Towards a Taxonomy for Security Threats on the Web Ecosystem

Carlo Silva, Ricardo Batista,
Ruy Queiroz, Vinicius Garcia
CIn, UFPE
Recife, PE, Brazil
Email: {cmrs, ruy, vcg}@cin.ufpe.br

Jose Silva, Daniel Gatti
TiDD, PUC-SP
Sao Paulo, SP, Brazil
Email: {jlcs, daniel}@pucsp.br

Rodrigo Assad, Leandro Nascimento,
Kellyton Brito, Pericles Miranda
DEINFO, UFRPE
Recife, PE, Brazil
Email: assad@usto.re

***Abstract*—The aim of this paper is to present a taxonomy for security threats on the Web ecosystem. We proposes a classification model based on 21 vectors divided into 8 distinct security threats, making use of levels of abstraction and criteria for discrimination which consider propagation and similarity in vulnerabilities. We also propose to estimate the risk factor and impacts on assets, considering data breaches, human aspects and service reliability. In addition, we validate the taxonomic model proposed through the catalogues of attacks facing the public. Thus, it was possible to observe its applicability for most of the attacks which appear before the public.**

***Keywords*—Web Application Vulnerabilities, Web Browser Vulnerabilities, Social Engineering, Taxonomy for Security Threats**

I. Introduction

Currently, the Web is gaining force as an efficient platform for developing services. Diverse organizations have become motivated in placing their applications in this environment, and with Cloud Computing, or more precisely with Software as a Service (SaaS) model, the tendency has strengthened. This advent has had leverage on the Web in such a way that its platform has been extended into a collaborative environment, providing a greater interactivity between services, users and devices [7]. However, the challenges for assets management have become notorious in such an environment, as can be observed in the frequent cases of vulnerabilities' being exploited [14], resulting in breaches to sensitive data, the identity of users and, consequently, the trustworthiness of those services which maintain assets.

Generally these incidents become names with an aim to put the failures into an inventory, facilitating the identification and prevention of the attacks, alongside vulnerabilities which result from them. Some organizations specialized in the subject, for example MITRE and Open Web Application Security Project (OWASP) strive to maintain and periodically renew a catalogue of these threats. Curiously, these artifacts do not describe the relationships which determine the common characteristics between threats. Such detailing would result in a categorization of failures for a determined pattern of behavior.

A good example for this level of detailing is taxonomy, responsible for presenting a classification for a particular ecosystem. We observe that, in view of the fact that the threats do indeed present similarities, the state of the art in the Web environment continues to require this kind of solution. Such an artifact has the potential to offer support for dealing with vulnerabilities, since their systematic classification considers behavior and characteristics which arise and have things in common, providing for a clinical overview of the menaces. We can cite the similarities in the variations of Buffer Overflow attacks (BoF) [12] which work into differing data structures; they share the same vector and, when used, reproduce similar impacts on assets.

This study aims to present taxonomy for security threats in the Web ecosystem. Besides the usual benefits of taxonomy, such as providing a streamlined terminology, our proposal establishes a model as a specific solution for the Web scenario, considering factors such as the propagation of threats, similarities between attacks, and the respective impacts these have on assets. Our taxonomy is a theory founded on data where we raise hypotheses, prove phenomena and, in the end, consolidate an artifact which classifies the ecosystem into 3 domains, 8 threats types and 21 attack vectors. Besides this, we evaluate our study through a methodology which investigates the proposed taxonomic model, considering the registered attacks which arise in catalogues and which are presented in the literature.

II. The Web as an Ecosystem

An ecosystem is defined as a set which designates the observation of behavior and interaction between different individuals in a single environment. Following Figure 1, the interaction between the set of services, devices and resources on the Web is what we define as an ecosystem. Consequently, it is a tool which possesses a great number of responsibilities. Its aim is to provide interaction between a user and a Web application, which is defined as the service which managing users assets.

According to [8], an asset is any and every information type where there is value added by its owner, and that when hampered carries with it serious consequences for those involved. In this scenario an individual denominated attacker appears. His aim is to execute illicit harm on assets. The motives are varied, being for fun, ideology, financial gains or political interests. In comparison with the user, his flow of interaction is more complex since he takes it upon himself to find behaviors which are not expected by the assets' administrators. His first step is to exploit vulnerabilities in the service, Web browser or in the users themselves, the chosen method of exploitation being what we call the attack vector.

978-1-5090-0223-8/16/$31.00

Fig. 1: The Web Ecosystem.

When vulnerability appears in a service, one supposes that the responsibility of the asset is totally directed towards its administrators. This falls in line with a major dilemma, where one expects that it should be in the hands of the management to adopt strong security measures for development and maintenance. However, the attacker can exploit vulnerabilities with the Web browser. In this context, the one responsible for security is not well defined since it is a matter of an environment susceptible to a lack of acuity on the part of users.

Finally there is another perspective when the attacker exploits the social structure surrounding the user or the service. Unlike the aforementioned vulnerabilities, it takes advantage of lack of care regarding the human factor, even being possible in a direct way by inducing techniques which intercept user engagement, or indirectly where the attacker analyzes the personal life of the victims to exploit loopholes in service measures.

Based on Figure 2, it is important to clarify the semantics of the security attributes involved, being divided into 3 distinct sets. The first treats Loss or Leakage of Identity, a category which ties together human aspects of service users, this category being divided into: Privacy (Pr), Non-Repudiation (Nr) and Anonymity (An). The second group, denominated Data Breach, is focused on data stored or traffic between the user and the service, being divided into: Confidentiality (Co), Integrity (In) and Availability (Av). And, finally, the Non-Compliance group tells of the service's trustworthiness, dividing itself into Auditability (Ad), Authenticity (Au) and Responsibility (Re).

III. Proposal

The taxonomy proposed makes use of supporting data [5] from the most common attacks present in the literature. Our support was constructed based on systematic revisions of the literature and empirical studies. The fact is that at a single point of vulnerability there can be unleashed various and distinct attacks. This is because they share behaviors and similarities in their method of execution. However, in spite of the similarities, each attack can hamper a variety of features, causing distinct impacts on the assets. In the wake of this, we aim to propose a model which offers service administrators a holistic vision of the management of these threats on the Web ecosystem.

Based on this theory, we believe our approach makes possible the construction of a modeling capable of estimating, identifying and preventing a considerable number of potential threats within Web ecosystem domains. The proposal considers scenarios and security attributes presented in Section II, where the semantics of the attacks is centralized, their vectors of propagation having a base, since it is focused on the method of exploiting vulnerabilities. The taxonomy proposed is presented in Figure 3, where the ecosystem is divided into three distinct domains: Service, Service Consumption and Social Engineering, where 8 threats group together resulting in 21 attack vectors.

A. Domain 1: Service

This domain refers to threats originating in the failures of technological resources present in the service domain, or in the application, due to the absence or inefficiency of a solid process of secure development. The domain is divided into five threats and 13 vectors which will be described in detail in the next section.

Threat 1 Missing Access Control and Identity Management: These are flaws which happen during authentication and service authorization. This threat is divided into 2 vectors, as follows:

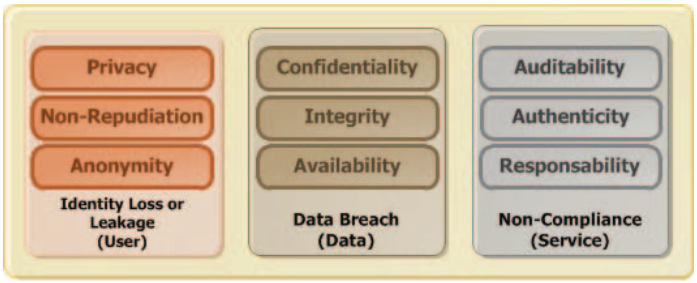

Fig. 2: Security attributes grouped by asset's category.

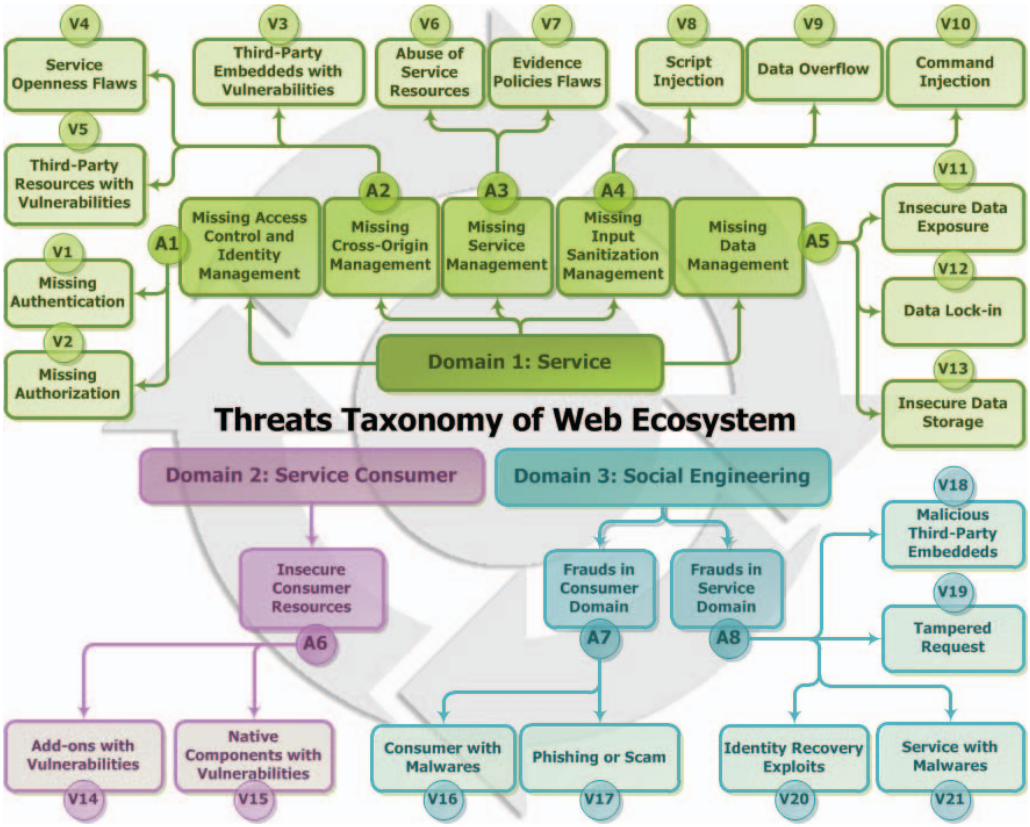

Fig. 3: Taxonomy of threats on the Web ecosystem.

Vector 1 Missing Authentication: These are flaws resulting from the management of identities, permitting to the attacker an identity which does not belong to him. An example is the absence of verification through two stages [16].

Vector 2 Missing Authorization: These flaws result from access control, permitting the attacker to become privileged in access to directories, functionalities or service modules which were not conceded to him. The classic examples are the Path Transversal attack [12] and flaws in the access control to the application.

Threat 2 Missing Cross-Origin Management These are flaws in the interoperability of a third-party application. This threat is divided into 3 vectors, as follows:

Vector 3 Third-Party Embedded with Vulnerabilities: Also called embedded, these resources are well distributed by content providers, the sharing of videos being a typical case. The flaw occurs when the consumer service does not expect vulnerabilities coming from the content provider incorporated in the presentation layer. These vulnerabilities, even though existing in third-party resources, for being present in the application domain, can propagate risks in assets. The case also applies to Web Widgets, Mashups [7], Websocket and Same-Origin Policy [1] flaws.

Vector 4 - Service Openness Flaws: This case applies when the service decides to be a provider of resources, that is to say, it desires to make its resources available through a streamlined and programming interface for the public, or restricted to certain consumers. Examples are the flaws in the development of an API or in the use of Cross-Origin Resource

[1] *Same-Origin Policy:* https://goo.gl/f8ZoVZ

Sharing[2].

Vector 5 - Third-Party Resources with Vulnerabilities: In this case, the service decides to be a consumer of third-party resources, very common in the adherence to solutions based on clouds or content providers. Examples of exploitation: API consumption or Web Services with vulnerabilities.

Threat 3 Missing Service Maintenance Management: These are flaws in the implantation or provision of the service for the end user. This threat is divided into two vectors:

Vector 6 Abuse of Service Resources: These are flaws originating in prevention policies for resources susceptible to use exploitation by the end user. The preoccupations come down to the physical or logical abuse of the service. Examples of exploitation: incorrect configurations on the Web server, Denial of Service (DoS), flaws in the server's operation system, expired certificates, vulnerable versions of Java installed in the server, or other cases which propagate zero-day attacks.

Vector 7 Evidency Policy Flaws: These are flaws originating in compliance policies to make evident activities and performance in applications. An example which illustrates this is when a service worker manages to sabotage the assets. Compliance should exist to aid in the forensic analysis, guaranteeing non-repudiation and responsibility. Another angle is when the service in question is not transparent on the privacy policy with user data.

Threat 4 Missing Input Sanitization Management: These are flaws in application input. This threat is divided into 3 vectors, as following:

Vector 8 Script Injection: These are scripts which, when injected, will be executed at the front-end of the application, that is to say, their propagation will be produced at the client-side. A classic example of this is Cross-Site Scripting (XSS).

Vector 9 Data Overflow: This occurs when the attacker inserts a quantity of information which is larger than what is supported in the storage space reserved for the application. A classic example of this is the BoF attack.

Vector 10 Command Injection: These are codes which, when injected, will be executed at the back-end of the application, in other words, their propagation will appear on the server-side. A classic example is SQL Injection (SQLi).

Threat 5 Missing Data Management: These are flaws originating in the exposure and storage of sensitive data. This threat is divided into 3 vectors:

Vector 11 Insecure Data Explosure: These are flaws resulting from the traffic and exhibition of sensitive data in an insecure way. Examples are the absence of HTTPS and Man-in-the-middle attacks (MiTM).

Vector 12 Data Lock-in: These flaws result from the user's data being locked in. It occurs when the owner of the information does not succeed in moving or removing his stored data.

Vector 13 Insecure Data Storage: These failures result from the absence of, or flawed, technique in storing the application's sensitive data. Common cases are also the non-compliance with storage policies in certain data types provided by the application. Other cases involve weak passwords or the persistence of plain-text for sensitive data.

[2]*Cross-Origin Resource Sharing:* http://www.w3.org/TR/cors/

B. Domain 2: Consumer

This domain describes the components or complements with known vulnerabilities installed on the browser. It is important to state that, in this context, a browser can be whatever other consumer type application which has a role on the client-side. Considering that the browser moves sensitive data constantly, this threat, when present, proportions an impact to the assets equivalent to one of the vulnerabilities existing in the service. The domain contains 1 threat and 2 vectors, as follows:

Threat 6 Insecure Consumer Resources: These are vulnerabilities which come from resident materials or from those originating on the client side. This threat is divided into 2 vectors, as follows:

Vector 14 Add-ons with Vulnerabilities: These are flaws where the consumer mechanism acquires add-ons, generally from outside, which propagate breaches during the use of data on the service. Examples are extensions or plug-ins with vulnerabilities installations in the Web browser.

Vector 15 Native Components with Vulnerabilities: These are failures where the consumer mechanism possesses vulnerabilities in its native components, such as the rendering engine, propagating breaches during service consumption.

Domain 3: Social Engineering Social engineering is a wide concept; however, it is important to highlight that in the scope of this research the illicit actions of users with bad intentions are intrinsically related, such as the persuasion of a user towards a certain action which results in the sensitive data breach. In contrast to the rest, these exploitations are not necessarily related to technological aspects, but to human factors, be it the artlessness of the consumer or vulnerable decisions in the policies which are defined by the service. This domain is divided into 2 threats and 6 vectors, as follows

Threat 7 Frauds in Consumer Domain: These are vectors which aim to realize fraudulent actions within the domain of the service. In this context, domain means the Web's own application, the application's server or provider of computing resource. It is important to understand that the method for taking advantage of the service will be realized inside an environment already previously foreseen by the service. Normally these frauds present themselves to the administrators of the service as components with diverse features, aiming to gain confidence in order to receive some privilege type, but with the real intention of stealing or intercepting assets. Another angle of entry is to investigate the consumer or the service in search of extracting some information which helps exploit flaws in the service's security policies. This threat is divided into 2 distinct vectors:

Vector 16 - Consumer with Malwares: These are flaws created by malicious software installed in the browser or on the client's machine.

Vector 17 - Phishing or Scam: These are flaws created by forging genuine services in which the user is lead to believe

that he finds himself within a safe and legitimate environment for passing on information.

Threat 8 Frauds in Service Domain: These are attack vectors whose intention is to realize fraudulent actions within the service domain. In this context, domain means the Web application itself, the application's server or the computational resource provider. It is important to understand that the method of exploitation will be realized within an environment which the service expects to find there. Normally these frauds present themselves to the service administrators as components with diverse functionalities, with the aim of achieving trust to receive some privilege type, but with the real intention of robbing or intercepting assets. Another approach is to investigate the circumstances of the consumer or inside the service, looking to extract some kind of information which helps in the exploitation of flaws in the service's safety policies. This threat is divided into 4 distinct vectors:

Vector 18 - Malicious Third-party Embedded These are flaws caused by external resources of malicious third-parties. Distinct from the vector "Third-Party Embedded with Vulnerability", here the resources are maintained by bad intentions.

Vector 19 Tampered Request: This flaw occurs due to the application of confidence for any consumer request. The attacker makes use of resources such as url, parameters, or screen elements which belong to the session of a particular user. Examples of this are falsifications of HTTP requests or parameter modification. A good technique for prevention is to make requests by the use of a token, besides solid policies controlling the use of HTTP headings. Distinct from the vector "Service Openness Flaws", this falsifies existing requests in the service, characterizing fraud.

Vector 20 Identity Recovery Exploitation: These flaws are caused by a weak policy for user account recovery, where the attacker succeeds in retrieving the credentials of a certain user through social engineering. It is in the interests of the service to adopt a strong policy of communication with its users, aiming to make it clear which data are personal and non-transferable.

Vector 21 Service with Malwares: Distinct from the vector "Third-Party Resources with Vulnerabilities", in this case the service makes use of third-party resources carrying malicious content, be it a service or a library, putting its assets at risk.

IV. Evaluation

As an evaluative measure for our proposal, we decided to use our taxonomy to establish classifications presenting diverse attacks published in the literature. This classification aims to identify similarities between various categories of a particular subject, making a classification by tables in accordance with characteristics and behaviors possible.

As a starting point, it was necessary to realize a survey of catalogues of attacks registered in the literature. Along these lines, top threat type catalogues are generally the most recommended for exposing emerging attacks. The second criterion was to identify the presence of parameters necessary for risk analysis. Considering these criteria, we found two available catalogues in the literature: OWASP Top Ten [13] and the CWE/SANS Top 25 Most Dangerous Software Errors [12]. The results of our evaluation are listed in Figure 4.

The main difference in our taxonomy in relation to the others published in the literature is that it is centered on the attack's propagation vectors. In other words it is not focused on aspects from where the vulnerability originated, but in the method of exploitation. One example is the XSS attack which comes from a flaw in the development of the application, however the way the vulnerability is exploited is done by the vector classified as "Script Injection", acting on the client-side. The SQLi attack, on the other hand, in spite of its similarity, works the server-side, being classified with the "Command Injection" vector. Similarly, BoF also originates from flaws in the input sanitization, but has its vector of propagation directed towards "Abuse of Service Resources".

In the case of Unvalidated Redirect and Forwards (URF), the vector is disseminated by the modification of a legitimate parameter of the application, aiming to redirect the user to another domain, possibly malicious, in this way considered a vector for Phishing or Scam. Even though they share similar characteristics, the Cross-Site Request Forgery (CSRF) is classified in the vector "Tampered Requests". The justification is that, to conclude its execution, the final steps of the URF are necessarily performed in a fraudulent domain. In other words, the victim needs to give continuity to the process without perceiving that he has been led to a hostile environment. Different to this is the CSRF, where everything is realized in the legitimate domain. Bad intentions manipulate the request provoking a behavior which was not expected by the service.

In our taxonomy we present some considerations which, even though they merit some reflection, we believe are based on solid grounding. One example is when we affirm that whatever artifice exploited by social engineering, considering the ecosystem, will result in a fraud. Along similar lines, attacks of the "Insecure Data Storage" or "Insecure Data Exposure" types entail similar violations; however, they are generated in the server-side and client-side contexts respectively. Another approach is when we look for levels of granularity in vector groupings, hoping to separate architectural responsibilities in the design of the service. One example is when we distinguish "Service Openness Flaws" and "Third-Party Resources with Vulnerabilities", in this way separating preoccupations in the distinct decisions between when one is the server or a consumer of resources.

Finally, the similarity between the vectors "Third-Party Embedded with Vulnerabilities" and "Malicious Third-Party Embedded" is notorious. However, in the first case the information provider does not intend there to be vulnerability. In the second case the actor with bad intentions simulates a genuine service in search of practicing illicit acts. It is important to understand that in each situation there exists a very particular means of exploitation, justifying their belonging to different vectors. Our taxonomy does not cover just the sensitive data breach, but also the social impacts, including on businesses.

A. Threats to the Evaluation

Our evaluation proposed a classification of attacks extracted from lists of the top threats genre, giving a reasonable total of classified attacks. In fact, catalogues of this kind

OWASP Top Ten 2013										
Attack	Vector	Co	In	Av	Pr	Nr	An	Ad	Au	Re
A1 Injection	Command Injection	✓	✓	✓	✓	✓	✓	✗	✓	✓
A2 Broken Authentication and Session Management	Missing Authentication	✓	✓	✓	✓	✗	✓	✗	✓	✓
A3 Cross-Site Scripting (XSS)	Script Injection	✓	✓	✓	✓	✗	✗	✗	✗	✓
A4 Insecure Direct Object References	Insecure Data Exposure	✓	✓	✓	✓	✓	✓	✗	✓	✓
A5 Security Misconfiguration	Abuse of Service Resources	✓	✓	✓	✓	✗	✗	✗	✗	✓
A6 Sensitive Data Exposure	Insecure Data Exposure	✓	✓	✓	✓	✗	✗	✗	✗	✓
A7 Missing Function Level Access Control	Missing Authorization	✓	✓	✓	✓	✗	✗	✗	✗	✓
A8 Cross-Site Request Forgery (CSRF)	Tampered Request	✓	✓	✓	✓	✗	✗	✗	✗	✓
A9 Using Components with Known Vulnerabilities	Abuse of Service Resources	✓	✓	✓	✓	✓	✗	✗	✗	✓
A10 Unvalidated Redirects and Forwards	Phishing or Scam	✓	✓	✓	✓	✗	✗	✗	✗	✓

Fig. 4: OWASP Top Ten 2013.

have a strong point in their approach on the most serious emerging attacks of the moment. However, since they take a very objective stance, such catalogues offer an approach which is not so quantitative, which made it difficult to present all possible classification foreseen by our taxonomy. This fact motivated us to look into catalogues with the greatest number of cases in order to evaluate model with greater precision.

V. Related Works

In this section works from the literature which have solutions correlated to the research proposal will be described. One of the motivations for the development of our taxonomy was to counter the lack of solutions for this type of problem by considering the Web as an ecosystem. Besides this, it is planned three criteria for abstraction: (i) Specific Purpose, (ii) Propagation and Similarities and (iii) Impacts on Assets.

For (i) we refer to the proposal of having a specific scenario for action. Some works propose solutions with a more general purpose, for example flaws in operational systems [1]. This type of proposal does have its benefits and often serves as a base for others, for example taxonomies for the UNIX operational system [3], [4] or social engineering threats [9], [10]. However, these approaches place themselves in the conceptual factor of the threat, without identifying intrinsic aspects of a scenario. Because our proposal limits itself to the Web ecosystem, it is possible to establish a relationship between devices, actors and domains involved. This makes it possible to cover threats which go beyond the domain of the application, taking the example of mechanisms of consumption.

In (ii) we describe the power of propagation produced by an attack and the behaviors in common which occur with distinct threats. Some works describe how, when and where a threat is acting [11] or look to investigate insecure implementations basing themselves on standards for encoding failures [2], [15]. However, in spite of covering propagation, these approaches do not relate shared similarities among distinct attacks. The main benefit of this would be to aid in prevention, since it facilitates in the distribution of responsibilities to identify a variety of vectors originating in a single threat, mitigating vulnerabilities through an approach guided by semantics.

Finally, (iii) studies the capacity of the taxonomic model to analyze risks and impacts for assets. Some works propose solutions based on scenarios of software encoding [15] or aspects between computers and the internet [6]. The main difference in our proposal is that it suggests greater granularity in the taxonomic model for classification through the outlining of the paths of 21 vectors. By establishing a semantic grouping our taxonomy proposes more precision over the propagation of exploitation and the identification of attributes whose security is at risk.

VI. Conclusion and Future Works

We believe that the main benefit of our taxonomy would be to offer efficiency in the management of vulnerabilities, since its approach well defines the behaviors of the existing attacks and considers their similarities, being able to minimize incidents provoked by still unregistered attacks. Consequently the taxonomy also offers support in intervention against incidents. And, for establishing a standard, its adoption favors better understanding of domains for service administrators.

We apply our taxonomy in catalogues published in vehicles which adhere to the problem of our proposal in a way that validates it. The objective has been to demonstrate that our taxonomy proposes an approach directed towards the exploration of methods of execution in attacks, establishing rules of classification which reflect the real scenario.

For future works, we aim to develop a threat modeling. This artifact is responsible for specifying good practices during cycles of software development, offering service administrators a process guided towards prevention and countermeasures. Instead of making use of conventional approaches, centered on software be it active or attacking our modeling will be centered on attack vectors, conditioned to the taxonomy.

VII. Acknowledgment

This work was partially supported by the National Institute of Science and Technology for Software Engineering (INES[3]), funded by CNPq and FACEPE, grants 573964/2008-4 and APQ-1037-1.03/08.

[3]http://www.ines.org.br/

REFERENCES

[1] R. P. Abbott, J. S. Chin, J. E. Donnelley, W. L. Konigsford, S. Tokubo, and D. A. Webb. Security analysis and enhancements of computer operating systems. 1976.

[2] G. Alvarez and S. Petrovic. A new taxonomy of web attacks suitable for efficient encoding. *Computers & Security*, 22(5):435–449, 2003.

[3] T. Aslam. A taxonomy of security faults in the unix operating system, 1995.

[4] M. Bishop. A taxonomy of unix system and network vulnerabilities. 1995.

[5] K. M. Eisenhardt. Building theories from case study research. *Academy of Management Review*, 14(4):532–550, 1989.

[6] S. Hansman and R. Hunt. A taxonomy of network and computer attacks. *Computers & Security*, 24(1):31–43, 2005.

[7] J. Hendler and T. Berners-Lee. From the semantic web to social machines: A research challenge for ai on the world wide web. *Artificial Intelligence*, 174(2), 2010.

[8] ISO/IEC. Information security management. Technical report, 2013.

[9] K. Ivaturi and L. Janczewski. A taxonomy for social engineering attacks. *International Conference on Information Resources Management*, 2011.

[10] K. Krombholz, H. Hobel, M. Huber, and E. Weippl. Social engineering attacks on the knowledge worker. In *Proceedings of the 6th International Conference on Security of Information and Networks*, SIN '13, pages 28–35, New York, NY, USA, 2013.

[11] C. E. Landwehr, A. R. Bull, J. P. Mcdermott, William, and S. Choi. A taxonomy of computer program security flaws. *ACM Computing Surveys*, 26:211–254, 1994.

[12] MITRE. Cwe/sans top 25 most dangerous software errors. *Available from: http://cwe.mitre.org/top25/*, 2011.

[13] OWASP. Top ten 2013. *Available from: https://goo.gl/VKz94B*, 2013.

[14] Secunia. Secunia vulnerability review. *Available from: http://goo.gl/ZrdrRh*, 2015.

[15] K. Tsipenyuk, B. Chess, and G. McGraw. Seven pernicious kingdoms: A taxonomy of software security errors. *IEEE Security & Privacy*, 3(6):81–84, 2005.

[16] J. Yu and P. Brune. No security by obscurity - why two factor authentication should be based on an open design. In *Security and Cryptography (SECRYPT), 2011*, 2011.