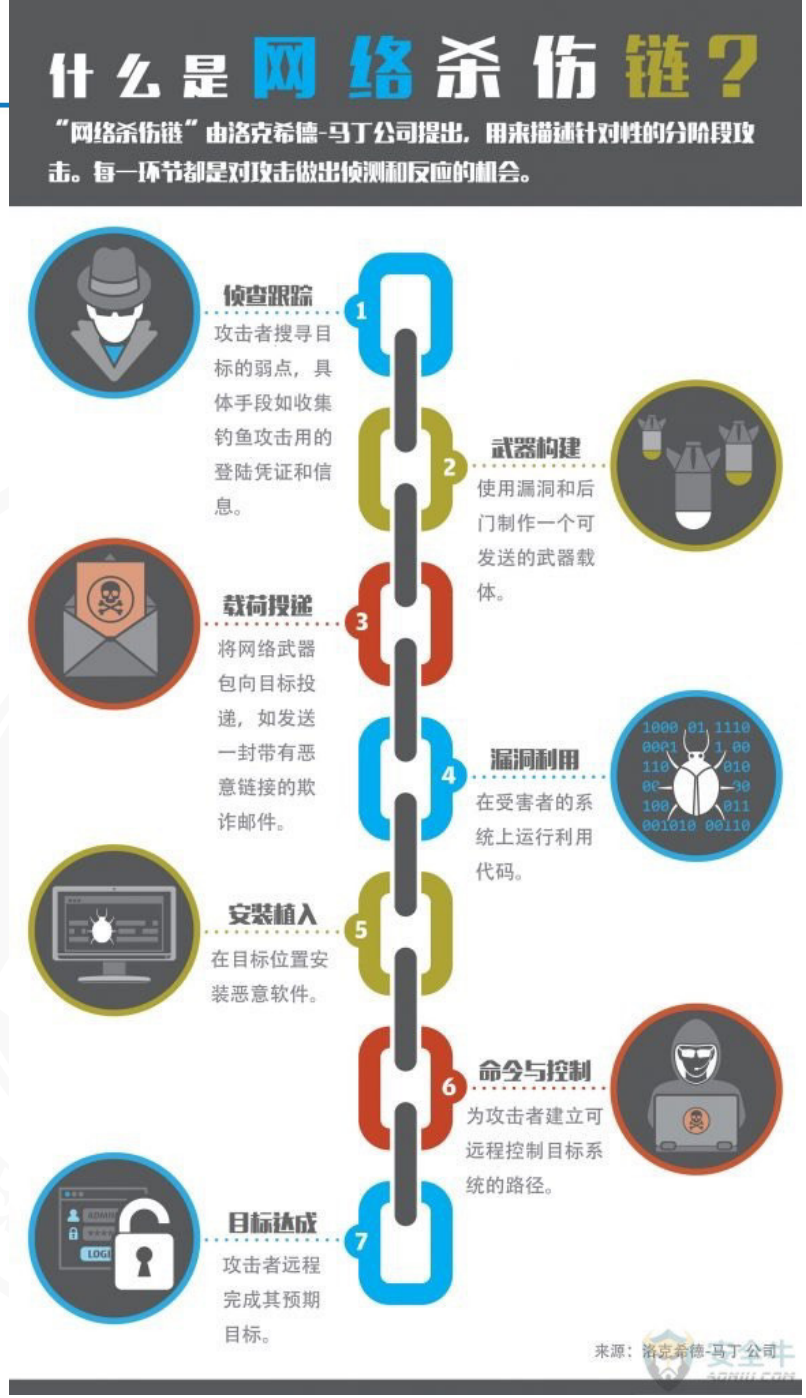
HERJAVEC GROUP



CYBERARK


洛克希德-马丁网络杀伤链





ATT&CK 威胁框架

- 概 况
  - ATT&CK威胁框架是在洛马杀伤链框架的基础上，构建的一套更细粒度、更易共享的攻击行为模型和知识库，具有更为**丰富的技术性细节**
  - **依据对真实APT攻击进行追踪分析**，提炼出技术点，不断积累知识库，促使该框架基于可能遇到的实际威胁来完善
  - **广泛获得** 安全研究人员与安全厂商的支持

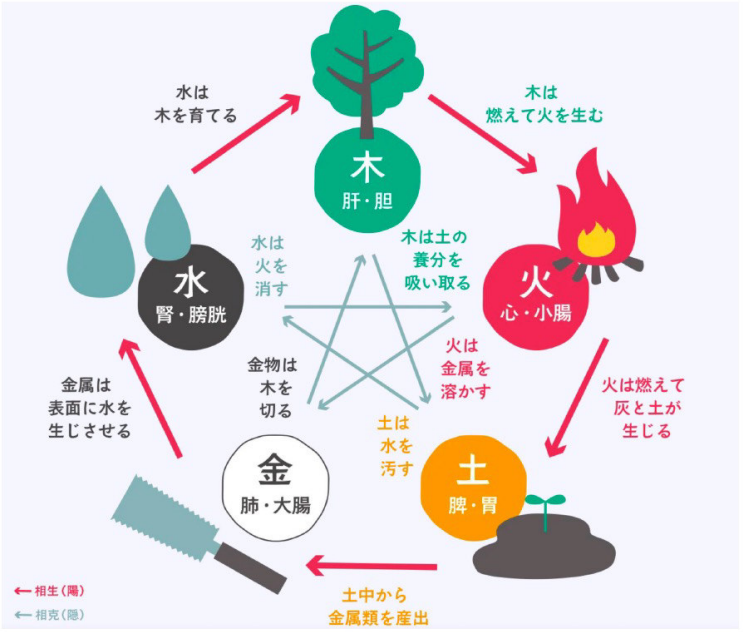
MITRE在2013年推出了ATT&CK™ 模型，它的全称是 Adversarial Tactics, Techniques, and Common Knowledge (ATT&CK)，它是一个站在攻击者的视角来描述攻击中各阶段用到的技术的模型。

MITRE’ s Adversarial Tactics 战术, Techniques 技术, and Common Knowledge (ATT&CK) is a curated knowledge base and model for cyber adversary behavior, reflecting the various phases of an adversary’ s attack lifecycle and the platforms they are known to target.

ATT&CK 威胁框架

初始访问	执行	持久化		提权	防御规避		凭证访问	发现	横向移动	收集	命令与控制	数据渗出	影响
水坑攻击	利用AppleScript	利用.bash_profile和.bashrc	利用LSASS 驱动程序	提纵访问令牌	提纵访问令牌	安装根证书	提纵账户	发现账户	利用AppleScript	捕获音频	利用常用端口	自动渗出数据	删除账户权限
利用面向公众的应用程序	利用CMSTP	利用辅助功能	修改现有服务	借助辅助功能	填充二进制文件	利用InstallUtil	查看Bash历史	发现应用程序窗口	利用应用程序部署软件	自动收集	通过可移动介质通信	压缩数据	损毁数据
利用外部远程服务	利用命令行	提纵账户	Netsh Helper DLL	利用AppCert DLL（注册表项）	利用BITS服务	利用Launchctl	暴力破解	发现浏览器书签	利用组件对象模型（COM）和分布式	收集剪贴板数据	利用连接代理	加密数据	造成恶劣影响的数据加密
添加硬件	利用HTML编译文件	利用AppCert DLL（注册表项）	新建服务	利用AppInit DLL（注册表项）	绕过用户账户控制（UAC）	LC_MAIN劫持	凭证转储	发现域信任	利用远程服务漏洞	收集信息库数据	使用自定义C2协议	限制传输数据大小	网页内容置换攻击
通过可移动介质复制	利用组件对象模型（COM）和分布式	利用AppInit DLL（注册表项）	启动Office应用程序	利用Windows应用程序兼容性框架	清除命令历史	伪冒	获取Web浏览器凭证	发现文件和目录	执行内部鱼叉式钓鱼攻击	收集本地系统数据	使用自定义加密协议	通过备选协议回传	擦除磁盘内容
使用鱼叉式钓鱼附件	利用控制面板项	利用Windows应用程序兼容性框架	路径拦截	绕过用户账户控制（UAC）	利用CMSTP	修改注册表	获取文件中的凭证	扫描网络服务	利用登录脚本	收集网络共享驱动数据	编码数据	通过C2信道回传	擦除磁盘结构
使用鱼叉式钓鱼链接	使用动态数据交换协议（DDE）	利用认证包	修改属性列表	DLL搜索顺序劫持	代码签名	利用Mshta	获取注册表中的凭证	发现网络共享	利用密码哈希认证	收集可移动介质数据	混淆数据	通过其他网络介质回传	篡点侧拒绝服务（DoS）
通过服务执行鱼叉式钓鱼攻击	通过API执行	利用BITS服务	端口敲门	Dylib劫持	投递后编译	删除网络共享连接	利用凭证访问漏洞	网络嗅探	利用Ticket认证	回传数据准备	前置域名	通过物理介质回传	损坏固件
入侵供应链	通过模块加载执行	使用Bootkit	端口监控	提示用户输入合法凭证提权	利用HTML编译文件	利用NTFS交换数据流（ADS）	强制认证	发现密码策略	利用远程桌面协议	收集电子邮件	使用域名生成算法（DGA）	定时传输	禁止系统恢复
利用受信关系	利用主机软件漏洞	添加浏览器扩展插件	利用PowerShell配置文件	利用事件监控守护进程	利用组件固件	混淆文件或信息	利用Hook	发现主机接入设备	拷贝远程文件	输入捕捉	使用备用信道		网络侧拒绝服务（DoS）
利用有效账户	利用图形用户界面（GUI）	更改默认文件关联	利用Rc.common文件	利用漏洞提权	组件对象模型（COM）劫持	伪造父进程	输入捕捉	发现权限组	利用远程服务	浏览器中间人攻击（MitB）	利用多跳代理		资源劫持
	利用InstallUtil	利用组件固件	重启应用程序	额外窗口内存注入（EWMI）	利用连接代理	修改属性列表	欺骗用户输入凭证	发现进程	通过可移动介质复制	获取屏幕截图	创建多级信道		操纵运行时数据
	利用Launchctl	组件对象模型（COM）劫持	冗余访问	利用文件系统权限漏洞	利用控制面板项	端口敲门	使用Kerberoasting技术	查询注册表	共享Webroot目录	捕获视频	使用多协议通信		禁用服务
	利用linux本地任务调度	创建账户	添加注册表运行键/启动文件夹项	利用Hook	使用DCShadow技术	Process Doppelgänging（仿冒合法进	利用Keychain	发现远程系统	SSH劫持		使用多层加密		操纵本地存储数据
	利用LSASS驱动程序	DLL搜索顺序劫持	利用计划任务	映像劫持	反混淆/解码文件或信息	替换进程内存	LLMNR/NBT-NS投毒和中继	发现安全软件	污染共享内容		端口敲门		系统关机/重启
	利用Mshta	Dylib劫持	利用屏幕保护程序	启动守护进程	禁用安全工具	进程注入	网络嗅探	发现软件	利用系统中的第三方软件		利用远程访问工具		操纵传输中的数据
	利用PowerShell	利用事件监控守护进程	利用SSP DLL（注册表项）	新建服务	DLL搜索顺序劫持	冗余访问	利用Password Filter DLL	发现系统信息	利用Windows管理员共享		拷贝远程文件		
	利用Regsvcs/Regasm	利用外部远程服务	利用服务器软件组件	伪造父进程	DLL旁路加载	利用Regsvcs/Regasm	收集私钥	发现系统网络配置	利用Windows远程管理服务		使用标准应用层协议		
	利用Regsvr32	利用文件系统权限漏洞	利用服务注册表权限漏洞	路径拦截	按条件执行	利用Regsvr32	利用Securityd内存	发现系统网络连接			使用标准加密协议		
	利用Rundll32	隐藏文件和目录	利用Setuid和Setgid位	修改属性列表	利用漏洞规避防御	使用Rootkit	窃取Web会话Cookie	发现系统所有者/用户			使用标准非应用层协议		
	利用计划任务	利用Hook	修改快捷方式	端口监控	额外窗口内存注入（EWMI）	利用Rundll32	双因子认证拦截	发现系统服务			利用不常用端口		
	使用脚本	利用Hypervisor	会话发起协议（SIP）和受信提供商劫持	利用PowerShell配置文件	修改文件和目录权限	使用脚本		发现系统时间			利用Web服务		
	利用windows服务	映像劫持	利用启动项	进程注入	删除文件	执行签名的二进制文件代理		虚拟化/沙箱逃逸					
	利用签名的二进制文件代理执行	利用内核模块和扩展	利用系统固件	利用计划任务	文件系统逻辑偏移	执行签名的脚本代理							
	利用签名的脚本代理执行	启动代理	利用Systemd服务	利用服务注册表权限漏洞	绕过Gatekeeper	会话发起协议（SIP）和受信提供商劫持							
	利用Source命令	启动守护进程	利用Windows时间服务	利用Setuid和Setgid位	修改组策略	软件加壳							
	加入空格隐藏扩展名	利用Launchctl	利用Trap命令	SID历史注入	隐藏文件目录	加入空格隐藏扩展名							
	利用系统中的第三方软件	添加LC_LOAD_DYLIB	利用有效账户	利用启动项	隐藏用户	模板注入							
	利用Trap命令	利用linux本地任务调度	使用Web Shell	利用Sudo命令	隐藏窗口	修改文件时间戳							
			利用Windows事件订阅										

网络攻防
相生相克



博弈冲突
打破次元

以前防御措施较为被动

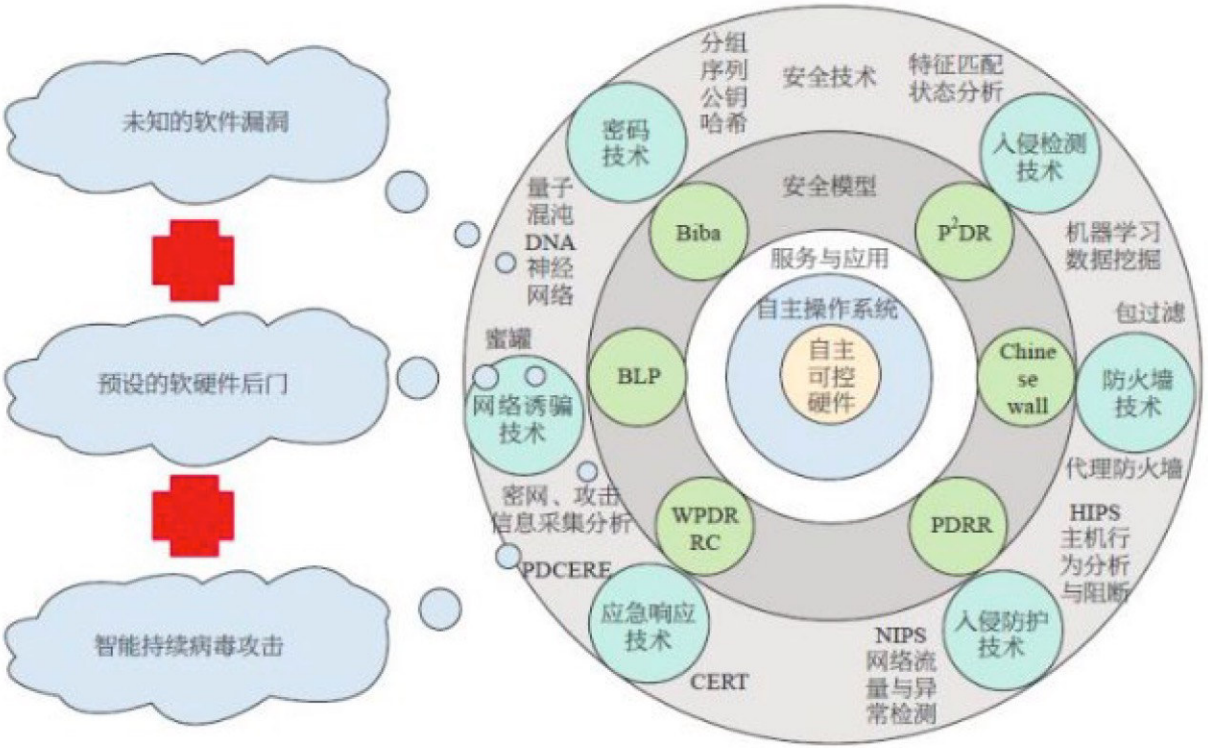


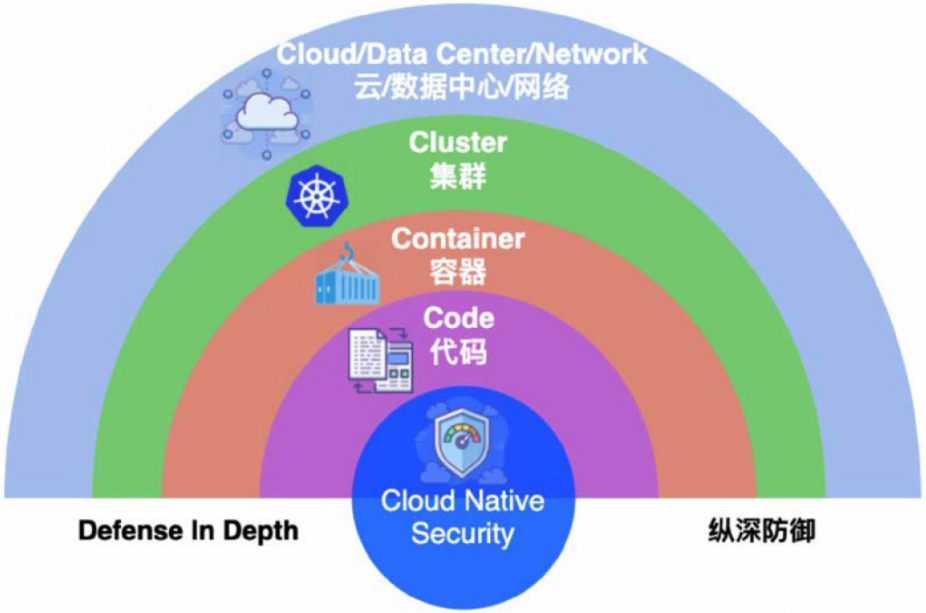
图1 现行的被动安全防御体系

I. 网络攻防 相生相克

II. 博弈冲突 打破次元

III. 人间真实 不得不防

IV. 未来战争 破局之道



防御应对措施——改变游戏规则

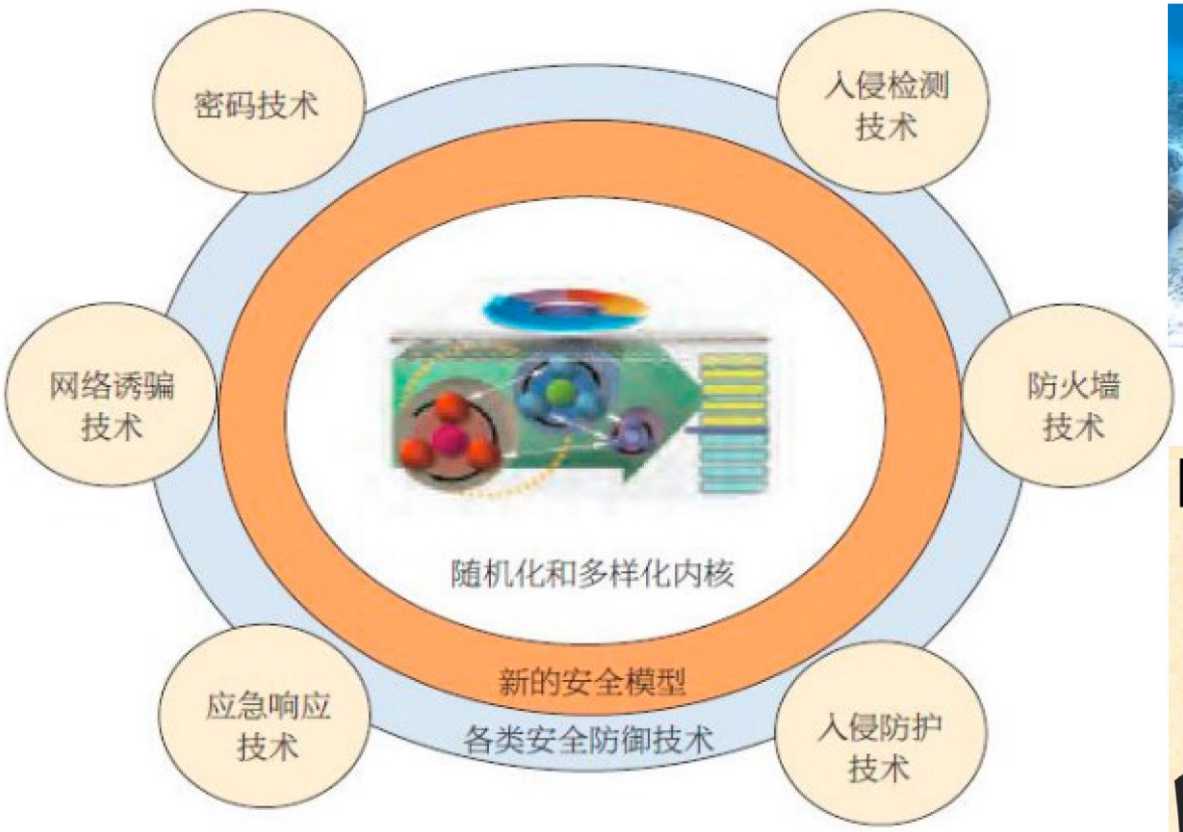
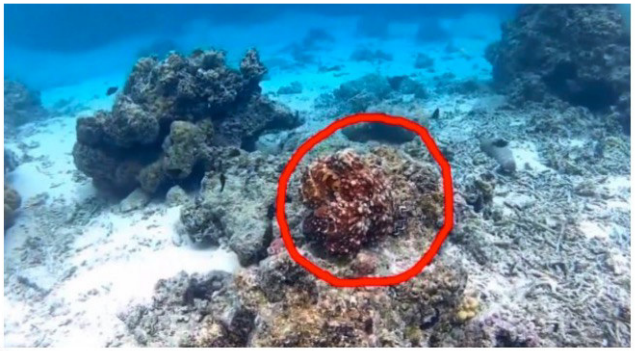
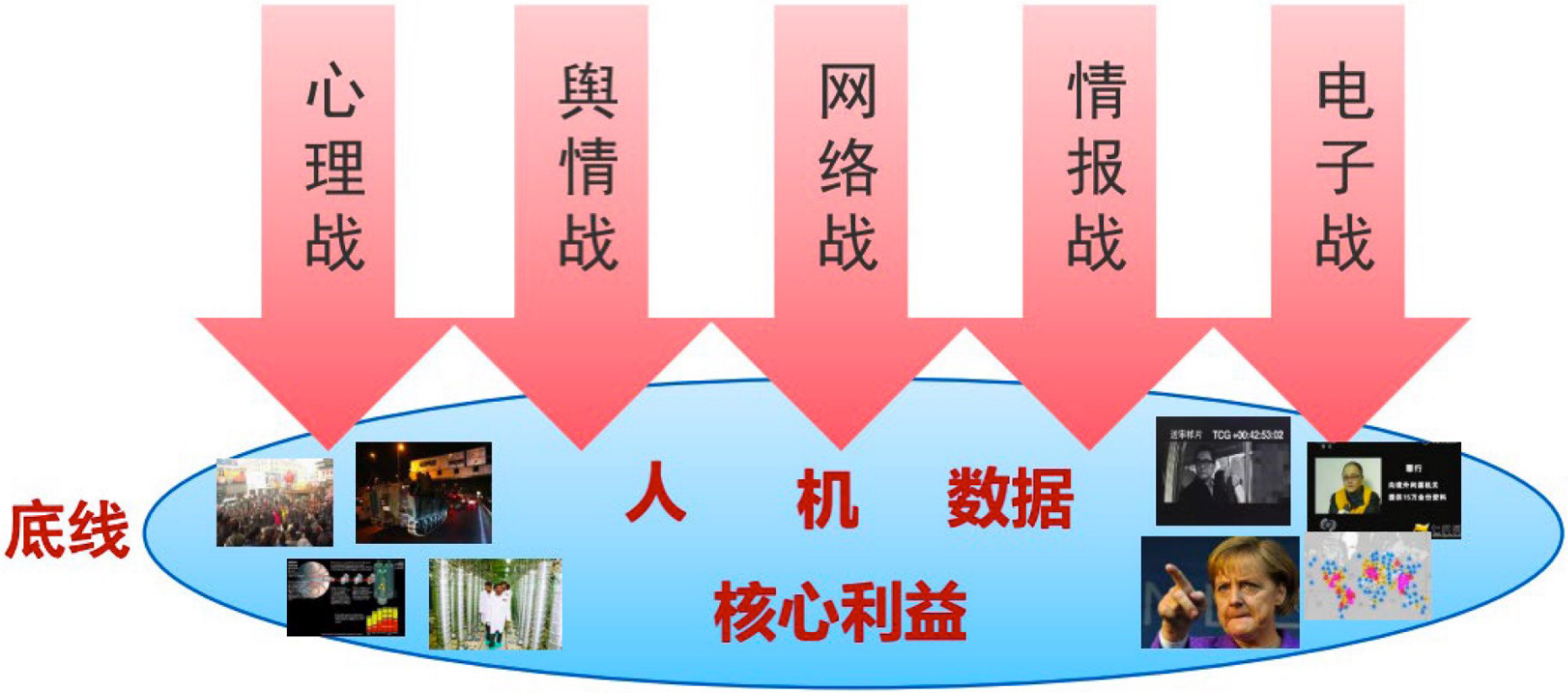


图2 拟态安全主动防御体系基础架构

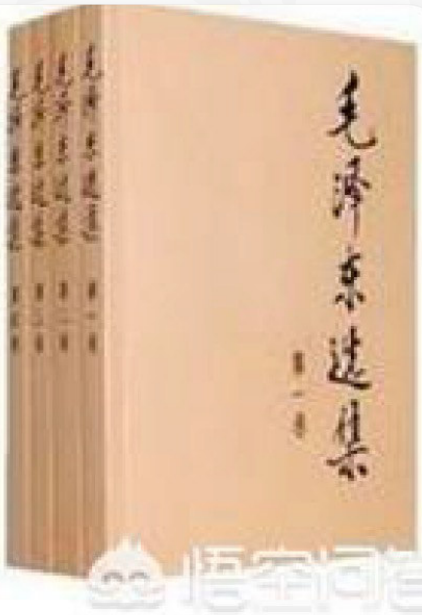


极限施压情况下的网络空间安全底线思维



用魔法打败魔法
人机合一的信息安全
是构建未来大同世界的基石

武器是战争的重要因素，但不是决定性的因素，决定的因素是人不是物。



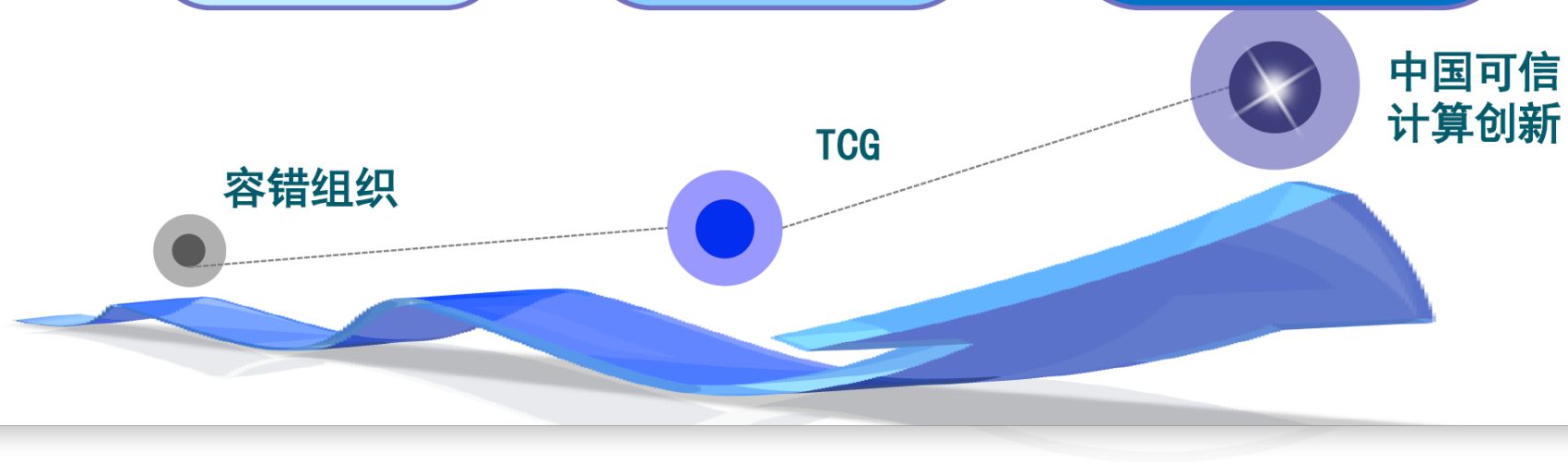
从信息确保、软件确保到系统确保

——寻找可信信息系统之路

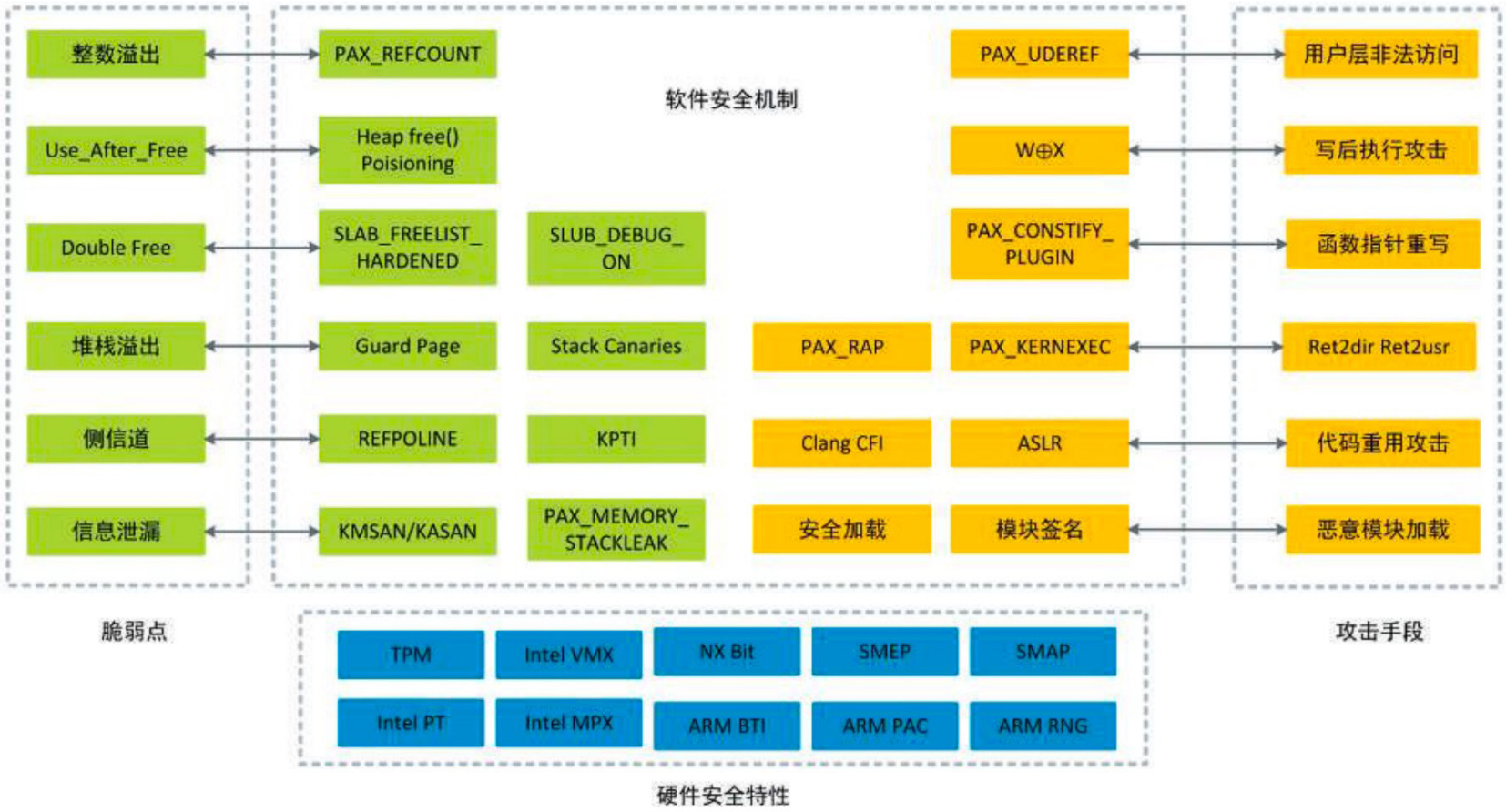
方滨兴, 2010年1月8日

可信计算

	可信1.0（主机）	可信2.0（PC）	可信3.0（网络）
特性	主机可靠性	节点安全性	系统免疫性
对象	计算机部件	PC单机为主	节点虚拟动态链
结构	冗余备份	功能模块	宿主+可信双节点
机理	故障诊查	被动度量	主动免疫
形态	容错算法	TPM+TSS	可信免疫架构



内置安全


整数溢出
Use_After_Free
Double Free
堆栈溢出
侧信道
信息泄漏
脆弱点
PAX_REFCOUNT
Heap free() Poisioning
SLAB_FREELIST_ HARDENED
Guard Page
REFPOLINE
KMSAN/KASAN
SLUB_DEBUG_ ON
Stack Canaries
KPTI
PAX_MEMORY_ STACKLEAK
软件安全机制
PAX_RAP
Clang CFI
安全加载
PAX_UDEREF
W⊕X
PAX_CONSTIFY_ PLUGIN
PAX_KERNEXEC
ASLR
模块签名
用户层非法访问
写后执行攻击
函数指针重写
Ret2dir Ret2usr
代码重用攻击
恶意模块加载
攻击手段
TPM
Intel VMX
NX Bit
SMEP
SMAP
Intel PT
Intel MPX
ARM BTI
ARM PAC
ARM RNG
硬件安全特性

网络空间安全基础最佳实践

1. CS自学指南 zhongyinmin@pku.edu.cn https://csdiy.wiki/
2. 给表弟的Web安全入门建议 https://sosly.me/index.php/2017/07/17/studywebsec/
3. Pikachu漏洞测试平台 https://github.com/zhuifengshaonianhanlu/pikachu
4. Web安全之机器学习入门 https://github.com/duoergun0729/1book

计算机科学基础：

- **编程思维、全栈思维、逻辑思维**
- **实践能力、工程能力、动手能力**

网络空间安全：

- **完整性、保密性、可用性、真实性**
  - ▶系统安全、软件安全、网络安全、网络空间安全

攻防博弈、内置安全：

- **漏洞缺陷的本质原因**
- **新设计、新思维、新方法**

感谢批评指正

THANKS

lixeon.lij@gmail.com

中国科学院信息工程研究所
INSTITUTE OF INFORMATION ENGINEERING,CAS