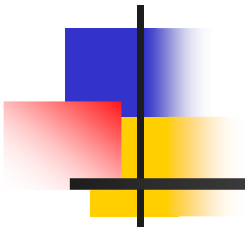


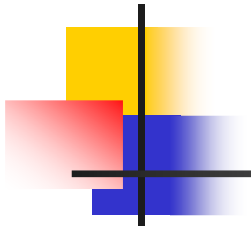
Protótipo de software para a monitoração de pacotes em uma conexão TCP/IP em ambientes Linux

Orientando:

Jorge L. Pompermayer Junior

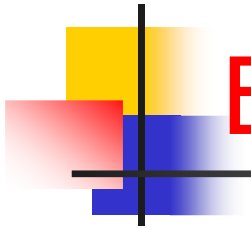
Orientador:

Francisco Adell Péricas



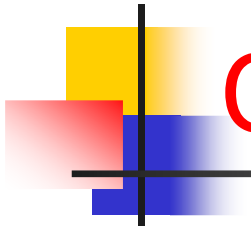
Roteiro

1. Escolha do tema
2. Introdução
3. Ethernet e Arquitetura TCP/IP
4. Segurança e *Sniffers*
5. Especificação do protótipo
6. Implementação do protótipo
7. Conclusões e considerações finais



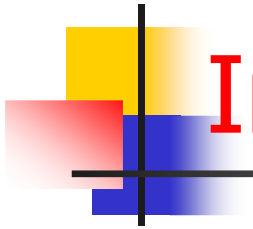
Escolha do Tema

1. Porque Redes?
   - Utilização profissional
   - Opção pessoal
2. Porque Segurança?
   - Utilização profissional
   - Curiosidade
   - Redes e segurança andam juntos
3. Porque Linux?
   - Características do SO
   - Utilização no dia-a-dia
   - Curiosidade



Objetivos

- Monitorar pacotes trafegando em uma rede *Ethernet*
- Interceptar e interpretar pacotes TCP/IP
- Analisar dados contidos nos pacotes
- Identificar situações suspeitas
- Armazenar ou não as informações monitoradas

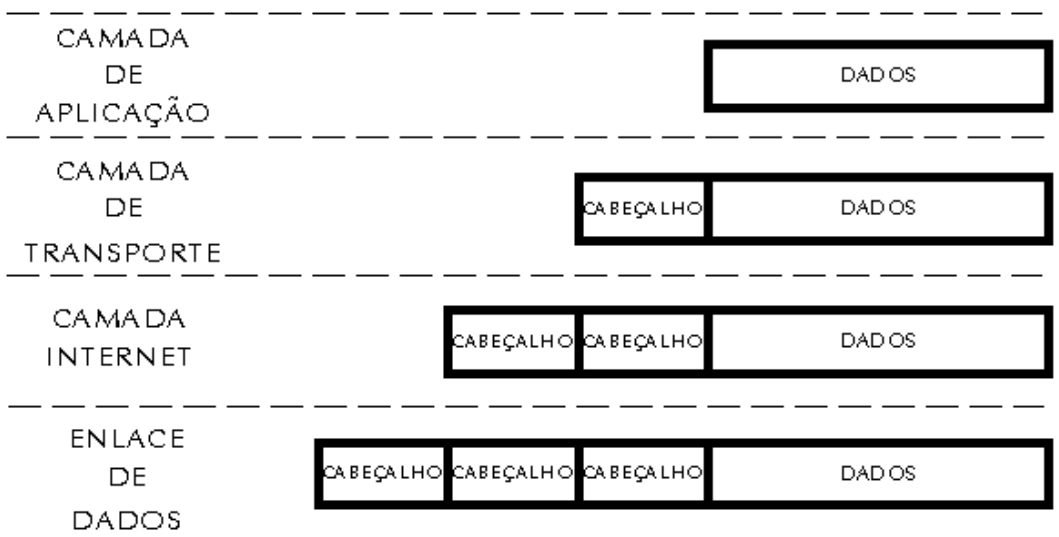


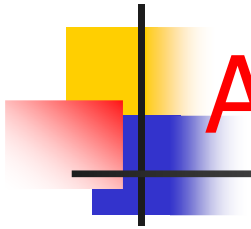
Introdução

- Necessidade crescente de compartilhar informações remotamente
- Riscos de tentativas de invasão e ataques por *hackers*
- Surge a necessidade da utilização de ferramentas de monitoração

Arquitetura TCP/IP

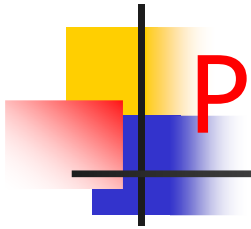
- Dividido em camadas:





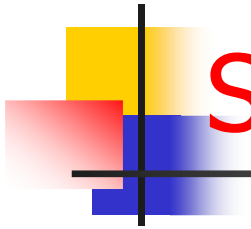
Arquitetura TCP/IP (cont.)

- Aplicação: protocolos de alto nível
- Transporte: comunicação entre hosts fim-a-fim, sendo orientados ou não à conexão: TCP, UDP
- Internet: roteamento e entrega dos pacotes IP: IP e ICMP
- Enlace: encapsular pacotes da camada Internet para o padrão da rede associada e vice-versa: *Ethernet*



Protocolos monitorados

- **IP:** transmissão de datagramas entre origem e destino
  - Não orientado à conexão
  - Não assegura seqüência de entrega dos datagramas
- **ICMP:** indica ocorrência de problemas no transporte de algum datagrama ou serve à operações de controle
- **UDP:** não orientado à conexão
  - Não assegura seqüência de entrega dos datagramas
- **TCP:** Orientado à conexão
  - Entrega confiável dos pacotes
  - Entrega seqüencial dos pacotes
  - Controle de fluxo
  - Recuperação de erros

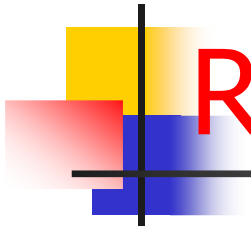


Segurança e Vulnerabilidades

- Descoberta de novas vulnerabilidades a todo momento;
- *Hackers X Crackers;*
- Atualização técnica por parte do administrador de redes;
- Falhas, problemas, configurações errôneas;
  - Falta de padrões técnicos;
  - Vulnerabilidades dos produtos;
  - Vulnerabilidades nas configurações;
- Políticas de operação;
- Técnicas de invasão;

Sniffers

- Ferramenta para captura de pacotes trafegando em uma rede
- Originalmente criada para depuração de problemas de rede
- Utilização por *hackers* para captura de senhas

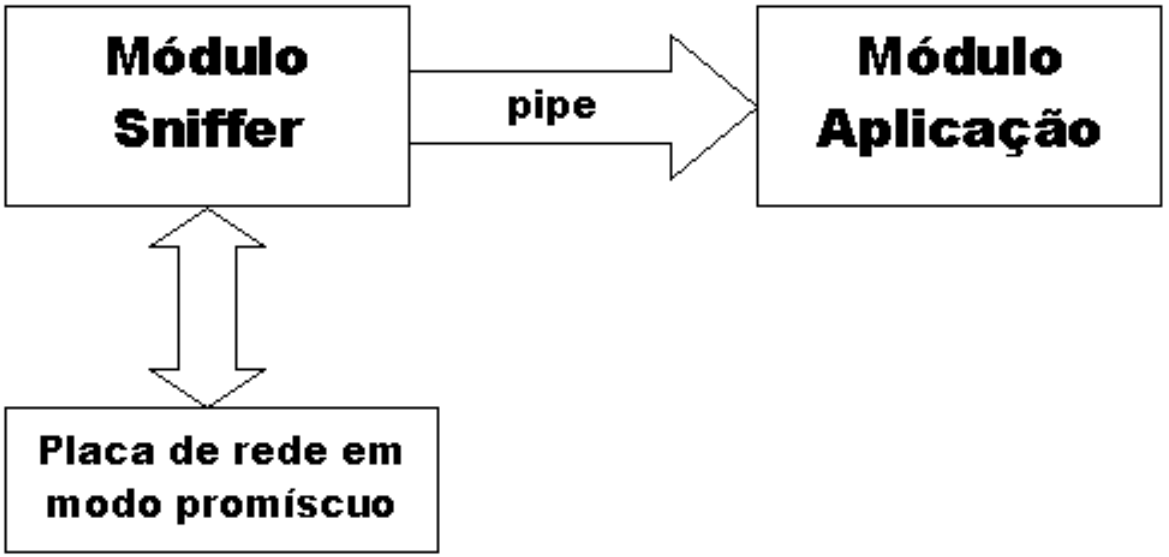


Requisitos do protótipo

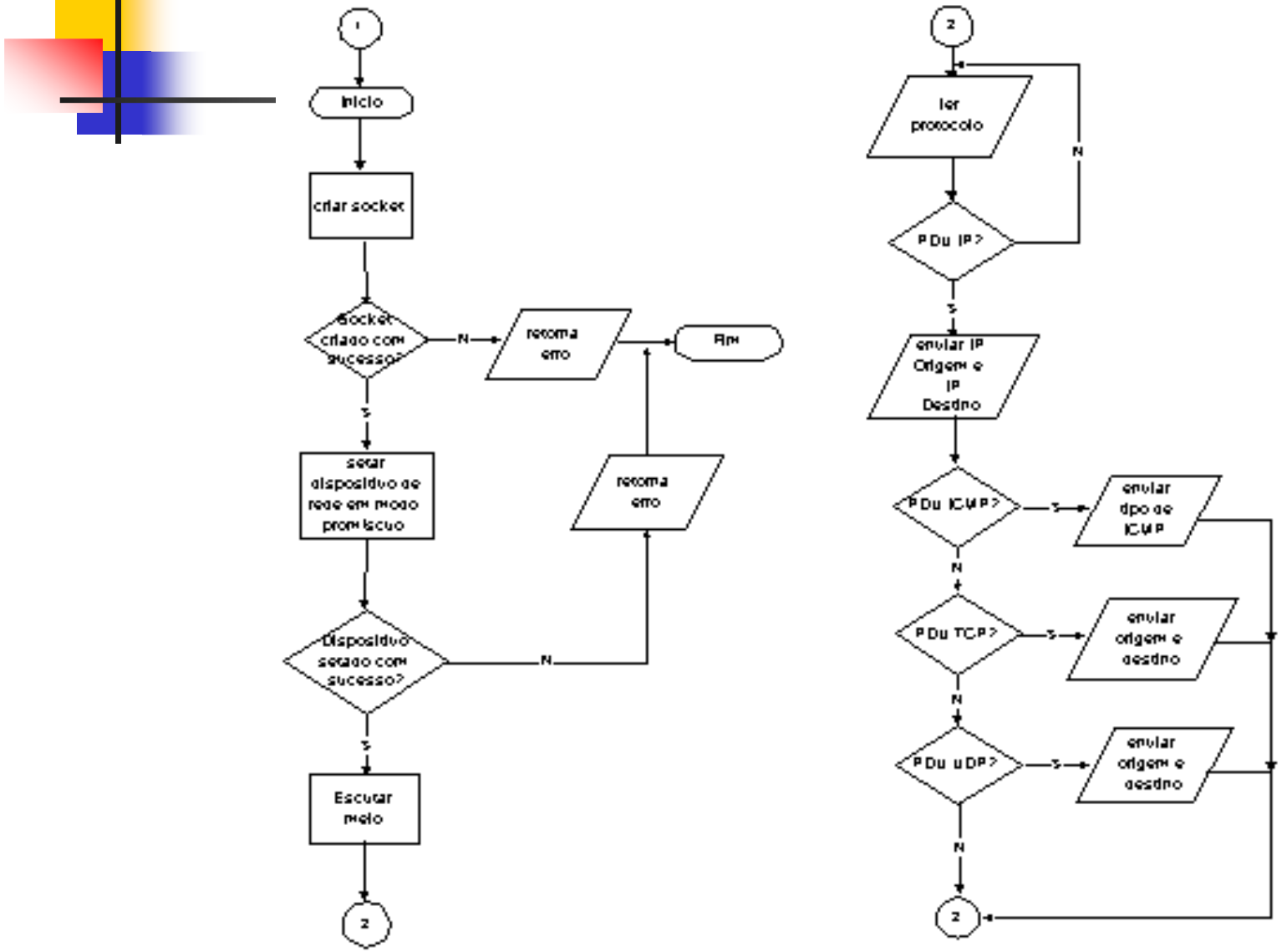
- *Host* linux conectado a uma rede *Ethernet*
- Capturar todos pacotes trafegando na rede e analisá-los conforme configuração de filtros
- Os protocolos a serem analisados pelo protótipo são: IP, ICMP ,TCP e UDP
- Permitir a visualizaçãos dos pacotes capturados
- Gerar logs, que poderão ou não ser armazenados

Especificação

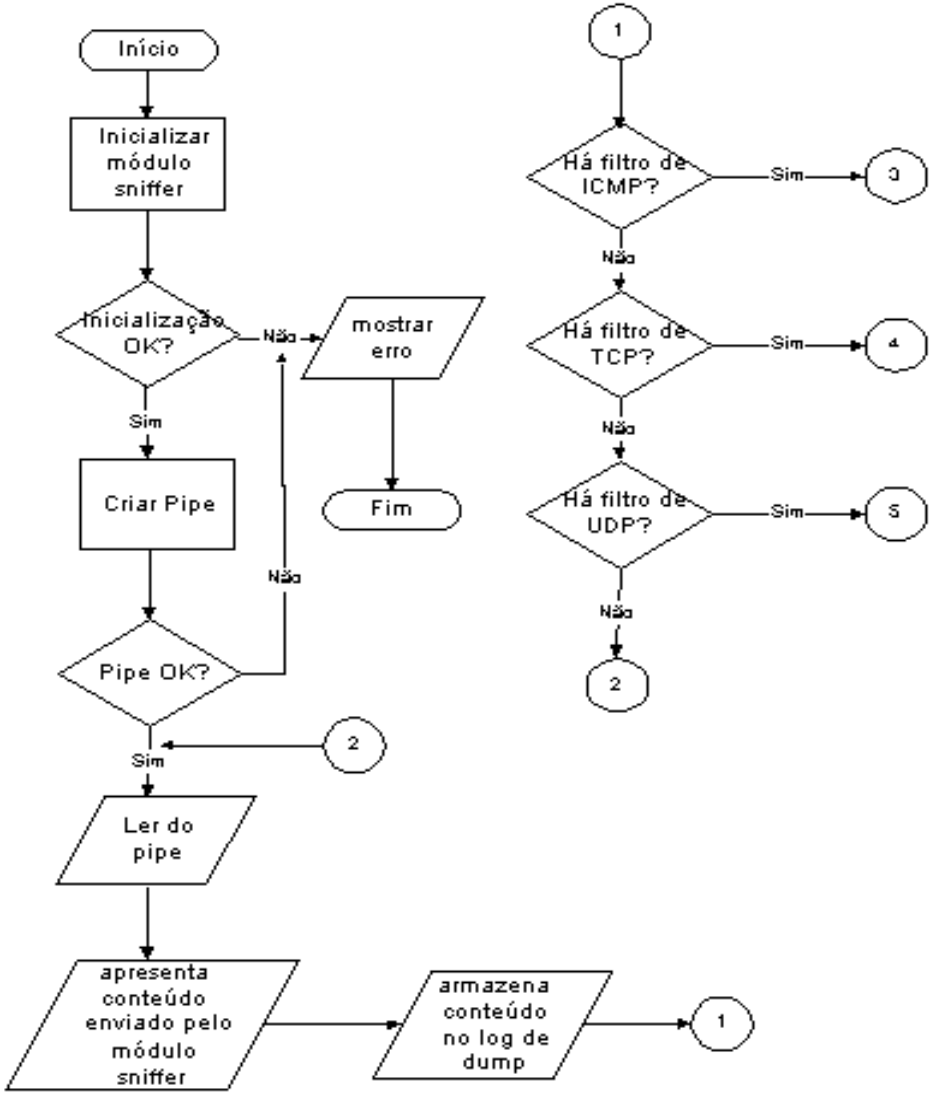
- O protótipo é composto por dois módulos: *sniffer* e aplicação

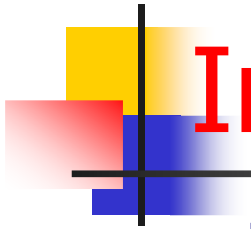


Especificação do módulo *sniffer*



Especificação do módulo aplicação



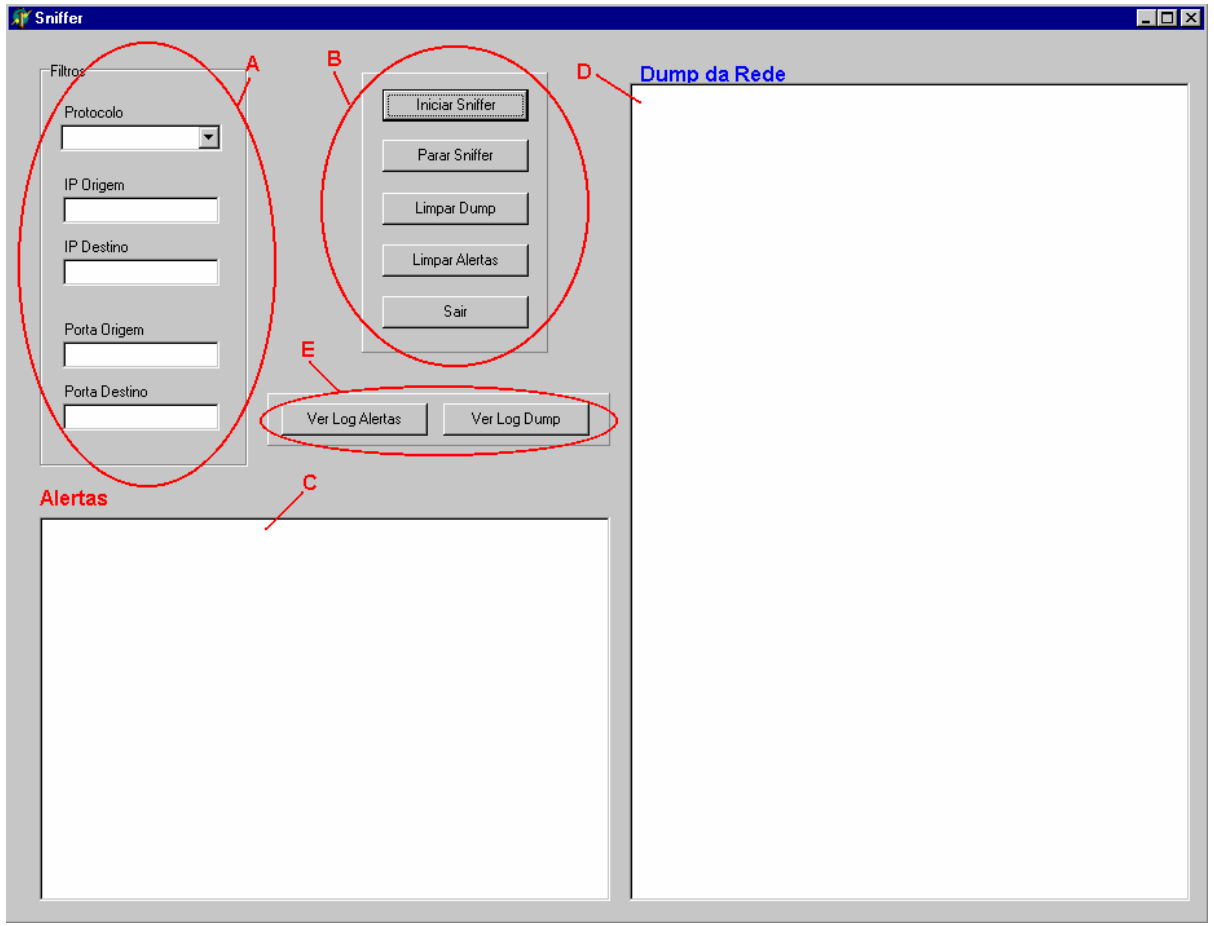


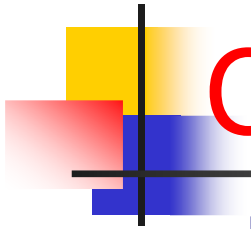
Implementação

- Módulo *sniffer* implementado em C
- Módulo aplicação implementado em *Kylix*
- Utilização de pipe entre os módulos

Implementação

- Tela do módulo aplicação





Conclusões

- Disponibilização de dados
- Atualização constante por parte do administrador
- Utilização de ferramentas adequadas
- Eficácia por parte do protótipo
- Aplicação, na prática, de conceitos teóricos
- Utilização do *Kylix*
- Utilização do compilador C

"Todo sistema é seguro,
até ser invadido pela primeira vez"

Comentários finais

- Dificuldades enfrentadas:
  - Peculiaridades dos protocolos
  - Documentação
- Limitações:
  - Reconhecimento de cabeçalhos dos protocolos IP, ICMP, TCP, UDP
  - Opções limitadas de filtros
- Extensões
  - Monitoração dos conteúdos dos pacotes
  - Expandir para outros protocolos