Computational Trust for Peer-to-Peer Web Services

Hisham Salah
Electrical and Computer Eng.
Virginia Tech, USA
hisham79@vt.edu

Mohamed Eltoweissy
Electrical and Computer Eng.
Virginia Tech, USA
toweissy@vt.edu

Ayman Abdel-Hamid
Arab Academy for Science
and Technology, Egypt
hamid@aast.edu

RESEARCH ABSTRACT

The integration of Peer-to-Peer architecture (P2P) and Web Services (WS) technologies offer a great potential towards highly scalable, distributed, interoperable virtual environments in which functionality is easily accessed, composed and even introduced. Unfortunately, the service provider autonomy, the volatile environment, and the priori unknown behavior of web services, in addition to the frequent fluctuations in their delivered quality and performance levels, challenge the assumption of trusting a service or a service composition to behave as expected [1]. Trust Management Systems (TMS)s were introduced to dilute such issues, where several studies investigated and generally confirmed that trust systems benefit both buyers and sellers in online trade communities (*eBay* and *Amazon*) [2]. Reputation management is a well known tool to establish trust across the web; where it records past behavior of the subject of reputation and uses it later on as a publicly predictor of its future behavior. In the contemporary literature, most trust systems utilize reputation management as means to establish trust. We extend that view to include reputation as one of a number of credentials used to predict future behavior. In addition, few efforts have been done towards TMS for P2P WS. Moreover, current trust systems suffer from several drawbacks including; 1) *Absence of personalization:* derivation of trust score is oblivious of requestors' point of view. For instance a file storage service may have users requiring frequent access while others requiring high performance. In the first case a trust score which reflects availability of the services is needed, while in the second case a trust score which reflects response time and bandwidth is needed. 2) *Absence of honesty enforcement policies and mechanisms:* Honesty may be enforced by offering incentive or by applying penalties. Offering incentives may increase participation, but does not guarantee "good" behavior. While applying penalties reduce misbehaving, but may not increase participation. Thus both techniques may be required. 3) *Absence of trust validation:* where most trust systems assume perfect networking infrastructure [3], and interprets any QoS degradation as a malicious action performed by the WS. 5) *Most trust systems collect trust data anonymously:* without any quality of data evaluation. Such data may be customized to serve the malicious needs of an adversary.

Given these drawbacks, it is required to establish an evolutionary TMS framework to enable the selection and composition of P2P WS based on credentials. Such framework may also serve as a test bed for studying and analyzing different trust systems. In this research, we explore computational trust issues in P2P WS environments, and develop a theoretical foundation and system for trust management in such environments.

Our research attempts to address the following questions; 1) How can we accommodate the diversity of trust requestors' subjective views of trust in a way which allows each requestor to receive trust scores that are compatible with his own preferences? 2) How can we manage collective trust versus individual trust in various forms of compositions? 3) How can we manage trust for services with incomplete credentials? e.g. a file storage service which offers storage space for a time period of 3years, may only complete after 3 years. Other credential establishment mechanisms are needed. 4) What is the impact of consumers' decision of (not) invoking a service hosted by a certain service provider on the level of trust in that provider and his future behavior? 5) What forms of data patterns that are to be gathered in the trust system? And what are their sources? In this research, we attempt to propose a novel TMS architecture for P2P WS which overcomes drawbacks of current TMSs, and serves as a framework for studying trust issues in P2P WS. The proposed TMS framework mainly consists of three basic services namely; Monitoring Service (MS), Data Management Service (DMS), and TMS framework. TMS takes the form of a replicated service which may exist on each peer, i.e. any peer is capable of providing trust management services. Each peer implements 2 infrastructure services on top of which the TMS operates, namely: DMS and MS. *The DMS* is responsible for managing the distributed storage/retrieval policy of trust data, where different Data Managers (figure 1) from different peers, connect together to form an overlay network which handles portioning, replication, and storage/retrieval processes of trust data. In such manner the trust computation process (performed by the TMS) is decoupled from trust data management processes. The DMS utilizes an important component which deals with semantics of web services namely Ontology Processor (OP). *The OP* is responsible for processing, storing, and downloading ontologies of different web services domains. It serves different TMS components requests for ontological data in order for them to deal with services from different domains. *The MS* is responsible for creating monitoring threads to capture and collect data about peers engaged in invocations. In addition, it records network operations including: bandwidth, congestion, memory loads…etc. Here we should note that securing DMS and MS components from adversary is outside the scope of the paper. Our goal here is to introduce a system which mitigates the risk behind dealing with priori unknown web services rather than indentifying different threats. *The TMS* framework (figure 2) mainly consists of five components each of which performs a crucial job. and interacts through a communication module with the 3 basic roles of SOA (provisioning, consumption, and discovery) in addition to DMS and MS roles. The basic components of TMS are as follows: *Communication module (CM):* implements two types of protocols, namely: Trust Data Storage/Retrieval

and Monitoring Data Retrieval Protocol (TDSTMDRP) and Trust Communication Protocol (TRCP). TDSTMDRP is used for exchanging control and data between TMS, DMS and MS, while TRCP is used to exchange data and control with different SOA roles. *X-based Trust Managers (TM):*represents different kinds of TMs in the system, each TM calculates one or more trust aspects, which are applied to trust requestor's preferred scoring method. Possible trust managers may include: Context-based TM, Feedback-based TM, and Experience based TM. *Trust Calculator (RC):* this component accepts trust calculation request through the CM, along with the requestor's preferences regarding trust aspects to be used in scoring and the scoring method. On receiving such request, RC requests TMs and MSs to provide data necessary for calculating trust aspects, and after calculating them, trust aspects are applied to the preferred trust score method to provide the final score value. *Verifier:* In the proposed architecture there exist two kinds of verification, namely: Local Verification and Cross TM Mechanism Verification. The Local Verification is the process of validating the calculated trust score inside each TM component. The Cross TM Mechanisms Verification is the process of calculating the degree of confidence in trust scores passed by TM Mechanism components. *Rehabilitator:* controls privileges and access rights during probation period of misbehaving peers.

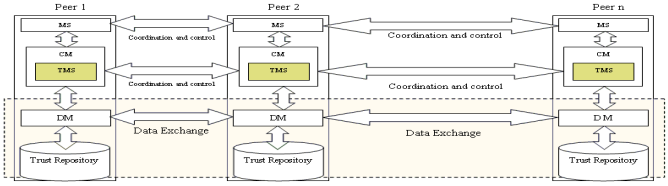

Fig.1 Trust Data overlay network

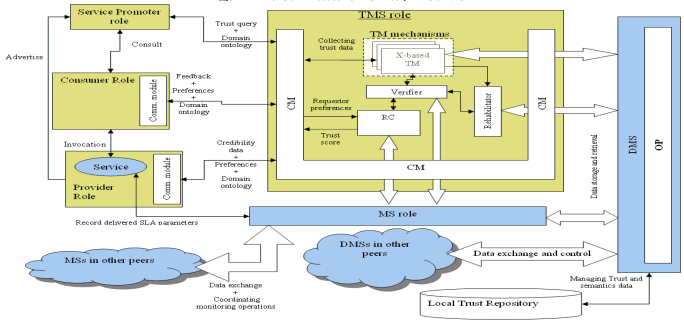

Fig.2 Trust Management System framework

Our proposed TMS holds the following main contributions; 1) Trust is defined as a collection of trust aspects. 2) TMS administrator has the capability to define new forms of TMs. Possible TMs may include: Feedback-based, Experience-based, and SLA-based. 3) Trust requestors have the capability to select both; trust aspects and trust scoring method to which trust aspects are applied to. 4)Collected data and information are verified in the TMS and produced in the form of credentials associated with quality of credential (QoC) which reflects degree of trust in credential it self. 5) Accepts services from different domains, where ontological data and information of different domains are stored and processed by the TMS. 6) Honesty enforcement is performed by a separate unit, which offers incentives and applying penalties to misbehaving individuals. 7) Separation between trust computation processes, and trust data management and monitoring processes. There are two main scenarios of interaction in TMS, but due page limit we'll only mention calculating trust score scenario of interaction. On receiving request, including; subject of trust Id, and trust calculation preferences parameters, CM forwards the Id to all TMs, triggering them to start compute trust aspects. In turn, each TM contacts the DMS —through CM— to retrieve required trust and ontological data. The DMS then starts comm./coord. with other DMSs to perform the task. After computing them, trust aspects are forwarded to the Verifier which contacts the DMS again —through CM— to retrieve verification data. After verification, reputation aspects are forwarded to the RC, which applies trust requestor's preferences to select among aspects and calculation methods to compute the final trust score. To examine the importance of verification unit, we conducted several experiments in order to explore the impact of low QoC on consumer's experience. The credential was setup as a simple boolean value which provides an expectation of the service having good or bad behavior. In addition, we defined the coincidence between the actual behavior of the service and the credential to be the quality of credential (QoC). The higher coincidence values the better expectation of service provider's behavior and vise versa. Figures 3 and 4 summarize results with services of population ratio 75 good to 25 bad. Each consumer selects a total of 200 services for invocation, and invokes services with credentials expecting good behavior while flips a coin to make decision about ones with credentials expecting bad behavior. From the figures 3 and 4 we may observe; 1) The higher the QoC value the higher number of good invocations consumers have. 2) The lower the QoC value the more cheating services consumers may invoke. 3) Random service selection yields better results in case of low QoC (credential coincidence of 60% or below). The experiments show that QoC is vital for computational trust systems.

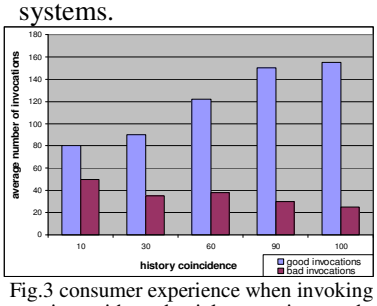

Fig.3 consumer experience when invoking services with credentials expecting good behav. and flipping a coin in case of serv. with credentials expecting bad behav.

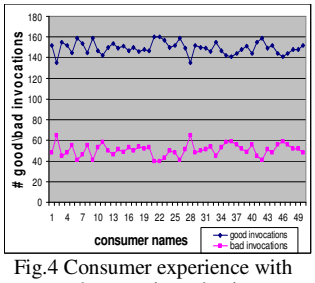

Fig.4 Consumer experience with random service selection

REFERENCES

[1]. P. Resnick and R. Zeckhauser. "Trust Among Strangers in Internet Transactions: Empirical Analysis of eBay's Reputation System", The Economics of the Internet and E-Commerce, Ed. M. Baye, Advances in Applied Microeconomics, 2002, vol. 11.

[2]. D. Artz and Y. Gil. "A survey of trust in computer science and the semantic web", Web Semantincs, 2007, vol. 5 (2) pp. 58–71

[3]. Mohamed Tamer Rafaei, Luiz Dasilva, and Mohamed Eltoweissy. **"**Reputation Management Systems in Ad Hoc Networks". Technical Report, 2006