Non Interference: Past, Present and Future

Riccardo Focardi
Dipartimento di Matematica Applicata ed Informatica
Università Ca' Foscari di Venezia
Via Torino 155,
I – 30173 Venezia Mestre (Italy)
e_mail: focardi@dsi.unive.it

Roberto Gorrieri
Dipartimento di Scienze dell'Informazione
Università di Bologna
Mura Anteo Zamboni 7,
I – 40127 Bologna (Italy)
e_mail: gorrieri@cs.unibo.it

Security is a crucial property of system behaviour. It generally requires some kind of control over the information flow among parts of the system. In order to analyze these possible flows it has been introduced the Non-Interference (NI) property [7]. In particular NI was introduced to detect all the possible flows from a group of users to another one. A lot of research has been done about NI in the following years (e.g., see [7, 13, 11, 4, 5]) with the aims of generalizing NI to non deterministic systems, comparing different definitions, automatically check some NI-like properties.

The Past: One of the main motivation for NI was to limit, and possibly to avoid, the damages produced by malicious programs, called *Trojan Horses*, which try to broadcast secret information. In the classic approach of *Discretionary Access Control* security (DAC), a Trojan Horse program can easily attack the system by simply modifying the security properties of user's objects. In fact, in DAC we have that every *subject* (i.e., an active agent such as a user), decides the access properties of its *objects* (i.e., passive agents such as files). An example of DAC is the file management in Unix where a user can decide the access possibilities of her/his files. If a user's program is a Trojan Horse (the user could have get it by ftp) then it can make whatever it wants of the user's files.

As a solution to this problem it could be possible to use a *Mandatory Access Control* (MAC for short), where some access rules are imposed by the system. An example of MAC is *Multilevel Security* [1]: every object is bounded to a security level, and so every subject is; information can flow from a certain object to a certain subject only if the level of the subject is greater than the level of the object. So a Trojan Horse, which operates at a certain level, has no way to *downgrade* information, and its action is restricted inside such a level. In general, Multilevel Security can be described through the following access rules: *No Read Up* (a subject cannot read data from an upper level object) and *No Write Down* (a subject cannot write data to a lower level object).

However, these access rules are not enough. It could be possible to indirectly transmit information using some system side effect. For example, if two levels – 'high' and 'low' – share some finite storage resource (e.g., a hard disk), it is possible to transmit data from level 'high' to level 'low' in the following simple way: for a high level transmitter, it is sufficient to alternatively fill or empty the resource in order to transmit a '1' or a '0' datum; simultaneously, the low level receiver tries to write on the resource, decoding every error message (the disk is full) as a '1' and every successful write as a '0'. It is clear that such indirect ways of transmission, called *covert channels*, do not violate the two multilevel access rules (see Figure 1). Therefore it is necessary to integrate a MAC discipline with a covert channel analysis (see e.g. [12]).

NI can be seen as an alternative, more general approach to this kind of security. It requires to control directly the whole flow of information, rather than the accesses of subjects to objects. In general, NI-like properties are based on a formal model of system behaviour and on a definition of information flow on such a model. By imposing some information flow rule, we can control any kind of transmission, be it direct or indirect.

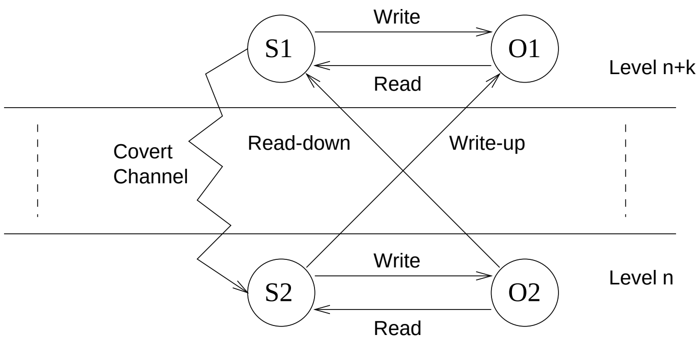

Figure 1: Information flows in multilevel security.

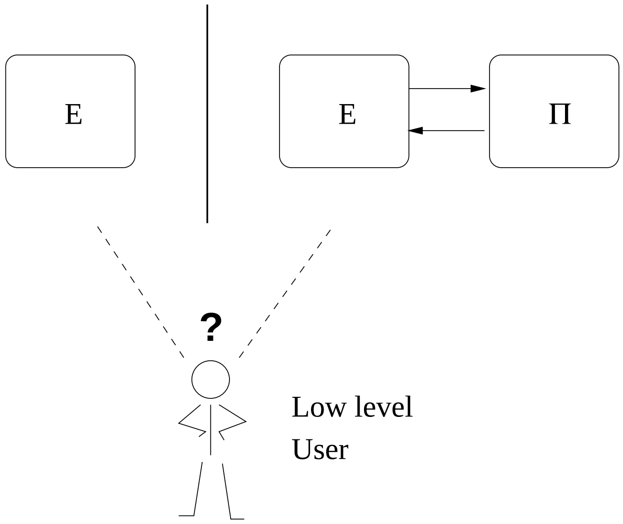

Figure 2: BNDC basic idea

The Present: In the literature, there are many different definitions of this kind based on several system models (see e.g. [7, 13]). In [4] we have uniformly rephrased them in the setting of *Security Process Algebra* (SPA, for short), then compared and classified, leading to our proposed notion of *Bisimulation Non Deducibility on Compositions* (*BNDC*, for short). SPA is an extension of CCS [9] – a language proposed to specify concurrent systems, whose programs are terms of an algebra comprising some operators for building systems bottom-up from smaller subsystems. The basic building blocks are the atomic activities, simply called *actions*; contrary to CCS, in SPA actions belong to two different security levels (or groups): *high* and *low*.

The basic idea of BNDC is very simple: we want to check the system against all high level potential interactions. In particular, we require that what a low level user sees of the system E is not modified by composing any high level process Π to E (see Figure 2). In this way we obtain that high level users cannot interfere with low level ones. In order to decide what a low level user can see of a system we need to select a notion of equivalence between processes. For BNDC we take the weak bisimulation equivalence; however the idea of Non Deducibility on Compositions is general and we can use other notions of equivalence (see [4, 2] for trace and failure based properties). The BNDC property can be expressed in SPA as follows:

$$E/Act_H \approx_B (E|\Pi) \setminus Act_H$$

where E/Act_H is what a low level user sees of the system ('$/Act_H$' hides all the high level actions), and $(E|\Pi) \setminus Act_H$ is the system E composed with the high level system Π ('$\setminus Act_H$' forces the communications on high level actions).

The Future: As we said above, the basic motivation for NI properties was the analysis of information flows inside systems. However NI is very general and we think that it could be also applied in other quite different settings. As an example, in some recent works [6, 3], BNDC-like properties are used to check the Needham-Shroeder public-key protocol. It is an authentication protocol which works on the following simplified schema (we suppose that the two users A and B already know the respective public keys):

$$\begin{array}{ll} A \longrightarrow B: & PK_B(N_A, A) \\ B \longrightarrow A: & PK_A(N_A, N_B) \\ A \longrightarrow B: & PK_B(N_B) \end{array}$$

where N_A and N_B are nonces generated by A and B respectively; PK_A and PK_B are the public keys for A and B.

In [8] it is shown the following attack to the protocol:

$$\begin{array}{ll} 1) A \longrightarrow E: & PK_E(N_A, A) \\ 2) E(A) \longrightarrow B: & PK_B(N_A, A) \\ 3) B \longrightarrow E(A): & PK_A(N_A, N_B) \\ 4) E \longrightarrow A: & PK_A(N_A, N_B) \\ 5) A \longrightarrow E: & PK_E(N_B) \\ 6) E(A) \longrightarrow B: & PK_B(N_B) \end{array}$$

where E is the enemy and $E(A)$ represents the enemy impersonating A. We have that A requests an authentication with E (this is the hypothesis for the attack) then E starts a protocol session with B. The two sessions are composed by messages 1, 4, 5 and 2, 3, 6 respectively. At the end of the two interleaved sessions A is convinced to comunicate with E (and this is true) but B is convinced to communicate with A. This happens becouse E is able to "use" A to decrypt the nonce N_B. A corrected version of the protocol include the name of the sender in the second message:

$$\begin{array}{ll} A \longrightarrow B: & PK_B(N_A, A) \\ B \longrightarrow A: & PK_A(N_A, N_B, B) \\ A \longrightarrow B: & PK_B(N_B) \end{array}$$

In this way it is not possible for E to forward the second message to A (message 4 in the attack) since it will be $PK_A(N_A, N_B, B)$ while (to be correctly forwarded to A) it should be $PK_A(N_A, N_B, E)$.

In [6, 3] the protocol is specified in SPA. Then it is automatically checked that the first version is not secure using some some BNDC-like properties. The basic idea which permits to express authentication requirements using NI-like properties is very simple: we want to check if the enemy is able to interfere with the correct execution of the protocol so what we do is to check if the actions executed by the enemy interfere with the actions which characterize the successful authentication. As an example in the attack we have that B is convinced to communicate with A but A has never sent an authentication request to B. So we have a succesful authentication of A with respect to B without a request from A. Thus, we discover that the enemy can interfere with the execution of some particular actions: the first representing an authentication request from A to B (can be identified by the first message) and the second representing the successful completion of the protocol by B (can be identified with the second message).

So we have seen that NI is quite general and can be succesfully applied to the analysis of protocols. Moreover it is possible to change the underlying model in order to deal with mobile processes. As an example in [14] some of the BNDC-like properties are redefined on the π-calculus [10] and analysed. One of the main feature of the π-calculus is the possibility to send on a channel another channel or a process. Thus, it can be used to specify mobile systems which change their structure dinamically. It is shown that, passing to the π-calculus, most of the relations between various BNDC-like properties are preserved. Thus, we think that all the researches we are presently carring out on SPA-based NI properties could be easily extended to such a model.

References

[1] D. E. Bell and L. J. La Padula. "Secure Computer Systems: Unified Exposition and Multics Interpretation". *ESD-TR-75-306, MITRE MTR-2997*, March 1976.

[2] R. Focardi. "Comparing Two Information Flow Security Properties". In *Proceedings of Ninth IEEE Computer Security Foundation Workshop, (CSFW'96), (M. Merritt Ed.)*, pages 116–122. IEEE press, June 1996.

[3] R. Focardi, A. Ghelli, and R. Gorrieri. "Using Non-Interference for the Analysis of an Authentication Protocol". Forthcoming.

[4] R. Focardi and R. Gorrieri. "A Classification of Security Properties for Process Algebras". *Journal of Computer Security*, 3(1):5–33, 1994/1995.

[5] R. Focardi and R. Gorrieri. "Automatic Compositional Verification of Some Security Properties". In *Proceedings of Second International Workshop on Tools and Algorithms for the Construction and Analysis of Systems (TACAS'96)*, pages 167–186, Passau (Germany), March 1996. Springer-Verlag, LNCS 1055.

[6] A. Ghelli. "Analisi di un Protocollo di Autenticazione con il Security Checker". Master's thesis, Università degli Studi di Bologna, March 1997.

[7] J. A. Goguen and J. Meseguer. "Security Policy and Security Models". In *Proceedings of the 1982 Symposium on Security and Privacy*, pages 11–20. IEEE Computer Society Press, April 1982.

[8] G. Lowe. "Breaking and Fixing the Needham-Schroeder Public-key Protocol using FDR". In *Proceedings of Second International Workshop on Tools and Algorithms for the Construction and Analysis of Systems (TACAS'96)*, pages 146–166, Passau (Germany), March 1996. Springer-Verlag, LNCS 1055.

[9] R. Milner. *Communication and Concurrency.* Prentice-Hall, 1989.

[10] R. Milner, J. Parrow, and D. Walker. "A Calculus of Mobile Processes". *Information & Computation 100*, pages 1–77, 1992.

[11] A. W. Roscoe, J. C. P. Woodcock, and L. Wulf. "Non-interference through Determinism". In *Proceeding of European Symposium on Research in Computer Security 1994 (ESORICS'94)*, pages 33–53. Springer-Verlag LNCS 875, 1994.

[12] C. R. Tsai, V. D. Gligor, and C. S. Chandersekaran. "On the Identification of Covert Storage Channels in Secure Systems". *IEEE Transactions on Software Engineering*, pages 569–580, June 1990.

[13] J. T. Wittbold and D. M. Johnson. "Information Flow in Nondeterministic Systems". In *Proceedings of the 1990 IEEE Symposium on Research in Security and Privacy*, pages 144–161. IEEE Computer Society Press, 1990.

[14] V. Zannarini. "Sicurezza in Sistemi Mobili". Master's thesis, Università degli Studi di Bologna, December 1995.