基于Squid+ICAP+PyICAP的简易DLP系统设计方案

1. 系统架构设计

```
[客户端]
   ↓（HTTP/HTTPS流量）
[Squid代理]
   ↓（ICAP协议）
[PyICAP服务]
   ↓（内容分析）
[DLP规则引擎]
```

2. 核心组件说明

A. Squid代理层

- 配置ICAP重定向：

```
icap_enable on
icap_service service_req reqmod_precache icap://127.0.0.1:1344/reqmod
icap_service service_resp respmod_precache icap://127.0.0.1:1344/respmod
adaptation_access service_req allow all
adaptation_access service_resp allow all
```

B. PyICAP服务层

- 使用Python实现的ICAP服务器框架
- 关键处理逻辑：
  - REQMOD：检查出站请求（上传文件等）
  - RESPMOD：检查入站响应（下载内容等）

C. DLP规则引擎

- 规则存储：JSON/YAML格式的规则文件
- 检测方式：

```
{
  "rule_id": "DLP-001",
  "name": "身份证号检测",
  "type": "regex",
  "pattern": "\\d{17}[0-9Xx]",
  "action": "block"
}
```

3. 核心功能设计

A. 内容检测模块

1. **关键词匹配**
   - 支持精确匹配（如"商业机密"）
   - 支持通配符（如"机密*文档"）
2. **正则表达式检测**
   - 敏感数据模式：
     - 身份证号：`\d{17}[0-9Xx]`
     - 银行卡号：`\d{16}|\d{19}`

- 手机号：`1[3-9]\d{9}`

3. **文件类型控制**
   - 扩展名黑名单：.zip, .rar等
   - 文件头检测（Magic Number）

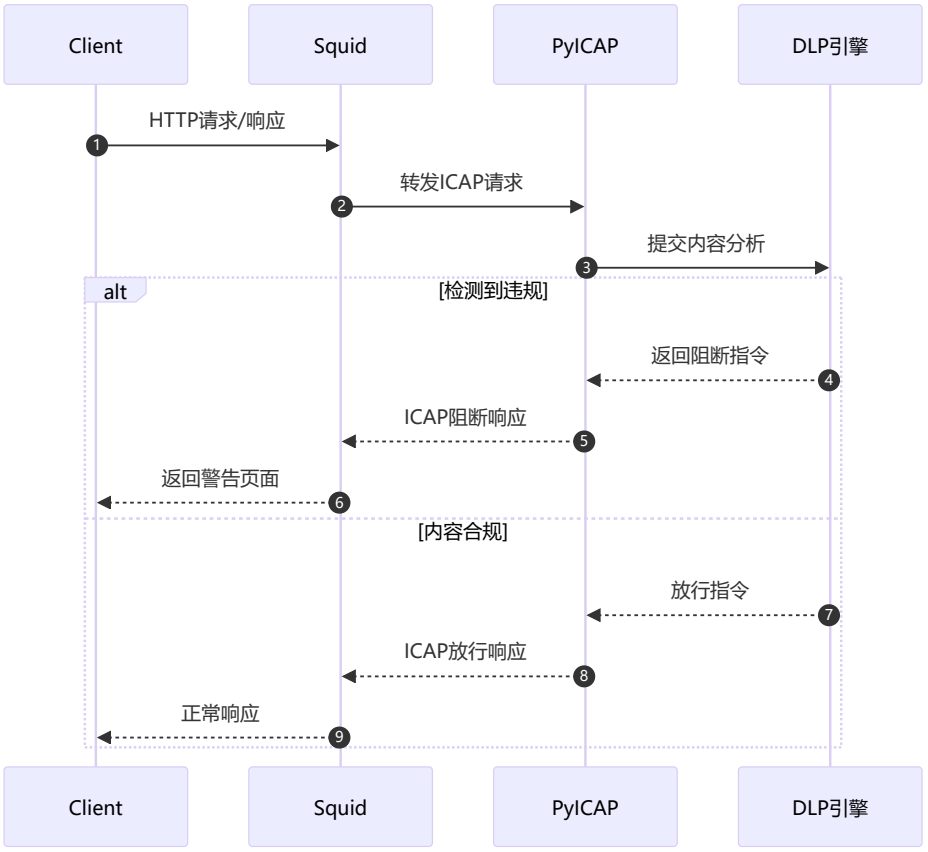
B. 处理动作设计		
触发条件	**动作类型**	**实现方式**
低风险匹配	记录日志	生成审计记录
中风险匹配	替换内容	返回警告页面
高风险匹配	阻断连接	返回403响应

C. 策略配置示例

```
policies:
  - name: 财务数据保护
    apply_to: ["finance/*"]
    rules:
      - type: keyword
        values: ["财务报表", "年度预算"]
        action: block

  - name: 个人信息保护
    apply_to: ["*"]
    rules:
      - type: regex
        pattern: "\d{17}[0-9Xx]"
        action: replace
        replace_with: "[ID_NUMBER_REDACTED]"
```

4. 数据处理流程



5. 扩展性设计

1. **插件机制**：支持动态加载检测模块
2. **API接口**：提供RESTful管理接口
3. **机器学习集成**：预留接口支持NLP模型

6. 性能优化措施

- 内容分块处理（避免大文件内存溢出）
- 正则表达式预编译
- 热点规则缓存

7. 日志审计设计

字段	说明
timestamp	事件时间
src_ip	源地址
matched_rule	触发规则
content_sample	内容片段(脱敏)
action_taken	执行动作

该设计方案通过Squid的ICAP接口实现网络流量拦截，利用PyICAP进行协议解析，结合可配置的DLP规则引擎完成内容检测，在不影响现有网络架构的情况下实现轻量级数据防泄露功能。