UDC 004.415.2.031.43

APPROACH TO THE DESIGN PATTERNS OF MULTICOMPONENT SYSTEM FOR DETECTING NETWORK ATTACKS

PAVEL ZHURAUKOU, EUGENIY SUKCHAREV
Polotsk State University, Belarus

The article touches upon the concepts of design services for collection and processing of network packets received by corporate systems, along with the principles of modern web applications development.

Introduction. Due to the fact that new network technologies penetrate into various spheres of human activities the questions regarding protection of computer networks from outside interference are becoming more relevant nowadays. The leading world IT-companies are developing various solutions for protection of cloud-based computing. The relevance of such investigations comes from two factors: firstly the modern features to connect computing resources to the Internet networks and secondly cloud services users, who pose a real threat to security. In this article two components of Intrusion Detection System will be considered, such as agent that collects and processes network packets (that is a Windows Service) that works on corporate system nodes and web application client.

Agents for collecting and processing network packets. The main purpose of the agent is to collect and analyze the quantity, type, IP address, from which the packet was sent, as well as the size of the received and transmitted packets to the host. The developed component should be implemented as a Windows Service, that is automatically started after loading the operating system.

The fundamental task of the developed service is 24/7 uninterrupted operation of the main components responsible for data collection and analysis. Otherwise, the system will stop to analyze the data, which may result in serious losses for the company, containing the attacked site. According to the statistics from Kaspersky Lab covering 2015 year, loss of the companies affected by network attacks based on unlimited amounts of targeted system resources amounts to approximately $400 thousand. On average, 61% of victims temporarily lost access to very important information, 38% were unable to perform usual actions, and 33% of companies reported about transactions and contracts losses. Due to these facts, analysis and monitoring of systems is required to be available at any time.

To achieve this goal it is necessary to consider and implement the mechanism that checks whether the agent is working all the time. Based on one of the items in the settings, the service should send a similar to ping query to the database, thereby informing that the DB is functioning correctly [2].

It is necessary to develop the infrastructure that would allow to continue the work of a subsystem even after an error in it. It is necessary to take a Task based approach when a separate Task is created for a particular event in the operating system as a basis for such a system. Tasks in the operating system allow to abstract from the current sequential execution of the program. A unit of work is being executed in a separate thread that allows to move away from the main context of the program. The main advantage of this solution in this situation is that if there is a runtime thread error, it will not affect the operation of the rest of the system. The system will continue working as if nothing has happened.

In the situation when any subsystem service has failed (an error occurred in the program), the same subsystem must correctly handle the error that occurred and continue to work correctly. One of the design patterns of programming with threads is the realization of the so-called task managers that are responsible for creation, running and error handling in these tasks. Task manager takes over the whole process of creating a task automatically wrapping the unit of work in the thread. This implementation can significantly reduce the amount of code, as well as to improve its readability. After the thread is created, it has to be run in a separate thread. The manager automatically launches the task after it is created. Finally, if the runtime thread error occurs, manager handles this error to prevent the crash of the entire system [4].

One of the aims for designing an agent is the ability to configure the service remotely. This functionality must be implemented as a part of the client web application. Moreover there is one more requirement. That is imperceptible change of the current settings of the agent while it is running. The service should automatically load the required settings from the database and apply them without breaking or stopping its work. This functionality must be implemented using cache mechanism that has to check the current settings over some period of time. The main criterion for choosing this approach is to reduce the load on the database.

As far as the requirements are concerned, the main task of the agent is to collect data for later analysis. The unit of data transmitted over the network is a packet network i.e. a block of data which is formed in a certain way. The main task of the attacker who wants to disrupt any node in the system is a multiple increase in the transmitted packet node, so the node is not able to cope with the large amount of information. As a result, legal users cannot partially or fully use the corporate system. Collecting information about the number of packets, as well as their content is the paramount task.

Capturing of packets must be implemented on the network card driver level, which uses the NDIS packet to read what the network card receives. There is the Pcap library that has all the low-level logic to work with these drivers implemented. It allows programmers to avoid the work directly with the driver. The mentioned above library is written in the C programming language. It is not always convenient when working with higher-level languages, such as Java and C#. Therefore it is good to use the wrapper Pcap on the SharpPcap language. SharpPcap implements all the functionality of the Pcap library, complementing it with the specific to the C# language features namely an event processing model of incoming packets. It is also needed to consider the size and type of packages while developing. Low-level library Pcap implements the processing of the following packet types: ARP, IPv4, IPv6, ICMP, ICMPv6, IGMP, UDP, TCP. All of the mentioned packages types should collected and counted by the developed system. For this purpose, a separate thread (task) performs quantification of each packet, as well as its size and the type in real time [4].

Filter subsystem should filter all the packages types. At the design stage, the concept of reuse of filters should be considered. The so-called reusable filters that can be applied to different devices in different network nodes in the enterprise system will allow to control the process of filter setting process. On the basis of the cache mechanism of additional upload settings agent, it is needed to implement cached upload of filters from the database to the network devices on the node where the agent is running.

Web Application. The main purpose of the client Web part is the ability to configure the agents as well as display information on their current load. Setting agents includes displaying information about the work at the sites of agents, output of agent errors allowing information security professionals to monitor the correctness of the agents, output of operations of different filter units. Besides, web application must show the real time data collected by the collection and processing agents, as well as to build a clear schedule.

Web application should be developed using the MVC pattern. MVC is a design pattern where application model, user interface and interaction with the user are divided into three separate components so that the change of one of the components brings about a minimal effect on the work of the other. The controller captures the event from the outside and in accordance with the logic it responds to this event with changing the model by calling the appropriate action. After that the model uses the event that it has changed, and when all the relevant to this event updates become available, they turn to the model and update the data that is displayed after that. Thus, adhering to this approach throughout the software development cycle makes the code scalable and easy to read. The scheme of the MVC approach is shown in Figure 1.

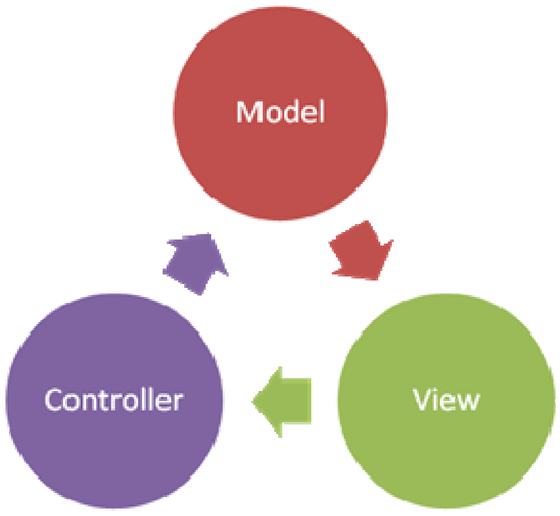

Fig. 1. The scheme of the MVC design pattern

Let's consider the implementation of the user interface. The total summary statistics for all the nodes should be displayed on the the main page. The diagrams with load nodes, packet types percentage, the number of packets on all nodes and the traffic amount should be displayed on the page for clearer perception. The model of the described page is displayed in Figure 2.

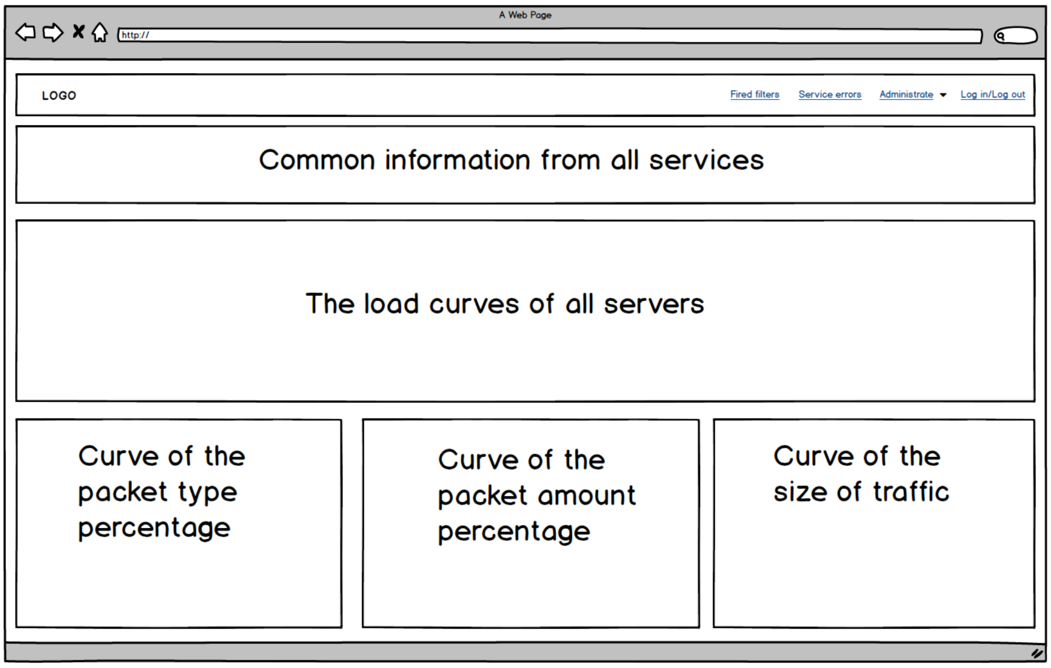
A Web Page
http://
LOGO
Fired filters
Service errors
Administrate
Log in/Log out
Common information from all services
The load curves of all servers
Curve of the packet type percentage
Curve of the packet amount percentage
Curve of the size of traffic

Fig. 2. The mock of the Web application home page

REFERENCES

1. Система обнаружения вторжений – Викиучебник [Электронный ресурс] / Wikimedia Foundation Inc.. – 2017. – Режим доступа: https://ru.wikibooks.org/wiki/Система_обнаружения_вторжений. – Дата доступа: 23.01.2017.
2. ping – Викиучебник [Электронный ресурс] / Wikimedia Foundation Inc. – 2017. – Режим доступа: https://ru.wikibooks.org/wiki/Ping. – Дата доступа: 20.01.2017.
3. Обработка исключений (библиотека параллельных задач) – MSDN – сеть разработчиков Microsoft [Электронный ресурс]. – 2017. – Режим доступа: https://msdn.microsoft.com/ru-ru/library/dd321409(v=vs.110).aspx. – Дата доступа: 21.01.2017.
4. Pcap – Викиучебник / Wikimedia Foundation Inc. [Электронный ресурс]. – 2017. – Режим доступа: https://ru.wikibooks.org/wiki/Pcap – Дата доступа: 21.01.2017.
5. Model-View-Controller – Викиучебник / Wikimedia Foundation Inc. [Электронный ресурс]. – 2017. – Режим доступа: https://ru.wikibooks.org/wiki/Model-View-Controller. – Дата доступа: 21.01.2017.