**Web进程链恶意入侵及SSH异常操作检测技术研究需求说明**

1. **问题背景**

Web进程链恶意入侵及SSH异常操作检测技术研究项目将应用在自研的主机型入侵检测系统，通过分析程序访问的内存、文件系统、日志存储等信息，结合算法能力，精确检测程序异常或恶意入侵行为，以为系统提供主动性的防御机制。

该项技术可通过捕获Linux系统下用户态和内核态通信过程中产生的信息，挖掘更深层次的特征，以进程为粒度进行入侵或异常行为的检测，能够抵御新型攻击手段，特别是隐蔽而又高危的APT攻击。

1. **问题描述**

针对Web进程链/SSH异常操作序列，检测上下文关系中的子进程异常。

* 1. Web进程链：同一Web服务应用的进程调用关系中，父进程为JAVA，其派生的所有子进程序列。
  2. SSH操作序列：同一台主机系统中，父进程为TTY，其派生的所有子进程序列。（可以当作是管理员的输入）
  3. 异常定义：根据历史行为建立基线模型，识别有异于正常行为的异常行为。
* 进程链举例：

正常样本：以下进程链为Tomcat服务器启动时派生的子进程：

1) p\_process(java) -> c\_process1

2) p\_process(java) -> c\_process2

3) p\_process(java) -> c\_process3

恶意样本：当服务器被黑客入侵时，正常进程命令中会夹杂恶意的命令进程，如：

1) p\_process(java) -> c\_process1

2) p\_process(java) -> malicious\_process1

3) p\_process(java) -> malicious\_process2

4) p\_process(java) -> c\_process2

5) p\_process(java) -> malicious\_process3

6) p\_process(java) -> c\_process3

* 攻击场景举例：

场景一：执行系统命令

1) p\_process(java) -> /opt/a.sh cmd1 cmd2 cmd3 正常样本

2) p\_process(java) -> cat /etc/passwd 恶意样本

3) p\_process(java) -> ifconfig恶意样本

4) p\_process(java) -> /opt/ b.sh cmd4 cmd5 cmd6 正常样本

5) p\_process(java) -> cd /tmp;ls –al –al恶意样本

6) p\_process(java) -> /opt/ b.py cmd7 cmd8正常样本

场景二：命令注入

1) p\_process(java) -> /opt /a.sh cmd1 cmd2 cmd3正常样本

2) p\_process(java) -> /opt /b.sh cmd4正常样本

3) p\_process(java) -> /opt /a.sh cmd1 cmd2 cmd3;ifconfig 恶意样本

4) p\_process(java) -> /opt /a.sh cmd1 cmd2 cmd3|cat /etc/passwd 恶意样本

5) p\_process(java) -> /opt /b.sh cmd4$(cat /etc/passwd) 恶意样本

6) p\_process(java) -> /opt /c.sh cmd5 cmd6 cmd7正常样本

1. **验收标准：**
   1. 交付件：
      * 原型系统
      * 算法源码（Python）
      * 设计文档及验证报告
   2. 功能目标：
      * 可对多个业务产生的Web进程链建立正常基线检测异常进程。
      * 可针对业务调整，模型具有动态更新能力，或者根据新的业务行为进行历史相关性学习。
      * 在检测出已知威胁的基础之上，需对未知的威胁有检测能力。

**4. 附件：样本数据说明**

（1）数据仅包含白样本，建议使用无监督模型。需人为构造黑样本来检测模型的效果。

（2）数据为从不同Agent采集的用户态和内核态通信进程信息，来源于/proc路径下的相关信息，以JSON格式保存，具体字段意义如下：

CreateTime: 进程创建时间

ppid：父进程id

pname：父进程名称

pcmdline：父进程命令

pexe：父进程执行路径

cpid：子进程id

cname: 子进程名称

ccmdline：子进程命令

cexe: 子进程执行路径

（3）目前Web进程链异常检测技术只需检测父进程为Java的子进程链异常。针对不同Web服务，如Tomcat、Nginx等，需要区分开。