

Database Isolation and Filtering against Data Corruption Attacks

Meng Yu, Wanyu Zang
Department of Computer Science
Western Illinois University

Peng Liu
College of Information Sciences and Technology
The Pennsylvania State University, University Park

Abstract

Various attacks (e.g., SQL injections) may corrupt data items in the database systems, which decreases the integrity level of the database. Intrusion detection systems are becoming more and more sophisticated to detect such attacks. However, more advanced detection techniques require more complicated analyses, e.g., sequential analysis, which incurs detection latency. If we have an intrusion detection system as a filter for all system inputs, we will introduce a uniform processing latency to all transactions of the database system. In this paper, we propose to use a “unsafe zone” to isolate user’s SQL queries from a “safe zone” of the database. In the unsafe zone, we use polyinstantiations and flags for the records to provide an immediate but different view from that of the safe zone to the user. Such isolation has negligible processing latency from the user’s view, while it can significantly improve the integrity level of the whole database system and reduce the recovery costs. Our techniques provide different integrity levels within different zones. Both our analytical and experimental results confirm the effectiveness of our isolation techniques against data corruption attacks to the databases. Our techniques can be applied to database systems to provide multizone isolations with different levels of QoS.

1 Introduction

While more and more data processing services are running through web services, transaction level attacks, such as SQL injections, become one of the major threats to the database systems because transaction level attacks can be done through a variety of ways. For example, the attacks can be done through web applications [8, 5, 14]. Among the OWASP top ten most critical web application security vulnerabilities [12], five out of the top 6 vulnerabilities can directly enable the attacker to launch a malicious transaction. The attacks can also be done through identity theft. Identity theft naturally leads to malicious transactions because a main purpose of using a stolen identity is for the attacker to

launch some malicious transactions to steal money, etc. As indicated in [2], since database management systems are an indispensable component of modern Internet services and mission-critical applications, it is more susceptible to malicious attacks from remote sites.

Transaction level attacks have been studied in a good number of researches, e.g., in [2, 1, 16, 18], where recovery from data corruption caused by attacks is the major concern. These work focuses on how to trace damage spreading and repair the damage inside a database or across database systems. It is a remedy when the integrity level of the database system has already been compromised.

Intrusion detection techniques [11], especially some intrusion detection techniques for database systems like [4, 13, 17], are able to detect the attacks in the first place. However, to deploy an intrusion detection system (IDS) in a Database Management System (DBMS), we need to consider the *detection latency* incurred by the IDS. The detection latency of IDS is mainly caused by the following reasons. First, when the IDS is deployed in a local host, the detection latency is mainly caused by the processing time to analyze the characters of inputs, e.g., to model the behaviors of inputs, or to recognize the patterns of inputs. Recent more complex intrusion detection techniques require accessing a large amount of knowledge created based on training data sets. Second, when an IDS is deployed in a distributed manner, the detection latency will be significantly longer due to the latency introduced by communications and message exchanges between different hosts.

Such detection latency is a big concern to the DBMSs. Assume a perfectly accurate IDS, we can use it as a filter to inspect all database input to guarantee that there is no malicious inputs being injected to the DBMS. However, such filtering will introduce the detection latency to all database transactions. The database users will suffer prolonged transaction processing time, which significantly degrade the performance of DBMSs. If we let all inputs bypass the IDS and let the IDS work as a simple *inspector*, when the IDS finds out malicious inputs, the integrity level of the databases has already been compromised, especially when corrupted data items are referred by innocent transactions, where the dam-

