A Distributed Community Approach for Protecting Resources in Digital Ecosystem

Ilung Pranata, Geoff Skinner, and Rukshan Athauda
Faculty of Science and IT, University of Newcastle
Callaghan, Australia

E-mail: ilung.pranata@newcastle.edu.au, geoff.skinner@newcastle.edu.au, rukshan.athauda@newcastle.edu.au

Abstract — Dynamic interactions occur excessively between the enterprises/entities in a Digital Ecosystem (DE) environment. These interactions promote resource sharing, task collaboration and application as services which are carried over the internet. Due to the open and loosely coupled nature and interactions, a DE poses several challenges, and in particular cyber security. These challenges span across from enterprise resources protection, comprehensive access control and identity management, strong and effective encryption mechanism, to the management of diversified security policies. However, current developments of security mechanisms for solving these challenges in DE are still in their infancy. On the other hand, security mechanism for protecting e-commerce transactions has matured over a number of years. Secure Socket Layer (SSL) is the preferred mechanism that has been widely adopted by many e-commerce websites. In this paper, we discuss our proposed solution for managing security in DE, and we further provide the comparison analysis between SSL and our proposed security mechanism for a DE environment.

I. INTRODUCTION

Since its first inception in 2002, the newly emerging concept of Digital Ecosystem (DE) has received increasing attention from researchers, businesses, ICT professionals and communities around the world. The DE concept is aimed at achieving a set of predetermined goals that resulted from a Lisbon summit in March 2000. The derived objectives primarily focus on dynamic formation of knowledge based economies [1]. Further, it was proposed that a knowledge based economy will lead to a creation of more jobs and a greater social inclusion in sustaining the world economic growth [2].

DE is a multi-dimensional concept that encompasses several current technology models such as collaborative environments [3], distributed systems [4], and grid technology [5]. The combination of concepts from these models provides DE the ability to deliver an open, flexible and loosely coupled resource sharing environment. These unique resource sharing characteristics are critical to preserve and improve the synergy among components and enterprises inside a DE environment. However, this combinational configuration also produces a number of complicated security issues. Such security problems need to be addressed before the full implementation of a DE concept can be realised. Unfortunately, after the review and evaluation of the current literature on the DE security dimensions, a number of deficiencies in DE security architecture, particularly in a scalable security mechanism which is capable to comprehend the various security protection and policy refinement are apparent.

A key challenge for enterprises that are involved and participating in a DE is to determine the right entities or users whom are able to access the services, resources, knowledge and information hosted in the DE by member enterprises. There are several key reasons why this challenge is particularly difficult to address. Firstly, the occurrences of multiple resources published and shared by each enterprise in a DE and secondly, the situation where various clients are able to access each individual resource. Due to these reasons, enterprises and more importantly DE’s, to ensure their increasing uptake and utility, urgently need a mechanism that effectively manages their clients’ access control and authorization permissions. Further, this mechanism must put the management of various security policies into consideration. Several centralized mechanisms face several issues in protecting the DE such as single point failure, administration overhead and access permission assignments [6, 7].

In this paper, we provide a detailed resource protection mechanism in DRPM, and incorporate a community based trust approach for its authentication and authorization process. This approach would promote the engagement of DE community in protecting DE resources while ensuring a fine-grained resource protection is upheld in a distributed approach. We further detail an approach introduced to protect the resource exchange in the proposed mechanism. The ultimate aim is to protect the resources from any unauthorized use in a Digital Ecosystem. The remainder of this paper is organized into five sections. Section 2 presents a brief overview of DRPM. Section 3 describes our extension to include community based trust into DRPM. Further, it also discusses a detailed implementation of DRPM. Section 4 provides an
evaluation to our proposed extension, and finally section 5 summarizes our research and suggests the future work of this research.

II. OVERVIEW OF DRPM

The Distributed Resource Protection Mechanism (DRPM) [8, 9] has recently emerged as a new mechanism to provide a fine-grained resource protection for a DE environment through strong authentication and authorisation. Similar to SSL, DRPM relies on the Public Key Infrastructure (PKI). However, the PKI implementation in DRPM mechanism differs significantly. DRPM focuses on two important workflows: the registration workflow and the resource access workflow. The registration workflow is the first step for any resource consumer or client to engage in a transaction with the resource provider. Three important steps occurred in the registration workflow: exchange of public keys, creation of client profile, and the generation of capability token. The first step of registration workflow is to exchange the public keys between the resource consumer and provider. These public keys are important for the encryption and signature verification in the later part of resource access. Once public keys are exchanged, both parties store it in its own secure repository.

The next step of registration process is the creation of client profile in resource provider end-point. It is necessary for the resource provider to gain the information of the prospective resource consumer during initial interaction. Such information is crucial for resource provider to understand who consumes its resources and for what purpose the resources are consumed. Armed with this information, resource provider feels more secure that its sensitive resources are protected from any misuse or malicious act. In DRPM, client profile is adopted to capture all required, but voluntarily provided, information about a client. Beside its functionality for capturing information about a client, client profile also facilitates the auditing process for any disputes that may occur.

Once client profile is created, the next step of registration process is to create a capability token that contains all access permissions given to the client. Capability token is generated by the resource provider and is sent to the client. A simple capability token would consist of the client ID, provider ID, resource access URL, validity time stamp, and all given resources and their respective permissions. Once generated, resource provider would generate the hash no of capability token and store it in its own server. The purpose of generating the hash no is for integrity checking during resource access. The capability token will then be encrypted by client public key that was obtained earlier and signed using provider private key. It is then sent to the client for the future resource access request.

During resource access workflow, client encrypts the capability token with resource provider public key and signs the token with its own private key. Further, a pre-master key is issued by the client. The purpose of the pre-master key is to create the session or symmetric key for data communication. Both capability token and pre-master key are then sent to the resource provider. In resource provider end-point, the decryption and signature verification processes on the presented capability token are performed. If the signature of capability token cannot be verified, the resource access is rejected. It is clear that the capability token and its signature verification provide the authentication and identity proving process of the client. After the signature is verified, the resource provider would generate the hash no of the presented capability token and the hash no obtained from the registration workflow. If the hash no verification is successful, the resource provider then grant the access to the resources based on the permissions contained in the capability token. Fig. 2 shows the step by step process of resource access workflow.

Fig. 2 DRPM resource access workflow.

III. EXTENSION TO DRPM

In this section, we propose an integration of community based trust services to the DRPM.

A. Community Based Trust Services

The existing internet mechanism such as SSL [10] heavily utilizes a central CA as the third party entity that provides digital certificate for both client and resource provider. A certificate is required to certify the authenticity of an entity which is vital before any
transaction occurs. In order for a certificate to be accepted, CA must be trusted by both client and resource provider. Although the utilization of CA is essential for the entity verification, several detriments could possibly limit such implementation in a DE environment. First, the implementation of CA creates barriers to entry for the Small and Medium Enterprises (SMEs) due to the cost ineffectiveness and hectic audit process.

In its practical implementation, an enterprise is required to pay a yearly subscription fee to the CA provider for each certificate that is issued for single enterprise service. Further, a complex audit process is conducted annually to verify the identity of an enterprise. Due to these reasons, the CA certificate is mainly utilized by a small number of large enterprises while hindering such implementation for the SMEs. Note that, this situation further deviate the core purpose of a DE concept which is to enable a greater involvement of SMEs by allowing SMEs to actively engage, collaborate, and gain advantage from an open socio economic environment [11, 12].

Second, Significant vulnerabilities would occur if CA security is compromised by the malicious hackers that are able to obtain signed certificates. In this case, the clients or resource consumers do not have any knowledge that the hacker certificate is bogus as it is signed by the legitimate CA. Such situation further allows the clients to unknowingly share the sensitive information to the hackers. This was the case that was faced by the Microsoft and VeriSign in 2001 which the unauthorised entity posed as Microsoft employee and requested VeriSign to issue two digital certificates [11]. This incident which allowed the attacker to post a virus has affected all Microsoft retail and business customers. The similar incident was also transpired recently in 2011 [12] where an attacker whose IP address originated from Iran forcefully requested several fraudulent certificates to be issued by Comodo certification services for the well-known websites, such as Google, Yahoo, Skype, Mozilla, and several others. The recent similar incident affected Diginotar in August 2011, and it forced the company to declare bankruptcy and abandoned its customers To address the above concerns, the idea presented in this paper is to integrate the community trust services, such as Web of Trust (WoT) [13] to DRPM workflow, particularly in the client registration process. We present this idea as an alternative approach for protecting resources in a DE environment.

Web of Trust (WoT) is a community endorsed certificate which provides a decentralized trust management in a digital community. In WoT, there is no central authority (such as CA) that every entity trust, instead each entity is able to sign others certificates or public keys to build an interconnected web of public keys. The identification of an entity is provided primarily by his public key which is digitally signed by any number of "introducers". Three degree of trustworthiness is introduced to reveal the reliability of the entity public key certificate: undefined, marginal and complete. Final decision for trusting the entity is rely on the user after examining the degree of trustworthiness. The prominent application of WoT is in Pretty Good Privacy (PGP) [14], which is used extensively to secure emails.

B. DRPM Extension Evaluation

The adoption of WoT into a DE environment is an alternative approach to ensure that the confidentiality, integrity and authentication of resources are being sustained when there is a non-existence of central CA. A DE virtual community comprises multiple interacting entities that mutually trust each other. Such community is an appropriate medium for the implementation of WoT. The similarity between DE concept and WoT lies on the power manifestation of the trusted virtual community, on which each entity provides the verification or endorsement to another entity. Further utilization of the WoT community involvement also facilitates the evolution concept of DE. WoT allows new entities or services to be trusted by the community without a procedure to follow the registration process on a central CA, while the outdated entities or services disappears without administrative overhead on a central CA.

As the decentralized trust mechanism in DE environment matures, the occurrences of CA as a single certificate provider can be minimized. Several research in [15] and [16] notably presents the decentralized trust mechanism in virtual communities. Further, the elimination of CA is also adopted by JXTA overlay [17] for securing the peer-to-peer networking. In DRPM, the decentralized trust mechanism will therefore enhances the verification of genuine resource provider during registration process. In addition to WoT extension, the utilization of PKI further ensures a fine grained access protection on the DRPM registration and resource access workflow. Public keys of both client and resource provider are being exchanged during the registration process. These keys are retained and re-used for the future resource access. In any case of public key is being compromised on which the generation of a new key is required, this key will be distributed to the designated business entities.

IV. ANATOMY OF SSL

Secure Socket Layer (SSL) [10] is an industry standard for securing e-commerce transactions. This technology is implemented in a secure HTTPS protocol no 443, and it has a significant dependent on the Certificate Authority (CA). In SSL, CA primarily functions as the trusted third party that proofs the identity and authenticity of the entities that are involved in a transaction. Essentially, any entity may generate a request to CA to prove its identity by submitting its self-generated public key. This is then followed by a thorough process of identity proving and registration that is conducted by CA. After such process, CA produces a certificate that contains its ID, the server ID and other important information such as
validity period, key algorithm, etc. CA then generates a certificate fingerprint to sign the newly generated certificate that serves as an instrument to indicate the authenticity of the certificate and its owner. On any secure HTTPS transaction, the provider entity always provides this certificate to its client. The certificate can be viewed on any notorious web browser such as Internet Explorer, Mozilla Firefox, etc. when secure transaction is used.

Client verifies the authenticity of provider certificate by validating the certificate signature that was generated earlier by the CA. The validation process requires the client to hold CA public key in its workstation. This means that trust must be established in advance between client and CA which prompts the client to retrieve the public key of all CAs that it trusts. To simplify the process of selecting which trusted CA, the majority of web browsers have performed the selection of the trusted notorious CAs and embedded their certificates into its application. If client accesses the provider whose certificate is not trusted in the browser e.g. in the case of enterprise private CA, manual process to embed the certificate must be performed. Further, an additional certification verification process is conducted toward the revoked certificates. Each CA also provides a list of the revoked certificates that prompts the browser not to trust the certificate that is signed by the CA. Client’s browser can obtain the revoked certificates by requesting a copy of Certificate Revocation List (CRL) from CA or through Online Certificate Status Protocol (OSGP).

SSL also ensures the confidentiality of the information that is transmitted through HTTPS protocol. Such confidentiality is enforced through the symmetric and asymmetric encryption processes. The asymmetric encryption process relies on Public Key Infrastructure (PKI) to securely transmit the symmetric key for data encryption in later part. Essentially, two main processes happened when secure HTTPS connection is initiated by the client: public key transfer process through certificate and symmetric key generation process. These processes are notoriously known as SSL handshake [18].

The SSL handshake process is further broken down into several child processes as follow:

1. Client sends a ClientHello message along with its supported cipher suites to the provider server requesting for HTTPS connection.
2. Server responds with ServerHello message with its selected cipher suite, random number and session ID.
3. Server then sends its certificate to the client that contains server public key and indicates the ServerHelloDone message.
4. Client generates a pre-master key and encrypts it using server public key. This pre-master key will be used to generate the symmetric key and MAC (Message Authentication Code) key.
5. Client then send ChangeCipherSpec message to indicate the server that any communication from now on would be encrypted using the secret key.
6. Server also responds with ChangeCipherSpec message to indicate the client that any communication from now on would be encrypted using the secret key.

In later development, TLS (Transport Layer Security) is introduced to replace SSL. TLS has a similar handshake process as SSL with major differences on its operating layer and PKI algorithm. TLS operates on transport layer of OSI (Open System Interconnection) [19] model and adopts Diffie-Hellman [20] algorithm while SSL operates on the session layer of OSI model and adopts RSA [21] algorithm.

V. COMPARATIVE ANALYSIS BETWEEN SSL AND DRPM

As have been discussed in previous section, SSL heavily depends on CA for authenticating and certifying the identity of an entity. The occurrence of CA in SSL is extremely vital for provider identification process. This could possibly generate significant risk of single point failure in DE environment. That is, any failure that happens on CA could halt the entire identification processes in DE environment, and without any identification processes in place the electronic transactions could not be performed as there is no trust that could be drawn between the resource consumer and provider. On the other hand, DRPM utilizes the WoT (Web of Trust) mechanisms where the public key of resource provider is signed by other entities in the ecosystem. This further reduces the reliance over the central CA to prove the provider certificate while increasing the participation of the entities inside a community to validate each other identities. The failure of a validating entity does not easily jeopardize the entire identification process as other validating entities are still exists in the environment.

Although SSL proves the identity of resource provider and authenticate it to the client, client
authentication on resource provider endpoint is limited in SSL. To authenticate client in SSL, client needs to create the public key and request CA to sign its public key. This is essentially a similar process with provider authentication, and it requires client to pay annual cost to the CA to maintain its certificate signature. To avoid such cost for the clients, most resource providers implement other means for authenticating its clients. The most popular solution is to generate a pair of username and password for the clients to remember. Client authentication is then performed by verifying the username and password during each login. The username and password are encrypted by symmetric key that was generated earlier during SSL handshake. DRPM takes different approach in authenticating the client by issuing the capability token and by verifying client signature. A client is authenticated if the presented capability token is genuine and it contains client signature. Figure 3 shows the comparative analysis on processes between SSL and DRPM. In addition, DRPM grants access to the client based the permissions that are contained in the capability token. As this functionality is not performed in SSL, it requires provider to implement other authorisation method such as Access Control List (ACL), Active Directory (AD) and etc.

VI. CONCLUSION AND FUTURE WORK

In this paper we have highlighted the needs for protecting enterprise resources from any unauthorized use in a DE environment. Further, we have re-examined the Distributed Resource Protection Mechanism (DRPM) for DE in providing a comprehensive resource protection. DRPM emphasizes decentralized authorization mechanism performed by individual resource providers. It is achieved by utilizing the client profile and capability token for its authorization permissions. We have also extended DRPM with Community protection. Finally, we discuss the existing e-commerce solution: SSL, and we provide the comparative analysis between SSL and DRPM. It has been shown that DRPM solves several limitations of SSL for its implementation in a DE environment.

REFERENCES


