Um Núcleo de Segurança Distribuído para Suporte a Protocolos Tolerantes a Intrusões

Pan Jieke, Miguel Correia, Nuno Ferreira Neves, Paulo Veríssimo

LASIGE, Faculdade de Ciências da Universidade de Lisboa

7 de Novembro de 2005

Parte I

1. Conceitos básicos
   - Segurança vs Confiabilidade
   - Tolerância a Intrusões
   - Arquitectura genérica do sistema
   - Wormhole

Parte II

2. Concretização do ChongDong
   - Arquitectura do sistema
   - Segurança do ChongDong
   - LIDS
   - Tornar o ChongDong local seguro com o LIDS
   - Proteger acesso à rede privada dos ChongDongs com o LIDS
   - Análise estática de código
   - Serviço ChongDong_TBA

Parte III

3 Exemplo e Resultados
- Consenso Tolerante a Intrusões
- Resultado

Parte IV

4 Conclusão

Parte I

Conceitos básicos

Segurança vs Confiabilidade

- Segurança: ênfase na prevenção.
  - ex. firewalls, controlo de acesso, criptografia...
- Confiabilidade: o sistema continua operacional mesmo que alguns componentes falhem.
  - ex. se o computador de bordo do avião falhar...
- Tolerância a Intrusões: aplicar o paradigma de tolerância a faltas no domínio de segurança.

Segurança vs Confiabilidade

- Segurança: ênfase na prevenção.
  - ex. firewalls, controlo de acesso, criptografia...
- Confiabilidade: o sistema continua operacional mesmo que alguns componentes falhem.
  - ex. se o computador de bordo do avião falhar...
- Tolerância a Intrusões: aplicar o paradigma de tolerância a faltas no domínio de segurança.

Segurança vs Confiabilidade

- Segurança: ênfase na prevenção.
  - ex. firewalls, controlo de acesso, criptografia...
- Confiabilidade: o sistema continua operacional mesmo que alguns componentes falhem.
  - ex. se o computador de bordo do avião falhar...
- Tolerância a Intrusões: aplicar o paradigma de tolerância a faltas no domínio de segurança.

Tolerância a Intrusões

- O que é aplicar o paradigma da tolerância a faltas no domínio de segurança?
  - Assumir e aceitar que o sistema permanece sempre mais ou menos vulnerável.
  - Assumir e aceitar que os componentes do sistema podem ser atacados e que alguns desses ataques terão sucesso.
  - Garantir que o sistema como um todo permanece seguro e operacional, ou seja, que não falha.

Tolerância a Intrusões

- O que é aplicar o paradigma da tolerância a faltas no domínio de segurança?
  - Assumir e aceitar que o sistema permanece sempre mais ou menos vulnerável.
  - Assumir e aceitar que os componentes do sistema podem ser atacados e que alguns desses ataques terão sucesso.
  - Garantir que o sistema como um todo permanece seguro e operacional, ou seja, que não falha.

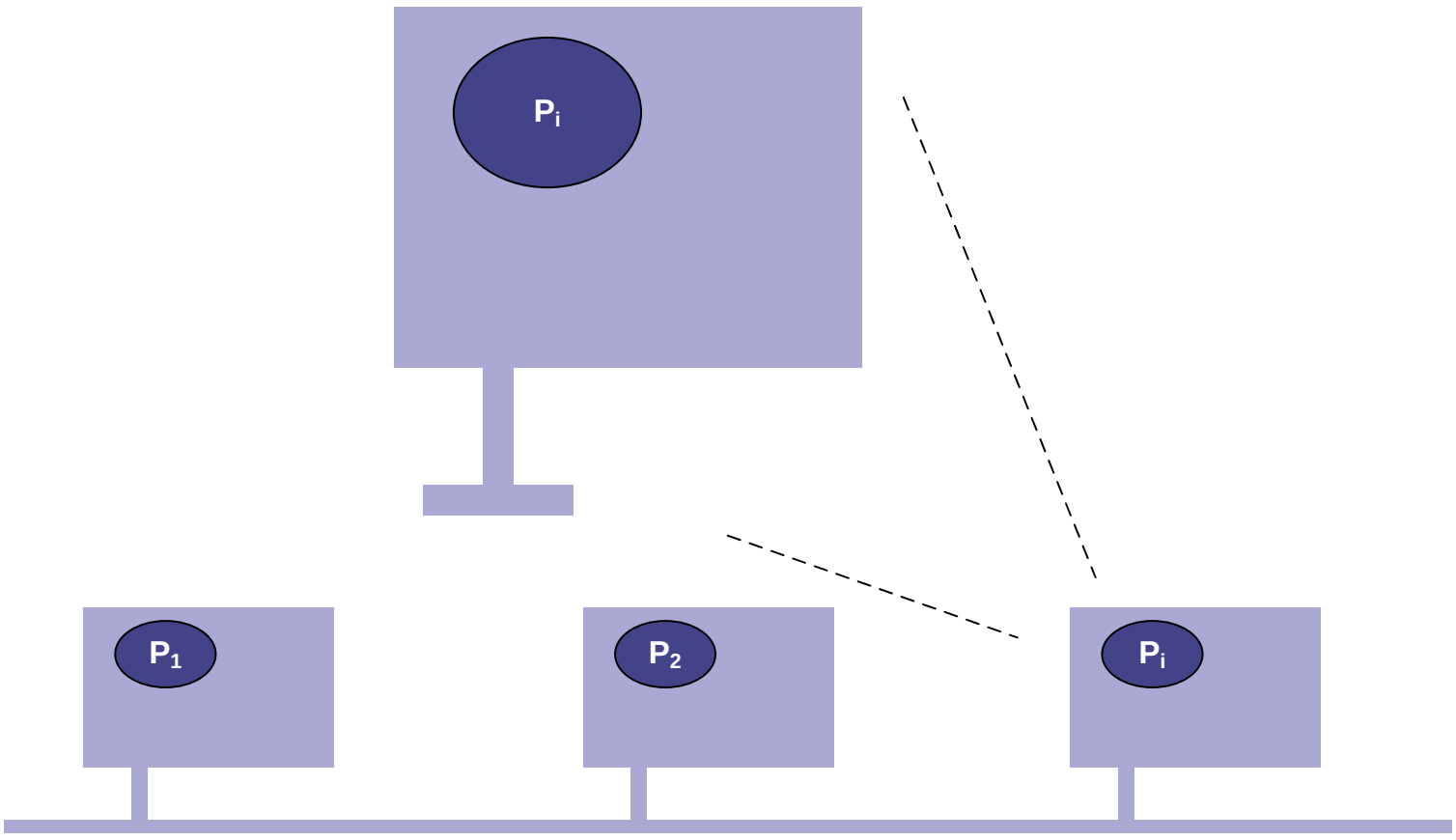
Tolerância a Intrusões

- O que é aplicar o paradigma da tolerância a faltas no domínio de segurança?
  - Assumir e aceitar que o sistema permanece sempre mais ou menos vulnerável.
  - Assumir e aceitar que os componentes do sistema podem ser atacados e que alguns desses ataques terão sucesso.
  - Garantir que o sistema como um todo permanece seguro e operacional, ou seja, que não falha.

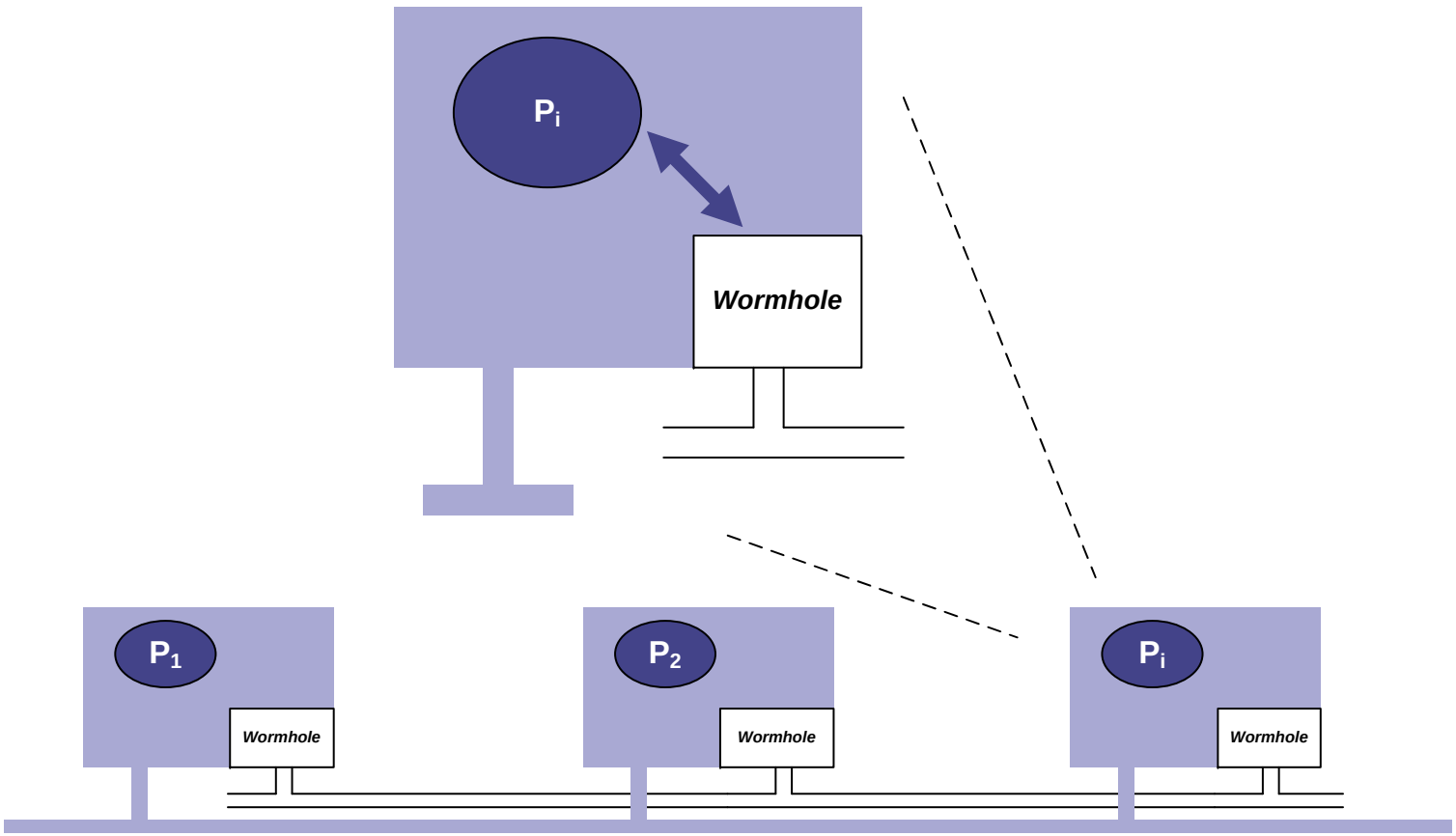
Tolerância a Intrusões

- O que é aplicar o paradigma da tolerância a faltas no domínio de segurança?
  - Assumir e aceitar que o sistema permanece sempre mais ou menos vulnerável.
  - Assumir e aceitar que os componentes do sistema podem ser atacados e que alguns desses ataques terão sucesso.
  - Garantir que o sistema como um todo permanece seguro e operacional, ou seja, que não falha.

Arquitectura genérica do sistema



Arquitectura genérica do sistema



Wormhole

- Considera-se que o sistema distribuído exibe incerteza em termos de segurança. O *Wormhole*, pelo contrário, é uma componente distribuída que não manifesta essa incerteza, é segura.
- Oferece operações seguras que devolvem resultados fiáveis e correctos.
- O *Wormhole* que vamos usar chama-se *ChongDong*.

Wormhole

- Considera-se que o sistema distribuído exibe incerteza em termos de segurança. O *Wormhole*, pelo contrário, é uma componente distribuída que não manifesta essa incerteza, é segura.
- Oferece operações seguras que devolvem resultados fiáveis e correctos.
- O *Wormhole* que vamos usar chama-se *ChongDong*.

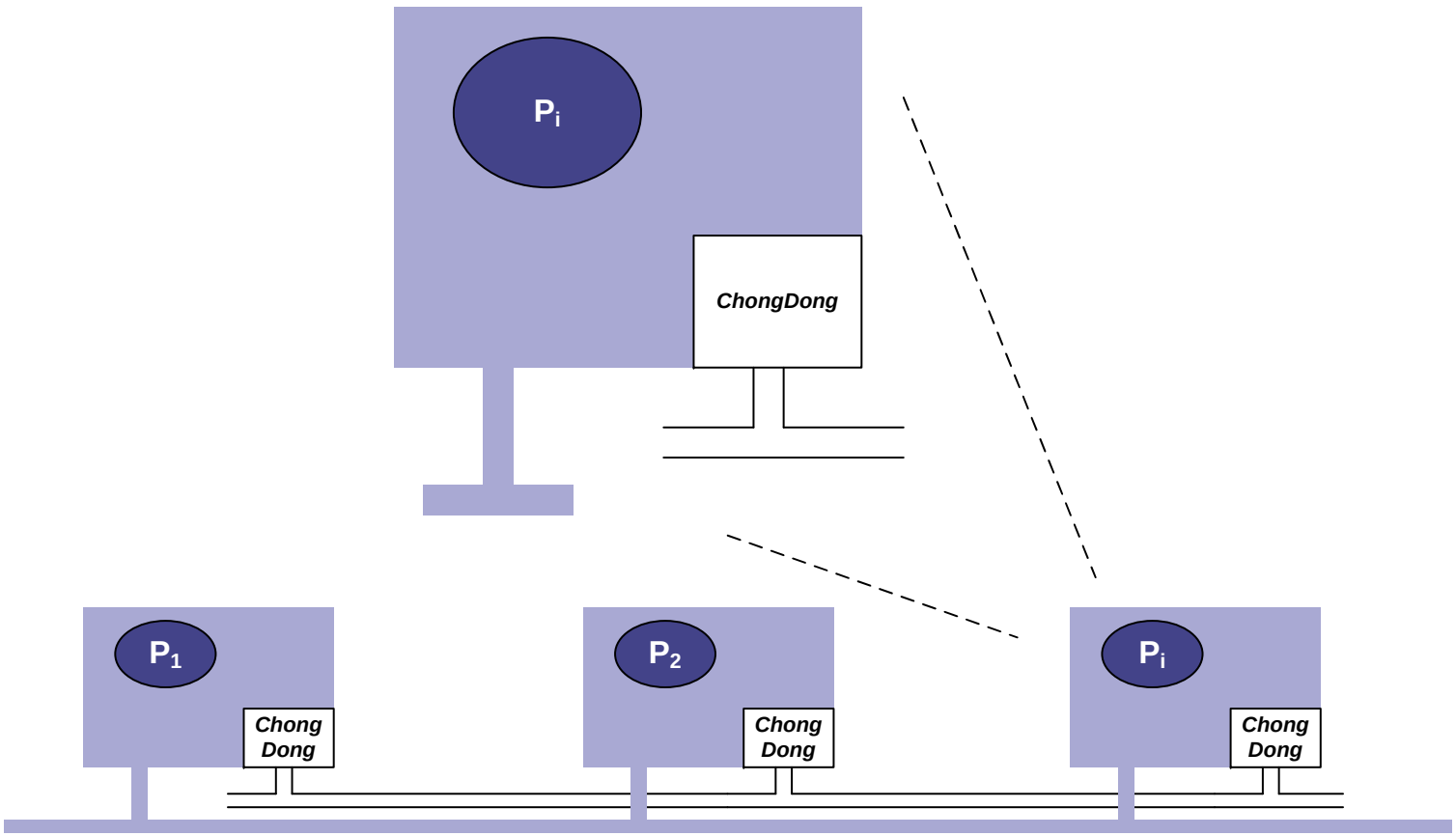
Wormhole

- Considera-se que o sistema distribuído exibe incerteza em termos de segurança. O *Wormhole*, pelo contrário, é uma componente distribuída que não manifesta essa incerteza, é segura.
- Oferece operações seguras que devolvem resultados fiáveis e correctos.
- O *Wormhole* que vamos usar chama-se *ChongDong*.

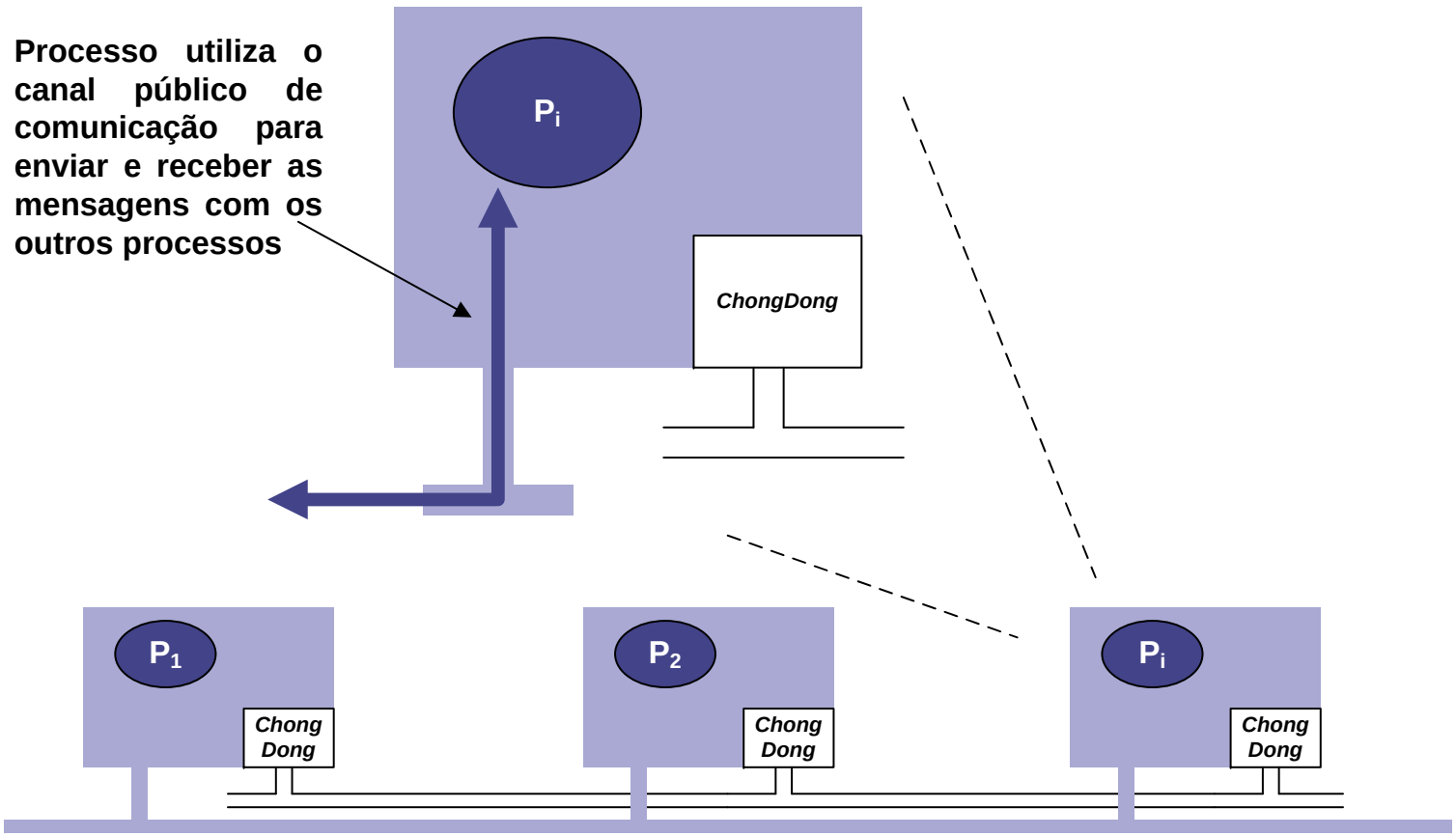
Parte II

Concretização do ChongDong

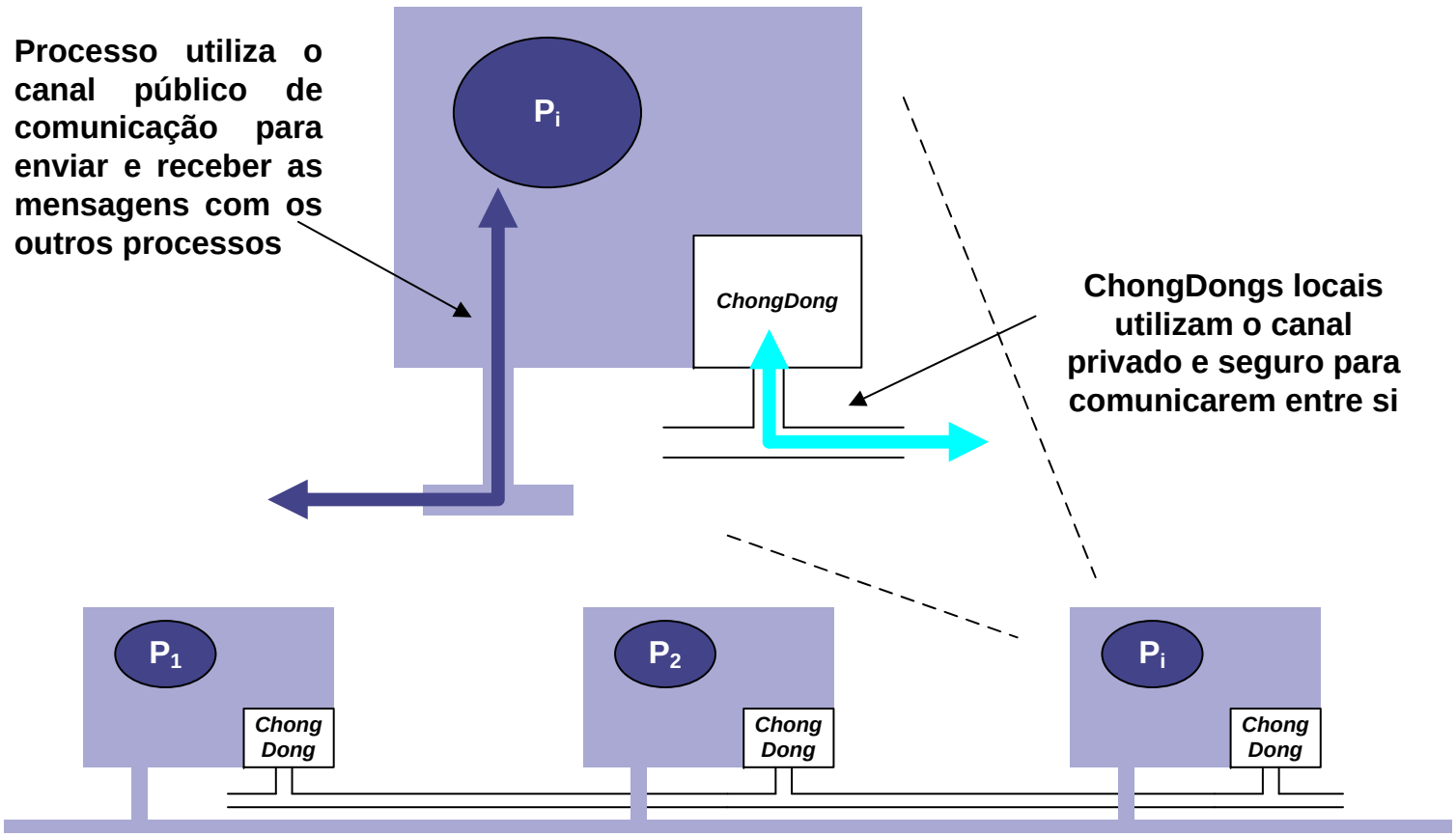
Arquitectura do sistema



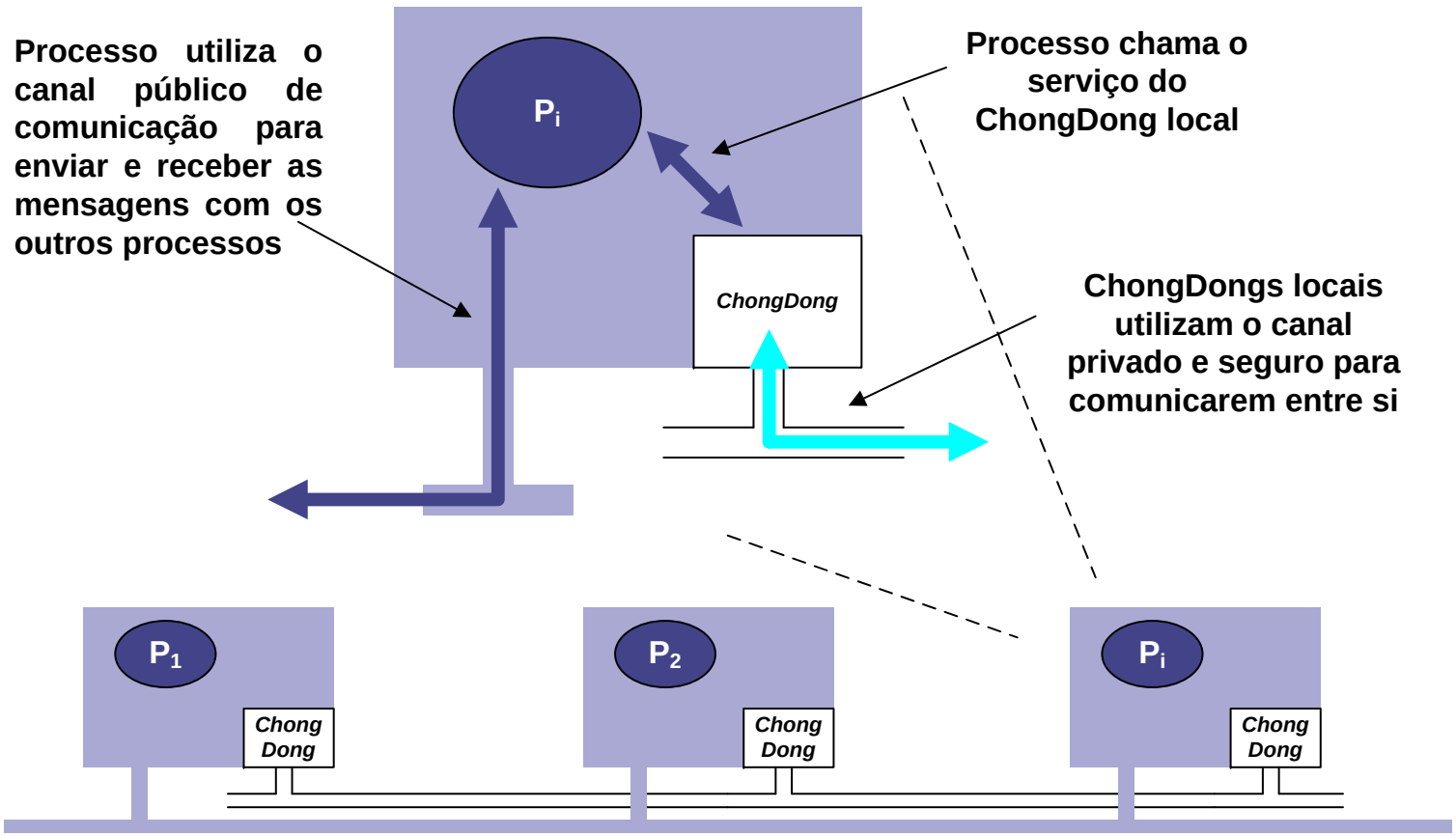
Arquitectura do sistema



Arquitectura do sistema



Arquitectura do sistema



Segurança do *ChongDong*

Problema

No sistema *Linux*, tendo o privilégio de *root* é permitido controlar todos os recursos.

Solução

Restringir o poder da conta do *root*, através de controlo de acesso.

Segurança do *ChongDong*

Problema

No sistema *Linux*, tendo o privilégio de *root* é permitido controlar todos os recursos.

Solução

Restringir o poder da conta do *root*, através de controlo de acesso.

LIDS

- É um *patch* de segurança para o *kernel* do *Linux*.
- Concretiza o princípio do privilégio mínimo para restringir as capacidades extensivas do *root*.
- Realiza o controlo de acesso imperativo *MAC (Mandatory Access Control)*.
- Com o mecanismo de controlo de acesso do *LIDS*, qualquer acesso aos recursos só deve correr com o mínimo privilégio.

Tornar o *ChongDong* local seguro com o *LIDS*

- A obtenção do identificador do processo do *ChongDong* deve ser inacessível para os utilizadores do sistema.
- O processo do *ChongDong* deve ser protegido de modo a que não sejam enviados sinais ao processo.
- O processo do *ChongDong* não deve ser interrompido.
- O código do *ChongDong* deve ser protegido contra modificação.

Tornar o *ChongDong* local seguro com o *LIDS*

- A obtenção do identificador do processo do *ChongDong* deve ser inacessível para os utilizadores do sistema.
- O processo do *ChongDong* deve ser protegido de modo a que não sejam enviados sinais ao processo.
- O processo do *ChongDong* não deve ser interrompido.
- O código do *ChongDong* deve ser protegido contra modificação.

Tornar o *ChongDong* local seguro com o *LIDS*

- A obtenção do identificador do processo do *ChongDong* deve ser inacessível para os utilizadores do sistema.
- O processo do *ChongDong* deve ser protegido de modo a que não sejam enviados sinais ao processo.
- O processo do *ChongDong* não deve ser interrompido.
- O código do *ChongDong* deve ser protegido contra modificação.

Tornar o *ChongDong* local seguro com o *LIDS*

- A obtenção do identificador do processo do *ChongDong* deve ser inacessível para os utilizadores do sistema.
- O processo do *ChongDong* deve ser protegido de modo a que não sejam enviados sinais ao processo.
- O processo do *ChongDong* não deve ser interrompido.
- O código do *ChongDong* deve ser protegido contra modificação.

Proteger acesso à rede privada do *ChongDong* com o *LIDS*

- Desactivar a administração da rede e torná-la apenas acessível para o *ChongDong*
- Desactivar a possibilidade de efectuar *broadcast* pela rede e torná-la apenas acessível para o *ChongDong*.
- Descativar o serviço de *binding* dos portos protegidos e torná-la apenas acessível para o *ChongDong*.
- Esconder a interface da placa de rede ou dar autorização de acesso a apenas alguns processos. (A versão corrente do *LIDS* ainda não suporta esta funcionalidade)

Proteger acesso à rede privada do *ChongDong* com o *LIDS*

- Desactivar a administração da rede e torná-la apenas acessível para o *ChongDong*
- Desactivar a possibilidade de efectuar *broadcast* pela rede e torná-la apenas acessível para o *ChongDong*.
- Descativar o serviço de *binding* dos portos protegidos e torná-la apenas acessível para o *ChongDong*.
- Esconder a interface da placa de rede ou dar autorização de acesso a apenas alguns processos. (A versão corrente do *LIDS* ainda não suporta esta funcionalidade)

Proteger acesso à rede privada do *ChongDong* com o *LIDS*

- Desactivar a administração da rede e torná-la apenas acessível para o *ChongDong*
- Desactivar a possibilidade de efectuar *broadcast* pela rede e torná-la apenas acessível para o *ChongDong*.
- Descativar o serviço de *binding* dos portos protegidos e torná-la apenas acessível para o *ChongDong*.
- Esconder a interface da placa de rede ou dar autorização de acesso a apenas alguns processos. (A versão corrente do *LIDS* ainda não suporta esta funcionalidade)

Proteger acesso à rede privada do *ChongDong* com o *LIDS*

- Desactivar a administração da rede e torná-la apenas acessível para o *ChongDong*
- Desactivar a possibilidade de efectuar *broadcast* pela rede e torná-la apenas acessível para o *ChongDong*.
- Descativar o serviço de *binding* dos portos protegidos e torná-la apenas acessível para o *ChongDong*.
- Esconder a interface da placa de rede ou dar autorização de acesso a apenas alguns processos. (A versão corrente do *LIDS* ainda não suporta esta funcionalidade)

Análise estática de código

- Tentativa de evitar até ao máximo possível as vulnerabilidades da concretização do *ChongDong*.
- A ferramenta usada para a análise estática de código - *Flawfinder*.
- Resultado:
  - Foram encontradas 59 possíveis vulnerabilidades, todas elas relacionadas com a função *memcpy*.
  - Como os tamanhos dos *buffers* são sempre verificados antes de execução da função *memcpy*, essas vulnerabilidades na realidade não existem.

Análise estática de código

- Tentativa de evitar até ao máximo possível as vulnerabilidades da concretização do *ChongDong*.
- A ferramenta usada para a análise estática de código - *Flawfinder*.
- Resultado:
  - Foram encontradas 59 possíveis vulnerabilidades, todas elas relacionadas com a função *memcpy*.
  - Como os tamanhos dos *buffers* são sempre verificados antes de execução da função *memcpy*, essas vulnerabilidades na realidade não existem.

Análise estática de código

- Tentativa de evitar até ao máximo possível as vulnerabilidades da concretização do *ChongDong*.
- A ferramenta usada para a análise estática de código - *Flawfinder*.
- Resultado:
  - Foram encontradas 59 possíveis vulnerabilidades, todas elas relacionadas com a função *memcpy*.
  - Como os tamanhos dos *buffers* são sempre verificados antes de execução da função *memcpy*, essas vulnerabilidades na realidade não existem.

Serviço *ChongDong_TBA*

- Serve de suporte à execução de protocolos tolerantes a intrusões.

_ChongDong_TBA_

- Cada processo propõe um valor ao *ChongDong* local.
- Os *ChongDong*'s trocam entre si os valores propostos.
- Executam um protocolo de consenso.
- Por fim devolve o valor proposto pelo maior número de processos.

Serviço *ChongDong_TBA*

- Serve de suporte à execução de protocolos tolerantes a intrusões.

ChongDong_TBA

- Cada processo propõe um valor ao *ChongDong* local.
- Os *ChongDong*'s trocam entre si os valores propostos.
- Executam um protocolo de consenso.
- Por fim devolve o valor proposto pelo maior número de processos.

Serviço *ChongDong_TBA*

- Serve de suporte à execução de protocolos tolerantes a intrusões.

ChongDong_TBA

- Cada processo propõe um valor ao *ChongDong* local.
- Os *ChongDong*'s trocam entre si os valores propostos.
- Executam um protocolo de consenso.
- Por fim devolve o valor proposto pelo maior número de processos.

Serviço *ChongDong_TBA*

- Serve de suporte à execução de protocolos tolerantes a intrusões.

ChongDong_TBA

- Cada processo propõe um valor ao *ChongDong* local.
- Os *ChongDong*'s trocam entre si os valores propostos.
- Executam um protocolo de consenso.
- Por fim devolve o valor proposto pelo maior número de processos.

Serviço *ChongDong_TBA*

- Serve de suporte à execução de protocolos tolerantes a intrusões.

ChongDong_TBA

- Cada processo propõe um valor ao *ChongDong* local.
- Os *ChongDong*'s trocam entre si os valores propostos.
- Executam um protocolo de consenso.
- Por fim devolve o valor proposto pelo maior número de processos.

Concretização do *ChongDong_TBA*

Detalhe da Concretização

- Difusão do valor proposto pelo processo.
- Quando tiver quorum valores propostos por diferentes processos, executam consenso.
- O protocolo de consenso usa o detector de falhas para tolerar crashes de máquinas.

Concretização do *ChongDong_TBA*

Detalhe da Concretização

- Difusão do valor proposto pelo processo.
- Quando tiver quorum valores propostos por diferentes processos, executam consenso.
- O protocolo de consenso usa o detector de falhas para tolerar crashes de máquinas.

Concretização do *ChongDong_TBA*

Detalhe da Concretização

- Difusão do valor proposto pelo processo.
- Quando tiver quorum valores propostos por diferentes processos, executam consenso.
- O protocolo de consenso usa o detector de falhas para tolerar crashes de máquinas.

Parte III

Exemplo e Resultados

Exemplo

- Sendo o objectivo do *ChongDong* suportar a execução de protocolos tolerantes a intrusões, foi concretizado um protocolo com essa propriedade - protocolo de consenso tolerante a intrusões.
- Os processos que correm o protocolo, utilizam uma rede pública de comunicação.
- Comunicam com o *ChongDong* existente em cada máquina local para obter o valor de acordo distribuído.

Exemplo

- Sendo o objectivo do *ChongDong* suportar a execução de protocolos tolerantes a intrusões, foi concretizado um protocolo com essa propriedade - protocolo de consenso tolerante a intrusões.
- Os processos que correm o protocolo, utilizam uma rede pública de comunicação.
- Comunicam com o *ChongDong* existente em cada máquina local para obter o valor de acordo distribuído.

Exemplo

- Sendo o objectivo do *ChongDong* suportar a execução de protocolos tolerantes a intrusões, foi concretizado um protocolo com essa propriedade - protocolo de consenso tolerante a intrusões.
- Os processos que correm o protocolo, utilizam uma rede pública de comunicação.
- Comunicam com o *ChongDong* existente em cada máquina local para obter o valor de acordo distribuído.

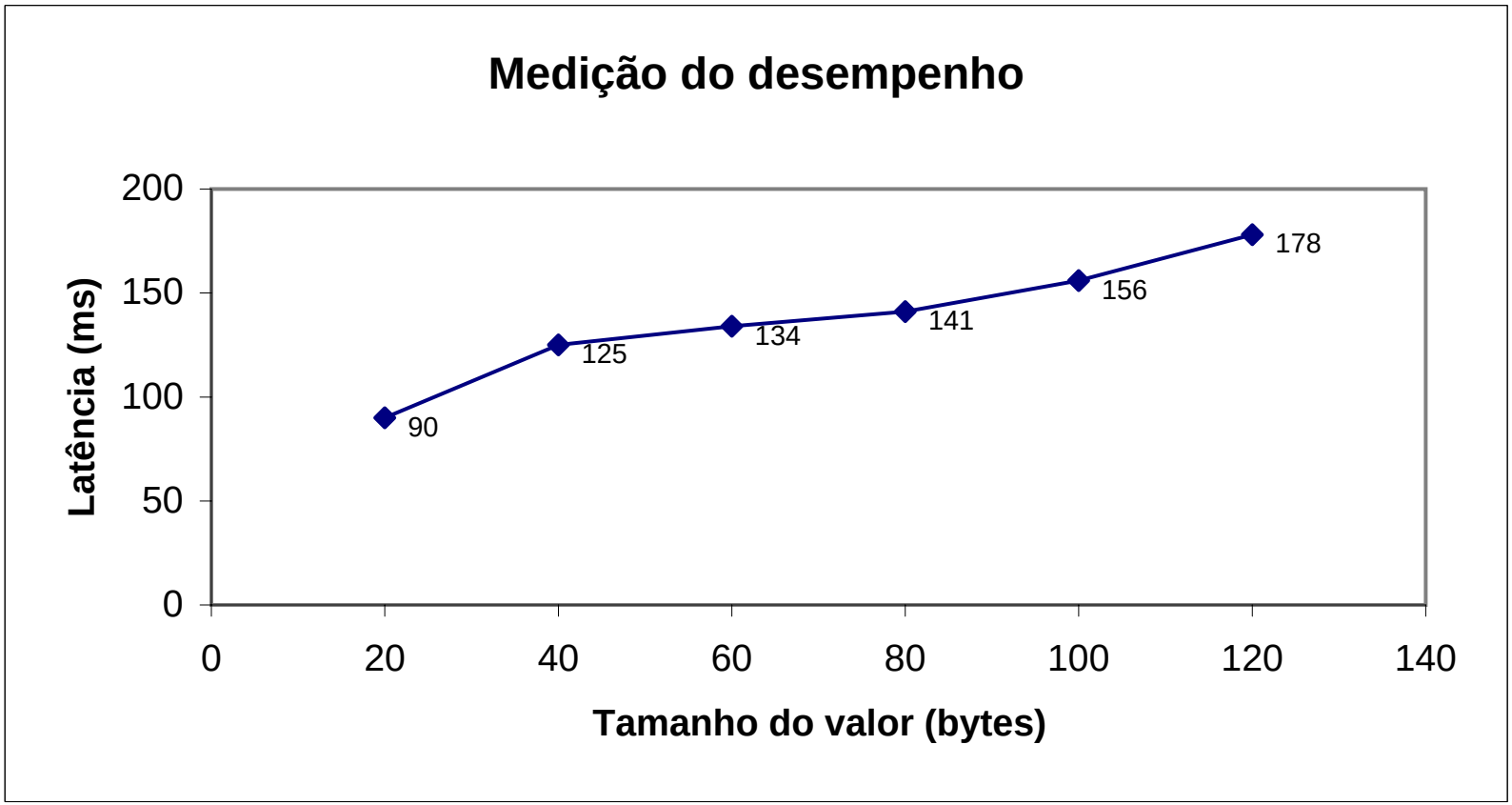
Consenso Tolerante a Intrusões

- Num ambiente assíncrono, os processos com intrusões podem comportar de forma arbitrária.
  - ex. os processo maliciosos podem tentar fazer concluio para quebrar as propriedades do consenso.
- O objectivo é garantir que os processos correctos conseguem decidir num valor com a presença de um subconjunto de processos maliciosos.

Consenso Tolerante a Intrusões

- Num ambiente assíncrono, os processos com intrusões podem comportar de forma arbitrária.
  - ex. os processo maliciosos podem tentar fazer concluio para quebrar as propriedades do consenso.
- O objectivo é garantir que os processos correctos conseguem decidir num valor com a presença de um subconjunto de processos maliciosos.

Resultado


Medição do desempenho
Latência (ms)
200
150
100
50
0
0
20
40
60
80
100
120
140
Tamanho do valor (bytes)
90
125
134
141
156
178

Parte IV

Conclusão

Conclusão

- Foi descrito o projecto de uma componente distribuída segura da classe dos *wormholes* com o suporte do LIDS.
- Foi mostrado como essa componente pode ser usada para concretizar um protocolo de consenso tolerante a intrusões.
- Foi avaliado o desempenho do protocolo de consenso.