1. Goal

Our goal is a tool for security analysis at implementation level. The security flaws that we look for are violations of secrecy and authentication. Further aspects of the analysis are:

- **Automation:** Minimise the need for manual guidance and annotations.
- **Soundness:** Don’t miss any security flaws as each of them makes security software useless.
- **Scalability:** The tools should work on libraries like OpenSSL, NSS or Kerberos.
- **Abstraction level:** Assume that the cryptographic building blocks (encryption, hashing, etc.) are implemented correctly.

2. Secrecy

A protocol fulfills **secrecy** if the attacker has no way to get information about a secret exchanged in the protocol.

![Secrecy](image)

Note that the attacker is assumed to have full control of the network.

3. Authentication

A protocol fulfills **authentication** if the participants have consistent views of events. A counterexample:

![Authentication](image)

4. Background

There are projects focusing on low-level properties of machine languages or on high-level properties of formal languages. However, there has been very little overlap.

<table>
<thead>
<tr>
<th>Machine Languages (C, Java)</th>
<th>Formal Languages ($\pi$-calculus, LySa)</th>
</tr>
</thead>
<tbody>
<tr>
<td>low-level properties (NULL dereference, division by zero)</td>
<td>VCC</td>
</tr>
<tr>
<td>high-level properties (secrecy, authentication)</td>
<td>ESC/Java</td>
</tr>
</tbody>
</table>

5. Motivation

Why isn’t it enough to verify abstract protocol models? The problem is that most security flaws are introduced during implementation. Consider a simple protocol for message integrity protection by hashing with a shared key:

```c
A \rightarrow m, hash(k_{AB}, m) \rightarrow B
B \rightarrow mac_len = recv(mac, MAXLEN);
if (!strcmp(mac, my_mac)) assert(valid(msg, msg_len));
```

6. Challenges

Why can’t we simply use general-purpose verification tools? The answer is that they only check for properties defined by the state of the program. Security properties, on the other hand, speak about the state of an unknown attacker.

8. Project

The project is done in collaboration with François DurGRESS and is supervised by Andrew Gordon (Microsoft Research), Jan Jürjens (Open University) and Bashar Nuseibeh (Open University). It is partially supported through the Microsoft Research PhD Scholarship programme.

This way a clean model can be extracted, suitable for verification with high-level tools like ProVerif or CryptoVerif.