Securing statically-verified communications protocols against timing attacks

Stephen Gilmore
LFCS, Edinburgh

Joint work with Mikael Buchholtz,
Jane Hillston and Flemming Nielson
Sometimes a modelling problem has two different aspects.
Sometimes a modelling problem has two different aspects.

- Security and performance
Sometimes a modelling problem has two different aspects.

- Security and performance
- Performance and mobility
Sometimes a modelling problem has two different aspects.

- Security and performance
- Performance and mobility
- Mobility and security
Sometimes a modelling problem has two different aspects.

- Security and performance
- Performance and mobility
- Mobility and security
- Mobility and scalability
Sometimes a modelling problem has two different aspects.

- Security and performance
- Performance and mobility
- Mobility and security
- Mobility and scalability

For such a problem it can suffice to undertake two, separate, complementary analyses.
Sometimes a modelling problem has two different aspects.
- Security and performance
- Performance and mobility
- Mobility and security
- Mobility and scalability

For such a problem it can suffice to undertake two, separate, complementary analyses.

Sometimes the problem seems to involve aspects of two, or more, overlapping concerns.
Performance and security are two interconnected problems.
Performance and security are two interconnected problems.

A machine which is broken because it has been hacked into is not performing well.
Performance and security

- Performance and security are two interconnected problems.
- A machine which is broken because it has been hacked into is not performing well.
- Some performance optimisations (caching) have security implications.
Motivations: Security

Why analyse security?

- Greater exposure than ever before because more applications are networked.
- Weak programming model in widespread use with many possible security exploits.
- Potential for significant loss of esteem if security is breached.
- Keep the hackers out.
Motivations: Security

- Why analyse security?
  - Greater exposure than ever before because more applications are networked.
Motivations: Security

- Why analyse security?
  - Greater exposure than ever before because more applications are networked.
  - Weak programming model in widespread use with many possible security exploits.

Keeping the hackers out.
Motivations: Security

- Why analyse security?
  - Greater exposure than ever before because more applications are networked.
  - Weak programming model in widespread use with many possible security exploits.
  - Potential for significant loss of esteem if security is breached.
Motivations: Security

Why analyse security?
- Greater exposure than ever before because more applications are networked.
- Weak programming model in widespread use with many possible security exploits.
- Potential for significant loss of esteem if security is breached.

Keep the hackers out.
Motivations: Performance

Why analyse performance? If you have any problems just buy a bigger machine. Hardware is so cheap now.
Motivations: Performance

- Why analyse performance? If you have any problems just buy a bigger machine. Hardware is so cheap now.
  - Users do not necessarily have the latest hardware.
Motivations: Performance

Why analyse performance? If you have any problems just buy a bigger machine. Hardware is so cheap now.

- Users do not necessarily have the latest hardware.
- If the device which you replace is not actually the bottleneck device you might just make your problems worse.
Why analyse performance? If you have any problems just buy a bigger machine. Hardware is so cheap now.

- Users do not necessarily have the latest hardware.
- If the device which you replace is not actually the bottleneck device you might just make your problems worse.
- What about when you want to advertise your (quantified) quality of service in a service-level agreement?
Motivations: Performance

- Why analyse performance? If you have any problems just buy a bigger machine. Hardware is so cheap now.
  - Users do not necessarily have the latest hardware.
  - If the device which you replace is not actually the bottleneck device you might just make your problems worse.
  - What about when you want to advertise your (quantified) quality of service in a service-level agreement?
- Keep the customers in.
For some unfathomable reason industrial users don’t routinely use process calculi.
For some unfathomable reason industrial users don’t routinely use process calculi.

A modelling language with a significant industrial base is the Unified Modelling Language (UML).
For some unfathomable reason industrial users don’t routinely use process calculi.

A modelling language with a significant industrial base is the Unified Modelling Language (UML).

We connected our analysis tools to the UML modelling tools.
Outline

1. Secure communications protocols
   - Timing attacks
   - Remote timing attacks

2. Static analysis for security properties using LySa
   - LySa process calculus
   - Static analysis with the LySatool

3. Dynamic analysis for performance properties using PEPA
   - The PEPA stochastic process algebra
   - Dynamic analysis with IPC/DNAmaca

4. Choreographer
   - The Choreographer analysis platform

5. Summary
Outline

1. Secure communications protocols
   - Timing attacks
   - Remote timing attacks

2. Static analysis for security properties using LySa
   - LySa process calculus
   - Static analysis with the LySatool

3. Dynamic analysis for performance properties using PEPA
   - The PEPA stochastic process algebra
   - Dynamic analysis with IPC/DNAmaca

4. Choreographer
   - The Choreographer analysis platform

5. Summary

Securing communications protocols against timing attacks
Modelling Methods and Tools, November 14, 2005
Purpose of a secure communications protocol

- Allows sender and receiver to exchange confidential messages.
- Authenticates the principals to confirm their identity.
An example: the Wide-Mouthed Frog protocol

1. A sends a message to S including the name of B and the new session key $K_{AB}$, encrypted under $K_{AS}$.
2. S decrypts this and sends the name of A and the key $K_{AB}$ to B, encrypted under $K_{BS}$.
3. A sends a message to B encrypted under $K_{AB}$.
An example: the Wide-Mouthed Frog protocol

1. A sends a message to S including the name of B and the new session key $K_{AB}$, encrypted under $K_{AS}$. 
An example: the Wide-Mouthed Frog protocol

1. A sends a message to S including the name of B and the new session key $K_{AB}$, encrypted under $K_{AS}$.
2. S decrypts this and sends the name of A and the key $K_{AB}$ to B, encrypted under $K_{BS}$. 
An example: the Wide-Mouthed Frog protocol

1. A sends a message to S including the name of B and the new session key $K_{AB}$, encrypted under $K_{AS}$.

2. S decrypts this and sends the name of A and the key $K_{AB}$ to B, encrypted under $K_{BS}$.

3. A sends a message to B encrypted under $K_{AB}$.
Timing attacks

- Attacks where information is leaked through an inference obtained from timing a secure interaction are known as **timing attacks**.
Timing attacks

- Attacks where information is leaked through an inference obtained from timing a secure interaction are known as **timing attacks**.

- Secure communications protocols depend on encryption algorithms which take a measurable time to execute.
Timing attacks

- Attacks where information is leaked through an inference obtained from timing a secure interaction are known as timing attacks.
- Secure communications protocols depend on encryption algorithms which take a measurable time to execute.
- If security-sensitive operations can be repeatedly timed then information about the secret keys used for decryption can be gained bit by bit until they are entirely known.
Timing attacks

- Attacks where information is leaked through an inference obtained from timing a secure interaction are known as timing attacks.

- Secure communications protocols depend on encryption algorithms which take a measureable time to execute.

- If security-sensitive operations can be repeatedly timed then information about the secret keys used for decryption can be gained bit by bit until they are entirely known.

- When secret keys become known then the confidentiality and authenticity offered by secure protocols are entirely lost.
Remote timing attacks

- It has been known for some years that timing attacks can be used to extract keys from weak computing devices such as smartcards.
Remote timing attacks

- It has been known for some years that timing attacks can be used to extract keys from weak computing devices such as smartcards.
- It was shown last year that remote timing attacks can be used to uncover secure keys stored on servers. [Remote timing attacks are practical, Brumley and Boneh, 12th USENIX Security Symposium, 2003].
Remote timing attacks

- It has been known for some years that timing attacks can be used to extract keys from weak computing devices such as smartcards.

- It was shown last year that remote timing attacks can be used to uncover secure keys stored on servers. [Remote timing attacks are practical, Brumley and Boneh, 12th USENIX Security Symposium, 2003].
  - Mounted an attack across multiple routers and switches on an OpenSSL-based web server.
Remote timing attacks

- It has been known for some years that timing attacks can be used to extract keys from weak computing devices such as smartcards.

- It was shown last year that remote timing attacks can be used to uncover secure keys stored on servers. [Remote timing attacks are practical, Brumley and Boneh, 12th USENIX Security Symposium, 2003].
  - Mounted an attack across multiple routers and switches on an OpenSSL-based web server.
  - Attack applies in networked, inter-process and virtual machine environments.
Remote timing attacks

- It has been known for some years that timing attacks can be used to extract keys from weak computing devices such as smartcards.
- It was shown last year that remote timing attacks can be used to uncover secure keys stored on servers. [*Remote timing attacks are practical, Brumley and Boneh, 12th USENIX Security Symposium, 2003*].
  - Mounted an attack across multiple routers and switches on an OpenSSL-based web server.
  - Attack applies in networked, inter-process and virtual machine environments.
  - Found that many crypto libraries completely ignore the timing attack.
Outline

1. Secure communications protocols
   - Timing attacks
   - Remote timing attacks

2. Static analysis for security properties using LySa
   - LySa process calculus
   - Static analysis with the LySatool

3. Dynamic analysis for performance properties using PEPA
   - The PEPA stochastic process algebra
   - Dynamic analysis with IPC/DNAmaca

4. Choreographer
   - The Choreographer analysis platform

5. Summary
LySa [Buchholtz, Nielson and Nielson, 2004] is a variant of Abadi and Gordon’s Spi-calculus which includes pattern matching on values at input and decryption.
LySa process calculus

LySa [Buchholtz, Nielson and Nielson, 2004] is a variant of Abadi and Gordon’s Spi-calculus which includes pattern matching on values at input and decryption.

\[
\begin{align*}
P_1 & | P_2 & \text{Parallel} \\
!P & & \text{Replication} \\
0 & & \text{Nil} \\
(\nu n) P & & \text{New} \\
\langle t_1, \ldots, t_k \rangle.P & & \text{Output} \\
(t_1, \ldots, t_j; x_{j+1}, \ldots x_k).P & & \text{Input} \\
& \text{decrypt } t \text{ as } \{t_1, \ldots, t_j; x_{j+1}, \ldots, x_k\}_{t_0} \text{ in } P & \text{Decrypt}
\end{align*}
\]
Structure of a LySa packet

LySa packets are tuples of information sent across a global network.

LySa packet

\[
\langle \text{header}, \text{payload}, \text{metadata} \rangle
\]

- Header: The sender is $A$ and the receiver is $S$.
- Payload: The name $A$ is sent in the clear and the name $B$ and the key $K_{AB}$ are sent encrypted under $K_{AS}$.
- Metadata: This is encrypted at $a_1$ to be decrypted at $s_1$. 

LySa process calculus
Static analysis with the LySatool
LySa packets are tuples of information sent across a global network.

- **Header**: The sender is $A$ and the receiver is $S$.

- **Payload**: The name $A$ is sent in the clear and the name $B$ and the key $K_{AB}$ are sent encrypted under $K_{AS}$.

- **Metadata**: This is encrypted at $a_1$ to be decrypted at $s_1$. 

LySa packets are tuples of information sent across a global network.

- **Header:** The sender is $A$ and the receiver is $S$.
- **Payload:** The name $A$ is sent in the clear and the name $B$ and the key $K_{AB}$ are sent encrypted under $K_{AS}$.
- **Metadata:** This is encrypted at $a_1$ to be decrypted at $s_1$. 
Structure of a LySa packet

LySa packets are tuples of information sent across a global network.

- **Header**: The sender is $A$ and the receiver is $S$.
- **Payload**: The name $A$ is sent in the clear and the name $B$ and the key $K_{AB}$ are sent encrypted under $K_{AS}$.
- **Metadata**: This is encrypted at $a1$ to be decrypted at $s1$. 
LySa model of the Wide-Mouthed Frog protocol
LySa model of the Wide-Mouthed Frog protocol

Principal A

\[
! (\nu K_{AB}) \langle A, S, A, \{ B, K_{AB} \} K_{AS}[\text{at } a1 \text{ dest } s1]\rangle . \\
(\nu \text{ message}) \langle A, B, \{ \text{message} \} K_{AB}[\text{at } a2 \text{ dest } b2]\rangle .0
\]
LySa model of the Wide-Mouthed Frog protocol

**Principal A**

\( !(\nu K_{AB}) \langle A, S, A, \{ B, K_{AB} \} K_{AS} [at a1 \text{ dest } s1] \rangle \).

\( (\nu \text{ message}) \langle A, B, \{ \text{message} \} K_{AB} [at a2 \text{ dest } b2] \rangle.0 \)

**Server S**

\( !(A, S, A; z).\text{decrypt } z \text{ as } \{ B; zk \} K_{AS} [at s1 \text{ orig } a1] \text{ in} \langle S, B, \{ A, zk \} K_{BS} [at s2 \text{ dest } b1] \rangle.0 \)
LySa model of the Wide-Mouthed Frog protocol

**Principal A**

\[ !(\nu K_{AB}) \langle A, S, A, \{ B, K_{AB} \} K_{AS}[\text{at }a1 \text{ dest }s1]\rangle. \]
\[ (\nu \text{ message}) \langle A, B, \{ \text{message} \} K_{AB}[\text{at }a2 \text{ dest }b2]\rangle.0 \]

**Server S**

\[ !(A, S, A; \ z).\text{decrypt } \ z \text{ as } \{ B; \ zk \} K_{AS}[\text{at }s1 \text{ orig }a1] \text{ in } \]
\[ \langle S, B, \{ A, zk \} K_{BS}[\text{at }s2 \text{ dest }b1]\rangle.0 \]

**Principal B**

\[ !(S, B; \ x).\text{decrypt } \ x \text{ as } \{ A; \ xk \} K_{BS}[\text{at }b1 \text{ orig }s2] \text{ in } \]
\[ (A, B; \ y).\text{decrypt } \ y \text{ as } \{ ; \ ym \} xk[\text{at }b2 \text{ orig }a2] \text{ in } 0 \]
Static analysis

- Secure communications protocols
- Static analysis for security properties using LySa
- Dynamic analysis for performance properties using PEPA
- Choreographer

Summary

LySa process calculus
Static analysis with the LySa tool

Static analysis

- Securing communications protocols against timing attacks
- Modelling Methods and Tools, November 14, 2005
Benefits of static analysis

- Static analysis applies to all possible attacks which a standard network attacker can apply and so it is as general as theorem proving.
- It generates informative counter-examples showing where the problems occur and so it is as useful as model checking.
- Static analysis is computationally inexpensive so stock hardware can be used to prove security properties of complex real-world communication protocols, and to discover previously-unknown flaws in them.
Benefits of static analysis

- Static analysis applies to all possible attacks which a standard network attacker can apply and so it is as general as theorem proving.
- It generates **informative counter-examples** showing where the problems occur and so it is as useful as model checking.
- Static analysis is computationally inexpensive so stock hardware can be used to prove security properties of complex real-world communication protocols, and to discover previously-unknown flaws in them.
Benefits of static analysis

- Static analysis applies to all possible attacks which a standard network attacker can apply and so it is as general as theorem proving.
- It generates informative counter-examples showing where the problems occur and so it is as useful as model checking.
- Static analysis is *computationally inexpensive* so stock hardware can be used to prove security properties of complex real-world communication protocols, and to discover previously-unknown flaws in them.
The LySa processes are annotated with authentication properties specifying intended origin and destinations of messages.
Use of the LySatool

- The LySa processes are annotated with authentication properties specifying intended origin and destinations of messages.
- The LySatool works by computing over-approximations to what a LySa process can do in all executions of the process executed in parallel with an arbitrary attacker.
Use of the LySatool

- The LySa processes are annotated with authentication properties specifying intended origin and destinations of messages.

- The LySatool works by computing over-approximations to what a LySa process can do in all executions of the process executed in parallel with an arbitrary attacker.
  - The analysis may report too many errors in protocols, but cannot report too few.
Use of the LySatool

- The LySa processes are annotated with authentication properties specifying intended origin and destinations of messages.
- The LySatool works by computing over-approximations to what a LySa process can do in all executions of the process executed in parallel with an arbitrary attacker.
  - The analysis may report too many errors in protocols, but cannot report too few.
  - Reporting too many errors does not pose a big problem in practice.
Outline

1. Secure communications protocols
   - Timing attacks
   - Remote timing attacks

2. Static analysis for security properties using LySa
   - LySa process calculus
   - Static analysis with the LySatool

3. Dynamic analysis for performance properties using PEPA
   - The PEPA stochastic process algebra
   - Dynamic analysis with IPC/DNAmaca

4. Choreographer
   - The Choreographer analysis platform

5. Summary
PEPA [Hillston, 1994] is a stochastic process algebra in which the rate at which each activity can be performed is quantified.
PEPA [Hillston, 1994] is a stochastic process algebra in which the rate at which each activity can be performed is quantified.

\[(\alpha, r).P\] Prefix
\[P_1 + P_2\] Choice
\[P_1 \bowtie L P_2\] Co-operation
\[P/L\] Hiding
\[X\] Variable
PEPA model of the Wide-Mouthed Frog protocol
The PEPA stochastic process algebra
Dynamic analysis with IPC/DNAmaca

PEPA model of the Wide-Mouthed Frog protocol

Principal $A$

\[
P_A \equiv (as, r_{as}).(ab, r_{ab}).P_A
\]
The PEPA stochastic process algebra
Dynamic analysis with IPC/DNAmaca

**Summary**

Secure communications protocols
Static analysis for security properties using LySa
Dynamic analysis for performance properties using PEPA
Choreographer

**PEPA model of the Wide-Mouthed Frog protocol**

**Principal A**

\[ P_A \overset{\text{def}}{=} (as, r_{as}).(ab, r_{ab}).P_A \]

**Server S**

\[ P_S \overset{\text{def}}{=} (as, \top).(sb, r_{sb}).P_S \]
Secure communications protocols
Static analysis for security properties using LySa
Dynamic analysis for performance properties using PEPA
Choreographer
Summary

The PEPA stochastic process algebra
Dynamic analysis with IPC/DNAmaca

PEPA model of the Wide-Mouthed Frog protocol

Principal A

\[ P_A \equiv (as, r_{as}).(ab, r_{ab}).P_A \]

Server S

\[ P_S \equiv (as, \top).(sb, r_{sb}).P_S \]

Principal B

\[ P_B \equiv (sb, \top).(ab, \top).P_B \]
A PEPA model is analysed relative to valuations which map the symbolic rates of the model to concrete values determined by estimation or measurement.
A PEPA model is analysed relative to valuations which map the symbolic rates of the model to concrete values determined by estimation or measurement.

Rates can be chosen to represent communication cost, computation cost, or an aggregate of these.
Dynamic analysis of PEPA models

- A PEPA model is analysed relative to valuations which map the symbolic rates of the model to concrete values determined by estimation or measurement.
- Rates can be chosen to represent communication cost, computation cost, or an aggregate of these.
- We can modify the protocol by adding delays to mask the difference between a faster interaction and a slower one.
Dynamic analysis of PEPA models

- A PEPA model is analysed relative to valuations which map the symbolic rates of the model to concrete values determined by estimation or measurement.
- Rates can be chosen to represent communication cost, computation cost, or an aggregate of these.
- We can modify the protocol by adding delays to mask the difference between a faster interaction and a slower one.
- Finally, we wish to determine whether or not two versions of the PEPA model of the protocol are sufficiently close that we would believe that a timing attack is impractical.
The Imperial PEPA Compiler and DNAmaca

- IPC (The Imperial PEPA Compiler) processes PEPA models to compile them into the input format of the DNAmaca Markov chain analyser.
The Imperial PEPA Compiler and DNAmaca

- IPC (The Imperial PEPA Compiler) processes PEPA models to compile them into the input format of the DNAmaca Markov chain analyser.
- IPC allows the modeller to attach stochastic probes to a PEPA model to mark the start and end of passages through the model.
The Imperial PEPA Compiler and DNAmaca

- IPC (The Imperial PEPA Compiler) processes PEPA models to compile them into the input format of the DNAmaca Markov chain analyser.
- IPC allows the modeller to attach stochastic probes to a PEPA model to mark the start and end of passages through the model.
- Via uniformisation, DNAmaca computes passage-time densities for this, allowing them to be presented as a cumulative density function (CDF) for the passage.
CDF plot of the WMF protocol

CDF plot of Wide-Mouthed Frog

Pr

Time in seconds

Reference
CDF plot of the WMF protocol

CDF plot of Wide-Mouthed Frog

delay = 0.5

reference
CDF plot of the WMF protocol

CDF plot of Wide-Mouthed Frog

Time in seconds

Pr

delay = 0.5

delay = 1

reference

Securing communications protocols against timing attacks

Modelling Methods and Tools, November 14, 2005
CDF plot of the WMF protocol

CDF plot of Wide-Mouthed Frog

Time in seconds

Pr

0 0.1 0.2 0.3 0.4 0.5 0.6 0.7 0.8 0.9 1

0 1 2 3 4 5 6 7 8 9 10

delay = 0.5
delay = 1
delay = 2
reference

Securing communications protocols against timing attacks

Modelling Methods and Tools, November 14, 2005
Outline

1. Secure communications protocols
   - Timing attacks
   - Remote timing attacks

2. Static analysis for security properties using LySa
   - LySa process calculus
   - Static analysis with the LySa tool

3. Dynamic analysis for performance properties using PEPA
   - The PEPA stochastic process algebra
   - Dynamic analysis with IPC/DNAmaca

4. Choreographer
   - The Choreographer analysis platform

5. Summary
Choreographer tool architecture

The succinct solver suite

Linear algebra solvers

Static analysis tools

UML
LySa
PEPA

LySatool analyser
The succinct solver suite

Dynamic analysis tools

PEPA Workbench
One feature of the methodology which we support with the Choreographer design platform tool is that modellers are able to express the models which are input to Choreographer in standard UML.
One feature of the methodology which we support with the Choreographer design platform tool is that modellers are able to express the models which are input to Choreographer in standard UML.

The analysis process is initiated by invoking Choreographer on a UML project archive. The formal content of the UML model is stored in such an archive in an XML-based interchange representation (XMI).
The analysis process

To enhance extensibility and flexibility (eg. to cope with new version of XMI) a modular approach is adopted whereby interaction is via shared files.
To enhance extensibility and flexibility (eg. to cope with new version of XMI) a modular approach is adopted whereby interaction is via shared files.
Secure communications protocols
Static analysis for security properties using LySa
Dynamic analysis for performance properties using PEPA
Choreographer
Summary

The analysis process

To enhance extensibility and flexibility (eg. to cope with new
version of XMI) a modular approach is adopted whereby
interaction is via shared files.
To enhance extensibility and flexibility (eg. to cope with new version of XMI) a modular approach is adopted whereby interaction is via shared files.
To enhance extensibility and flexibility (eg. to cope with new version of XMI) a modular approach is adopted whereby interaction is via shared files.
To enhance extensibility and flexibility (eg. to cope with new version of XMI) a modular approach is adopted whereby interaction is via shared files.

It is essential that results are reported in terms which make sense to the software designer, i.e. in terms of the original UML model.
To enhance extensibility and flexibility (eg. to cope with new version of XMI) a modular approach is adopted whereby interaction is via shared files.
Secure communications protocols
Static analysis for security properties using LySa
Dynamic analysis for performance properties using PEPA

Choreographer

Summary
The Choreographer analysis platform

Securing communications protocols against timing attacks
Secure communications protocols
Static analysis for security properties using LySa
Dynamic analysis for performance properties using PEPA

Choreographer

Summary
The Choreographer analysis platform

Securing communications protocols against timing attacks
Modelling Methods and Tools, November 14, 2005
Secure communications protocols
Static analysis for security properties using LySa
Dynamic analysis for performance properties using PEPA

Choreographer

The Choreographer analysis platform

Modelling Methods and Tools, November 14, 2005
Secure communications protocols
Static analysis for security properties using LySa
Dynamic analysis for performance properties using PEPA

Summarize

The Choreographer analysis platform

Securing communications protocols against timing attacks

Modelling Methods and Tools, November 14, 2005
Secure communications protocols
Static analysis for security properties using LySa
Dynamic analysis for performance properties using PEPA

Choreographer
Summary

Choreographer: detecting errors

Securing communications protocols against timing attacks

Modelling Methods and Tools, November 14, 2005
Secure communications protocols
Static analysis for security properties using LySa
Dynamic analysis for performance properties using PEPA
Choreographer

Summary
The Choreographer analysis platform

Example of reflection: security violations
Example of reflection: security violations
Example of reflection: security violations
Outline

1. Secure communications protocols
   - Timing attacks
   - Remote timing attacks

2. Static analysis for security properties using LySa
   - LySa process calculus
   - Static analysis with the LySatool

3. Dynamic analysis for performance properties using PEPA
   - The PEPA stochastic process algebra
   - Dynamic analysis with IPC/DNAmaCa

4. Choreographer
   - The Choreographer analysis platform

5. Summary
In the design of novel communications protocols it is necessary to consider both security and performance. It is helpful to have a systematic method of analysing protocols with automated support.
In the design of novel communications protocols it is necessary to consider both security and performance. It is helpful to have a systematic method of analysing protocols with automated support.

Security and performance are interrelated issues:
- Time-dependent behaviour can be used to attack a protocol.
- Developers who are concerned with achieving peak performance view security measures as an overhead.
In the design of novel communications protocols it is necessary to consider both security and performance. It is helpful to have a systematic method of analysing protocols with automated support.

Security and performance are interrelated issues:
- Time-dependent behaviour can be used to attack a protocol.
- Developers who are concerned with achieving peak performance view security measures as an overhead.

By using the LySato tool to check origination and destination of messages and the Imperial PEPA Compiler and DNAmaca for the computation of passage-time quantiles we have been able to guard against certain types of network-based attacks.
The industrial partners had no previous experience of using the LySatool and the PEPA Workbench and their use of them was solely via the Choreographer extraction/reflection discipline.
Experiences of the industrial users

- The industrial partners had no previous experience of using the LySa tool and the PEPA Workbench and their use of them was solely via the Choreographer extraction/reflection discipline.
- Almost all of the problems reported by our industrial users were in
The industrial partners had no previous experience of using the LySa tool and the PEPA Workbench and their use of them was solely via the Choreographer extraction/reflection discipline.

Almost all of the problems reported by our industrial users were in

- the UML constructs used in the input model; and
The industrial partners had no previous experience of using the LySatool and the PEPA Workbench and their use of them was solely via the Choreographer extraction/reflection discipline.

Almost all of the problems reported by our industrial users were in

- the UML constructs used in the input model; and
- communication of the UML model from the UML tool to the extractor.
Reflections on the experience

- Our anticipation of the difficulties for the industrial users was quite far removed from the actual difficulties encountered.
Reflections on the experience

- Our anticipation of the difficulties for the industrial users was quite far removed from the actual difficulties encountered.
- The fact that many of the errors were related to UML processing surprised us.
Reflections on the experience

- Our anticipation of the difficulties for the industrial users was quite far removed from the actual difficulties encountered.
- The fact that many of the errors were related to UML processing surprised us.
- We had assumed that the asymptotic complexity of the analysis procedures used in performance analysis would be a problem for models of industrial scale.
Reflections on the experience

- Our anticipation of the difficulties for the industrial users was quite far removed from the actual difficulties encountered.
- The fact that many of the errors were related to UML processing surprised us.
- We had assumed that the asymptotic complexity of the analysis procedures used in performance analysis would be a problem for models of industrial scale.
- In fact, the state-space explosion proved not to be a problem for the use of Choreographer by our industrial partners. The models which they built were much smaller than we had anticipated.
Conclusions

- Provided a development environment for high-level security and performance analysis of computer systems.
Conclusions

- Provided a development environment for high-level security and performance analysis of computer systems.
- Industrial partners willing to use tools and enthusiastic about them.
Conclusions

- Provided a development environment for high-level security and performance analysis of computer systems.
- Industrial partners willing to use tools and enthusiastic about them.
  - Discovered security flaws in their protocols using the tools, and modified them to fix the problems.
Conclusions

- Provided a development environment for high-level security and performance analysis of computer systems.
- Industrial partners willing to use tools and enthusiastic about them.
  - Discovered security flaws in their protocols using the tools, and modified them to fix the problems.
  - Discovered performance problems with their designs and re-worked these to get better performance.
Conclusions

- Provided a development environment for high-level security and performance analysis of computer systems.
- Industrial partners willing to use tools and enthusiastic about them.
  - Discovered security flaws in their protocols using the tools, and modified them to fix the problems.
  - Discovered performance problems with their designs and re-worked these to get better performance.
- All found the UML tools to be troublesome.