Researcher develops intelligent encryption libraries

September 26 2016

Prof. Dr.-Ing. Mira Mezini. Photo: Katrin Binner

Sometimes, when two people or software applications are communicating via the Internet, a third party is listening. Cryptographic protocols could prevent this situation, but software developers often find it difficult to correctly integrate them into applications. This is the reason why researchers at the TU Darmstadt want to automate

encryption.

Andrea wants to send her friend Stefan a message via the Internet. To prevent anyone else from reading it, she communicates with Stefan and agrees on a secret code with him that only he and she will be able to decipher. Should the message fall into the wrong hands, it will consist of nothing more than an incomprehensible string of characters that cannot be deciphered without the key. However, what Andrea and Stefan don't know is that a spy has inserted himself between them.

The secret code that they believe they established between themselves was actually generated by the spy. He sent it to both participants by pretending to be the other one in each case. Now he can read all of their messages and, for example, ask Andrea for an important password. She feels safe, as she thinks she's sending it to her friend.

This scenario gives a rough idea of a man-in-the-middle attack, during which an attacker manipulates Internet communication. In information technology, the relevant key or code is part of a so-called encryption algorithm. It is designed to protect data that is transmitted back and forth either within a single application or between different applications. Encryption protocols usually run in the background without the user noticing. If, for example, we visit an online shop, our browsers and the online shop automatically negotiate a unique key that is used to mathematically encrypt the data, be it details of the order or bank details. A third party could not do anything useful with the data without the key.

Protection of sensitive data

We encounter cryptography constantly: when using Apps, when sending emails, when communicating via messenger services or during in-house corporate communications – whenever sensitive data is involved.

Encryption protocols ensure a certain level of security under certain assumptions concerning the abilities of the potential attackers, but only if the designer of an encryption protocol implements it correctly and the application developers have used and integrated it correctly in their code.

And, it is in this context that one finds problems that should not be ignored. Configuring cryptographic components is difficult and one cannot expect software developers to be cryptography experts: they make mistakes. Between 2013 and 2015 alone, 1769 security vulnerabilities registered in the USA's National Vulnerability Database (nvd.nist.gov) were the result of such mistakes – as such, issues involving the integration of encryption protocols in applications were the fourth most frequent source of such registered vulnerabilities.

This is not surprising, if one considers several studies presented the most renowned scientific conferences relating to the area of cyber security over the past few years. These demonstrate that software integration is an important point of weakness, especially since the use of components of cryptographic libraries requires knowledge of too many details that application programmers often do not posses.

For example, software developers need to ensure that the individual steps of an encryption protocol are executed in a specific order, for which concrete recommendations are available depending on what is to be protected. However, software developers frequently lack the time to read the relevant manuals.

Another source of errors are so-called digital certificates that verify the validity of a given key. Developers sometimes disable the verification process for their software certificates, in order to speed up testing, but then forget to re-enable it for the production system. According to Mira Mezini, Head of the Software Engineering Research Group at the TU Darmstadt: "these are both common errors, even among serious software

providers; and those are just two examples among many". Intruders always have the advantage that they only need to discover a single security vulnerability to be able to steal data, while developers are faced with the enormous challenge of closing all possible security gaps.

Provided by Technische Universitat Darmstadt