

Towards Data Security in Affordable Data Warehouses

Marco Vieira
CISUC/DEI
University of Coimbra
3030-290 Coimbra, Portugal
mvieira@dei.uc.pt

Jorge Vieira
CISUC/DEI
Critical Software SA
3045-504 Coimbra, Portugal
jvieira@criticalsoftware.com

Henrique Madeira
CISUC/DEI
University of Coimbra
3030-290 Coimbra, Portugal
henrique@dei.uc.pt

Abstract

The Data Warehouse Striping (DWS) technique is a round-robin data partitioning approach especially designed for affordable data warehousing environments based on clusters of low-cost computers running low-cost open-source software, which guarantees a nearly optimal speed up and scale up when new nodes are added to the cluster. However, low-cost software does not provide the security capabilities needed to protect critical business data. For example, most open-source Database Management Systems (DBMS) lack in providing efficient data encryption mechanisms essential to guarantee data confidentiality. This paper presents our current work on developing an architecture for affordable data warehouses with extended data protection capabilities and tolerance against nodes Denial of Service (DoS) attacks. The approach uses data encryption, spurious data, signatures, and redundancy to guarantee full data protection even when an attacker gets administrative access to one or more cluster nodes.

1. Introduction

A data warehouse (DW) is an integrated and centralized repository that offers high capabilities for data analysis and manipulation [3]. Data Warehouses represent nowadays an essential source of strategic information for many enterprises. In fact, as competition among enterprises increases, the availability of tailored business information that helps decision makers during decision support processes is of utmost importance.

In data warehousing the data is organized according to the multidimensional model [3], which includes facts and dimensions. Facts are numeric or factual data that represent a specific business or process activity and each dimension represents a different perspective for the analysis of the facts. The multidimensional model is typically implemented as one or more star schema made of a large central fact table surrounded by several dimensional tables related to the fact table by foreign keys [3].

Data warehouses are repositories that usually contain high volumes of data integrated from several different operational sources. Thus, the data stored in a DW can

range from some hundreds of Gigabytes to dozens of Terabytes. In order to properly handle high volumes of data, allowing performing complex data manipulation operations, enterprises normally use high performance systems to host the DW. The most common choice is systems that offer massive parallel processing capabilities [4], [1], as Massive Parallel Processing (MPP) systems or Symmetric MultiProcessing (SMP) systems. Due to the high price of this type of systems, some less expensive alternatives have already been proposed and implemented. One of these alternatives is the Data Warehouse Striping (DWS) technique [2].

In a simplified view, the DWS technique consists in the distribution of the data of a data warehouse over a cluster of low-cost computers, providing near linear speedup and scale up when adding new nodes to the cluster. To achieve low-cost, the data warehouse cluster is based on open-source software and the computers can be shared with other applications (whose typically do not exploit all computational resources of the machines). However, open-source software (and Database Management Systems (DBMS) in particular) normally does not provide the full security capabilities needed to protect critical business data. Furthermore, sharing the computers with other applications increases the risk of security attacks as several users can have administrative access to the machines.

One of the main problems faced by system administrators is the protection of the data against unauthorized access or corruption due to malicious actions. In fact, due to the impressive growth of the internet, security attacks have become one vital concern in any information infrastructure. Database security arises from the need to protect from: 1) intentional unauthorized attempts to access private data, and 2) loss or corruption of critical data due to malicious actions. Other concerns include protecting against undue delays in accessing or using data, or even against malicious interferences that may cause Denial of Service (DoS).

Security is an integrative concept that includes the following properties [7]: confidentiality, authenticity, integrity, and availability. In this paper we present our current work on achieving high data security in affordable data warehouses. The goal is to endow DWS with the capa-

bilities needed to fulfill all the data security properties, providing extended data protection capabilities and tolerance against nodes DoS attacks.

Data confidentiality is achieved by encrypting the dimensions data. Facts data is not encrypted due to performance issues (encryption in large tables is a heavy process that typically ruins the system performance [8]). Nevertheless, to improve confidentiality, facts data is obfuscated by adding spurious records to the fact tables in order to mislead the attacker. Data authenticity and integrity are guaranteed by using signatures in all records in the data warehouse and concurrent detection of malicious data modifications. Finally, data availability is achieved using replication.

The structure of the paper is as follows. Section 2 presents the DWS technique. Section 3 discusses the approach for data protection in DWS. Section 4 concludes the paper and presents the intended research directions.

2. Using the DWS technique to build affordable data warehouses

Low cost platforms based on the DWS approach and open source technology can be employed to allow small and medium enterprises to acquire and use data warehousing technology. Our goal is thus to develop a technology that allows a dramatic reduction of the hardware, software, and administration cost when compared to traditional data warehouses based in high-end servers and proprietary software. As shown in Figure 1, our proposal is to use parallel query processing in low-cost clusters of computers running inexpensive open-source software,

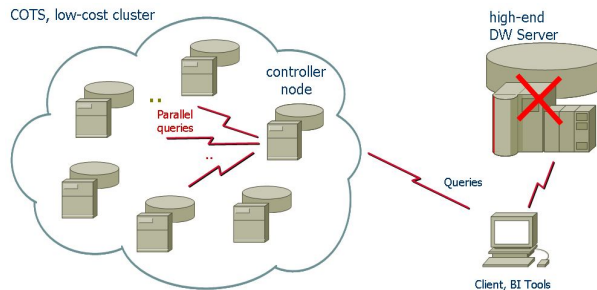


Figure 1. Architecture for low-cost data warehouses.

In the DWS technique [2] the data of each star schema of a data warehouse is distributed over an arbitrary number of nodes having the same star schema (which is equal to the schema of the equivalent centralized version). The data of the dimension tables is replicated in each node of the cluster (i.e., each dimension has exactly the same rows in all the nodes) and the data of the fact tables is distributed over the fact tables of the several nodes using strict row-by-row round-robin partitioning or hash parti-

tioning (see Fig. 1). It is important to emphasize that the replication of dimension tables does not represent a serious overhead because usually the dimensions only represent between 1% and 5% of the space occupied by all database [3]. In the rare cases in which the star schema has a very large dimension it is possible to accommodate that dimension in the DWS cluster by using selective loading techniques [5] or encoding techniques [6].

DWS data partitioning for star schemas balances the workload by all computers in the cluster, supporting parallel query processing as well as load balancing for disks and processors. The experimental results presented in [2] show that a DWS cluster can provide an almost linear speedup and scale up.

In a DWS cluster typical OLAP (On-Line Analytical Processing) queries are executed simultaneously by all the nodes available and the results are merged by the DWS middleware. As the use of a large number of inexpensive nodes increases the risk of having node failures that impair the computation of queries, DWS uses selective replication of data over the cluster nodes to guarantee full availability when one or more nodes fail.

3. Data protection in DWS nodes

The goal of our work is to endow DWS with the mechanisms needed to assure data authenticity, integrity, confidentiality, and availability. This section presents our current research thoughts on how to achieve this goal.

3.1. Assuring data authenticity and integrity

Data authenticity and integrity can be guaranteed by using signatures in all records in the data warehouse. Each record in each table must have an associated signature that allows DWS to distinguish original data from tampered data. Obviously the signatures generation and verification must be controlled by the DWS middleware.

Record based signatures (i.e., signatures based on all columns of each record) may not be the most interesting approach. In fact, using record based signatures requires that all columns from a record are read in order to verify the signature, which may not be the best approach in terms of performance. Using one signature for each column in each record is an alternative; however it brings a storage space problem that also influences performance. Our goal is to investigate the possibility of having a single signature that can be applied to validate each column individually and also to validate the entire record at once, while maintaining high-performance.

The DWS middleware has to generate the signatures when inserting or modifying data in the data warehouse. Those signatures are used later during queries execution

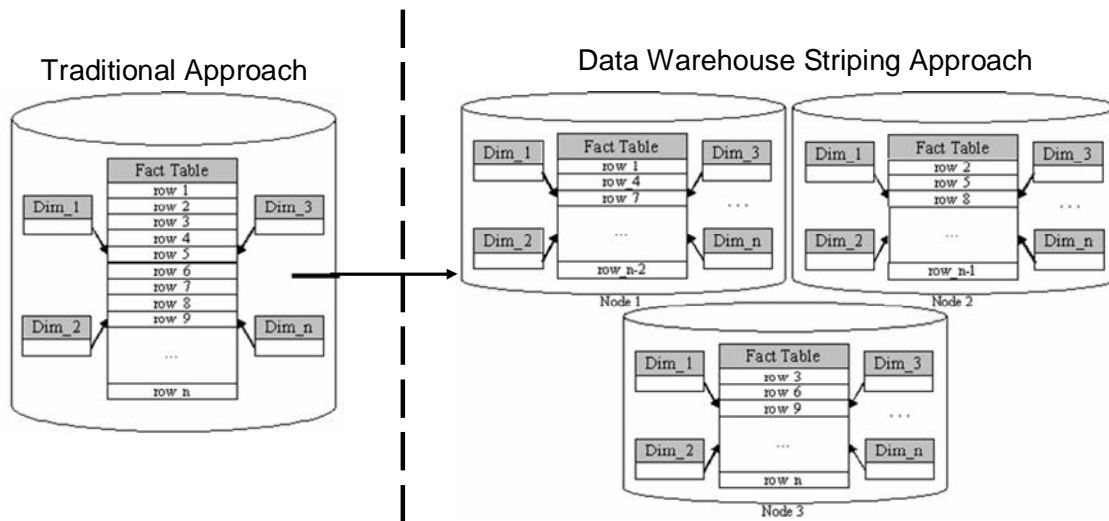


Figure 1. Data Warehouse Striping Technique.

to guarantee that the data is authentic and has not been maliciously changed. If an authenticity or integrity problem is detected then the system must assure that that data is not used (see section 3.3 on assuring data availability).

3.2. Assuring data confidentiality

To provide communication in a secure way and avoid the access to data transferred over the network, most DBMS provide encryption mechanisms for data communication. In some cases, mechanisms that allow the encryption of the data stored in the database tables are also provided. These mechanisms are quite useful as they allow protecting the data even when someone maliciously accesses the system. However, using encryption for data storage is a heavy process that may ruin the performance of the system. In fact, previous work shows that even the encryption algorithms provided by the well-known and quite sophisticated Oracle DBMS cause very high performance degradations (representing in some cases an increase of 1000% in the response times).

Due to this performance issue, data encryption should be used only for tables with a small number of records. Tables that store large amounts of data must not be encrypted, which means that an alternative approach is required to guarantee the confidentiality of the data in those cases.

Our approach to achieve data confidentiality in DWS clusters consists in encrypting the dimensions data, which typically resides in small size tables (usually the dimensions only represent between 1% and 5% of the space occupied by all database). One of the research efforts needed is to identify the data encryption mechanisms to be used (the ones that offer higher security with lower performance impact). An important aspect is that

primary keys and foreign keys in dimensions do not need to be encrypted as they are typically filled with synthetic values. The combination of encryption with encoding techniques [6] to reduce the size of very large dimensions is also going to be explored.

Facts data cannot be encrypted due to performance issues. In fact, as a fact table may store several gigabytes of data, the use of data encryption on those tables would ruin the queries response times (although some critical fact columns may have to be encrypted despite of the performance problems). However, as in DWS the facts data is fragmented across the cluster nodes using round-robin or hashing distribution, the confidentiality issue is minimized (additionally facts data cannot be easily related to the dimensions as these will be encrypted). In fact, when an attacker gets admittance to one or more nodes he only has access to portions of the data that are typically meaningless without their counterparts. However, there may be some cases in which the attacker may retrieve some critical facts information using solely the data stored in the attacked node (it obviously depends on how the data distribution is performed). This way, to improve confidentiality, facts data will be obfuscated by adding spurious records to the fact tables in order to mislead the attacker. The original and spurious records can be distinguished based on the use of different data signatures (which must also guarantee the authenticity of the data even for the spurious records).

Note that, adding spurious records to the fact tables may cause a performance overhead. However, we believe that this overhead will be considerably smaller than the overhead caused by data encryption. Further investigation is obviously needed to identify the number of extra records to include in each table and how to vary the facts values to effectively mislead the attacker.

To improve the privacy of the data, table names and column names (among other database objects) may also be encrypted. This difficult the task of the attacker as it becomes more difficult to understand the meaning of each table and column. Obviously the DWS middleware must be able to translate the user queries that use the original names into queries using the encrypted table and column names.

3.3. Assuring data availability

A DWS cluster is typically based on inexpensive nodes. However, the use of a large number of inexpensive nodes increases the risk of having node failures that impair the computation of queries. This way, DWS includes a redundancy mechanism, named RAIN (Redundant Array of Inexpensive Nodes), able to tolerate failures of several cluster nodes (the number of node failures tolerated depends on the configuration used). The RAIN technique is based on the selective replication of data and comprises two redundancy schemes: simple redundancy (RAIN-0) and striped redundancy (RAIN-S). The simple redundancy approach consists of replicating the facts data from each node in other nodes of the cluster. The striped replication is an evolution of the simple replication where the facts data from each node is randomly distributed in $N-1$ sub-partitions (where N is the number of nodes) and each sub-partition is replicated in at least one of the other nodes.

Although the RAIN mechanism was initially designed to tolerate node failures it can also be used to guarantee data availability in the presence of data attacks. In fact, the RAIN mechanism is already able to automatically identify unavailable nodes and redirect the queries to the alternative replicas. Obviously, this allows DWS to tolerate attacks that cause nodes DoS.

For the attacks that affect the data integrity (i.e., where the attacker maliciously modifies the data), we need to add a mechanism able to concurrently detect malicious data modifications. This mechanism, based on the data signatures used to guarantee authenticity and integrity (see section 3.1), must automatically disable nodes when malicious modifications are detected. The RAIN mechanism can then be used to redirect the queries to the available replicas. An important aspect is that the detection mechanism must be built in such way that can not be tampered by the attacker (e.g., triggers are not a good approach as the attacker can easily deactivate them).

4. Conclusion and future work

In this paper we discuss our current research work on achieving high data security in DWS clusters. Data con-

fidentiality is achieved by encrypting the dimensions data. Facts data is obfuscated by adding spurious records to the fact tables in order to mislead the attacker. Data authenticity and integrity are guaranteed by using signatures in all records in the data warehouse and concurrent detection of malicious data modifications. Finally, data availability is achieved by using data replication.

Several interesting aspects have to be researched to achieve our goals. In fact, we need to study different encryption mechanisms in order to select one (or several) that provide high confidentiality with low performance costs. Facts obfuscation by using superfluous records is also an interesting challenge. Some aspects that have to be investigated include the number of extra records to include in each table and how to vary the facts values to effectively mislead the attacker. Concerning signatures we need to identify the best method for the DWS context. A mechanism to concurrently detect unauthorized data modifications based on signatures has also to be defined.

Obviously one of the main research efforts to be undertaken is related to the evaluation of the approach in terms of the impact in the DWS cluster performance. In fact, we need to compare the baseline performance (DWS without security mechanisms) with the performance achieved when using our approach and with the performance achieved when using data encryption in the fact tables.

References

- [1] Agosta, L., "Data Warehousing Lessons Learned: SMP or MPP for Data Warehousing", DM Review Magazine, 2002.
- [2] Bernardino, J., Madeira, H., "A New Technique to Speedup Queries in Data Warehousing", ABDIS-DASFA, Symp. on Advances in DB and Information Systems, Prague, 2001.
- [3] Kimball, R., Ross, M., "The Data Warehouse Toolkit: The Complete Guide to Dimensional Modeling (Second Edition)", Ed. J. Wiley & Sons, Inc, ISBN: 0471200247, 2002.
- [4] Sun Microsystems, "Data Warehousing Performance with SMP and MPP Architectures", White Paper, 1998.
- [5] Costa, M., Madeira, H., "Handling big dimensions in distributed data warehouses using the DWS technique", ACM Seventh Intl Workshop on Data Warehousing and OLAP, DOLAP 2004, Washington, D.C., USA, 2004.
- [6] Vieira, J., Bernardino, J., Madeira, H., "Efficient compression of text attributes of data warehouse dimensions", 7th Intl Conference on Data Warehousing and Knowledge Discovery - DaWak, Copenhagen, Denmark, 2005.
- [7] C. Cachin et al, "Reference model and use cases, MAFTIA deliverable D1", MAFTIA Project IST-1999-11583, 2000.
- [8] Vieira, M., Madeira, H., "Towards a security benchmark for Database Management Systems", Intl Conf. on Dependable Systems and Networks, DSN2005, Japan, 2005.