A Novel Approach for Component based Application Logic Event Attack Modeling

Faisal Nabi[1], Jianming Yong[1], and Xiaohui Tao[2]
(Corresponding author: Faisal Nabi)
School of Management and Enterprise, University of Southern Queensland[1]
West St, Darling Heights QLD 4350, Australia
School of Sciences, University of Southern Queensland, Australia[2]
(Email: u1104061@umail.usq.edu.au)
(Received Aug. 4, 2019; Revised and Accepted Dec. 6, 2019; First Online Feb. 28, 2020)

Abstract

An Event that targets a particular system is required to identify through a novel approach of vulnerability modeling. Current research does not support Event Attack Modeling in component based application logic vulnerabilities. To find such vulnerabilities, it is important to identify the component that triggered the Event to exploit the system. This research proposes the Event Based Attack Modeling, especially in a scenario of component based software subversion logic attack category Business Application Logic. This will help to design and reuse of component from existing application's functional logic.

Keywords: Attack Modeling; CBS reuse; E-Commerce Application; Event Attack Method; Security Modeling

1 Introduction

Event based inter-component applications interact with each other through a passing message inter-communication mechanism [12]. This controlled by a distinct component that is called the event dispatcher, which performs its role as an intermediary between components where condition s are set for the system or application. In this process data communication is called an events that is generated from input communi-cation between components [11]. There are two more type of events, event parameters and event procedures that invoke the individual procedure called the event handlers. In an application, an event attack is occurred when any component of an application is mismatched with its design specification at integration stage. This may result of design fault, because of event-based interruption, which then can create a loophole to exploit the particular system, generated by an attack event during the inter-communication of event parameters [2].

The security vulnerability can arise in the environment that supports the event attack method. The source of the vulnerability can be based on object (component) that is able to generate the event send without any restriction and can be easily crafted into an event sequence for other objects (components) to circumvent the entire logic [9,27].

Event Interception is a phase of condition in which a victim object is identified and intercept the events destined to it. To be able to intercept the event sent to an object permits the attacker to breach the confidentiality of one direction of object (component) communication within the system [1,2,11,27]. In recent years there have been many application attacks based on logical flaws, such as logic flaw or design faults. There is a specific strategy that is required to deal with logical vulnerabilities, such logical attacks are classified as subversion attack. This attack is occurred because of logical flaw in design component based application and its interfaced based integration fault [4,7,25,27].

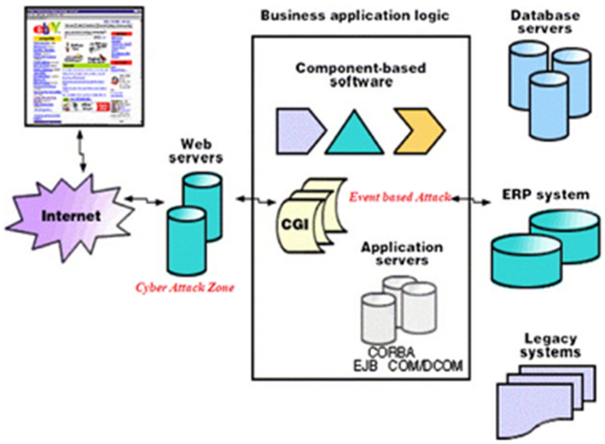

Figure 1: Component based application logic event attack scenario

Therefore, we classify this problem as an Event Attack View. In this case, specification refers to conventional attack, threat, vulnerability. This classifies the attack method, and attack model of identified vulnerability that is known as a subversion attack. In the field of cyber se-

curity Attack Event information is considered as at-tack related data that is derived from various sources. An at-tack event is defined as targeting assets by using attack method, which then exploits the functionality of application business process or circumvents the flow logic. It is very hard to detect the design flaw based vulnerabilities through traditional scanning tools; this is why such vulnerabilities never classified to deal with in terms of the application logic [5, 7].

In this research, we propose Event Based Attack Modeling for design flaw based vulnerability, called Subversion Attack (Component based application logic flaw) by using a Banking case study. The purpose of this re-search is to simplify the process of vulnerability modeling to understand the life cycle of vulnerability. This could help the developers while designing and reusing design specification of business components from existing application components and their underlining application logic. An Event Attack refers to a security problem that exploits the event based inter component communication model [5]. The definition of Event Attack: A malicious component that generates an event of circumvention in order to exploit the target's application logic or functionality. This intercepts communication by forcing the targeted component to send back an inappropriate call or calling away from application functional logic [5].

2 Problem Statement

The focus of this research is to analyze the Event at-tack model and the Subversion attack that falls in the category of business logic vulnerability. Specially considering the security breach scenario real life case study related to Barclay bank, as well as the re-usability design description of component.

The research question, how can Event Attack Modeling simplify the application logic vulnerability, subversion attack? This question is answered by the example of real time case study research method, using Event At-tack Modeling technique.

This real-life case study is a good example of a design flaw in application logic due to the reuse of a component caused component subversion. In this example, the developer reused the same component that was already incorporated in the registration functionality elsewhere within the application, violating the assumptions of the component developer. This mistake lead to the introduction of an application-level flaw that allowed an attacker to access another client's bank accounts. The approach taken to be analyzed, this problem is one that the Event Attack Modeling Technique will be able to helpful to detect design flaws and/or fault free component-based application logic in the middle tier of the n-tier architecture as depicted in Figure 1.

2.1 Research Philosophy

The research philosophy is taken as applied science that is basically an application of existing scientific knowledge to practical applications such as technology, concerning the theory of Event of inter component-communication model. It uses theory, knowledge, method and technique for a particular state of the art [28]. This discussion about Component-based State of the Art in relation to the philosophy of its application & design pattern. The research philosophy also defines and investigates about state of the art technology in Event interaction between the component software de-signs, which is adopted from an applied science philosophy to formulate a solution for business logic vulnerability. In this process, it is very important to understand that design question in the light of research philosophy, can help to conduct the research in the field of Attack Modeling & Security domain by ensuring that research-er's work is going in a right direction and their work is rigorous and insightful.

2.2 Research Gap

In the light of current research and recently studied literature review, [6, 14, 18, 21] and [17] in the domain of cyber and network vulnerability modeling. The research Gap clearly finds an interest to improve the business logic security, specially "Design Flaw" in a service oriented e-commerce applications, that is composed with integrated components. The research gap identified the significance of application logic vulnerability class and category "Subversion attack" cause of Design Flaw, because automated vulnerability analysis and detection tools cannot detect it. This is reason why such vulnerabilities are always oversighted by the application developers. The developers are always keen to reuse existing component core logic from current business logic of the system. This may often cause of mistake while integrating component code solution and designing new functionality.

2.3 Research Design and Method

This research is based on exploratory method where no scientific foundation is available for supporting techniques. The current research and literature review highlights the gap between the current approach and previously designed models or frameworks for logical vulnerabilities. Therefore, we have proposed (Event Attack Modeling) such a technique that could deal with application level logic vulnerabilities. This would help to detect early design faults at the time of integration of components and design fault free new applications. The re-search design also follow previous modeling techniques to justify the newly proposed technique. This simplifies the problem detection process and method.

2.4 Current Approaches in Attack Modeling

There have been several techniques used for vulnerability modeling. These techniques are Attack Graph [26], Attack-Vector [22], Attack-Surface [22], Diamond model [13], OWASP's threat model [13] and Kill Chain [15]. Each technique has its own properties and speciality to identify and model the attack process path way through out the system and network. For example, Attack Graph technique is used for network related vulnerability and system exploitation modeling based on scenario of security issues. Through this technique one can identify the process and pathway of security breach cause within the network as shown in Figure 2.

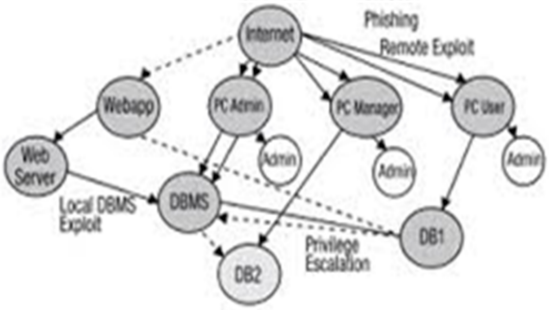

Figure 2: Attack graph with attack path against system

3 Studying Case Profile & Event Attack Modeling

This real life case is a good example of a design flaw in application logic due to the reuse of a component caused component subversion. In this example, the developer reused the same component that was already incorporated in the registration functionality elsewhere within the application, violating the assumptions of the component developer. This mistake leads to the introduction of an application-level flaw that allows an attacker to access an other client's bank accounts (component code Figure 3).

```
class CCustomer
{
 String firstName;
 String lastName;
 CDoB dob;
 CAddress homeAddress;
 long custNumber;
... }
```

Figure 3: *C* customer component code

3.1 Component Application Logic Design Fault

The registration functionality incorporated with the *CCustomer* component that consist of "(use case logic + *Process* and *Entity* Type Logic)" within the application, including core functionally. This process allows the user to authenticate and grant access to the application components such as "My Account component", "View Balance component", "Funds transfers component", "Select Bank Account component, Debit Credit component and other information component. After having authenticated user itself to the application through the registration process, the same Object instantiate and saves in the session key information related to the identity. The components of application within functionally referenced information related to the $^{\star}CCustomer(Component)^{\star}$ object in order to carry out its actions because the $^{\star}CCustomer(Component)^{\star}$ object is candidate component (*Process* and *Entity* Type logic) within the majority of application — for example, account details shown on the main page of the user was generated based on the customer unique number that contained within this component. In the way composition or reuse of the component, code was already used within the application. It clearly shows that the developer assumption leads to a flaw in the reuse of application logic design. This caused the birth of a vulnerability to subversion attack on application business logic. It was a serious mistake and subtle to detect and exploit.

3.1.1 Class of Vulnerability

The "*Subversion Attack*" characterization of vulnerability flaw falls under the application logic, and attack method is to exploit the workflow of business logic, this process subvert business process. At implementation level it is classified as design logic flaw, which then finally characterized as "Subversion of logic" attack.

Subversion of logic. Class: Programme logic flaw;

Server application: (Target agent);

Attack method: (Exploit the work flow);

Subvert application logic: (Attack cause);

Implementation level: (Application design logic flaw classification);

Vulnerability: Subversion of logic.

Therefore, we modeled the Event oriented subversion life cycle that displays the logic diversion of business logic in a small chain of inter-component based communication application model, caused by CBS Flaw.

The above mentioned Figure 4 displays an event attack model scenario, class is subversion attack that falls under business application logic vulnerability, based on component based software that may be flawed in CBS. This fault may have effects on service calls and flow of the function that depends on event based call to other

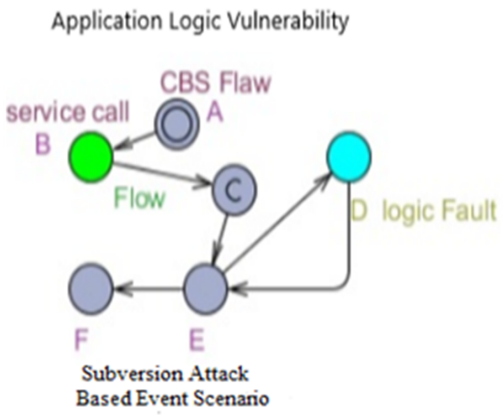

Figure 4: Subversion attack event scenario

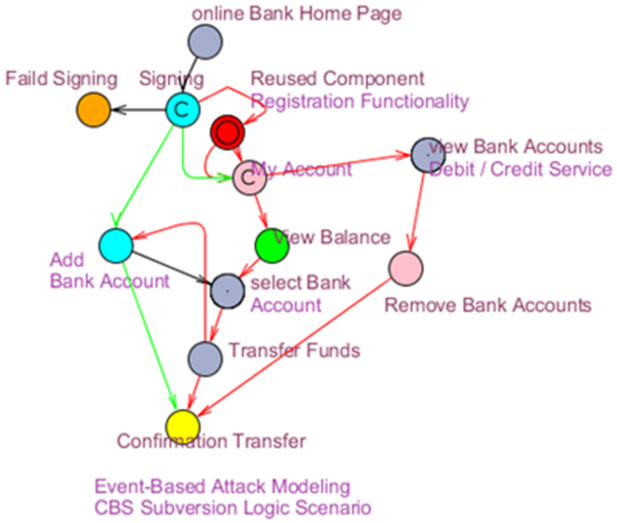

Figure 5: Event attack subversion logic scenario

objects within the system. As it is shown in the above Figure 4, C is condition that must correspond to component D before processing to normal application logic flow to proceed the E. Therefore, D component is a logic fault that does not let the service flow according to normal flow of CBS call service,this is reason why such faults cannot be detected by automated code & system vulnerability scanning tools,and such faults or flaws fall under the classification of logical vulnerability.

3.2 Case Scenario Based Experimental Study

We have further investigated the scenario of this attack keeping in view the above mentioned example related to a security breach of Bank case study. This is caused by a logical design flaw within the system while reusing component from existing application logic. This is called "Subvert Event based Attack" on the banking application. The developers always oversight such attacks on the application's business logic, even though it is a serious vulnerability. It is hard to detect through code scanning and automated detection tools. Therefore, such a technique is required that could simplify the projection of this vulnerability, through the approach of Event based Attack modeling. The proposed technique seems to be a new and effective technique for early detection of such attack at design level of application.

The above-mentioned Figure 5 displays the complete life cycle of the Event Attack Model. In the model C indicates to a condition, If **sign-in**, Pass log in to **My Account** Condition to allow access into the system, **Else** Failed **sign in**. This is the general case of scenario system logic for sign in. However, the major mistake is done by the application developer of the banking system reused same component that was already incorporated in the registration functionality elsewhere within the application. This mistake causing subversion of logic and by pass the condition that is set on **My Account (component)** this violated the assumptions of the component developer and caused the system under attack. This attack also subvert the other components of the application service flow as shown in Figure 5. Any intrusion detection tool cannot detect this sort of attack known as a class of application logic subversion attack. Therefore security scanning automated software, fail to discovery and un-automate this class of vulnerability. The reused component in the application is spotted in **Red Color**, which reflects the service flow diversion and allow an Event to trigger a logical attack by passing session and controls security mechanism of an application related to other service components as displayed in Figure 5. Therefore, above model cycle of an attack is modeled through a Event Attack Modeling technique in scenario of Component-based Software subversion logic Fault.

3.3 Theoretical Analysis of Proposed Approach

In the light of cyber attack theory a successful attack relies on information to be processed by attacker, in case of when an attack is underway and it is measured by modifying as a result related to attack. Therefore, in-formation is a most important element of any cyber at-tack theory [25].

As, it is confirmed that in the theory of cyber attack, first attack is defined and then attacker knowledge related to information parameters and configuration parameters are derived in order to mitigate the system from potential damage [10].

Therefore we formalized the theory of cyber attack into proposed approach event attack modeling.In this pro-cess, first identified the attacker and then measured the attack information parameters, through that an event is occurred as a fault logic, service component triggered to flow diversion and allow an Event trigger by passing session and controls security mechanism. This is demonstrated through scenario based event attack modeling Figure 5 that helped to diagnose the vulnerability life-cycle. This

Figure 6: Cyber attack theory model

gives the knowledge related to information attack parameters, and component configuration parameters that decides the attack vector related to vulnerability of application logic class (subversion logic attack).

Therefore, it is concluded that above mentioned technique is very useful for attack modeling in the light of cyber attack theory.

3.4 Systematical Comparison of the Proposed Scheme

The current approaches of attack modeling are based on attack graph and vector modeling techniques [22, 26], these techniques models focus on the network or system vulnerability based modeling that deals with the different attacks targeting the network [10], but the lack of software application scenario based modeling.In this, scenario an approach is immanent for application based vulnerability modeling technique. Therfore, the proposed scheme is presented, event based attack modeling that targets the service component triggered to flow diversion of application logic in component-based system. The proposed scheme is comparably sounder as compare to any other modeling technique for software based application and its core logic flow.

3.5 Discussion

We have seen that the proposed technique is very helpful in detecting the event that triggered the subversion attack within the application and its component at the integration level, which clearly depicts the vulnerability and its effects on other components of the application and underling business logic. We also have evaluated the other techniques such as Attack Graph and Attack Vector. The Attack Graph is use to identify the vulnerability in the networks and system, and Attack Vector can provide the path way projection through hacker exploitation attempt which targets the network servers by payload or malicious input. It is also modeled through Attack Vector Modeling technique. It has been noticed that none of these techniques meet the requirement of logical attack modeling and simulation [18].

Where as proposed technique is useful to model the case scenario of banking application through Event Based Attack modeling. That is spotted in red color the component with fault service flow, calling C condition **My Account** component within the application that cause exploitation.

4 Related Work

There are numbers of approaches target the security in event based inter-component applications [3, 19, 23, 24]. For example, Simeon *et al.* [27] took into account the security vulnerabilities in event-based applications and systems, explained the conditions that can be made of them,in result of inter-communication fault. In simple term, current security solutions more rely on encryption, static code analysis, and runtime ACL techniques. Whereas, on the other hand, there have been many techniques adopted to attack modeling such as the Diamond Model [13], Attack Tree [20], Attack Vector [22], Attack Surface [16], Kill Chain [15] and Attack Graph [26]. However, all of these techniques fail to ad-dress the logical vulnerabilities detection or modeling framework, because these techniques are network vulnerability modeling and address the network security issues related to the system. Therefore, such a technique needs to introduce that can deal with missing gap between application and system level vulnerability modeling. This will fill the research gap related to logical vulnerabilities in application logic (Component-based Software) [8].

5 Conclusions

Attack modeling is a most useful technique in ana-lysing the attacks and early mitigation of the problem. This is why many techniques are introduced to deal with the attack modeling in the system network domain. The logical vulnerabilities are flaw in design or fault in logic. It is hard to detect and modeled. Therefore such a technique is required that could deal with the logical flaw based vulnerability. In this paper, we have introduced a novel approach of modeling called "Event Attack Modeling" that used Uppaal Tool to model the vulnerability and its attack flow through attack-triggered component within the application in real time scenario. This will help the developers design their application free from logical flaws and design faults, while reusing design specification of component from existing application.

References

[1] A. A. Al-khatib, W. A. Hammood, "Mobile malware and defending systems: Comparison study," *International Journal of Electronics and Information Engineering*, vol. 6, no. 2, pp. 116–123, 2017.

[2] H. Al-Mohannadi, Q. Mirza, A. Namanya, I. Awan, A. Cullen, J. Disso, "Cyber-attack modeling analysis techniques: An overview," *The 4th International*

Conference on Future Internet of Things and Cloud Workshops, 2016. DOI: 10.1109/W-FiCloud.2016.29.

[3] L. Aniello, R. Baldoni, C. Ciccotelli, G. A. D. Luna, F. Frontali, and L. Querzoni, "The overlay scan attack: Inferring topologies of distributed pub/sub systems through broker saturation," in *Proceedings of the 8th ACM International Conference on Distributed Event-Based Systems (DEBS'14)*, pp. 107—117, 2014.

[4] A. Anurag, "Network neutrality: Developing business model and evidence based net neutrality regulation," *International Journal of Electronics and Information Engineering*, vol. 3, no. 1, pp. 1–9, 2015.

[5] Bank of England, "An introduction to cyber threat modelling", Industry report, Bank of England Publication, 2016. (`https://www.cyentia.com/library-item/an-introduction-to-cyber-threat-modelling/`)

[6] M. Bentounsi, S. Benbernou, M. J. Atallah, "Security-aware business process as a service by hiding provenance," *Computer Standards & Interfaces*, vol. 44, pp. 220–233, 2016.

[7] BSIMM, "Attack models with bsimm frameworks," 2016. (`https://www.bsimm.com/framework/intelligence/attack-models/`)

[8] E. M. Hutchins, M. J. Cloppert, and R. M. Amin, "Intelligence-driven computer network defense informed by analysis of adversary campaigns and intrusion kill chains," *Leading Issues in Information Warfare & Security Research*, vol. 1, pp. 80, 2011.

[9] S. Islam, H. Ali, A. Habib, N. Nobi, M. Alam, and D. Hossain, "Threat minimization by design and deployment of secured networking model," *International Journal of Electronics and Information Engineering*, vol. 8, no. 2, pp. 135–144, 2018.

[10] S. Jajodia and S. Noel, *Advanced Cyber Attack Modeling, Analysis, and Visualization*, George Mason University, Mar. 2010. (`https://csis.gmu.edu/noel/pubs/2009_AFRL.pdf`)

[11] N. J. Kim, M. S. Gong, G. S. Lee, "An attack-target-method schema for cyber attack event database," *IEEE International Conference on Electronic Information and Communication Technology (ICEICT'16)*, 2016. DOI: 10.1109/ICEICT.2016.7879705.

[12] Y. K. Lee, D. Nam, N. Medvidovic, *Identifying Inter-Component Communication Vulnerabilities in Event-based Systems*, Technical Report: USC-CSSE-17-801, 2016.

[13] X. Lin, P. Zavarsky, R. Ruhl, and D. Lindskog, "Threat modeling for CSRF attacks," in *IEEE 16th International Conference on Computational Science and Engineering*, vol. 3, pp. 486-–491, 2009.

[14] A. K. Luhach, S. K. Dwivedi, C. K. Jha, "Designing and implementing the logical security framework for e-commerce based on service oriented architecture," *International Journal on Soft Computing (IJSC'14)*, vol. 5, no. 2, 2014.

[15] P. K. Manadhata, J. M. Wing, "An attack surface metric," *IEEE Transactions on Software Engineering*, vol. 37, no. 3, pp. 371-–386, 2011.

[16] M. Mulazzani, S. Schrittwieser, M. Leithner, M. Huber, and E. Weippl, "Dark clouds on the horizon: Using cloud storage as attack vector and online slack space," in *USENIX Security Symposium*, pp. 65-–76, 2011.

[17] F. Nabi, "Designing a framework method for secure business application logic integrity in e-commerce systems," *International Journal of Network Security*, vol. 12, no. 1, pp. 29-–41, Jan. 2011

[18] F. Nabi and M. M. Nabi, "A process of security assurance properties unification for application logic," *International Journal of Electronics and Information Engineering*, vol. 6, no. 1, pp. 40–48, 2017.

[19] F. Petroni, L. Querzoni, R. Beraldi, and M. Paolucci, "Exploiting user feedback for online filtering in event-based systems," in *Proceedings of the 31st Annual ACM Symposium on Applied Computing (SAC'16)*, pp. 2021-–2026, 2016.

[20] C. Phillips and L. P. Swiler, "A graph-based system for network-vulnerability analysis," in *Proceedings of the 1998 Workshop on New Security Paradigms*, pp. 71-–79, 1998.

[21] L. Seinturier, P. Merle, R. Rouvoy, D. Romero, V. Schiavoni, and J. B. Stefani, "A componentbased middleware platform for reconfigurable service-oriented architectures," *Software Practice and Experience*, vol. 42, no. 5, pp. 559—583, 2017.

[22] B. Schneier, "Attack trees," *Dr. Dobb's Journal*, vol. 24, no. 12, pp. 21—29, 1999.

[23] B. Shand, P. Pietzuch, I. Papagiannis, K. Moody, M. Migliavacca, D. Eyers, and J. Bacon, "Security policy and information sharing in distributed event-based systems," *Reasoning in Event-Based Distributed Systems*, pp. 151—172, 2011.

[24] M. Srivatsa, L. Liu, and A. Iyengar, "Event-guard: A system architecture for securing publish-subscribe networks," *ACM Transactions on Computer Systems (TOCS'11)*, vol. 29, no. 4, pp. 10:1—10:40, Dec. 2011.

[25] A. Tayal, N. Mishra and S. Sharma, "Active monitoring & postmortem forensic analysis of network threats: A survey," *International Journal of Electronics and Information Engineering*, vol. 6, no. 1, pp. 49–59, 2017.

[26] United States. Joint Chiefs of Staff, *Joint Tactics, Techniques, and Procedures for Joint Intelligence Preparation of the Battlespace*, 2000. (`http://purl.access.gpo.gov/GPO/LPS49610`)

[27] S. (simos) Xenitellis, "Security vulnerabilities in event driven systems," in *Proceedings, Security in the Information Society: Visions and Perspectives*, pp. 147–160, 2001.

[28] A. Yaghmaie, "How to characterise pure and applied science," *International Studies in the Philoso-Phy of Science*, vol. 31, no. 2, pp. 133–149, 2017.

Biography

Faisal Nabi is a PhD researcher at University of Southern Queensland.He has also received Honorary PhD in Computer Science from Brock University St. Catharines, Ontario, Canada. Faisal's research interests are e-commerce security and software security.

Jianming Yong is Professor of school of information systems.He has received his PhD from SwinburneUT.He is also member of IEEE professional.His research areas are Cloud Computing, Big Data Security and Privacy, Data Integration, Workflow systems, Information system security, Network management, Web service for SMEs, Digital Identity Management.

Xiaohui Tao is Associate Professor in School of Sciences, University of Southern Queensland, Australia. His research interests include Natural Language Processing, Text Mining, Knowledge Engineering, and Health Informatics. During his research career, Tao has gained a wealth of knowledge and experience in dealing with massive datasets and delivering solution to complex research problems, and made many contributions to Ontology Learning, Web Intelligence, Data Mining, and Information Retrieval. His research results have been published in 90+ refereed papers, many of them are on highly ranked journals such as IEEE TKDE, KBS, PRL and conferences such as ICDE, PAKDD and CIKM. He has been a Program Chair of many International Conferences and Workshops.