1. 引言（Introduction）
   - 研究背景：Web应用面临的安全威胁及入侵恢复的必要性。
   - 现有入侵恢复系统的局限性：需要修改源代码、仅支持单一数据存储。
   - 研究目标：提出Sanare，一种无需修改源代码的插件式入侵恢复系统。
2. 系统模型与威胁模型（System Model & Threat Model）
   - 系统模型：
     - Sanare 是一个用于恢复 Web 应用程序的入侵恢复系统
     - 支持多种数据存储（数据库、文件系统、外部Web服务）。
   - 威胁模型：
     - 攻击仅通过HTTP请求进行（t1）
     - 错误的HTTP请求可能导致数据库、文件系统或外部Web服务破坏数据（t2, t3）。
     - 攻击者无法篡改Sanare或应用代码（t4, t5）
3. Sanare系统架构（Sanare Architecture）

   application container （只能通过Sanare请求）和 Sanare container

   两个数据库：日志存储（各类代理的数据整合，便于搜索和分析）和样本存储（用于深度学习）

   Sanare container核心组件：
   - HTTP代理：拦截并记录HTTP请求。
   - 数据库代理：记录数据库操作。
   - 文件系统代理：记录文件操作。
   - Web服务代理：记录外部Web服务请求。
   - Sanare管理器：协调恢复过程。

4. Matchare：深度学习匹配方案（Matchare: Deep Learning Matching Scheme）
   - 问题定义：匹配HTTP请求与数据操作（数据库、文件系统、Web服务）。
   - 模型设计：
     - 使用深度卷积神经网络（Deep CNN）。
     - 特征提取：HTTP请求特征、数据操作特征（如操作类型、参数值等）。
     - 预处理：归一化（对数值变量进行缩放，使其均位于较小的区间内的过程，提高训练效率）和分类特征编码。

5. Sanare的执行阶段（Execution Phases）

   学习阶段：
   - 模拟HTTP请求并记录操作（每一个模拟的请求都会引发操作并被记录在数据库中）
   - 训练Matchare模型（要训练三个不同的模型，帮助其理解执行相同 HTTP 请求时数据库语句、文件系统操作和 Web 服务请求的差异）
   - 首先，从样本数据库中获取样本，然后对其进行预处理（参见 4.3 节），最后对模型进行训练（如本节所述）。训练好的模型随后被存储起来，以便在恢复过程中用于对 {HTTP 请求，数据操作} 对进行分类。

- 正常阶段：
  - 记录所有操作，定期备份。
- 损害评估阶段：
  - 使用Matchare识别恶意操作（扫描日志数据库并把入侵期间的操作与恶意请求检测是否匹配）
  - 构建依赖图分析间接影响的操作（验证 HTTP 请求之间的依赖关系，因为它们将用于推断它们引发的数据库语句之间的依赖关系）
- 损害修复阶段：
  - 执行补偿操作：数据库（不同数据库有不同的适配器）文件系统（结合使用了存储在外部云存储中的文件系统备份和存储在日志数据库中的文件系统操作）Web服务（提供一个允许撤消先前执行的操作的接口）
  - 保证数据一致性（如果仅执行部分补偿操作，则应用程序将变得不一致，其状态的一部分反映攻击的影响，另一部分反映恢复的影响）

6. 实现与部署

Sanare基于Python和Flask实现，使用MongoDB存储日志

实验评估显示，Sanare在WordPress、GitLab和ownCloud上性能开销低于18%，恢复单次入侵仅需数秒。

7. 实验评估

性能：平均请求处理速度下降12%–17%。

存储：100万次操作日志占用17–28 GB存储。

恢复时间：单次入侵恢复1.8–6秒，100次入侵恢复约3分钟。

匹配准确率：Matchare的精确度和召回率均超过97.5%。

8. 现有入侵恢复系统的分类与比较。

Sanare的创新点：支持多数据存储、无需修改代码、深度学习匹配。

9. 结论（Conclusion）

- Sanare的核心贡献：
  - 插件式入侵恢复系统。
  - Matchare深度学习匹配方案。
  - 支持多数据存储和外部Web服务。
- 未来方向：优化性能，扩展应用场景。