БГТУ и Дню белорусской науки (с международным участием), Минск, 03–14 февраля 2020 г. – Минск: БГТУ, 2020. – С. 10–14.

УДК 003.26+004.9

Ja. Wilk [1], P.P. Urbanovich[2]
[1]The John Paul II Catholic University of Lublin, Poland
[2]Belarusian State Technological University, Minsk, Belarus

USERS DATA PROTECTION ON THE BASIS OF A WEB APPLICATION FOR STATUS MONITORING OF COMPUTER SERVICES

Abstract. *The article (using a practical example) analyzes the main features of the choice of technologies and design tools, which, in combination with the implemented encryption and hashing tools, can provide comprehensive protection of user data based on a web application.*

Я. Вильк[1], П.П. Урбанович[2]
[1]Люблинский Католический университет Яна Павла II
Люблин, Польша
[2]Белорусский государственный технологический университет
Минск, Беларусь

ЗАЩИТА ДАННЫХ ПОЛЬЗОВАТЕЛЕЙ НА ОСНОВЕ ВЕБ-ПРИЛОЖЕНИЯ ДЛЯ МОНИТОРИНГА СТАТУСА КОМПЬЮТЕРНЫХ СЕРВИСОВ

Аннотация*. В статье (на практическом примере) проанализированы основные особенности выбора технологий и средств проектирования, которые в сочетании с реализованными средствами шифрования и хеширования могут обеспечить комплексную защиту пользовательских данных на основе веб-приложения.*

Nowadays, the Internet has become quite a specific and amazing place that allows to connect tens or even hundreds of thousands of people around the world. However, for this reason, it has also become a place that is hard to control and huge steps should be taken to increase the security of the users themselves and the data sent over the network [1, 2].

Small percentage of Internet users realize that simply visiting a specific website, downloading of a file or even reading messages from an e-mail box can actually allow a criminal to access user's private data. It is true that there are dedicated anti-virus software to prevent this from happening, but most of them are based on one and the same pattern. It involves scanning the suspicious item against a publicly available virus database and catching similarities. On the other hand, the available security in the operating system of the user's device, for example in the form of a firewall, as in the case of Windows, should be sufficient when it comes to simple office activities.

Methods of securing against attacks by cybercriminals do not guarantee complete security and freedom of using the network [2, 3]. The Internet is a huge medium which, due to its independence, freedom, but also mechanisms of operation, it is difficult or even impossible to fully supervise and censor it in the event of inconvenient data circulating on the Internet. Therefore, a very important factor influencing security when using the Internet is, above all, awareness, because without it even the best anti-virus software may turn out to be completely useless.

Cybercrime evolves depending on the available technology, and the actions of the experts in the trade make it that they are getting bolder and harder to detect. Currently, the most popular threats on the Internet include:

- websites and messages phishing for private data,
- fake software in the form of keyloggers or scripts,
- fake websites without appropriate certificates confirming their identity,
- no reliability and secure transfer via SSL.

However, taking into account the fact, that everyone using the network is an Internet user, it is in their interest to adequately secure not only access to content, but also communication in the event of their own provision of services. This article focuses on some aspects of securing of own websites and web applications.

Our application is designed to present and describe the basic processes accompanying the security aspect, on the example of a working website. The following technologies have been used for this purpose:

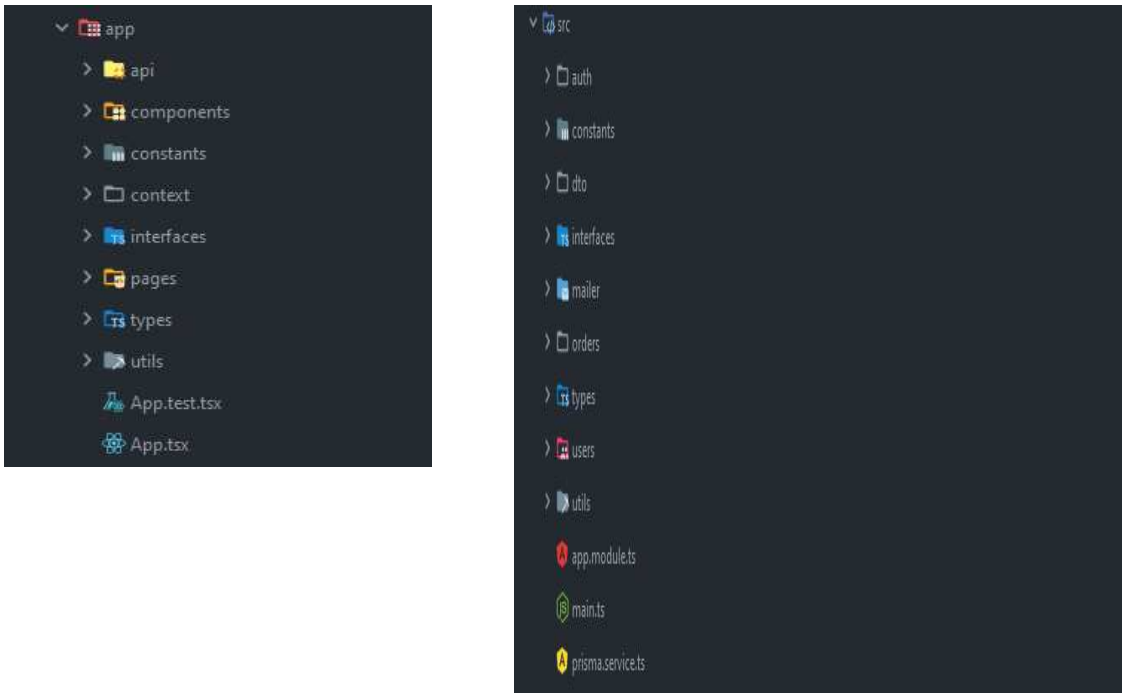
- React v17.0.2 – library dedicated to building views using JavaScript language,

- Script v4.1.2 and SCSS preprocessor – technologies that do not have much in common with each other, but are an extension of the existing ones, which are the backbone of every Internet application,

- NestJS v7.6.13 – framework for creating Back-Ends using JavaScript code in NodeJS technology. It has full support for the aforementioned TypeScript, and also includes a huge number of built-in tools that will help you create the right architecture for the server part of the application in an easy and pleasant way,

- Prisma v2.28.0 – the last of the most important pieces of the puzzle. It is a specially designed ORM (Object-Relational Mapping) that enables communication between the server layer and the database.

The application uses the popular design architecture that allows easily grouping relevant components, searching for them and providing a clear hierarchy. In a way, it is a pattern borrowed from such a framework as Angular, and also partially used in MVC projects written in C# or ASP.NET. Fig. 1 shows the catalog structure of the project, which allows for quick modification, adding new functionalities, as well as managing the existing ones. Appropriate grouping of individual code fragments makes it easier to select the appropriate component, and thus – corrections making.



a) b)

Fig.1 – Project architecture: a) Front-End, b) Back-End

Fig. 1 shows a domain architecture, where each directory corresponds to one specific functionality, both complex and simple. In the context of the

security aspect, it is worth emphasizing that *pages* – it contains components that can be used to build entire subpages; an example of such a view is the login page, registration page or the panel for the administrator; *auth* – the directory responsible for the authentication functionality of and user authentication. It has, among other things, watchdog functions that are used in controllers, as well as all logic related to logging in and registration of users.

The process of user authentication to specific parts of the application can take place in various ways, using several different external solutions or one, consistent, inaccessible to the outside system for authorizing people using the application. In the case of the described project, a mechanism called JWT (JSON Web Token) was used.

Authorization tokens have been used for a long time and are most often found in systems based on REST (REpresentational State Transfer). Everything is based on encrypted data, which in turn is hidden in the appropriate string of characters. It is random and the parts that store specific information are separated by dots. It is a standard that cannot be changed to maintain integrity between different, independent systems. In the case of the described project, this mechanism was used in two places: authentication and authorization.

JSON Web Token is saved in the form of a cookie, which in turn is located in the browser's memory and will function until its expiry date disappears or the browser data is cleared. What is very important when designing systems using tokens as one of the main mechanisms of authorization and authentication of users is the fact that there is a many-to-one relationship between the user and the person using the application [4]. This means that one user may have several tokens assigned to himself due to the use of different browsers, and as you know, the data between them is not, anyway, exported and imported by default.

The application also uses an element that allows you to confirm the creation of an account on the website. A special message is sent to the user's inbox, which includes a link to which the authorization token has been added directly. Thanks to this, after executing the *GET* query, this string will be read by the server and the relevant data will be extracted to start the verification process. In this case, the default HMAC SHA-256 hash function provided by the NestJS framework was used, along with a 50-bit secret key which is necessary for the encryption and verification process. It is a standard, and at the same time very hard to break, hash function, which is used in almost all ready-made systems. The declaration of the corresponding module takes literally a few lines (Fig. 2), and this is because most of the default settings provided by

NestJS are sufficient to run a working and secure server for web applications. However, the *JwtModule.register ()* function has many more settings that you can use for your own needs at any time.

```
@Module( metadata: {
    imports: [JwtModule.register( options: { secret: process.env['JWT_SECRET'] })],
    providers: [AuthService, PrismaService, MailerService],
    exports: [AuthService],
    controllers: [AuthController],
})
export class AuthModule {}
```

Fig.2 – The method of declaration of JSON Web Token of the module in the described application

The application sectors cannot be compared to micro-services that have become very popular in recent years. This is due to the fact that, unlike micro-services, domains are not completely independent of each other, functions are performed inside them, the declaration of which is found in a completely different part of the project.

The application is divided into two roles. One of them is the role of the user who has access to specific views and limited permissions, and the other is the role of employees, which includes both regular employees and supervisors. The second role also has access to a limited number of views, while their rights are no greater than in the case of clients.

Applications written in JavaScript, as well as the project described in this paper, have two types of protection against possible errors. The first one is the use of the TypeScript (TS) language itself, which, as mentioned earlier, adds the possibility of typing JavaScript components, and also opens the way to a completely different way of creating applications in this language. In addition, a huge number of Front-End tools support the native operation of the TS language, making it possible to use various tools that can be combined. In this work, the TS language ensures the integrity of structures, which already at the security stage allows to define how the containers for data that are transferred between components or between the client and the server are defined. Thanks to this, it is impossible to assign those values that do not match a given structure type and avoid errors at the very stage of writing the code, in which instead of, for example, the age of the user, which is a numerical value and only such, is assigned a string of characters, which means that data consistency is disrupted.

In addition, the security of data flow, from the side of the running application, is supervised by validation rules for each form to which the user has access. Regardless of your role, the Front-End part does not allow you to send data if they do not meet certain criteria. The project uses the Yup library, which allows you to create extensive validation rules for almost any type of field in the form.

References

1. Ochrona informacji w sieciach komputerowych / pod red. prof. P. Urbanowicza. – Lublin: Wydawnictwo KUL, 2004. – 150 s.
2. Mitnik, K. Niewidzialny w Sieci. Sztuka zacierania śladów / K. Mitnik. – Wydawnictwo Pascal, 2017. – 528 s.
3. Bentkowski, M. A. Bezpieczeństwo aplikacji webowych / M. A. Bentkowski [and etc.]. – Kraków, 2019.
4. Urbanowicz, P. Bazy danych: teoria i praktyka / P. Urbanowicz, M. Płonkowski, D. Urbanowicz. – Lublin: Wydawnictwo KUL, 2010. – 382 s.

УДК 621.3

А.А. Ференец
КНИТУ-КАИ, Казань, Россия

ПРОБЛЕМЫ ПЕРЕХОДА К ДЕЦЕНТРАЛИЗОВАННОЙ ЭНЕРГОСИСТЕМЕ

Аннотация. *В последние несколько лет основная повестка мировых форумов и конференций – это борьба с изменением климата, что влияет на фокус энергополитики государств. Экологичность энергодобычи и энергопотребления становится ключевым показателем энергосистемы, который возможно обеспечивать засчет возобновляемых источников энергии, переход на которые ведет к децентрализации энергосистемы.*

A.A. Ferenets
KNRTU-KAI, Kazan, Russia

PROBLEMS OF TRANSITION TO A DECENTRALIZED ENERGY SYSTEM