No 46 – March 2009

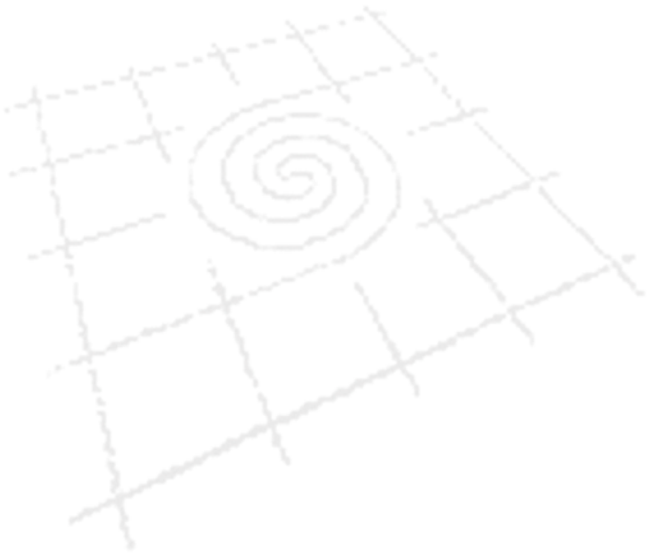
NIRSA Working Paper Series

NIRSA
National Institute For Regional and Spatial Analysis
An Institiúid Náisiúnta um Anailís Réigiúnach agus Spásúil

NUI MAYNOOTH
Ollscoil na hÉireann Má Nuad

THE SOCIOLOGY OF TRUSTED SYSTEMS: THE EPISTEME AND JUDGMENT OF A TECHNOLOGY

Stephano De Paoli
Aphra Kerr
Cristiano Storni



John Hume Building, National University of Ireland, Maynooth,
Maynooth, Co Kildare, Ireland.
Áras John Hume, Ollscoil na hÉireann, Má Nuad,
Má Nuad, Co Chill Dara, Éire.
Tel: + 353 (0) 1 708 3350 Fax: + 353 (0) 1 7086456
Email: nirsa@nuim.ie Web: http://www.nuim.ie/nirsa

HEA
Higher Education Authority
An tÚdarás um Ard-Oideachas
Funded under the
Programme for Research
in Third Level Institutions (PRTLI),
administered by the HEA

NDP
NATIONAL DEVELOPMENT PLAN

THE SOCIOLOGY OF TRUSTED SYSTEMS: THE EPISTEME AND JUDGMENT OF A TECHNOLOGY

Stefano De Paoli[1]

Sociology Department and NIRSA

National University of Ireland Maynooth - Stefano.DePaoli@nuim.ie

Aphra Kerr

Sociology Department and NIRSA

National University of Ireland Maynooth Aphra.Kerr@nuim.ie

Cristiano Storni

Interaction Design Centre

University of Limerick cristiano.storni@ul.ie

ABSTRACT

The goal of this paper is that of taking a first step toward a socio-technical conceptualization of trusted systems. In our view this might help in overcoming interdisciplinary differences and enhancing a common vocabulary for discussing trust issues for the Future of the Internet. In particular our main research question is to understand "to what extent and in which forms existing trusted systems embody social assumptions?" In order to answer this question we propose a new definition of Trusted Systems as situated Episteme: an apparatus of devices that set the conditions of possibility of certain practices while denying other practices. The conceptualization is augmented using the concept of technological mediation taken from the approach known as Actor-Network Theory (ANT). Our approach takes at its starting point the idea that it is possible to use sociological (from ANT) concepts to analyse and investigate the basic elements of Trusted Systems. This analysis opens up new possibilities for the sociological enquiry of Trust on a more micro, socio-technical level. In particular the paper puts forward the idea of Trust as result of the system design.

[1] The authors would like to acknowledge the support of the HEA under the PRTLI 4 programme and their partners on the 'Serving Society: Future Communications Networks and Services' project (2008-2010). This paper is a reduced version of the original, prepared for the TRUST 2009 conference. In particular we would like to thank David Malone for the proof reading of some technical details of the extended version of this paper.

1. INTRODUCTION

This work is part of an interdisciplinary research project that involves computer scientists, mathematicians, designers and sociologists. The goal of the project is to build an holistic view on the process know as "*The Future of the Internet*"[2]: the consideration that something has to be done for solving the existing mismatches between the original design of the Internet network and the current needs of various stakeholders. This view should take in to account different disciplinary view points, mixing them in an innovative framework for the next generation Internet. Interestingly, it emerged early on that different disciplines have different understandings of the key challenges facing the Internet, particularly as they relate to control over the users, information security and Trust. The latter in particular, seems to have different meaning for different disciplines with the existence, therefore, of problems in term of the vocabulary to be used for speaking about The Future of the Internet.

This paper is part of a concrete effort to bridge disciplinary gaps, and toward the construction of a holistic, comprehensive and interdisciplinary framework. Of particular concern for the authors of this paper is the extent to which we can delegate security, protection and control over the users to technological solutions, such as Trusted Systems (TSs hereafter), what governance principles get coded into these solutions and what the implications might be for end users.

The main research questions of this paper are "*to what extent and in which forms existing Trusted Systems embody social assumptions*?" and "*what are the implications for the design of Trusted Systems?*". In order to answer these questions we define, following Foucault (1977) and Law (2004) TSs as situated Episteme: an apparatus that set the conditions of existence of certain practices. Our method of analysis consists in a deconstruction of TSs, in particular based on the approach known as Actor-Network Theory (ANT) with a focus on the process of technological mediation (Latour, 1991; 2005).

In term of analysis our interest focuses on the main concepts of TSs and, in particular, those of Security Policy and Security Mechanism (i.e. the policy enforcement). In this paper we briefly analyze some of these concepts as they relate to a case study of Discretionary Access Control, that of the Multics system, as a concrete example that will help us in identifying the social dimension of TSs.

The paper is organized as follows. Initially we define Trust and Trusted Systems. Then we move toward the construction of an interpretative framework on the basis of which we analyze some examples. We finally move to a discussion on design implications and to the conclusion of the paper.

2. WHAT IS TRUST AFTERALL?

The problem here is in the very conceptualization of Trust. In most sociological literature Trust is seen as the attribute that prefigures the success or not of social interactions between individual and collective social agents (see for example Luhmann, 1979 and Giddens, 1990). In this sense, Trust is understood as a purely social feature that does not belong to natural or technological discourse. Therefore a Trust relationship implies a world in which the social is almost immaterial, and transcendental and Trust is closely comparable to an "invisible" element explaining why some interactions are successful and some others not.

[2] See for example the Bled Declaration See http://www.future-internet.eu/publications/bled-declaration.html

In Computer Science (CS hereafter) that of modelling Trust has become a central problem. While some more or less recent contributions in literature clearly take inspiration in their design from the social world (Jøsang, 1997; Nielsen and Krukow, 2003), it is also possible to affirm that Trust in this "environment" means something totally different from the sociological definition. To a certain extent the word Trust in CS does not mean the relationship between people and organizations in computer mediated settings and virtual teams/communities (Jarvenpaa and Leider, 1998). Trust is rather the reliability of a system to not suffer from failure and breaches under several threats such as for example crackers, viruses or Trojan horses. For this reason Nissenbaum pointed out that Trust in CS has more the resemblance of surety (i.e. reducing the risk to enter in a dangerous relations), whereas Trust in sociology is deeply centred on the notion of risky relationships (Luhmann, 1979).

The previous considerations changes the way we should understand Trust for The Future of the Internet. This is highlighted for example by De Paoli and Kerr (2009) in their review and comparison of the definitions of Trust is Sociology and Computer Science. The authors pointed out that most sociological work has seen Trust as a human attitude which a human being (the *trustor*) places in another human being (the *trustee*) in the form of a relationship (e.g. Sztompka, 1999). However in CS literature there is no such distinction and it is suggested that a TS – a machine - can be the *trustor* in a trust relationship (see for example DoD, 1983).

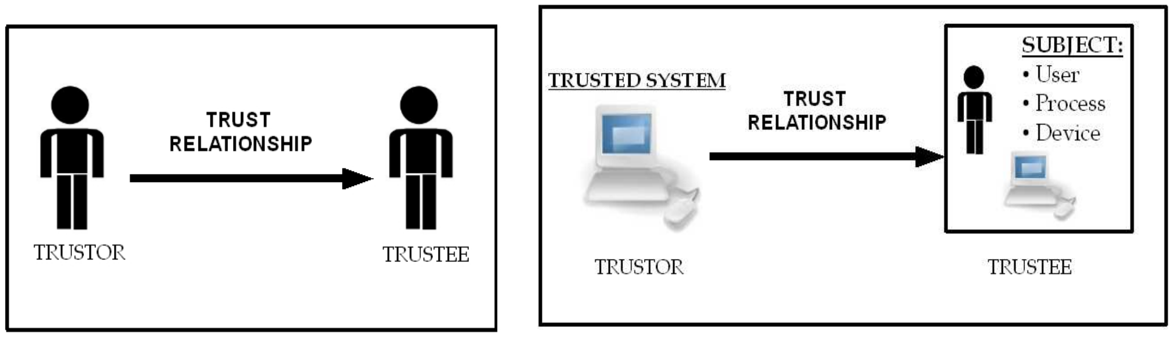


Figure 1 – Comparison between Trust notions (from De Paoli and Kerr, 2009)

These considerations modify the common sociological assumptions on Trust – which focuses exclusively on human beings - and require us to substantially change the way we conceptualize this concept. What we have here is s process in which the enforcement of moral rules is delegated to technologies that enact them back by imposing some prescriptions onto humans. These prescriptions are imposed back onto human in the form of "sentences" (i.e. the metaphor machine as text) that look very much like a programming language (Latour, 1992). These sentences have the form of "do this", "do that", "behave in this way", "do not do that" and so on and so forth.

A good approach to conceptualizing Trust would in-fact be that of seeing it as a socio-technical effect (a result) and not an individual (pre)condition (cause) for interaction. This means that we will have to focus on at least two aspect of the design rational of TSs. The first is that Trust is not purely human and in-between humans so that we should catch it, model it and implement it into a machine that replicates this purely social aspect. Trust is indeed already socio and technical in nature, it is not pure therefore there is nothing to transfer from the pure social to the pure technical. Rather we should understand how this already presents two ingredients: the human and the technical, and how these are intertwined in the production and reproduction of Trust (or mistrust) and then, only then, trying to develop means to support these - already in play - socio-technical logics.

The second would be that, since Trust is a collective socio-technical product, it has to be built together with the interaction and not as its presupposition. It does not precede action rather it emerges along with it. Maybe the issue is to better conceptualize the "filters" that judge humans as trustable or not, for the design of systems that might support human production and understanding of Trust and mistrust in practice in different domains.

3. TRUSTED SYSTEMS AS EPISTEME: a new definition

Having briefly defined Trust, we will now start to define a Trusted System. According to Artz and Gil (2007) today TSs cover different areas of computer protection and security and most notably the areas defined by them as "Security Policy", "Reputation Systems" and "Trust in semantic web". However it is easier for the goal of this paper to start from early works in the area computer security. The building blocks of Trust in CS were shaped during the 1960s and 1970, and starting this analysis from there is probably the easiest and straightforward way to proceed. We draw here in particular on some definitions that underline different aspects of TSs, useful for our discussion. For example, according to Nibaldi (1979, p. 1):

"*Trusted computer systems are operating systems capable of preventing users from accessing more information than that to which they are authorized.*"

This definition points out an important element of any TS: the ability of the system to prevent users from accessing certain types of information. A preventive action therefore will occur before the user's access to the information. This is interesting especially if we consider that *prevention*[3] means "*The action of keeping from happening or making impossible an anticipated event or intended act*.". TSs have therefore a "temporal" power of undertaking actions that will render certain events impossible, hence reducing the range of possible events: this enacts what philosophers have called the conditions of possibility (*Bedingungen der Möglichkeit*). In fact what we have here is the enactment of a temporal horizon in which what it is possible and what is not possible is, to a certain extent, anticipated by the system itself. In other words what will be possible and what will not be possible is brought in to existence (Heidegger, 1977) by the technology (Ciborra, 2002). Given these considerations it might be interesting to conceptualize TSs as a form of *Episteme*. According to Foucault (1977, p. 197) the Episteme is:

"*the strategic apparatus which permits of separating out from among all the statements which are possible those that will be acceptable within, I won't say a scientific theory, but a field of scientificity, and which it is possible to say are true or false. The episteme is the 'apparatus' which makes possible the separation, not of the true from the false, but of what may from what may not be characterised as scientific*."

[3] See definition at OED http://dictionary.oed.com/cgi/entry/50188315?single=1&query_type=word&queryword=prevention&first=1&max_to_show=10

With Episteme Foucault meant a rather high level apparatus (*Dispositif*) composed of an heterogeneous ensemble of discourses, institutions, architectural forms, regulatory decisions, laws, administrative measures and so on. The interplay between these elements seems therefore to set the conditions of existence of a certain knowledge discourse. What can be said in terms of positive statements depends therefore on the apparatus.

In his book *After Method*, Law (2004, p. 35) suggested a more modest and small scale definition of Episteme in which the conditions of possibility are enacted by local/situated devices. For this reason the conditions of possibility are empirically grounded and locally situated rather than being set by the foucauldian, modern, all encompassing Episteme. As an example Law quotes the book *Laboratory Life* by Latour and Woolgar (1986) in which the conditions of existence (production of reality) and therefore the existence of a certain scientific fact (like, for example, the existence of an Hormone) depends on the support that this fact receives from the apparatus of inscription devices (i.e. the laboratory apparatus). In this way the Episteme is a situated, ensemble of devices that enact certain practices while denying certain other practices.

Another definition can help us to conceptualize TSs as an apparatus of "devices" that enacts some conditions of possibility. The following passage is taken from Bell and Lapadula (1976, p. 9), for which in TS:

"*The essential problem is to control access of active entities to a set of passive (that is, protected) entities, based on some security policy. Active entities are called subjects […]; passive entities are called objects.* "

This definition points out several crucial features of the TSs as Episteme. It is important to briefly analyze them. First of all it is argued that the role of TSs is to control what are called active entities or subjects. Previously we have observed that TS prevents users from doing things, but in fact it is more likely that each active entity in the form of a user as well as a computer process is prevented from doing things (see fig. 1). Indeed, the key point is that TSs are defined in term of "*control exercised via security policies on active entities*". This is the Archimedean point of the problem. Three elements are at stake here: (1) the rule or security policy itself, which is static[4]; (2) the issue of control of a certain rule, which is a dynamic process; (3) the formal definition of the active entities using the system. These three elements constitute the apparatus of TSs.

The rule (1) is in fact a guideline on how/where/why/when a subject can and cannot access a piece of information (this is the prescription). Policies describe the conditions of possibility to obtain Trust, and can also prescribe outcomes if certain conditions are met. The control of the rule/policy (2) is instead the enforcement of the rule which is imposed onto subjects (the machine imposes the prescriptions): this is the security mechanism.

Finally we should clarify the differences between the subjects inside the TSs (3) and the subject outside them. Subjects inside the TSs are formally defined as statement in programming languages, these statements participate of course in the definition of outside subjects. However the latter may escape from their formal definition and always betray the design rationale. This is in fact the difference between the semiotic reader (the one inscribed in the text) and the flesh and blood reader (Latour, 1993). In other words subject as formally defined and subject as active entities are not necessarily exactly the same. TSs design (the writing) is also an attempt to

[4] In Web-Services, policies can be created dynamically. However, once the policy has been formulated, this is again a static rule.

channel the reader/subject (the flesh and blood) and convince them to accept their formal (semiotic) definition.

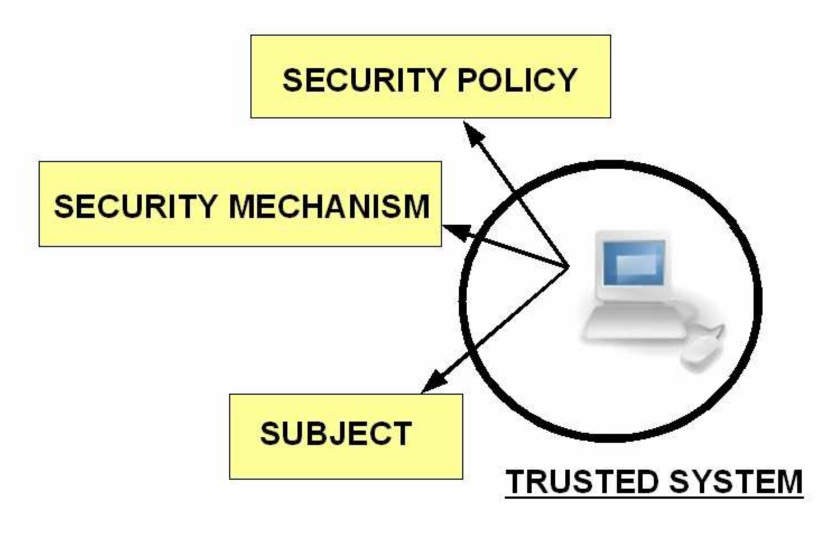


Figure 2 –Trusted Systems Episteme

Finally it is important to focus on a third definition of TSs, this time taken from the Wikipedia (Trusted System, 2009):

“a *trusted system is one whose failure may break a specified security policy*”

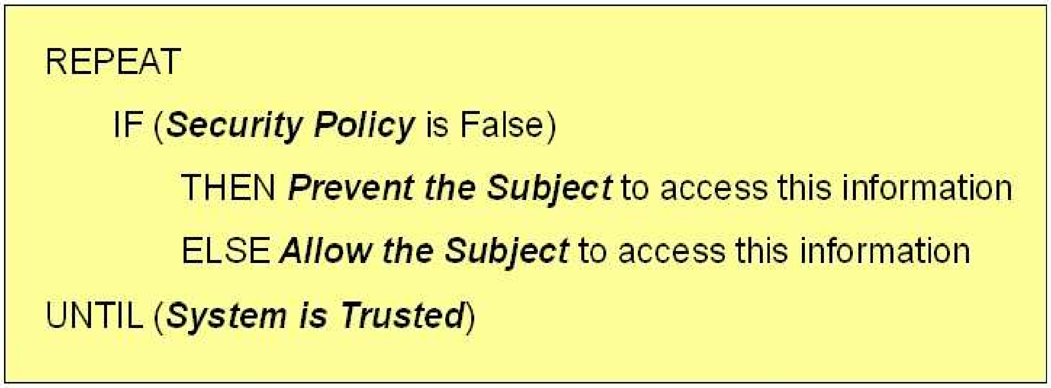
This definition is interesting because it clearly points out the cyclic self-referentiality of TSs Episteme. The reliability and validity of this Episteme will last until the enforcement of policies is broken and therefore suffer a failure. There are many sources of failure such as, the exploit of design flaws, the existence of Trojan horse and viruses and so on and so forth. The TSs Episteme therefore enacts a cyclic temporal horizon, a loop that will last until a failure in the system occur.

Mechanism	“Mechanism” refers to the features of a computer system that together enforce the protection policy. “ (Nibaldi, 1979, p. 9)
Policy	“A protection policy outlines a set of guidelines for determining how computer resources in general and information in particular may be shared..” (Nibaldi, 1979, p. 3)
Subject	“An active entity, generally in the form of a person, process, or device that causes information to flow among objects or changes the system state.” (DoD, 1983)

Table 1 – Definitions of TSs elements

4. TRUSTED SYSTEMS AS TECHNOLOGICAL MEDIATION

From here on it will be possible to proceed with a deconstruction of TSs by exploring their technical details. A simple way of formally describing the TSs Episteme is using a Pseudo-Code conditional expression (IF-THEN-ELSE) embedded in a loop cycle (REPEAT-UNTIL), like the following[5]:



```
REPEAT
    IF (Security Policy is False)
        THEN Prevent the Subject to access this information
        ELSE Allow the Subject to access this information
UNTIL (System is Trusted)
```

This formalization provides the basis for a comparison of the Trusted Episteme with the process of technological mediation (Latour, 1991). We use here, in particular, the description of this process provided in Lash (2002, pg. 53), which seems to adapt well to the Trusted Episteme described before.

According to Lash the goal of the Latourian technological mediation is weaving a morphism. This morphism is in fact the practice of shaping the form of some kind of entities by translating them from one point to another. This is also the meaning of technological mediation in which technologies enacts certain conditions of possibilities while making some others impossible. Mediation is indeed the creation of a link that did not existed before and that to some degrees modifies two elements or agents (Latour, 1999): during the mediation certain links are made stronger while other are made weak[6].

According to Lash the technological mediation in Latour entails three different steps. Each of them can be used to account for the Trusted Episteme:

1. *First, the creation of the morphism, of an analogy, to measure or judge or 'sort', or classify one entity by another.*

This first step of weaving a morphism is the judgment given to one entity by another. The condition "IF (**Security Policy** is False)" of the Pseudo-Code consists in fact in a judgment on whether a certain entity is a trusted subject (Bell and LaPadula, 1976) based on a a security

[5] This formalization is inspired by Serrano et al. (2009).

[6] This is for example well described in Callon (1986) research on Scallops, whereas the links between these and a series of entities (e.g. fishermen, the predator fishes) are made weak by the mediation of a cultivation technique.

policy. At this point a boolean variable[7] handles this judgement: true if the security policy is false; and false if the security policy is true. The IF condition itself probably contains the fundamental difference of the “information society”: the 0 (false) and 1 (true) difference. As we will see this first level difference is augmented by a phenomenology of second level differences: a division of labor between entities which is built on top of the 0 and 1 “division of labor”.

From here we can move toward the second step of this technological mediation process:

2. *To weave a morphism entails a second step of sending, or ‘passing’ the message to other actants, to other point in the network.*

The result of the judgement is sent or passed to the other elements/actants of the Trusted Episteme, respectively the THEN and the ELSE in the Pseudo-Code. At this point the decision on whether the subject is trusted or not has already been taken, and the new actants here have only the task of acting as intermediaries[8], they just: (a) prevent the subject from accessing the information if the subject is not trusted (THEN); (b) channel the subject toward the information if the former is trusted (ELSE).

Finally, as we saw earlier, the action of enforcing the rule is applied more than once:

3. *To weave means thirdly to create the net, the web, the network, and to extend it.*

The Trusted Episteme is a cyclical temporal horizon in which what is possible and what is not possible is continuously decided. This loop will last until Trust is broken by a disruptive event (i.e. UNTIL the Pseudo-Code clause “**System is Trusted**” is true). A web or a network of trusted entities with their practices is hence brought into existence by applying continuously the IF-THEN-ELSE conditional expression. Inside this loop “Trust” is therefore created and extended to the system/network (e.g. the company network; the Internet). We have here what is often defined as a mobilization of resources (Callon, 1986). The success of the morphism will last until all the resources that compose it are mobilized.

5. THE EPISTEME OF ACCESS CONTROL: the Multics system example

In this paragraph a concrete case study is described as a way to account for the argument that TSs embody social assumptions. During the 1950s and the 1960s a typical computer installation – a mainframe computer - consisted of a rather large machine, occupying entire floors of a building. These mainframes were used mainly for research purposes with a restricted group of programmers/researchers working on them. With the growth of the users base a data processing method known as Timesharing was introduced[9]. In Timesharing OS the CPU time is equally divided among the users, hence creating a centralized network (fig. 5). Using “stupid terminals”

[7] It is a primitive datatype that can assume one of the two values true or false.
[8] In Latour terms an Intermediary is something that transport meaning without transformation (Latour, 2005, pg. 39).
[9] This was meant to substitute a pre-existing data processing known as Batch (see on this Silberschatz, 2004).

users had access to computer resources (storage and CPU), hence permitting each user "*to behave as though he were in sole control of a computer*" (McKarthy , 1992).

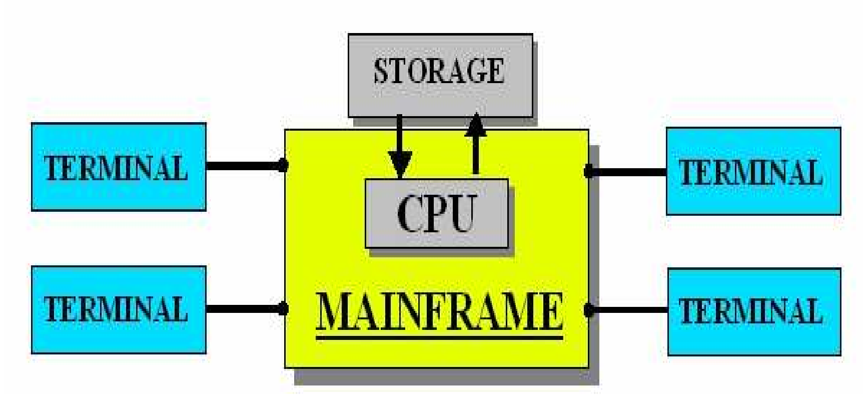


Figure 5– Timesharing system

The creation of this technology raised a series of security and protection concerns about how to "regulate" dozens or even hundreds of users using the OS. A book on Timesharing design (Watson, 1973), described some security and protection design principles as follows:

1. "*Protection of the system from user processes. It is imperative that users not be able to take actions which would stop the system from running or destroy information essential to the system.*
2. *Protection of the users from each other. A user must not be allowed to take an action which would harm the operation of another user's process.*"

In an OS protection refers "*to a mechanism of controlling the access of programs, processes or users to the resources defined by the computer system*" (Silberschatz, 2004). It is this mechanism that is interesting for us because, as we shall see, it embodies a division of labor between users and administrators of computer systems. We can observe this briefly describing the protection mechanism of a well known Timesharing OS: the Multics system.

The Multics was developed at MIT (from late '60 till late '80). What is important for us is to observe that the memory space (storage in figure 5) of the Multics was segmented. Each segment (i.e computer file) can be considered as the «*cataloging unit of the storage system, and it is also the unit of separate protection*» (Saltzer, 1974). Each segment therefore embodied a single security and protection level[10]. Associated with each segment there was an Access Control List (ACL): a list of each user of the system and the related permission to access the segments[11].

The user identity, established at login, is checked against the ACL of the thing being accessed (http://www.multicians.org/mga.html). This check against the ACL is the judgment done by the system on the subject (users or processes) identity: this is the first step of the Mutlics technological mediation. The ACL can be modeled using a matrix (table 1), in which we have the users and the segments. For each file there were different modes of access and in particular: None (access denied), R (read only), W (write only) and E (execute). In table 2 example the User_1 can only Read the file 1 & 3, while she is allowed to Write over the file2. The User_2 can instead

[10] Multics had a hierarchical file system, similar to that of UNIX .
[11] Due to limited space, we provide here a simplified account of Multics protection mechanisms.

also Write on File1&3, while she has not rights at all on file 2. So, we can imagine that User_2 gave to User_1 (this is in fact a DAC) the right to read but not to write the files 1 & 3. User_1 instead gave no right at all to User_2 for using file2.

The figure 6 represents a Multics segment. The sections 1, 3 and 4 t represent different part of the Multics segment (for example the specification of the physical address, section 1.), it is the section 2. which is of particular interest to us. In section 2 there are three bits that independently control the access to the segment. These 3 simple bits constitute the core element of the control mechanism of Multics: it is because of the setting of these bits that we have an overall division of labor.

	File1	**File2**	**File3**
User_1	Read	Read & Write	Read
User_2	Read & Write	None	Read & Write

Table 2 – Example of Access Control Matrix

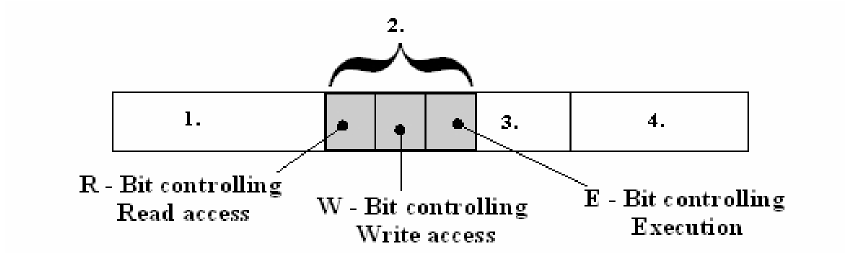

2.
1.
3.
4.
R - Bit controlling Read access
W - Bit controlling Write access
E - Bit controlling Execution

Figure 6 – Control bits of Multics segments, adapted from Saltzer (1974)

In order to understand how these control bits operate we can observe the following examples (7). The first one graphically describes the situation of User_2 & File1 of the matrix (R&W access to file1), while the second describes the situation of the User_1 & File 1 (Read only access). These examples illustrate the Timesharing protection design principle number **2.** and how the inner working protection mechanism worked. We are here in presence of the extension of the judgement to the whole "network", creating a set of general relationships between trusted subjects. The file protection bits were able to prevent the users from damaging other users' files: User_1 is not allowed to write over file1. We are here in presence of a certain set of conditions of possibility that are brought into existence by the system itself. However this form of protection does not characterize the whole conditions of possibility because we still have to look at the design principle number **1.**.

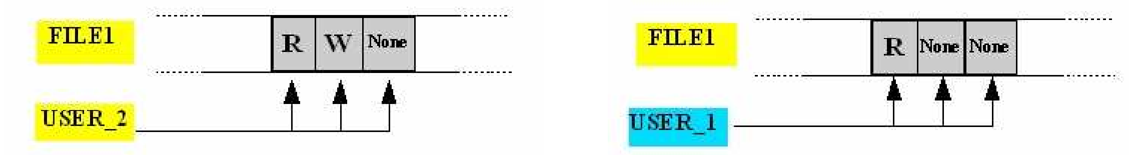


Figure 7 – Examples of Multics segment access control.

What is worth noting is that within the Multics there was a functional separation between the "normal" users and the "administrator" (like "users" and "root" in UNIX). While both were considered users by the system (i.e. both had username and password), the administrators had particular privileges over the system (Van Vleck, 1973):

- *"they have access to certain segments which regular users do not.*
- *the system will grant certain requests for them which it will not grant for normal users;*
- *some special abilities, such as the privilege of being able to patch the system, are available to system administrators."*

Some of these privileges, again, were determined by the control bits. The examples in figure 8 clarify this, where the normal users (in this specific case) do not have the right to write over the OS files or to install a computer application on top of the OS. Normal users had rights only to their specific segments and only administrators were allowed to install new programs and to modify and patch the OS.

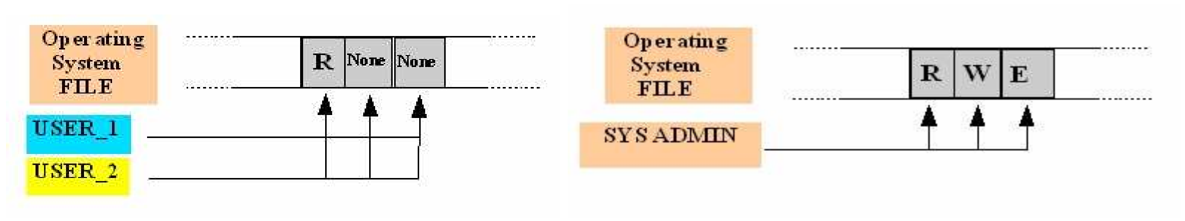


Figure 8 – Control bits and division of labor between users and administrators

The control bits clearly limit what the users can do with the OS and constitute a concrete implementation of the protection design principle **1**. Again we are here in presence of the extension of the judgment (*is this subject an administrator or not?*) to the whole network: the relationships between the whole system's users and the system administrators and between the users and the system itself depend on this. Again there is the enactment of some conditions of possibility: while the protection design principles had an acceptable goal (i.e. to prevent the user to harm other user's file or the whole system), this also leads to a very specific user and user's practices definitions: the user which is not allowed to modify the OS files, and therefore is not trusted to do that.

6. DISCUSSION: implications for the design of the next generation TSs

At this point it is important to provide some reflections for the design of the next generation of Trusted Systems for the Future of the Internet. One useful way to approach this problem is to reflect on what we have called the Trusted Episteme and its relation to users' practices and Trust.

What is at stake here is the relationship between innovation and stability as related to Trust. Strong TSS enact stability in terms of computer system and network usage, because their goal is to control the users' actions and prevent what is judged as bad behaviour or cheating. In this way, however, whatever falls outside the scope of policies is prevented from "legally" happening. In many cases this is good because it prevents disruptions and failure as well as prevents the action of malicious entities such as viruses. However in some cases this might greatly affect the ability of the user to innovate.

For example in the case of Multics all the users were prevented from modifying the OS files, a right belonging to the administrators only. However among the Multics users community some users were skilled programmers and had experience of working on systems without security and protection (Incompatible Timesharing System, 2009). The comparison between systems tells us something interesting:

The result was [*in systems without protection*], that whenever something in the system was broken, you could always fix it. You never had to sit there in frustration because there was NO WAY, because you knew exactly what's wrong, and somebody had decided they didn't trust you to do it [*in systems with protection such as Multics*] . You don't have to give up and go home, waiting for someone to come in in the morning and fix the system when you know ten times as well as he does what needs to be done. (Stallman, 1987b) – italic added.

This statement identifies an interesting element of the opposition between stability and innovation. If you want a system to be stable or to keep the necessary level of reliability (think about a military application) then these protection and security systems prevents the access to non-trusted subjects. However it could be that these subjects will be able to innovate (in the example the user can fix a bug in the system), but is prevented from doing so.

This situation is pretty much similar to what the philosopher of science Thomas Kuhn called scientific paradigms and the shift between paradigms as in normal and revolutionary science. In normal science there is no attempt to challenge the assumptions of the paradigm (the Episteme) and we are in a situation of stability and “accumulation” of knowledge. In revolutionary science however greater weight is given to anomalies. Certain anomalies (for example the Theory of Relativity) can have the power to radically modify the actual paradigm (the Mechanics) in a process of innovation (paradigm shift). TSs Episteme should probably work in this way. They have to sustain stability, since this is probably their main goal, however they have also to open themselves to positive anomalies that will enhance the users’ innovation.

A second possible implication for design is related to Trust itself. In CS exists a thing called “Trust modeling”: the idea that “social Trust” (see figure 1) can be modeled inside the system. The point here is to “capture” some sort of “Trust social model” and implement it (with appropriate reductionism) inside TS. It does not matter if this implementation has to do with Security Policy Trust or Reputation based Trust. Whether this strategy of implementing “social Trust” is good or not is not what is at stake here. The key point is instead the design process itself.

Taking something, formalizing it and including it into the system as if the social is totally separated from the technical is problematic. In terms of modeling Trust, this strategy clearly resembles the Artificial Intelligence Cartesian paradigm of a computerized thinking mind that is modeled as separated from the situation (Simon, 1995). However Trust, instead of a formal plan, should probably be conceptualized as a situated action (Suchman, 1987). As such the situated actions “paradigm” especially in its radical Heideggerian tradition (Ciborra and Hanseth, 1998; Ciborra, 2004), calls for the idea that things are brought into existence in the situatedness (*Befindlichkeit*) and that the technological systems have an important role in doing this (though often negative). Here it is where the definition of Trusted System as Episteme can give us a possible new approach in "modeling" Trust. We think that Trust can be built as part of the conditions of possibility, as a result of situated interactions and not as the precondition of a model that was formalized far from the situation. We made this point already in section 2 of the paper, but now it is clear that the "social Trust" toward which computer scientist try to converge should be grounded in the temporal horizon of prevention/non-prevention actions of the apparatus. Trust should then be brought into existence rather than formalized as totally separated from the situation.

CONCLUSION

This paper has put forward the idea that Trusted Systems can be conceptualized as small scale Episteme. We consider this as a first step that could help us in overcoming some of the interdisciplinary differences in conceptualizing Trust for the Future of the Internet. Future research will probably need to address the importance of the "judgment", as crucial element of TSs design.

REFERENCES

Artz, D. and Gil, Y. (2007) 'A Survey of Trust in Computer Science and the Semantic Web', *Journal of Web Semantics* 5(2): 58-71.

Bell, E.D. and LaPadula, L.J. (1976) 'Secure Computer Systems: Unified Exposition and MULTICS Interpretation', Belford, MA: MITRE Corporation.

Bell-La Padula model. (2009, January 22). In *Wikipedia, The Free Encyclopedia*. Retrieved 09:15, February 23, 2009, from http://en.wikipedia.org/w/index.php?title=Bell-La_Padula_model&oldid=265732371

Callon, M. (1986a) 'Some elements of a sociology of translation: domestication of the scallops and the fishermen of St. Brieuc Bay', in *Power, Action and Belief: a New Sociology of Knowledge?*, Law J. (ed.), London, Boston and Henley, Routledge and Kegan Paul:. 196-233.

Ciborra, C.U. (2002) *The Labyrinths of Information: Challenging the Wisdom of Systems*, Oxford University Press, Oxford.

Ciborra C. (2004) 'Encountering information systems as a phenomenon', in Avgerou C., Ciborra C. e Land F. (2004), *The Social Study of Information and Communication Technology*, Oxford, UK, Oxford University Press

Ciborra C. e Hanset H. (1999) From Tool to Gestell: Agendas for Managing the Information Infrastructure, in "Information Technology and People", 11 (4): 305-327.

De Paoli and Kerr (2009) Bridging Disciplinary Differences: Conceptualizing Trust for the Future of the Internet, (draft version here, http://eprints.nuim.ie/1145/)

Department of Defense, (1983) *Trusted Computer System Evaluation Criteria*. Retrieved 08 October 2008, URL: http://nsi.org/Library/Compsec/orangebo.txt

Foucault, M. (1977) *Power/Knowledge*, Random House.

Giddens, A. (1990) *The Consequences of Modernity*. Cambridge: Polity Press.

Heidegger M. (1977) *Letter on Humanism*, Retrieved 08 October 2008, URL: http://www.wagner.edu/departments/psychology/filestore2/download/101/MartinHeideggerLETTER_ON_HUMANISM.pdf

Jøsang, A., (1997) 'Prospectives for Modelling Trust in Information Security', In *Proceedings of the Australasian Conference on Information Security and Privacy*. Sydney, NSW: Springer LNCS: pp. 2-13.

Incompatible Timesharing System. (2009, January 9). In *Wikipedia, The Free Encyclopedia*. Retrieved 12:52, February 24, 2009, from http://en.wikipedia.org/w/index.php?title=Incompatible_Timesharing_System&oldid=262998408

Jarvenpaa, S.L. and Leidner, D.E., (1998) 'Communication and Trust in Global Virtual Teams', *Journal of Computer-Mediated Communication* 3(4), Retrieved June 2008, URL: http://jcmc.indiana.edu/vol3/issue4/jarvenpaa.html

Lash S. (2002), *Information Critique*, Sage.

Latour, B. (1992) 'Where are the missing masses? The sociology of a few mundane artifacts', in Biker W. and Law J. (eds.) *Shaping Technology/ Building Society*, pp. 225-258. Cambridge, Mass.: MIT Press.

Latour, B. (1993), 'Pasteur on Lactic Acid Yeast: A Partial Semiotic Analysis', *Configurations*, 1(1): 129-146.

Latour B. (1991), *We have never been modern*, Cambridge, Mass: Harvard University Press.

Latour, B. (1999b), *Pandora's Hope*, London: Harvard University Press.

Latour, B. (2005). *Reassembling the Social: An Introduction to Actor-Network-Theory*. Oxford: Oxford University Press.

Latour and Woolgar (1986), *Laboratory Life*, Princeton, NJ: Princeton University Press.

Law J. (2004), *After Method*, Routledge.

Lee, S.E. (1999). Essays about Computer Security. Retrieved October 07, 2008, URL: http://www.cl.cam.ac.uk/~mgk25/lee-essays.pdf.

Luhmann, N. (1980) *Trust and Power*, New York: John Wiley.

McCarthy, J. (1992) 'Reminiscences on the History of Time-Sharing' in.. *Annals of the History of Computing*, 14(1): 19-24.

Nibaldi, G.H. (1979) *Proposed Technical Evaluation Criteria for Trusted Computer Systems.* Bedford, MA: MITRE Corporation.

Nielsen, M. & Krukow, K., (2003). Towards a formal notion of trust. In *PPDP '03: Proceedings of the 5th ACM SIGPLAN international conference on Principles and practice of declarative programming*. New York, NY, USA: ACM Press: 4–7.

Nissenbaum, H. (2001) 'Securing Trust Online: Wisdom of Oxymoron?', *Boston University Law Review*, 81(3): 101-131.

Russell, D. and Gangemi, G.T. (1991). *Computer Security Basics.* Sebastopol CA: O'Reilly.

Saltzer, J.H. (1974), 'Protection and the control of information sharing in multics', Communications of the ACM, 7(7): 388 - 402.

Simon, H. (1995) 'Artificial Intelligence: an empirical science', *Artificial Intelligence*, 77: 95-127

Serrano M. et al (2009), 'Trust and Reputation Policy-Based Mechanisms for Self-Protection in Autonomic Communications', paper submitted to ATC 2009.

Silberschatz, A.(2004) *Operating Systems Concepts with Java*, Addison-Wesley.

Stallman, R. (1987b), 'Lecture at KTH', presented at Royal Institute of Technology, Stochom Sweden, 30 October 1986, URL: www.gnu.org/philosophy/stallman-kth.html

Suchman L. (1987), *Plans and Situated Action*, New York: Cambridge University Press.

Sztompka, P. (1999*) Trust: A sociological Theory*, Cambridge: Cambridge University Press.

Van Vleck, T. (1973) *Multics System Administrators' Manual*, URL: http://www.bitsavers.org/pdf/honeywell/multics/AK50-0_sysAdmin_Feb73.pdf

Trusted System. (2009, March 5). In *Wikipedia, The Free Encyclopedia*. Retrieved 11:45, March 5, 2009, from http://en.wikipedia.org/w/index.php?title=Trusted_system&oldid=275115789

Watson, R. (1973) *Timesharing System Design Concepts*. New York: McGraw-Hill Company.