Modern Types of Databases for SIEM System Development

Sergiy Gnatyuk[1], Rat Berdibayev[2], Ivan Azarov[1], Nazerke Baisholan[3], and Iryna Lozova[1]

[1] National Aviation University, 1 Liubomyr Huzar ave, Kyiv, 03058, Ukraine
[2] Almaty University of Power Engineering and Telecommunication, 126/1 Baytursynuli str., Almaty, 050013, Kazakhstan
[3] Al-Farabi Kazakh National University, 71 al-Farabi ave, Almaty, 050040, Kazakhstan,

Abstract

Today SIEM systems are used to prevent information loss in computer systems and networks. There are many approaches to databases construction in SIEM. But there is no any information about most effective (optimal) approach or their comparison. From this viewpoint, the paper is devoted to the analysis of existing types of modern databases and their management systems used in SIEM systems, as well as comparative characteristic of their capabilities and differences, advantages and disadvantages. The analysis identified the optimal types of databases that are used in existed SIEM systems and meet most of the criteria of modern SIEM systems and have the greatest number of advantages. The actual types of databases for future development of new advanced SIEM systems are also given in this research study. It will be used in research project realization devoted to open source SIEM development.

Keywords

Database, DBMS SIEM, SQL, NoSQL, NewSQL, SIEM, cybersecurity, SIEM development, cyber threat.

1. Introduction

In the modern world, the number of cyber threats is constantly growing; in particular, SIEM systems are used to prevent information loss. For the development of a future SIEM system, various types of existing modern databases that are used in SIEM systems are considered. A database is an ordered collection of structured data that is stored electronically in a computer system. The database is controlled by a database management system (DBMS). Data together with the DBMS, as well as the applications that are associated with them, are called a database system or database. Data in modern types of databases is usually stored as columns and rows that form a table. This data can be easily managed, added, modified, deleted, updated, monitored and organized. Most databases use Structured Query Language (SQL) to write and query data.

2. Analysis of Modern Approaches and Problem Statement

There are many different types of databases [1–3]. Choosing the best database for a specific SIEM system depends on how the data will be used. Database types are templates and structures that are used to organize data in a database management system (DBMS) [1–5].

The purpose of the work is an analysis of the types of databases used by SIEM systems to define most effective and use it for future SIEM development.

Consider the following types of databases that use SIEM systems.

CPITS-II-2021: Cybersecurity Providing in Information and Telecommunication Systems, October 26, 2021, Kyiv, Ukraine
EMAIL: s.gnatyuk@nau.edu.ua (S. Gnatyuk); r.berdybaev@aues.kz (R. Berdibayev); azarovphone@gmail.com (I. Azarov); baisholan@gmail.com (N. Baisholan); ilozovaya@gmail.com (I. Lozova)
ORCID: 0000-0003-4992-0564 (S. Gnatyuk); 0000-0002-8341-9645 (R. Berdibayev); 0000-0002-6810-8152 (I. Azarov); 0000-0002-8134-0466 (N. Baisholan); 0000-0002-7224-4763 (I. Lozova)

3. The Simplest Types of Databases

Let's start with three types of databases that can still be found in specialized environments, but have largely been replaced by reliable and efficient alternatives.

3.1. Simple Data Structures

The first and simplest way to store data is text files. The method is still used today to work with small amounts of information. A special character is used to separate fields: a comma or semicolon in csv dataset files, a colon or space in * nix-like systems:

root:x:0:0:root:/root:/bin/bash
bin:x:2:2:bin:/bin:/usr/sbin/nologin
sys:x:3:3:sys:/dev:/usr/sbin/nologin
sync:x:4:65534:sync:/bin:/bin/sync
games:x:5:60:games:/usr/games:/usr/sbin/nologin

Consequences:

- the type and level of complexity of the stored information is limited;
- it is difficult to establish connections between data components;
- lack of concurrency features;
- practical only for systems with low read and write requirements;
- used to store configuration data;
- no need for third party software.

Examples:

- /etc/passwd and /etc/fstab on * nix systems;
- csv files.

3.2. Hierarchical Databases

In contrast to text tables, in the next type of database, links between objects appear. In hierarchical databases, each record has one parent. This creates a tree structure in which records are classified according to their relationship to the parent record chain, the structure of hierarchical databases is shown in Figure 1.

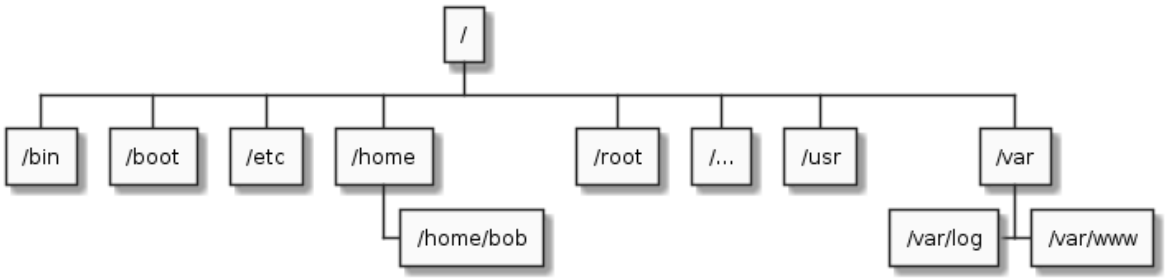

Figure 1: Structure of hierarchical databases

Consequences:

- information is organized in a tree structure with ancestor-child relationships;
- each record can have at most one parent;
- links between records are made in the form of physical pointers;
- it is impossible to implement a many-to-many relationship.

Examples:

- file systems;
- DNS;
- LDAP.

3.3. Network Databases

Networked databases extend the functionality of hierarchical databases: records can have more than one parent. This means that you can model complex relationships, the structure of network databases is shown in Fig. 2.

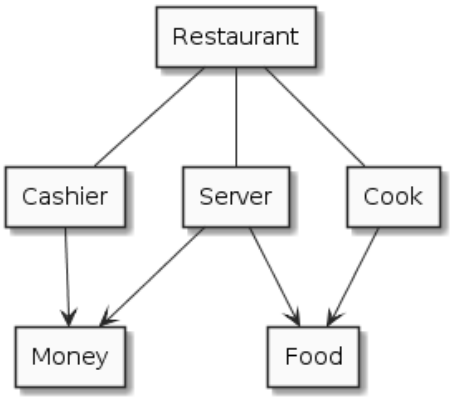

Figure 2: Structure of network databases

Consequences:

- information is organized in a tree-like structure with ancestor-child relationships;
- limited by the same access patterns as hierarchical databases.

Example:

- IDMS.

4. Relational Databases

4.1. SQL Databases

Relational databases are the oldest type of general purpose database still in widespread use. Data in a relational database is organized in tables, consisting of columns and rows. Each column in a table has a name and type. Each row represents a separate record or data element in the table, containing a value for each of the columns, the structure of relational databases is shown in Fig. 3.

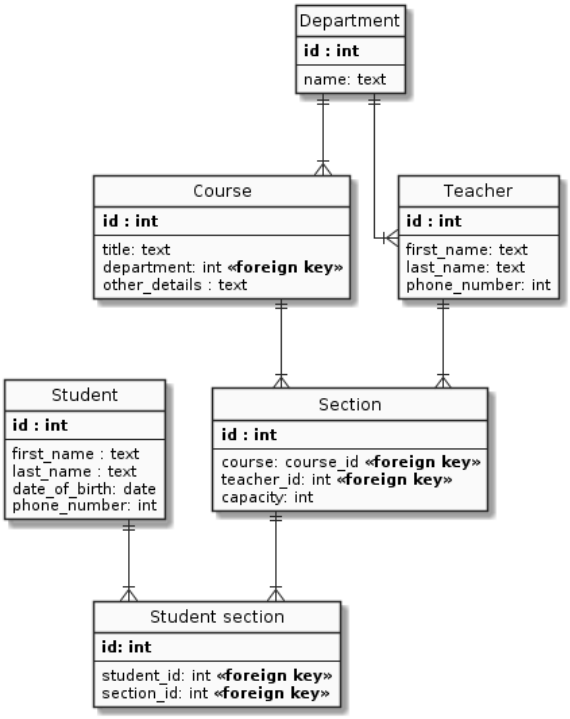

Figure 3: Structure of relational databases

Consequences:

- a field in a table, called a foreign key, can contain references to columns in other tables, which allows them to be joined;
- highly organized structure and flexibility makes relational databases powerful and adaptable to various types of data;
- structured query language (SQL) is used to access data;
- a reliable choice for many applications;
- fast and efficient access to structured information.

Examples:

- MySQL;
- MariaDB;
- PostgreSQL;
- SQLite;
- MSSQL;

4.2. OLTP Databases

An OLTP database is a database designed to perform business transactions performed by multiple users, the structure of OLTP databases is shown in Fig. 4.

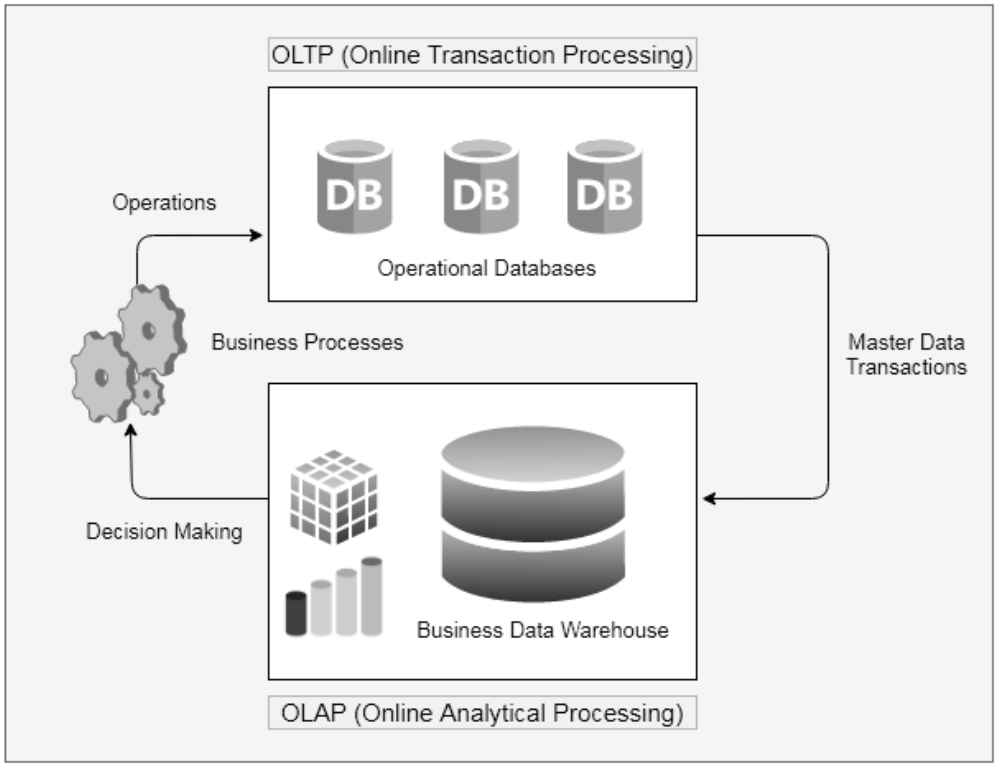

Figure 4: Structure of OLTP databases

Advantages

- High reliability and reliability of data as a result of the transactional approach.
- The transaction either completes completely and successfully, or fails and the system reverts to its previous state. For any outcome of the transaction, data integrity is not violated.

Disadvantage

- OLTP systems are optimized for small discrete transactions. But requests for some complex information (for example, quarterly dynamics of sales volumes for a certain product model in a certain branch), which are typical for analytical applications (OLAP), will generate complex table joins and whole tables view. One such request will take a lot of time and computer resources, which will slow down the processing of current transactions.

Relational databases use such SIEM systems: IBM QRadar, LOGRHYTHM, AlienVault USM, AlienVault OSSIM, Splunk, FortiSIEM, Wazuh, SolarWinds, ManageEngine, RuSIEM, Prelude OSS, Prelude SIEM, Sagan, Maxpatrol, EventTracker, Trustwave SIEM Enterprise, McAfee (ESM) [6-8].

5. NoSQL Databases

NoSQL is a group of database types that offer approaches other than the standard relational pattern. NoSQL refers to either “non-SQL” or “not just SQL” to clarify that sometimes a SQL-like query is allowed.

A NoSQL database, or non-relational database, provides the ability to store and process unstructured or semi-structured data (as opposed to a relational database, which defines the structure of the data it contains). The popularity of NoSQL databases is growing as web applications proliferate and become more complex.

5.1. Key-Value Databases

In key-value databases, to store information, you provide a key and a data object that you want to store. For example, JSON object, image or text. To request data, you send a key and receive a blob, the structure of NoSQL databases is shown in Fig. 5.

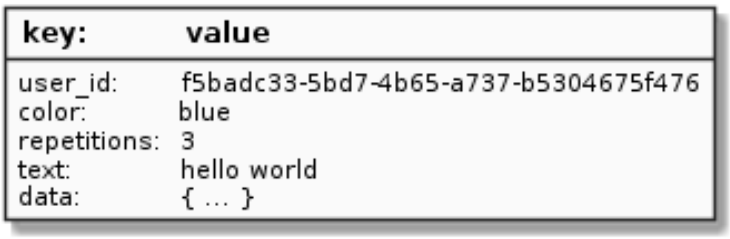

Figure 5: Structure of key-value databases

Consequences:

- storage facilities provide fast and low-cost access;
- often store configuration data and information about the state of data represented by dictionaries or hashes;
- there is no rigid scheme of the relationship between data, therefore, different types of data are often stored in such databases at the same time;
- it is the developer's responsibility to define the key naming scheme and ensure that the value is of the appropriate type / format.

Examples:

- Redis;
- memcached;
- etcd.

5.2. Document Database

Document databases (also document databases or document repositories) share the basic semantics of accessing and retrieving key and value stores. Such databases also use a key to uniquely identify data. The difference between key-value stores and document databases is that instead of storing blobs, document databases store data in structured formats – JSON, BSON or XML, the structure of document databases is shown in Fig. 6 [9–10].

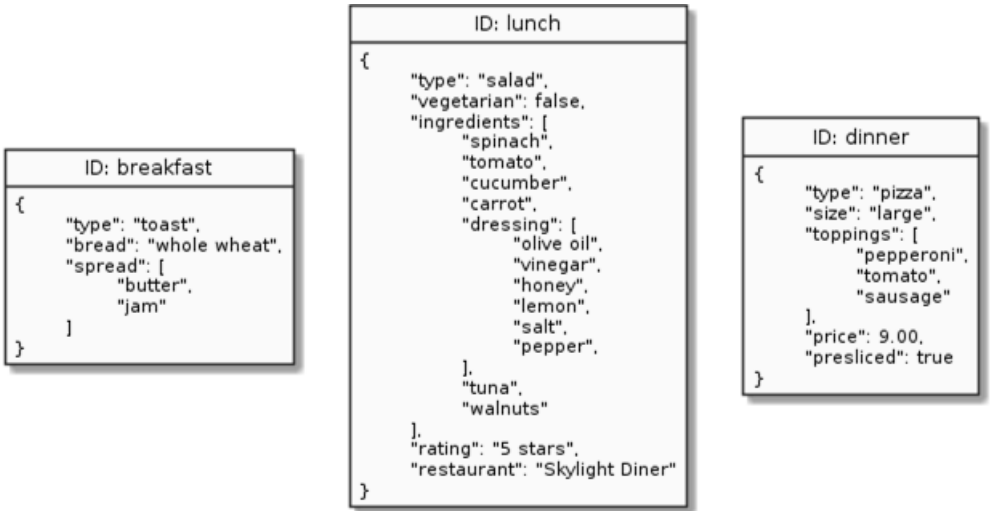

Figure 6: The structure of document databases

Consequences:

- the database does not prescribe a specific format or schema;
- each document can have its own internal structure;
- document databases are a good choice for rapid development;
- at any time, you can change the properties of the data without changing the structure or the data itself.

Examples:

- MongoDB;
- RethinkDB.

5.3. Graph Database

Instead of mapping relationships to tables and foreign keys, graph databases establish relationships using nodes, edges and properties, the structure of graph databases is shown in Fig. 7.

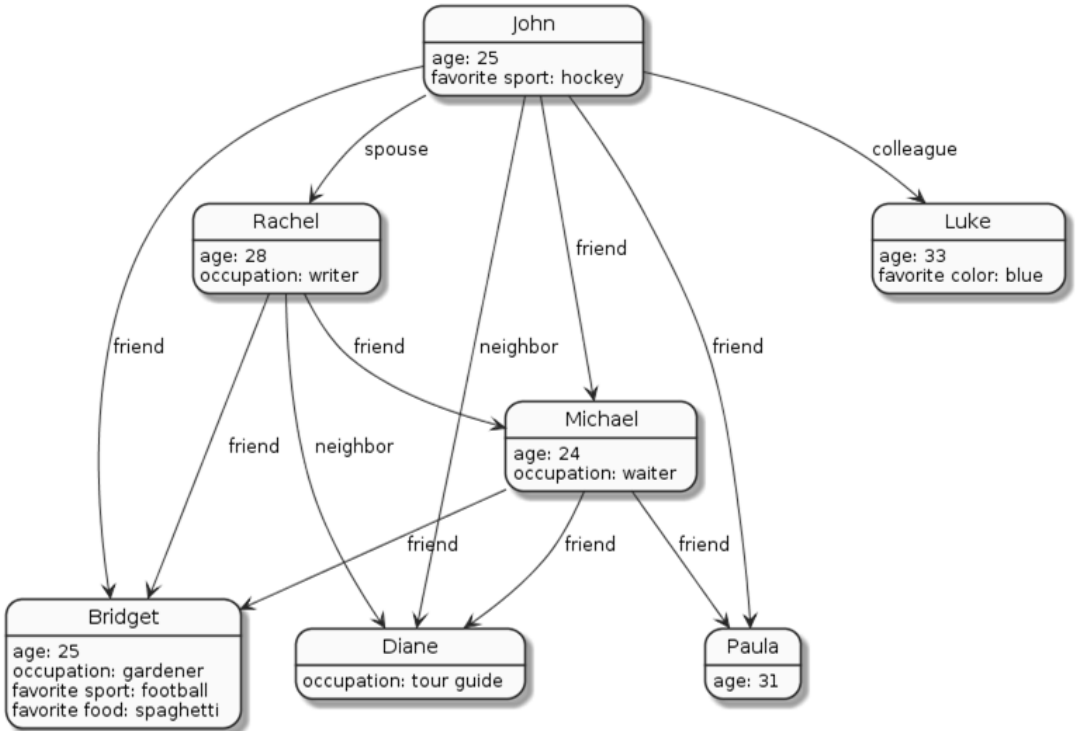

Figure 7: The structure of graph databases

Graph databases represent data as individual nodes, which can have any number of properties associated with them. A graph database stores data in the context of entities and relationships between entities.

Consequences:

- look similar to network;
- focus on relationships between elements;
- explicitly displays relationships between data types;
- do not require a step-by-step walk to move between elements;
- there are no restrictions on the types of links represented.

Examples:

- Neo4j;
- JanusGraph;
- Dgraph.

5.4. Columnar Databases

Columnar databases (also non-relational column stores or wide-column databases) belong to the NoSQL database family, but look similar to relational databases. Like relational databases, columnar databases store data using rows and columns, but with a different relationship between elements.

In relational databases, all rows must conform to a fixed schema. The schema determines what columns will be in the table, data types, and other criteria. Column bases have structures called "column families" instead of tables. Families contain strings, each of which defines its own format. A string consists of a unique identifier used for searching, followed by a set of column names and values, the structure of columnar databases is shown in Fig. 8.

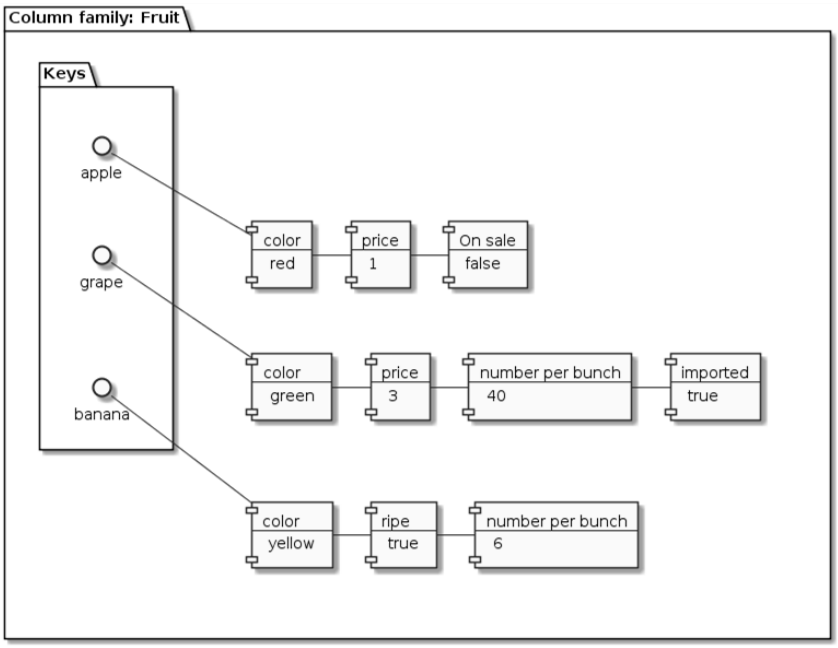

Figure 8: The structure of columnar databases

Consequences:

- DBs are convenient when working with applications that require high performance;
- record data and metadata are available by one identifier;
- it is guaranteed that all data from a row is placed in one cluster, which simplifies data segmentation and scaling.

Examples:

- Cassandra;
- HBase.

5.5. Time Series Databases

Time series databases are designed to collect and manage items that change over time. Most of these databases are organized into structures that record values for one element. For example, you can create a table to keep track of the CPU temperature. Inside, each value will consist of a time stamp and a temperature reading. There can be several metrics in the table; the structure of time series databases is shown in Fig. 9.

Time	CPU Temp	System Load	Memory Usage %
2019-10-31T03:48:05+00:00	37	0.85	92
2019-10-31T03:48:10+00:00	42	0.87	90
2019-10-31T03:48:15+00:00	33	0.74	87
2019-10-31T03:48:20+00:00	34	0.72	77
2019-10-31T03:48:25+00:00	40	0.88	81
2019-10-31T03:48:30+00:00	42	0.89	82
2019-10-31T03:48:35+00:00	41	0.88	82

Figure 9: Structure of time series databases

Consequences:

- write-oriented;
- are designed to process a constant stream of input data;
- performance depends on the number of items being tracked, the polling interval between writing new values, and the actual data payload.

Examples:

- OpenTSDB;
- Prometheus;
- InfluxDB;
- TimescaleDB.

NoSQL databases use following SIEM systems: AlienVault USM, AlienVault OSSIM, MozDef, Maxpatrol, SearchInform SIEM.

6. Combined Databases

NewSQL and multi-model databases are different types of databases, but solve one group of problems caused by polarized SQL approaches or NoSQL strategies. Why not combine the benefits of both groups?

6.1. NewSQL Databases

NewSQL databases inherit the relational structure and semantics, but are built using more modern, scalable constructs. The goal is to provide more scalability than relational databases and higher consistency guarantees than NoSQL. The trade-off between consistency and availability is the fundamental problem of distributed databases, described by the CAP theorem [11-12].

Consequences:

- the possibility of horizontal scaling;
- high availability;
- great performance and replication;
- small functionality and flexibility;
- considerable consumption of resources and the need for specialized knowledge to work with the database.

Examples:

- MemSQL;
- VoltDB;
- Spanner;

- Calvin;
- CockroachDB;
- FaunaDB;
- yugabyteDB.

6.2. Multi-Model Databases

Multi-model databases are databases that combine the functionality of several types of databases. The advantages of this approach are obvious - the same system can use different representations for different types of data.

The co-location of data from several types of databases in one system allows performing new operations that would otherwise be difficult or impossible. For example, multi-model databases can allow users to access and manage data stored in different types of databases within a single request, and also maintain data consistency when performing operations that change information in several systems at once [13].

Consequences:

- help to reduce the load on the DBMS;
- allow you to expand to new models as your needs change without changing the underlying infrastructure;
- provide continuous access and easy distribution of data;
- have linear scalability and are easy to develop.

Examples:

- ArangoDB;
- OrientDB;
- Couchbase.

7. Object-Oriented Databases (OODB)

Information in an object-oriented database is presented in the form of an object, as in object-oriented programming, the structure of OODB databases is shown in Fig. 10.

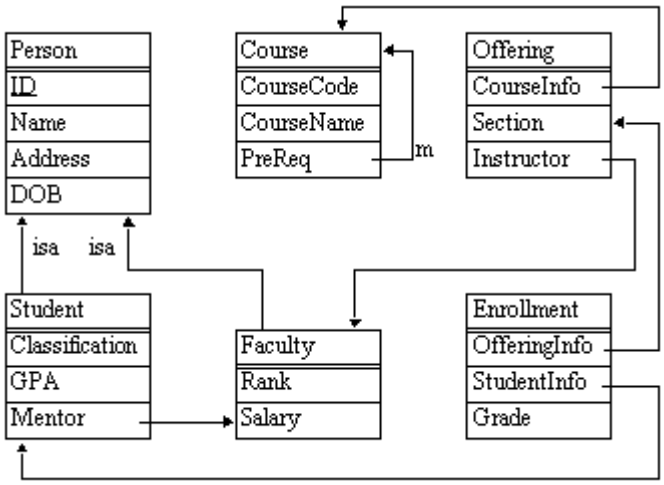

Figure 10: Structure of OODB databases

8. Cloud Databases

A cloud database is a collection of structured or unstructured data hosted on a private, public, or hybrid cloud computing platform [14–17]. There are two types of cloud database models: traditional database and database as a service (DBaaS). In the DBaaS model, administrative tasks and maintenance are performed by the cloud provider, the structure of the cloud databases [18–20] is shown in Fig. 10.

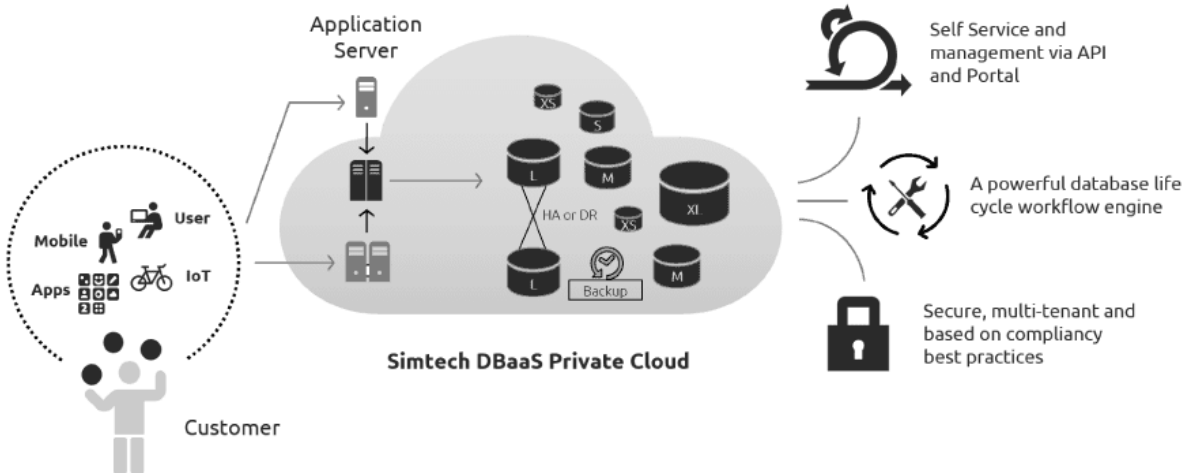

Figure 11: The structure of cloud databases

Cloud database types use such SIEM systems: HPE ArcSight Splunk Ixia ThreatARMOR, Micro Focus ArcSight, Trustwave SIEM Enterprise.

Results of DBMS analysis in various SIEM is shown in Table 1.

Table 1

DBMS used in various SIEM

SIEM	DBMS
IBM QRadar	Ariel database, PostgreSQL, SQLite
LOGRHYTHM	Oracle, SQL Server, MySQL
HPE ArcSight	Own development CORR-E
Splunk	DB2 / Linux, Informix, MemSQL, MySQL, AWS Aurora, Microsoft SQL Server, Oracle, PostgreSQL, AWS RedShift, SAP SQL Anywhere, Sybase ASE, Sybase IQ, and Teradata
McAfee (ESM)	MSSQL, Oracle, MySQL, Data Access Server (DAS), DB2 / UDB
AlienVault USM	RedisDB, MySQL
AlienVault OSSIM	RedisDB, MySQL
FortiSIEM	PostgreSQL
Ixia ThreatARMOR	Rap Sheet
MozDef	RabbitMQ, MongoDB, Elasticsearch, Kibana
Wazuh	MySQL, PostgreSQL
Prelude OSS	MySQL, PostgreSQL
Prelude SIEM	MySQL, PostgreSQL
Sagan	MySQL / PostgreSQL
Maxpatrol	ElasticSearch, MongoDB, MS SQL Express
SolarWinds	MSSQL, Oracle, MySQL, MariaDB.
ManageEngine	Oracle, SQL, DB2, & MySQL
EventTracker	Microsoft SQL Server
Micro Focus ArcSight	Own development CORR-E
Trustwave SIEM Enterprise	Microsoft SQL Server, Microsoft SQL Azure, ORACLE, SYBASE, MySQL, IBM, DB2, Hadoop
BlackStratus SIEMStorm	Own development
SearchInform SIEM	MongoDB
RuSIEM	MySQL / Oracle / MS SQL

9. Conclusion

The large number of data types that are stored, speed and performance requirements have led to the expansion of database types. At the same time, each of them continues to be in demand in its niche, where relationships between data are associated with a certain database structure scheme.

To select databases when creating SIEM systems, it is necessary to take into account the convenience of storage, the speed of obtaining and using data. It is necessary to provide integration with other system modules and external API to provide database support for most DPI systems (both software and hardware).

It is recommended to use links of several (hybrid types) databases such as SQL and NoSQL, which will allow you to preserve the convenience of storing data and their classification, as well as high speed of obtaining large amounts of information due to preliminary indexing.

These recommendations will be used in research project realization devoted to SIEM development.

10. Acknowledgement

This work is carried out within the framework of research grant №AP06851243 "Methods, models and tools for security events and incidents management for detecting and preventing cyber attacks on critical infrastructures of digital economics" (2020-2022), funded by the Ministry of Digital Development, Innovation and Aerospace Industry of the Republic of Kazakhstan.

11. References

[1] Vielberth M. and Pernul G. "A Security Information and Event Management Pattern". 12th Latin American Conference on Pattern Languages of Programs (SugarLoafPLoP 2018), November 2018, 12 p. 27.

[2] Agrawal K., Makwana H. "A Study on Critical Capabilities for Security Information and Event Management". International Journal of Science and Research (IJSR). Vol. 4 Issue 7, July 2015 Rock, pp. 1893-1896.

[3] Henrik Karlzén, "An Analysis of Security Information and Event Management Systems". Department of Computer Science and Engineering Chalmers University of Technology University of Gothenburg, Göteborg, Sweden, January 2009. Available on: http://publications.lib.chalmers.se/records/fulltext/89572.pdf

[4] Ribolovlev D., Karasov S., Polyakov S. "Classification of emergency management systems for incidents without baking". Food of cyber security, №3 (27), 2018, pp. 47-53.

[5] Ariel Query Language Guide, IBM QRadar 7.3.3 (2013 and 2019). Available on: https://www.ibm.com/docs/en/SS42VS_7.3.3/com.ibm.qradar.doc/b_qradar_aql.pdf

[6] SIEM Analytcis: http://www.siem.su/compare_SIEM_systems.php

[7] J. Lee, Y. Kim, J. Kim and I. Kim, "Toward the SIEM architecture for cloud-based security services," 2017 IEEE Conference on Communications and Network Security (CNS), Las Vegas, NV 2017, pp. 398-399, doi: 10.1109 / CNS.2017.8228696.

[8] I. Bachane, Y. I. K. Adsi and H. C. Adsi, "Real time monitoring of security events for forensic purposes in Cloud environments using SIEM," 2016 Third International Conference on Systems of Collaboration (SysCo), 2016, pp. 1-3, doi: 10.1109/SYSCO.2016.7831327.

[9] B. AlSabbagh and S. Kowalski, "A Framework and Prototype for A Socio-Technical Security Information and Event Management System (ST-SIEM)," 2016 European Intelligence and Security Informatics Conference (EISIC), 2016, pp. 192-195, doi: 10.1109/EISIC.2016.049.

[10] A. Serckumecka, I. Medeiros and A. Bessani, "Low-Cost Serverless SIEM in the Cloud," 2019 38th Symposium on Reliable Distributed Systems (SRDS), 2019, pp. 381-3811, doi: 10.1109/SRDS47363.2019.00057.

[11] M. Nabil, S. Soukainat, A. Lakbabi and O. Ghizlane, "SIEM selection criteria for an efficient contextual security," 2017 International Symposium on Networks, Computers and Communications (ISNCC), 2017, pp. 1-6, doi: 10.1109/ISNCC.2017.8072035.

[12] R.-V. Mahmoud, E. Kidmose, A. Turkmen, O. Pilawka, J.M. Pedersen, “DefAtt - Architecture of Virtual Cyber Labs for Research and Education”, 2021 International Conference on Cyber Situational Awareness Data Analytics and Assessment (CyberSA), pp. 1-7, 2021.
[13] Yu. Danik, R. Hryschuk, S. Gnatyuk, “Synergistic effects of information and cybernetic interaction in civil aviation”, Aviation, Vol. 20, №3, pp. 137-144, 2016.
[14] Berdibayev R., Gnatyuk S., Yevchenko Yu., Kishchenko V. “A concept of the architecture and creation for SIEM system in critical infrastructure”, Studies in Systems, Decision and Control, Vol. 346, 2021, pp. 221-242.
[15] O. Oksiiuk, V. Chaikovska and A. Fesenko, “Security Technique for Authentication Process in the Cloud Environment,” 2019 IEEE International Scientific-Practical Conference Problems of Infocommunications, Science and Technology (PIC S&T), 2019, pp. 379-382, doi: 10.1109/PICST47496.2019.9061248.
[16] Gnatyuk S., Berdibayev R., Avkurova Z., Verkhovets O., Bauyrzhan M. “Studies on cloud-based cyber incidents detection and identification in critical infrastructure”, CEUR Workshop Proceedings, 2021, Vol. 2923, pp. 68-80.
[17] J. Lee, Y. S. Kim, J. H. Kim and I. K. Kim, “Toward the SIEM architecture for cloud-based security services,” 2017 IEEE Conference on Communications and Network Security (CNS), 2017, pp. 398-399, doi: 10.1109/CNS.2017.8228696.
[18] V. Buriachok, et al., Invasion Detection Model using Two-Stage Criterion of Detection of Network Anomalies, Cybersecurity Providing in Information and Telecommunication Systems (CPITS), pp. 23–32, Jul. 2020.
[19] Lukova-Chuiko N., Fesenko A., Papirna H. and Gnatyuk S. “Threat hunting as a method of protection against cyber threats”, CEUR Workshop Proceedings, Vol. 2833, pp. 103-113, 2021.
[20] Astapenya V., Buriachok V., Sokolov V., Skladannyi P. and Ageyev D. “Last mile technique for wireless delivery system using an accelerating lens”, Proceedings of 2020 IEEE International Conference on Problems of Infocommunications Science and Technology, PIC S and T 2020, pp. 811-814, 2021. doi:10.1109/PICST51311.2020.946788.