A Quantitative Methodology for Security Risk Assessment of Enterprise Business Processes

Jaya Bhattacharjee, Anirban Sengupta and Chandan Mazumdar
Centre for Distributed Computing, Jadavpur University, Jadavpur, Kolkata, India

Keywords: Business Process, Risk Assessment, Task, Threat, Vulnerability.

Abstract: Business processes help to realize the business objectives of an enterprise. Security breach of business processes may lead to un-fulfillment of objectives, loss of revenue, and possible shutdown of the corresponding business venture. Hence, it is important to ensure that the security properties of critical business processes are protected from attacks and failures. Effective protection mechanisms can be designed only after identifying security risks to business processes. However, existing methodologies mostly focus on the detection of risks to individual hardware, software, network and information assets. They do not cater to risks that are specific to business processes. This paper attempts to address this gap in research by describing a technique for identifying the components of a business process and quantitatively assessing their security risks.

1 INTRODUCTION

An enterprise is usually defined as an organization that has been created to execute one or more business ventures (ed. Soanes and Stevenson, 2011). Such ventures are realized with the help of certain types of activities that are referred to as business processes (ISO/IEC, 2011). Business processes are responsible for building / generating products of an enterprise, as well as for delivering services to clients. Examples include processes for building a car, developing software, or providing online money transfer services to a bank's customers. Thus, the very existence of an enterprise depends on the proper functioning of its business processes. Security breach of business processes would lead to un-fulfilment of an enterprise's security objectives, loss of revenue, and possibly shutdown of the corresponding business venture.

Hence, it is extremely important to ensure that the security properties of critical business processes are protected from malicious attacks and accidental failures. However, effective protection mechanisms cannot be designed without identifying the specific security risks to business processes. Historically, there has been lots of research into methodologies for identifying risks to hardware, software, network and information assets of an enterprise (Peltier, 2010; Bhattacharjee et al., 2013). These methodologies are successful in evaluating the risks to assets that are required for executing business processes.

However, there is another class of security risks that exist owing to vulnerabilities within the basic structure of business processes, and are independent of the underlying assets of an enterprise. This includes both internal as well as external risks. For example, lack of training on use of privileged access rights (internal risk) could lead to users divulging their credentials, thus leading to serious security breaches. Again, high attrition of personnel owing to better opportunities elsewhere (external risk) could seriously jeopardize enterprise operations. Barring a few attempts (ENISA, 2015; MEHARI, 2010), there has been hardly any significant research for formulating techniques for the identification of such business process-specific risks. This leads to incomplete design of protection mechanisms, thus exposing an enterprise to various security breaches. This paper attempts to fill this research gap by proposing a quantitative methodology for the assessment of security risks (both internal and external) to enterprise business processes.

Rest of this paper is organized as follows. Section 2 presents a survey of related work. Section 3 details the components of a business process and their inherent relationships. Section 4 describes the

Bhattacharjee, J., Sengupta, A. and Mazumdar, C.
A Quantitative Methodology for Security Risk Assessment of Enterprise Business Processes.
DOI: 10.5220/0005739703880399
In *Proceedings of the 2nd International Conference on Information Systems Security and Privacy (ICISSP 2016)*, pages 388-399
ISBN: 978-989-758-167-0

vulnerabilities in, and threats to, business processes, and proposes a quantitative methodology for security risk assessment. While Section 5 discusses the benefits of the proposed methodology, an illustrative case study is included in Section 6. Finally, Section 7 concludes the paper. It may be noted that some of the terms and definitions used in Section 4 have been adopted from Bhattacharjee et al., (2013); Bhattacharjee et al., (2014).

2 RELATED WORK

Several researchers have discussed the security issues of enterprise business processes. Some of them have also proposed techniques for modelling risk-aware business processes. We discuss some of the significant contributions in these areas.

Marchesini and Viganò (2011) discussed an approach for the formal analyses of business processes that need to comply with security requirements like authorization constraints, or separation or binding of duties. They observed that a business process has two levels: the workflow level dealing with the control of the flow (and the manipulation of data) and the policy management level describing access rules and permissions. They introduced a notion of knowledge hierarchy within the entities of a business process that is involved in the interaction among workflow and policy management levels. An entity's state of knowledge represents the entity's view of the business process. The authors have attempted to include information about sets of security-critical tasks at different levels of hierarchy that can be used to control the process execution and enforce security properties.

Armando and Ponta (2011) discussed about authorization requirements of security-sensitive business processes. In a business process, agents can be dynamically delegated to perform tasks they were not initially authorized to execute. Considering this, they proposed a new approach for the specification and automatic analysis of security-sensitive business processes. They have used model checking to analyze the specification of the workflow and of the associated security policies separately.

Both of the above methods address the authorization aspects of business processes. They have not considered other important security issues like confidentiality, integrity and availability requirements of processes.

Lowis and Accorsi (2011) proposed a method to search and analyze the vulnerabilities of SOA-based business processes and services. They have proposed six attack effects for business processes corresponding to confidentiality, integrity and availability parameters: start, stop, steer, split, spot and study. An attacker can start or stop a process and may harm availability. He can steer or split a process and can harm integrity. Finally, ability of an attacker to spot or study a process can harm its confidentiality. Though the method analyzes vulnerabilities within business processes, it does not explicitly address threats or compute risk values.

Tjoa et al., (2011) proposed a formal model that considers relations between threats, detection mechanisms, safeguards, recovery measures and their effects on business processes. Business process is represented by a set of resources, activities and their attributes. Then threats to the attributes of different elements of business process are identified and their preventive, blocking and reactive measures are stated by the model.

Khanmohammadi and Houmb (2010) proposed a business process based risk assessment methodology and focused on business goals rather than assets. Business and their control processes are identified during the initial phase. Then vulnerabilities within these processes are identified and threats to those vulnerabilities are analyzed. Finally, risk is computed considering the degree of exposure of vulnerabilities, effects of installed security controls, threat levels and process value.

Jakoubi et al., (2010) presented a technique for risk-aware business process management. It consists of five distinct phases: Perform Program Management, Determine As-Is Situation, Re-engineer Processes, Implement Processes and Review and Evaluate. However, the methodology is mostly verbose and does not suggest any quantitative or formal technique for the computation of risks to business processes.

The above discussion shows that though some techniques for analyses of risks to business processes have been presented, most of them are verbose, qualitative approaches. They do not strive to model such risks quantitatively. Besides, there is also a lack of understanding of the internal structure of a business process that is so essential for developing a quantitative approach. Though BPMN (Business Process Model and Notation) provides graphical notations for enabling enterprises to model their business processes (OMG, 2011), there has been limited adoption of the approach in case of information security. The research presented in this paper attempts to fill these gaps by describing a technique for identifying the components of a business process and computing values of their security risks.

3 BUSINESS PROCESS

Business process, as stated above, refers to the primary activities that are essential for achieving the business objectives of an enterprise. It consists of a set of tasks (or activities) that can be arranged in a linear sequence, in parallel, as a conditional structure, or repetitive (loop) structure. These are depicted in Figures 1(a), 1(b), 1(c) and 1(d), respectively, where T_i denote tasks and C_j denote specific conditions. In addition to depicting simple if…else kind of instances, conditional task structures may also be used to represent situations where alternate paths exist. For example, a bank may offer cash withdrawal facilities via multiple means: through cheques, withdrawal forms, or ATM. The underlying procedure for each of them may be similar, though they differ in their actual implementations. Such a situation illustrates alternate task structures, where the actual path traversed would depend on the customer. On the other hand, parallel task structures are used in cases where multiple tasks can be executed simultaneously. This can occur when such tasks (or their predecessors) have the same *ancestor*, and they (or their successors) have the same *descendant*. In Figure 1(b), T_9 and T_{10} are parallel tasks; both of them have task T_8 as their ancestor, and task T_{11} as their descendant.

To illustrate the concept of business processes and tasks lucidly, let us consider the loan management process of a bank (Figure 2). This process contains different types of task structures (sequential, parallel, conditional and repetitive) as stated above. A task can be as simple as writing a standard, pre-defined report or it may consist of complex functions as illustrated by the "loan processing" task in Figure 2. Thus, a task can be viewed as comprising of a set of functions or sub-routines, all of which must be carried out to complete the corresponding task. The structure of an individual task is shown in Figure 3. In order to execute the functions of a task, three types of conditions may need to be fulfilled:

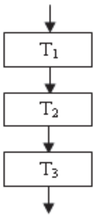

Figure 1(a): Sequential tasks.

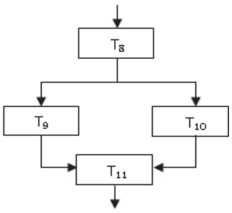

Figure 1(b): Parallel tasks.

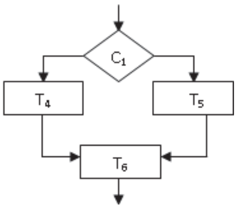

Figure 1(c): Conditional tasks.

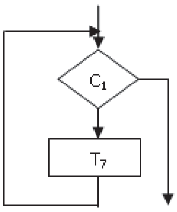

Figure 1(d): Loop.

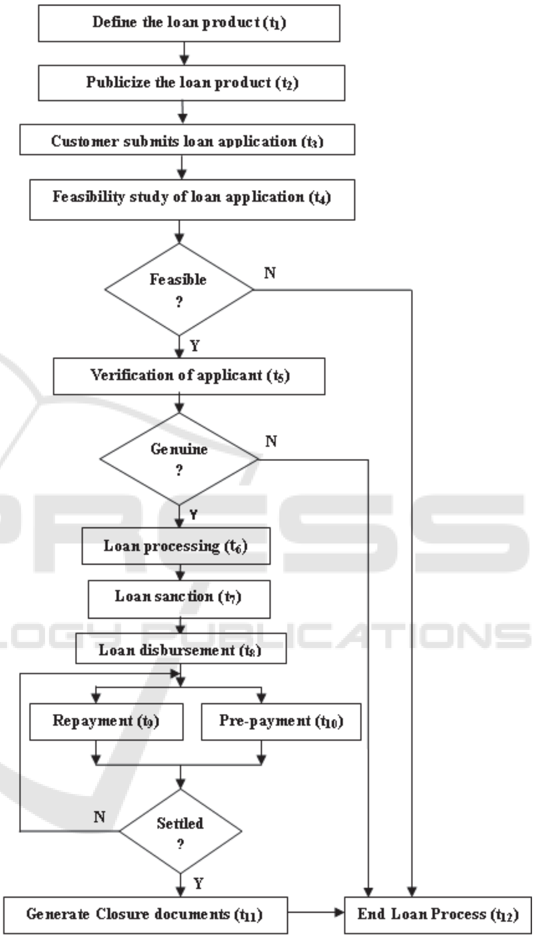
Define the loan product (t1)
Publicize the loan product (t2)
Customer submits loan application (t3)
Feasibility study of loan application (t4)
Feasible ?
N
Y
Verification of applicant (t5)
Genuine ?
N
Y
Loan processing (t6)
Loan sanction (t7)
Loan disbursement (t8)
Repayment (t9)
Pre-payment (t10)
N
Settled ?
Y
Generate Closure documents (t11)
End Loan Process (t12)

Figure 2: Loan management process.

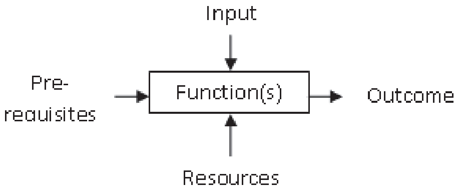

Figure 3: Structure of a task.

1) ***Completion of Pre-requisites.*** Some tasks may need to be triggered by certain other tasks / business processes. Similarly, in a sequential task

structure, a task, say t_i, can commence only after task t_{i-1} has been completed. These instances illustrate the fact that there might be some pre-requisites (usually other tasks, as shown here) that need to be fulfilled before a specific task can be executed.

2) ***Availability of Resources.*** One or more resource(s) may be required to carry out a task. Resources can be of several types: infrastructure (hardware, software, physical space or site, network components, etc.), money, time, agent (human or software agent who will execute the task), and competence. Competence can be viewed as a function of knowledge, skills, and attitude. An agent must possess the requisite competence to be able to carry out the task with the help of available infrastructure. Money may be needed to acquire the necessary infrastructure or employ agents. Certain amount of time is usually required to carry out a task. This may include the time needed to prepare the infrastructure and acquire relevant competence, besides the time spent in actual execution of the task.

3) ***Availability of Input.*** A task may need one or more input(s) to execute its functions. Such input may either be received from the environment, or produced as output by another task that has already been completed. Generally, inputs can be of two types: data items and control items. Since, control items have already been taken care of by pre-requisites (triggers or natural control flow owing to completion of the previous task), only data items are considered as input elements in the proposed task structure.
   It may be noted that it is not necessary that all of the three types of conditions described above are required by each and every task. A particular task might not need some (or any) of these conditions. The outcome of a task might be tangible or intangible. Different types of outcome are as follows:

1) ***Data Item (Output).*** A task might produce one or more data items (report, list, etc.) as output. These data items may either serve as *input* to another task, or may be used by an enterprise for other purposes. For example, there may be a task to generate the balance-sheet of an enterprise that is subsequently published for public viewing.

2) ***Trigger.*** As stated above, the completion of a task might trigger the commencement of another. Thus, an outcome of a task may simply be a signal for another task / process to begin execution.

3) ***Acknowledgement.*** Another type of outcome may be an acknowledgement that the task has completed its execution. This may be transmitted to the process / task that had initiated the task in question, based on which further actions may be performed.

Thus, task t_i can be represented as

$$t_i = \{inp, pr, res, fn, out\} \quad (1)$$

where, *inp* denotes a set of inputs, *pr* denotes pre-requisite set, *res* denotes resource set, *fn* denotes a set of functions and *out* denotes the outcome set. This implies that, in essence, task comprises of a set of functions that *transform* inputs to corresponding outcomes, with the help of resources and pre-requisites.

Business process, BP_l, can be represented as

$$BP_l = \{t_i, \rightarrow \mid i > 0\} \quad (2)$$

where, t_i denotes a set of tasks (as defined above) and → denotes a flow relation (Sun et al., 2006; Denning, 1976) between the tasks. The flow relation signifies the order of execution of tasks within a business process, and can be represented as $t_i \rightarrow t_j$, which means that task t_j should follow task t_i. As stated above, the flow relation between two tasks can be either one of the following: sequential, iterative, conditional or parallel. It is absolutely essential for the tasks to follow the pre-defined flow relation, else the outcomes of the tasks and hence, of the business process, may either be incorrectly generated, or not generated at all. For example, Figure 2 shows the following flow relation between tasks: "Verification of applicant" → "Loan processing" → "Loan sanction" → "Loan disbursement". Now, if it so happens that the flow relation is erroneously altered and the following sequence of tasks is executed: "Loan processing" → "Loan sanction" → "Loan disbursement" → "Verification of applicant". This could lead to disastrous consequences where the loan might get disbursed to an ineligible, even fraudulent, applicant.

Business processes are usually perceived as the most valuable primary assets of an enterprise (ISO/IEC, 2011). If a business process gets adversely affected, the very existence of the enterprise might be at stake. Hence, it is important to derive a methodology for computing values of the components of business processes and identifying probable risks, so that appropriate security measures can be deployed to protect them.

The technique for deriving values of tasks and assessing risks to business processes is described in the following section.

4 RISKS TO BUSINESS PROCESS

The risk assessment methodology begins by computing security values of the component tasks of business processes.

4.1 Valuation of Tasks

The proposed valuation technique hinges on identifying the confidentiality, integrity and availability requirements of a task. Confidentiality (C) of an entity refers to the need for protecting its secrecy. In case of a task, confidentiality may mean that unauthorized entities should not gain access to the elements of a task. It could also mean that unauthorized entities should not even be aware of the existence of a task. In technical terms, the former condition is referred to as “the ability to study” the task, while the latter is known as “the ability to spot” the task (Lowis and Accorsi, 2011). Enterprises like defense and space research organizations, which have very strict requirements for *task confidentiality*, need to implement controls wherein unauthorized users are not able to detect even the existence of the task. It may be possible for an entity to “study” the following elements of a task:

- Input
- Pre-requisites
- Resources
- Functions
- Outcome

Table 1: Assigning confidentiality values of tasks.

C-value	Interpretation
5	Only authorized entities can spot the task
4	Unauthorized entities can spot the task, but may not study any of its elements (input, pre-requisites, resources, functions, outcome)
3	Unauthorized entities may study the outcome of a task
2	Unauthorized entities may study all task elements except its functions
1	Unauthorized entities may study all task elements

The above elements may be used to define a graded system for assigning confidentiality values of tasks. Table 1 shows the proposed C-value assignment scheme. The following assumptions have been made:

- If an entity is not allowed to “spot” a task, it implies that the entity will also not be able to “study” it;
- “Study” of the outcome of a task is considered less critical than “study” of any other element of the task; and
- “Study” of the functions of a task is considered to be the most critical activity.

Integrity (I) requirement of a task refers to the need for protecting its accuracy and completeness. A task should exist in its entirety and all of its component elements should be accurate. Specifically, this implies that all inputs, pre-requisites, resources and functions of the task are present and are fulfilling their objectives correctly; similarly, all outcomes of the task are being generated correctly as per requirements. Table 2 shows the proposed I-value assignment scheme.

Table 2: Assigning integrity values of tasks.

I-value	Interpretation
5	All elements of the task (input, pre-requisites, resources, functions, outcome) are *always* accurate and complete
4	All elements of the task remain accurate and complete *whenever needed* for execution
3	Whenever the task is executed, its outcome is obtained *accurately* (but may not be complete)
2	The outcome of the task is obtained accurately *majority* of the time (during a period of measurement)
1	Integrity requirement of the task is *negligible*

Availability (A) of a task means that the task should remain accessible and usable. The former implies that authorized entities are able to access the task without hindrance (e.g. there is no “denial of service”), while the latter means that the task can be used to fulfill its intended objective. Table 3 shows the proposed A-value assignment scheme.

It is important to note that value adjustment due to inter-task dependencies is not carried out explicitly in the proposed technique. This is owing to the fact that such adjustment has already been taken care of implicitly during computation of values of individual tasks. For example, C-value considers all elements of a task that includes its pre-requisites and inputs. A task, say t_j, can be dependent on another task, say t_i, in one or more of the following ways:

- the outcome of t_i may serve as input to t_j;
- the completion of t_i may serve as a pre-requisite for the initiation of t_j.

As is obvious, both of the above have been considered during the valuation of task t_j.

After computation of the security values of tasks, vulnerabilities within the tasks, and their corresponding threats, need to be identified.

Table 3: Assigning availability values of tasks.

A-value	Interpretation
5	All required elements of the task (input, pre-requisites, resources, functions, or outcome) are *always* accessible and usable
4	All required elements of the task are *always accessible*, and remain *usable whenever needed* for execution
3	All required elements of the task remain accessible and usable *whenever needed* for execution
2	All required elements of the task remain accessible and usable *majority* of the time (during a period of measurement)
1	Availability requirement of the task is *negligible*

4.2 Vulnerabilities

Vulnerability is defined as an inherent weakness in an entity that can be exploited by threat(s) to breach security of that entity (ISO/IEC, 2014). Improper configuration of tasks and lack of checks and controls may expose the different task elements to security risks. Following are some examples of vulnerabilities within tasks:

- Non-availability of manpower may hamper task execution;
- Lack of training of concerned personnel may lead to incorrect execution of tasks;
- Improperly configured infrastructure may hinder task execution, or lead to unauthorized access and attacks;
- Lack of input validation may result in improper task execution, or generation of erroneous outcome;
- Lack of monitoring and management of pre-requisites may lead to non-initiation of a task;
- Improperly configured functions may generate erroneous outcome, or may not produce any outcome;
- Lack of output validation may cause leakage of confidential information to unauthorized entities.

Based on the component elements of a task, vulnerabilities may be categorized as follows:

- Input vulnerabilities;
- Vulnerabilities owing to pre-requisites;
- Resource vulnerabilities;
- Function vulnerabilities; and
- Output vulnerabilities.

It is important to devise means for identifying vulnerabilities in tasks and analyzing their criticalities. All identified vulnerabilities may not be equally critical. The gravity of vulnerabilities may be recognized from the values of two vital attributes, namely severity and exploitability. These are described in the following sub-sections.

4.2.1 Severity

Severity (Sev) of vulnerability indicates how bad the vulnerability is (Bhattacharjee et al., 2013). It is determined by the amount and type of impact that can occur if the vulnerability is successfully exploited by corresponding threat(s). Types of impact may be broadly classified as loss of confidentiality, integrity and/or availability of tasks. Breach of these security parameters may be caused by unauthorized entities when they gain illegal access as follows:

- Confidentiality of a task may be breached if unauthorized entities can “spot” or “study” the elements of the task;
- Integrity may be breached if entities can illegally “alter” or “obliterate” the elements of the task;
- Availability may be breached if entities can illegally “alter” the task elements or make them “inaccessible”.

It is important to note that vulnerabilities within the elements of a task might also jeopardize the security of other, dependent tasks. For example, consider the case where the outcome of a task, say t_i, is later used as an input to another task, say t_j. A vulnerability in t_i is exploited by a threat leading to illegal alteration of its outcome, and hence causing breach of its integrity. Moreover, this also impacts the input to t_j, which may render it useless, leading to loss of availability of t_j.

In the proposed methodology, severity is computed on a 5-point scale, as shown in Table 4. The assumption stated in Section 4 holds here as well; that is, if an entity can “study” a task, it implies that the entity will also be able to “spot” it.

Table 4: Severity values of vulnerabilities.

Severity	Interpretation
Very High (5)	Vulnerability in task allows unauthorized entities to study and/or alter/obliterate element(s) of multiple tasks and/or make them inaccessible.
High (4)	Vulnerability in task allows unauthorized entities to study and/or alter/obliterate element(s) of the task and/or make them inaccessible.
Medium (3)	Vulnerability in task allows unauthorized entities to spot element(s) of multiple tasks.
Low (2)	Vulnerability in task allows unauthorized entities to spot element(s) of the task.
Very Low (1)	Vulnerability in task does not allow any significant access to unauthorized entities.

4.2.2 Exploitability

Exploitability is another important attribute of vulnerability that denotes the ease with which the vulnerability can be exploited by threat(s) (Bhattacharjee et al., 2013). It can be determined with the help of 3 factors as follows:

1) ***Access Vector (AR).*** This indicates the type of access that is required in order to exploit the vulnerability of a task (say t_j). It can assume values on a 3-point scale: (i) if the vulnerability can be exploited only from within the task t_j, AR = 1; (ii) if the vulnerability can be exploited from a *neighbouring task*, that is a task (say t_i) which has a flow relation with task t_j (that is $t_i \rightarrow t_j$), AR = 2; (iii) if the vulnerability can be exploited from a *non-neighbouring task*, AR = 3.
2) ***Attack Complexity (AC).*** This factor signifies the amount of difficulty that needs to be encountered for exploiting vulnerabilities. Higher is the complexity, lower is the corresponding exploitability of vulnerability. Different resources may be required for exploiting vulnerabilities. Types of resources are: (i) financial resource; (ii) manpower; (iii) knowledge or expertise; (iv) tools, techniques, and infrastructure; and (v) time. The following scheme is proposed for computing AC: (i) if at least four of the five resources listed above are required for launching an attack, then AC = 3; (ii) if two or three of the resources listed above are required for launching an attack, then AC = 2; (iii) for all other cases, AC = 1.
3) ***Authentication Level (AL).*** This denotes whether (and how many) authentication is needed for gaining access to the task element that contains the vulnerability. It can assume values on a 3-point scale: (i) if multiple instances of authentication are required before being granted access to the task element, then AL = 1; (ii) if a single instance of authentication is needed, then AL = 2; (iii) if no authentication is needed, then AL = 3.

It is obvious from the above discussion that exploitability of vulnerability is directly proportional to access vector and authentication level, and inversely proportional to attack complexity. It can be computed as:

$$\mathrm{Exp}(v) = \mathrm{ceil}((AR * AL) / (3 * AC)) \qquad (3)$$

Here, Exp(v) denotes exploitability of vulnerability v and ceil denotes ceiling function. While, the denominator AC is multiplied by 3 to scale down the value of Exp(v), use of the ceiling function ensures that the value never reaches 0. Intuitively, if a vulnerability exists, it is definitely exploitable (though, the ease of exploitability may vary from one vulnerability to another); the ceiling function reflects this notion.

Thus, $\mathrm{Exp}(v)_{(max)} = \mathrm{ceil}((3 * 3) / (3 * 1)) = 3$

and $\mathrm{Exp}(v)_{(min)} = \mathrm{ceil}((1 * 1) / (3 * 3)) = 1$

Hence,

$$\mathrm{Exp}(v)\ \varepsilon\ \{1, 2, 3\} \qquad (4)$$

4.3 Threats

Threat (ISO/IEC, 2014) is defined as the potential cause of an unwanted incident, which may result in harm to a task (and hence, business process). Threats exploit vulnerabilities to cause impairment to elements of a task. Historically, threats have been categorized based on the type of entity that initiates the corresponding incident, namely nature (e.g. earthquake, lightning, etc.), environment (e.g. forest fire, corrosion, etc.), and human beings (e.g. malware, information theft, etc.). Such entities are referred to as primary threat agents; they may, or may not, need other (secondary) agents to inflict harm.

After identifying a particular threat, it is important to compute the probability that the threat will actually give rise to an incident. A threat might exist in a passive state without causing any untoward incident. For example, a volcano may remain dormant for ages, without causing any harm to its surrounding regions. On the other hand, cases of malware infections are on the rise and are causing serious damage to software and information assets. The proposed methodology computes threat probability, referred to as likelihood of occurrence of threats, as described in the following section.

4.3.1 Likelihood of Occurrence

Likelihood of occurrence (LOC) of threats can be predicted by combining data about past threat occurrences with current values of threat parameters (Bhattacharjee et al., 2014). Specifically, five factors are considered: (i) past occurrences (p_t) of threat-related incidents; (ii) proximity of threat agents to task elements (a_t); (iii) existence of motive of an agent (mvn_t); (iv) resources to realize a threat (res_t); and (v) efficacy of controls implemented (if any) to mitigate threat (e_c). Combining these, likelihood of threat t, for human-induced threats, is obtained as follows:

$$\mathrm{LOC}(t) = \mathrm{ceil}((p_t + a_t + mvn_t + res_t) / (2 + e_c)) \qquad (5)$$

Divisor 2 is used in Equation 5 to scale down the value of LOC. For natural and environmental threats, LOC is given by:

$$\mathrm{LOC}(t) = \mathrm{ceil}((p_t + a_t + res_t) / (2 + e_c)) \qquad (6)$$

The methods of computation of threat parameters are as described in (Bhattacharjee et al., 2014). The computation of a_t needs some attention, though. In case of human-induced threats, both physical and logical access may be crucial. While physical access is denoted by *reachability* of threat agents, logical access is manifested in task authorizations. If a human threat agent can *reach* the task element, as well as has *authorization* to access it (maybe acquired illegally), then the value of a_t is 3. If the agent can either reach the task element or has authorization on it, but not both, then the value of a_t is 2. Finally, if the agent can neither reach the task element nor has authorization on it, then the value of a_t is 1. It has been shown in (Bhattacharjee et al., 2014) that:

$$\mathrm{LOC}(t)\ \varepsilon\ \{1, 2, 3, 4, 5\} \qquad (7)$$

4.4 Security Concern

After identifying vulnerabilities within task elements and corresponding threats that can exploit them, the proposed methodology computes breachability and security concern values of threat-vulnerability pairs. Breachability defines the potential of a threat being able to exploit a given vulnerability (Bhattacharjee et al., 2013). It is computed as:

$$\begin{gathered} B(t, v) = \mathrm{RoundOff}(\alpha * \mathrm{LOC}(t) + \beta * \mathrm{Exp}(v)) \\ \text{such that } (\alpha + \beta) = 1 \end{gathered} \qquad (8)$$

Weights can be customized based on the enterprise requirements. From Equations 4, 7 and 8, it is obvious that

$$B(t, v)\ \varepsilon\ \{1, 2, 3, 4, 5\} \qquad (9)$$

Security Concern (SC) value is computed considering the breachability and severity of vulnerability. It denotes the impact that can occur if vulnerability v is exploited by corresponding threat t and is given by:

$$SC(t, v) = \mathrm{ceil}(B(t, v) * \mathrm{Sev}(v) / 5) \qquad (10)$$

Since $\mathrm{Sev}(v) \in \{1, 2, 3, 4, 5\}$, it can be seen from Equations 9 and 10 that:

$$SC(t, v)\ \varepsilon\ \{1, 2, 3, 4, 5\} \qquad (11)$$

4.5 Risk

Risk is first computed for individual tasks of a business process (BP). These are then aggregated to derive risk values for the entire business process.

4.5.1 Risk to Task

After having computed security concern values of all threat-vulnerability (t-v) pairs for all elements of a task, they are grouped into three categories – confidentiality concern (C-concern), integrity concern (I-concern) and availability concern (A-concern). The category C-concern contains all those security concern values that have been derived from such t-v pairs which can breach the confidentiality of a task element. Similarly, I-concern and A-concern categories contain values corresponding to integrity and availability parameters of a task element, respectively. If a t-v pair can breach multiple security parameters, then the pair will contribute security concern to multiple categories. For example, the threat "unauthorized data modification" can exploit the vulnerability "lack of access control" to breach integrity and availability of the input element of a task. This gives rise to both I-concern as well as A-concern for the input element.

The security concern categories for task ti can be denoted by C-concern(t_i), I-concern(t_i) and A-concern(t_i). Three separate risk values are obtained for a task – confidentiality risk, integrity risk and availability risk. These are computed as follows:

$$\begin{gathered} \text{C-risk}(t_i) = \mathrm{ceil}(\text{C-value}(t_i) * \max(\text{C-concern}(t_i)) / 5) \\ \text{I-risk}(t_i) = \mathrm{ceil}(\text{I-value}(t_i) * \max(\text{I-concern}(t_i)) / 5) \\ \text{A-risk}(t_i) = \mathrm{ceil}(\text{A-value}(t_i) * \max(\text{A-concern}(t_i)) / 5) \end{gathered} \qquad (12)$$

Here, C-value(t_i), I-value(t_i) and A-value(t_i) represent the confidentiality, integrity and availability values of task t_i as described in Section 4. These values are combined with the maximum values of C-concern(t_i), I-concern(t_i) and A-concern(t_i) to obtain confidentiality, integrity and availability risk values, respectively, of task t_i. It can be seen from Equation 11 and Section 4 that

$$\begin{gathered} \text{C-risk}(t_i)\ \varepsilon\ \{1, 2, 3, 4, 5\} \\ \text{I-risk}(t_i)\ \varepsilon\ \{1, 2, 3, 4, 5\} \\ \text{A-risk}(t_i)\ \varepsilon\ \{1, 2, 3, 4, 5\} \end{gathered} \qquad (13)$$

4.5.2 Risk to Business Process

The next step in the proposed methodology is to compute the risks to a business process. It has been shown in Section 3 that a business process consists of a set of tasks that follow one or more of the following flow relations: sequential, iterative, conditional or parallel flow. The process for combining risk values of tasks is described below.

1) ***Sequential Tasks.*** Risk values of sequential tasks (Figure 1(a)) are combined by computing their simple average. This helps in deriving a reasonable risk value that is neither too high (which would be the case if the max. risk value of component tasks was considered), nor too low, but reflects the actual state of the business process. One could be tempted to use weighted average with weights being assigned to tasks based on their criticality. However, it may be recalled that during computation of task values, confidentiality, integrity and availability requirements of tasks have already been considered, and hence the same exercise need not be repeated here. Thus, the combined risk value for a set of sequential tasks of a business process will be given by:

$$\begin{aligned} C\text{-risk}(t_1,...t_n) &= \text{ceil}(\Sigma\ C\text{-risk}(t_i) / n) \\ I\text{-risk}(t_1,...t_n) &= \text{ceil}(\Sigma\ I\text{-risk}(t_i) / n) \\ A\text{-risk}(t_1,...t_n) &= \text{ceil}(\Sigma\ A\text{-risk}(t_i) / n) \end{aligned} \tag{14}$$

Here, *C*-risk($t_1,\dots t_n$) denotes confidentiality-risk for sequential tasks $t_1,\dots t_n$. Similarly, *I*-risk($t_1,\dots t_n$) and *A*-risk($t_1,\dots t_n$) represent integrity-risk and availability-risk for $t_1,\dots t_n$, respectively.

2) ***Iterative Tasks.*** A set of iterative tasks (Figure 1(d)) can be viewed as a sequence of tasks whose risk values *may* vary between iterations. Hence, their combined risk value is obtained in the same manner as for sequential tasks. For example, if two tasks in sequence, say t_1 and t_2, are repeated n times, then it can be assumed that there are actually 2n tasks, say t_{11}, t_{21}, t_{12}, t_{22},…, where t_{11} denotes task t_1 during the first iteration, t_{21} denotes task t_2 during the first iteration, t_{12} denotes task t_1 during the second iteration, and so on. Also, their risk values may change over iterations. Hence, their combined risk values will be given by:

$$\begin{aligned} C\text{-risk}(t_1, t_2) &= \text{ceil}(\Sigma(C\text{-risk}(t_{1i}) + C\text{-risk}(t_{2i})) / n) \\ I\text{-risk}(t_1, t_2) &= \text{ceil}(\Sigma(I\text{-risk}(t_{1i}) + I\text{-risk}(t_{2i})) / n) \\ A\text{-risk}(t_1, t_2) &= \text{ceil}(\Sigma(A\text{-risk}(t_{1i}) + A\text{-risk}(t_{2i})) / n), \\ &\text{where } i = 1, ..., n \end{aligned} \tag{15}$$

Here, *C*-risk(t_1, t_2) denotes confidentiality-risk for iterative tasks t_1 and t_2. Similarly, *I*-risk(t_1, t_2) and *A*-risk(t_1, t_2) represent integrity-risk and availability-risk for t_1 and t_2, respectively.

3) ***Conditional Tasks.*** In cases where one among several tasks can be executed, depending on the evaluation result of a condition, the risk value of the task that is greatest among them, is considered. For example, in Figure 1(c), one among tasks T_4 and T_5 will be executed depending on the result of evaluation of condition C_1. Hence, the maximum of the risk values of T_4 and T_5 will be considered during computation of risk of the corresponding business process.

Hence, resultant risk of conditional tasks will be given by:

$$\begin{aligned} C\text{-risk}(t_1,...t_n) &= \max(C\text{-risk}(t_1),...,C\text{-risk}(t_n)) \\ I\text{-risk}(t_1,...t_n) &= \max(I\text{-risk}(t_1),...,I\text{-risk}(t_n)) \\ A\text{-risk}(t_1,...t_n) &= \max(A\text{-risk}(t_1),...,A\text{-risk}(t_n)) \end{aligned} \tag{16}$$

Here, *C*-risk($t_1,\dots t_n$) denotes the resultant confidentiality-risk for conditional tasks $t_1,\dots t_n$. Similarly, *I*-risk($t_1,\dots t_n$) and *A*-risk($t_1,\dots t_n$) represent resultant integrity-risk and availability-risk for $t_1,\dots t_n$, respectively.

4) ***Parallel Tasks.*** In cases where several tasks can be executed simultaneously, the resultant risk value is obtained as a simple average of the risk values to those tasks. For example, in Figure 1(b), tasks T_9 and T_{10} will be executed in parallel. Hence, the simple average of risk values of T_9 and T_{10} will be considered during computation of risk of the corresponding business process.

Hence, resultant risk of parallel tasks will be given by:

$$\begin{aligned} C\text{-risk}(t_1,...t_n) &= \text{ceil}(\Sigma(C\text{-risk}(t_i) / n) \\ I\text{-risk}(t_1,...t_n) &= \text{ceil}(\Sigma(I\text{-risk}(t_i) / n) \\ A\text{-risk}(t_1,...t_n) &= \text{ceil}(\Sigma(A\text{-risk}(t_i) / n), \\ &\text{where } i = 1, ..., n \end{aligned} \tag{17}$$

Here, *C*-risk($t_1,\dots t_n$) denotes confidentiality-risk for parallel tasks $t_1,\dots t_n$. Similarly, *I*-risk($t_1,\dots t_n$) and *A*-risk($t_1,\dots t_n$) represent integrity-risk and availability-risk for $t_1,\dots t_n$, respectively.

As shown in Figure 2, a business process can be a combination of various kinds of task structures. The composite risk values of the entire business process can be obtained as follows:

- Replace individual sets of parallel tasks with a single task having risk value as stated in Equation 17;
- Replace individual sets of conditional tasks with a single task having risk value as stated in Equation 16;
- Expand sets of iterative tasks to form a simple sequence of tasks having individual risk values;
- Finally, compute the risk values of the business process, which now contains a single sequence of tasks, using Equation 14.

The risk values of a business process are denoted by *C*-risk(BP_j), *I*-risk(BP_j) and *A*-risk(BP_j), which can be interpreted as confidentiality-, integrity- and availability-risk of business process BP_j.

5 DISCUSSION

The proposed methodology computes security risks to business processes of an enterprise. Risks that have the potential to breach confidentiality, integrity and availability of business processes are analyzed at individual task level and their values are calculated. These values are then aggregated to derive consolidated risk values for corresponding business processes. Thus, three separate risk values, namely confidentiality risk, integrity risk and availability risk, are computed for each business process. It has been assumed that maintenance of correct order of the tasks of a business process, denoted by the flow relation →, is the responsibility of an enterprise. As discussed in Section 3, improper configuration of the flow relation can lead to disastrous consequences. Also, change in the flow relation actually gives rise to a business process that is different from the desired one as is obvious from the definition of business process given by Equation 2. Hence, in such a scenario, the risk that would be computed would, in fact, pertain to the new (incorrectly configured) business process.

This risk assessment methodology will help an enterprise to decide on risk mitigation strategies that can protect its more critical security parameters. Owing to the generic nature of the methodology, both external and internal risks can be addressed. Besides, the steps of risk computation can be re-traced to identify the tasks, and their specific elements, that are the biggest contributors to the risk values. Another approach could be to maintain records of the threat-vulnerability pairs, their values and the corresponding security parameters that they can breach. Such an exercise can be easily implemented by following the steps of the proposed methodology as has been described in earlier sections. This would help in designing a better risk mitigation technique that can be effective in addressing the concerns of the enterprise.

It may be noted that the proposed methodology provides a technique for evaluating business processes, vulnerabilities and threats, and combining them to derive risk values. Major categories of vulnerabilities have also been described in Section 4.2. However, specific vulnerabilities and threats vary between enterprises and enterprise-sectors. Hence, actual identification of these risk factors is the task of the operational team that is involved in the computation of business risks.

A detailed case study that demonstrates the steps of the proposed methodology is included in the following section.

6 CASE STUDY

The case study refers the Loan Management Process shown in Figure 2. It can be seen that tasks t_1 to t_8 follow a sequential flow wherein completion of t_1 is a pre-requisite for initiation of t_2, and so on. Similarly, t_8 is a pre-requisite for t_9 and t_{10} which, in turn, lead to task t_{11}. Finally, execution of t_4 or t_5 or t_{11} should be completed before execution of task t_{12}.

The following assumptions may be made: task t_1 requires infrastructure, money and time, while tasks t_3 to t_{12} require all resources stated in Section 3. Besides, all the tasks require some input and generate some outcome.

Considering the pre-requisites, resources, input, and outcome of tasks and guidelines provided in Table 1, Table 2 and Table 3, appropriate C, I and A values can be assigned to the tasks of loan management process as shown in Table 5.

Table 5: C, I and A values of the tasks of loan management process.

Task_ID	C-value	I-value	A-value
t_1	4	4	4
t_2	3	3	4
t_3	3	3	4
t_4	3	4	4
t_5	3	4	4
t_6	4	4	4
t_7	4	4	4
t_8	4	4	4
t_9	3	4	4
t_{10}	3	4	4
t_{11}	3	4	4
t_{12}	3	4	4

Table 6: Vulnerabilities in the tasks of loan management process.

Task_ID	Vulnerability	Sev	Exp
t_1, t_9, t_{10}, t_{11}	Improperly configured functions may generate erroneous outcome, or may not produce any outcome (v_1)	3	2
t_4, t_5	Lack of training of concerned personnel may lead to incorrect execution of tasks (v_2)	4	2
t_6, t_7, t_8, t_9, t_{10}	Improperly configured infrastructure may hinder task execution (v_3)	3	2
t_{11}, t_8, t_9, t_{10}	Lack of output validation may cause leakage of confidential information to unauthorized entities (v_4)	3	2

Threats that can exploit the above vulnerabilities are listed in Table 7. LOC values of threats are computed applying Equations 5 and 6.

Applying Equation 8 and assuming $\alpha = \beta = 0.5$, B(th_1, v_1) = 0.5 * 4 + 0.5 * 2 = 3.

Similarly, B(th_2, v_2) = 3, B(th_3, v_3) = 3, and B(th_3, v_4) = 3.

Applying Equation 10, SC(th_1, v_1) = (3 * 3) / 5 = 2.

Table 7: Vulnerabilities and threats.

Vulnerability	Threat	LOC
v_1	Error in use (th_1)	4
v_2	Corruption of data (th_2)	3
v_3	Illegal processing (th_3)	3
v_4	Illegal processing (th_3)	3

Similarly, SC(th_2, v_2) = 3, SC(th_3, v_3) = 2, and SC(th_3, v_4) = 2.

It may be seen that threat th_1 can exploit vulnerability v_1 to breach the availability of tasks t_1, t_9, t_{10} and t_{11}. So, applying Equation 12,

A-risk(t_1) = (4 * 2) / 5 = 2, *A*-risk(t_9) = 2, *A*-risk(t_{10}) = 2, and *A*-risk(t_{11}) = 2.

Threat th_2 can exploit vulnerability v_2 to breach the availability of tasks t_4 and t_5. Hence, *A*-risk(t_4) = (4 * 3) / 5 = 3; *A*-risk(t_5) = (4 * 3) / 5 = 3.

Threat th_3 can exploit vulnerabilities v_3 and v_4 to breach the availability of tasks t_6, t_7 and t_8. So, *A*-risk(t_6) = 2, *A*-risk(t_7) = 2, and *A*-risk(t_8) = 2.

Applying Equation 14, combined risk of sequential tasks t_6, t_7 and t_8 = 2.

Applying Equation 16, resultant risk of conditional tasks t_4 and t_5 = 3.

Applying Equation 17, resultant risk of parallel tasks t_9 and t_{10} = 2.

Assuming that tasks t_2, t_3 and t_{12} do not have any identified risks, the composite risk value of the loan management process = (2+2+3+2+2) / 5 = 3.

7 CONCLUSION AND FUTURE WORK

A methodology for assessing security risks of enterprise business processes has been presented in this paper. The paper begins by stating a formal definition of business process based on tasks and their flow relation. The types of flow relations, namely sequential, parallel, conditional and iterative, have been clearly illustrated. The component elements of a task and their functionalities have been detailed. The manner in which a task can be related to another task has also been described. Such relation is defined either by the dependence of a task on the outcome of another task, or due to triggering of one task by another. A technique for deriving the confidentiality, integrity and availability requirements of tasks has been proposed. The paper also models the vulnerabilities within, and threats to, task elements, along with their attributes. These include severity and exploitability of vulnerabilities, and likelihood of occurrence of threats. Finally, procedures for computing security concern and risk values of tasks and business processes have been detailed. The paper concludes with a case study that illustrates the proposed methodology in its entirety.

It is hoped that the technique presented in this paper will help implementers to correctly model, and identify and assess critical risks to, business processes. This will, in turn, enable the design and implementation of an effective protection mechanism for enterprise business processes.

Future work is geared towards the development of a tool based on the proposed methodology. Results of risk assessment using the tool will help in validating the technique that will lead to further improvement of the mechanism. Another interesting extension could be the formulation of techniques for combining business process risks and asset-based risks to derive the overall value of enterprise risk. This could be useful in the preparation of dashboards for viewing by top management and making critical investment decisions.

ACKNOWLEDGEMENTS

This research was partially supported by grants allocated by the Department of Electronics and Information Technology, Govt. of India.

REFERENCES

Armando, A., Ponta, S. E., 2011. Model Checking of Security-Sensitive Business. In *CRiSIS'11, 6th International Conference on Risks and Security of Internet and Systems*. IEEE, pp. 66-80.

Bhattacharjee, J., Sengupta, A., Mazumdar, C., 2013. A Formal Methodology for Enterprise Information Security Risk Assessment. In *CRiSIS'13, 8th International Conference on Risks and Security of Internet and Systems*. IEEE, pp. 1-9.

Bhattacharjee, J., Sengupta, A., Mazumdar, C., 2014. A Formal Methodology for Modeling Threats to Enterprise Assets. In *ICISS'14, 10th International Conference on Information Systems Security*. Springer, pp. 149-166.

Denning, D. E., 1976. A Lattice Model of Secure Information Flow. *Communications of ACM*, ACM. Vol. 19, No. 5, pp. 236-243.

ENISA. 2015. *Ebios*. Available at: https://www.enisa.europa.eu/activities/risk-management/current-risk/risk-management-inventory/rm-ra-tools/t_ebios.html (Accessed: 5 June 2015).

ISO/IEC JTC 1 IT SC 27, 2011. *Information technology – Security techniques - Information security risk management, ISO/IEC 27005:2011*, ISO/IEC. Geneva, 2nd Edition.

ISO/IEC JTC 1 IT SC 27, 2014. *Information technology – Security techniques - Information security management systems – Overview and vocabulary, ISO/IEC 27000:2014*, ISO/IEC. Geneva, 3rd Edition.

Jakoubi, S, Tjoa, S., Goluch, S., Kitzler, G., 2010. Risk-Aware Business Process Management—Establishing the Link Between Business and Security. *Book of Complex Intelligent Systems and Their Applications*, Springer Science+Business Media, LLC. Vol. 41, pp. 109-135.

Khanmohammadi, K., Houmb, S. H., 2010. Business Process-based Information Security Risk Assessment. In *4th International Conference on Network and System Security*, IEEE, pp. 199-206.

Lowis, L., Accorsi, R., 2011. Vulnerability Analysis in SOA-Based Business Processes. *IEEE Transactions On Services Computing*, IEEE. Vol. 4, No. 3, pp. 230-242.

Marchesini, S., Viganò, L., 2011. A Hierarchy of Knowledge for the Formal Analysis of Security-Sensitive Business Processes. In *CRiSIS'11, 6th International Conference on Risks and Security of Internet and Systems*. IEEE, pp. 1-10.

MEHARI, 2010. *Risk analysis and treatment Guide*. Available at: https://www.clusif.asso.fr/fr/production/ouvrages/pdf/MEHARI-2010-Risk-Analysis-and-Treatment-Guide.pdf (Downloaded: 10 August 2015).

OMG, 2011. *Business Process Model and Notation, BPMN*, Object Management Group. Massachusetts, 2nd edition.

Peltier, T. R., 2010. *Information Security Risk Analysis*, CRC Press. Florida, 3rd Edition.

Soanes, C., Stevenson, A. (ed.), 2011. *Concise Oxford English Dictionary*, Oxford University Press. New York, 12th Edition.

Sun, S. X., Zhao, J. L., Nunamaker, J. F., Sheng, O. R. L., 2006. Formulating the Data-Flow Perspective for Business Process Management. *Information Systems Research*, INFORMS. Vol. 17, No. 4, pp. 374-391.

Tjoa, S., et al., 2011. A Formal Approach Enabling Risk-Aware Business Process Modeling and Simulation. *IEEE Transactions On Services Computing*, IEEE. Vol. 4, No. 2, pp. 153-166.