ҚАЗАҚСТАН РЕСПУБЛИКАСЫ ҒЫЛЫМ ЖӘНЕ ЖОҒАРЫ БІЛІМ МИНИСТРЛІГІ

«Л.Н. ГУМИЛЕВ АТЫНДАҒЫ ЕУРАЗИЯ ҰЛТТЫҚ УНИВЕРСИТЕТІ» КЕАҚ

Студенттер мен жас ғалымдардың
«ǴYLYM JÁNE BILIM - 2023»
XVIII Халықаралық ғылыми конференциясының
БАЯНДАМАЛАР ЖИНАҒЫ

СБОРНИК МАТЕРИАЛОВ
XVIII Международной научной конференции
студентов и молодых ученых
«ǴYLYM JÁNE BILIM - 2023»

PROCEEDINGS
of the XVIII International Scientific Conference
for students and young scholars
«ǴYLYM JÁNE BILIM - 2023»

2023
Астана

УДК 001+37
ББК 72+74
G99

«ǴYLYM JÁNE BILIM – 2023» студенттер мен жас ғалымдардың XVIII Халықаралық ғылыми конференциясы = XVIII Международная научная конференция студентов и молодых ученых «ǴYLYM JÁNE BILIM – 2023» = The XVIII International Scientific Conference for students and young scholars «ǴYLYM JÁNE BILIM – 2023». – Астана: – 6865 б. - қазақша, орысша, ағылшынша.

ISBN 978-601-337-871-8

Жинаққа студенттердің, магистранттардың, докоранттардың және жас ғалымдардың жаратылыстану-техникалық және гуманитарлық ғылымдардың өзекті мәселелері бойынша баяндамалары енгізілген.

The proceedings are the papers of students, undergraduates, doctoral students and young researchers on topical issues of natural and technical sciences and humanities.

В сборник вошли доклады студентов, магистрантов, докторантов и молодых ученых по актуальным вопросам естественно-технических и гуманитарных наук.

УДК 001+37
ББК 72+74

ISBN 978-601-337-871-8

Қолданылған әдебиеттер тізімі

1. *"The Web Application Hacker's Handbook: Finding and Exploiting Security Flaws" авторлары Даффиц (Dafydd Stuttard) және Маркус (Marcus Pinto).*
2. Безопасность приложений — Википедия (wikipedia.org)
3. Web Technologies of the Year 2021 (w3techs.com)
4. [All levels] DVWA Cross Site Request Forgery (CSRF) - YouTube
5. Мұнда кітіп көрсетіңіз диаграма алынған.

ӘОЖ 004

WEBSOCKET ПРОТОКОЛЫН ТАЛДАУ ЖӘНЕ ОНДАҒЫ ОСАЛДЫҚТАР

Есенпулов Айбек Бекзатұлы
aib.esen.02@bk.ru
Л.Н.Гумилев атындағы ЕҰУ Ақпараттық қауіпсіздік жүйелері мамандығының студенті,
Астана, Қазақстан
Ғылыми жетекші – Е.Қайупов

WebSocket - бұл браузер мен веб-сервер арасында тұрақты байланысты пайдаланып, хабар алмасуға арналған TCP қосылымының үстіндегі байланыс протоколы. WebSocket әрбір онлайн чаттар мен онлайн-таблицаларды құруда және басқа да нақты уақыттағы хабар алмасу (в режиме реального времени) қажет IT-өнімдер мен проектілерде кеңінен қолданылады.

WebSocket - ті пайдаланатын кейбір танымал қосымшалардың мысалдары:

WhatsApp: WhatsApp серверлері мен пайдаланушының құрылғысында орнатылған клиенттік қолданба арасында нақты уақыттағы хабар алмасу үшін WebSockets пайдаланады.

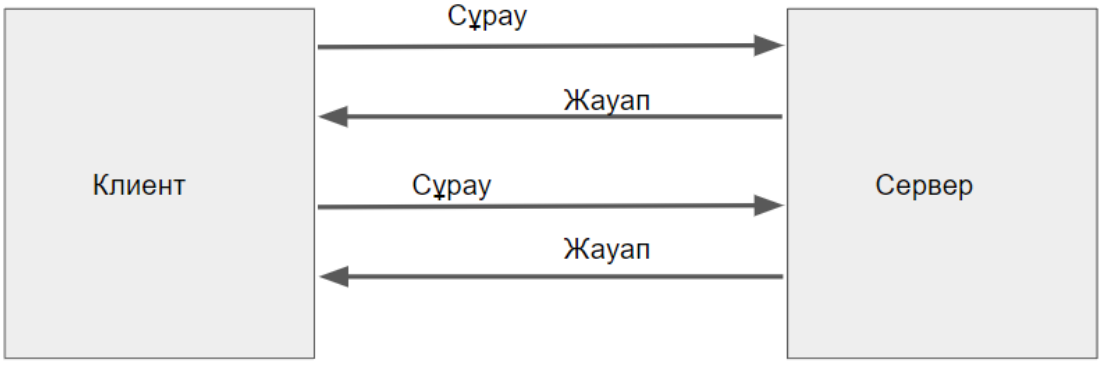
Slack: Slack пайдаланушыларына нақты уақыттағы хабар алмасу және бірлесіп жұмыс істеу мүмкіндіктерін беру үшін WebSockets пайдаланады.

Trello: Trello тақталарды, карталарды және хабарландыруларды нақты уақытта жаңарту үшін WebSockets пайдаланады.

Twitter: Twitter өзінің пайдаланушыларына твиттер, ретвиттер, ұнатулар және басқа әрекеттер үшін нақты уақытта жаңартуларды қамтамасыз ету үшін WebSockets пайдаланады.

Asana, Uber, Netflix, Twitch және тағы басқа да өзіміз пайдаланып жүрген қосымшалар.

WebSocket протоколы қалай жұмыс жасайтынын түсіну үшін, алдымен HTTP арқылы жүзеге асатын байланысқа назар аударайын.



Сурет 1 - HTTP арқылы клиент-серверлік байланыс

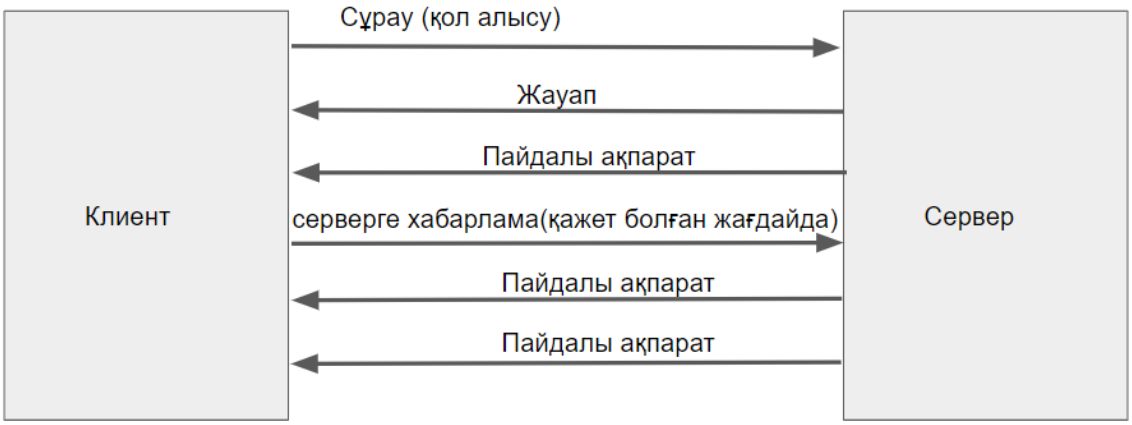
Суретте көріп тұрғанымыздай бір сұрауға бір жауап және одан кейін байланыс үзіледі. Клиент-серверлік архитектура HTTP протоколы арқылы осылай байланысады.

HTTP ерекшіліктері:

HTTP сұрауға жауап бергеннен кейін байланысты үзеді.

HTTP клиенттерді тақырыпта (HTTP Headers) жасағысы келетін әрекетті алдын-ала келісуге міндеттейді (GET, POST, PUT, DELETE).

Біз WebSocket арқылы жүргізілген байланста клиентті серверге бір рет қосамыз, содан кейін сервер бізге қажет болған кезде жауап бере алады:



Сурет 2 - WebSocket арқылы байланыс

Суретте көріп тұрғанымыздай клиент серверге әдеттегі TCP сұрауын жібереді, біз серверге қосылғымыз келетінін жеткізіп, серверден жауап күтеміз. Бұл процесс қол алысу (Handshake) деп аталады. Мысалы көптеген клиенттердің бірі серверге қосылуға сұраныс жібереді, бұл жағдайда сервер клиентке қарапайым жауап жібереді де басқа қосылып қойған клиенттерге пайдалы ақпараты бар пакеттерді жібереді.

Практикалық жүйеді мысал келтіріп өтейін. Клиент ретінде JavaScript фреймворкы React Js, Сервер ретінде Node Js ті қолданамын.

Мысал ретінде қарапайым онлайн режимдегі хабарлама алмасуға арналған чат болсын.

Серверді дайындап алайық:

```
const webSocketsServerPort = 8000;
const webSocketServer = require('websocket').server;
const http = require('http');
const server = http.createServer();
server.listen(webSocketsServerPort);
// Клиенттерді сақтауға арналған объект
const clients = {};
// WS - объектісін құрамыз
const wsServer = new webSocketServer({
  httpServer: server
});
wsServer.on('request', function (request) {
  var userID = getUniqueID();
  const connection = request.accept(null, request.origin);
  clients[userID] = connection;
  connection.on('message', function(message) {
    if (message.type === 'utf8') {
      // барлық қосылған Клиенттерге хабар тарату
      for(key in clients) {
        clients[key].sendUTF(message.utf8Data);
      }
    }
  })
});
```

Клиенттегі код:

```
// Байланыс орнатамыз
const client = new W3CWebSocket('ws://127.0.0.1:8000');
  // Серверден келетін пайдалы ақпарат, үнемі тыңдауда болады
  client.onmessage = (message) => {
    const dataFromServer = JSON.parse(message.data);
```

Веб-браузерден (Клиент) жіберілген сұрауға көз жүгіртейік.

Кодта көріп тұрғанымыздай WebSocket протоколы ws://127.0.0.1:8000/ адресінде ашылған. WebSocket ws:// URL схемасын және қауіпсіз қосылымдар үшін wss:// URL схемасын пайдаланады.

Біз жіберген сұрау:

Request Method: GET
Connection: Upgrade
Host: 127.0.0.1:8000
Origin: http://localhost:3000
Sec-WebSocket-Extensions: permessage-deflate; client_max_window_bits
Sec-WebSocket-Key: wHaoc9qLnMy7QelDGmvK7A==
Sec-WebSocket-Version: 13
Upgrade: websocket

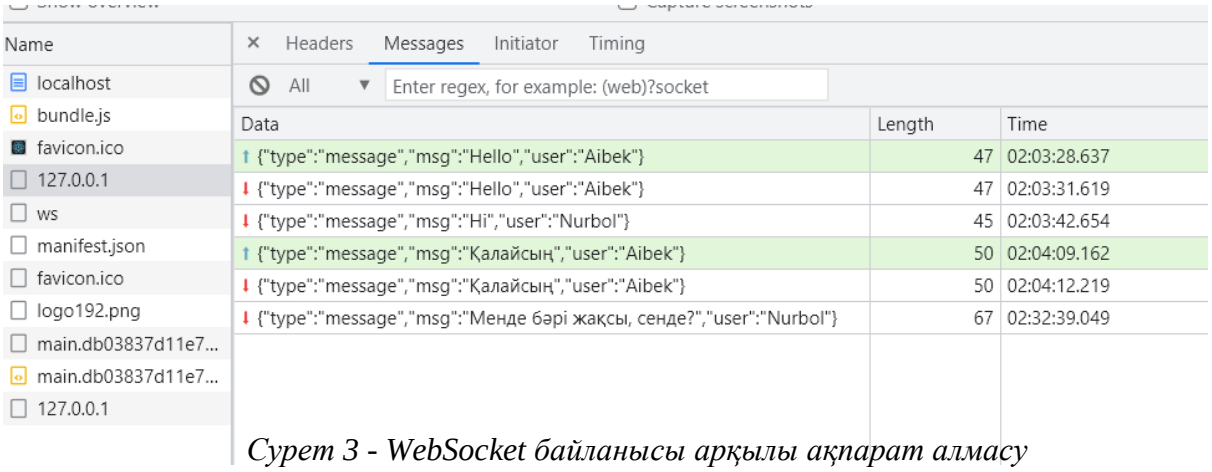
Серверден келген жауап(байланыс сәтті орнатылғаннан соң):

Connection: Upgrade
Origin: http://localhost:3000
Sec-WebSocket-Accept: GJFiv9wQ3cqgX5LsO8qwB/wMmXo=
Upgrade: websocket
Status Code: 101 Switching Protocols

Серверден келген жауаптан байқағанымыздай жауап сәтті аяқталғанымен біз 200 кодын емес 101 кодын көріп тұрмыз. 101 коды Switching Protocols - протоколдың ауысқанын білдіреді.

Байланыс орнатылды енді onMessage үнемі тыңдалымда болатын функция арқылы әрбір қосылған клиент басқа клиенттер жіберген ақпаратты алып отыратын болады.

Әрбір жіберілген және басқа клиенттер жіберген ақпаратты біз браузерде болатын әзірлеуші құралын ашу арқылы бақылай аламыз (developers tool).



Сурет 3 - WebSocket байланысы арқылы ақпарат алмасу

Жалпы WebSocket протоколының артықшылықтарын түсіндіре алдым деп ойлаймын.

WebSocket протоколындағы осалдықтар

Сайт аралық тоскауылға назар аударайын яғни Cross-Site WebSocket Hijacking (CSWSH) - бұл WebSocket-те болуы мүмкін осалдық түрі. Бұл WebSocket байланысына бағытталған жалған веб-сайтты сұрау шабуылының (CSRF) ерекше түрі, бұл шабуылдаушының орнатылған WebSocket байланысын ұстап қалуына және серверге ерікті деректерді жіберуіне әкеліп соғуы мүмкін.

Қалыпты WebSocket қосылымында клиент сервермен байланыс орнату үшін HTTP сұрауын жібереді және байланыс орнатылғаннан кейін клиент те, сервер де нақты уақыт режимінде деректерді бір-біріне жібере алады. WebSocket қосылымдарының проблемасы-олар бастапқыда CSRF қорғаныс механизмдерін қамтымайды, бұл оларды CSWSH шабуылдарына осал етеді.

CSWSH шабуылы пайдаланушы сервермен белсенді WebSocket байланысы бар веб-сайтқа кірген кезде пайда болады. Содан кейін шабуылдаушы пайдаланушы жақа яғни веб-сайтқа қандай да бір скрипт енгізу арқылы WebSocket сұрауын өзі қалаған жақа бағыттай алады. Бұл шабуылдаушыға ұрланған WebSocket қосылымын пайдаланып серверге зиянды деректерді жіберуге мүмкіндік береді, деректерді ұрлауға, деректерді басқаруға немесе басқа қауіпсіздік бұзушылықтарына әкелуі мүмкін.

CSWSH шабуылдарының алдын алу үшін әзірлеушілер WebSocket сұрау көзін тексеру, WebSocket сұрауларына таңбаларды (токендерді) қосу және пайдаланушы деректерін қорғау үшін шифрлау және аутентификация сияқты тиісті қауіпсіздік механизмдерін енгізу арқылы әртүрлі әдістерін жүзеге асыра алады.

CSWSH шабуылдарының алдын алудың тағы бір жолы-WebSocket қосылымдары үшін бірдей дереккөз саясатын (SOP) енгізу. Бұл саясат WebSocket байланыстарын тек сол домен мен байланыс орнатылған порт арасында орнатуды талап етеді. Бұл шабуылдаушыларға WebSocket сұрауларын басқа доменнен немесе заңды пайдаланушының WebSocket қосылымынан басқа порттан жіберуге мүмкіндік бермейді.

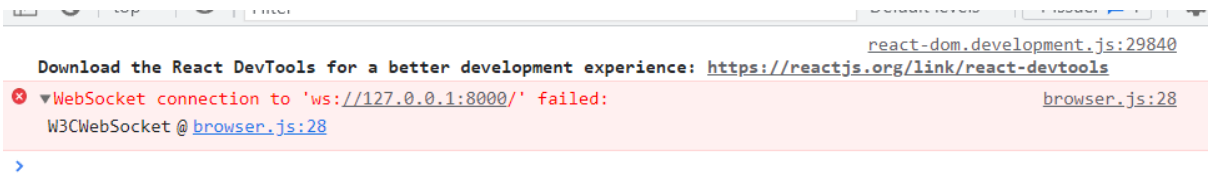
Мысал келтірейін. Серверге бірқатар өзгерістер енгіземін.

const allowedAddress = 'http://localhost:3000'; // рұқсат етілген клиенттің мекенжайын жазамыз

және бірден келіп түскен сұранысты тексереміз:

```
wsServer.on('request', function (request) {
    const connectionAddress = request.remoteAddress;
    if (connectionAddress !== allowedAddress) {
        console.log((new Date()) + ' Connection from ' + connectionAddress + ' rejected.');
        request.reject();
        return;
    }
}
```

Егер рұқсат етілген мекенжайларды осылай серверде алдын ала анықтап алуға болады. Клиенттік веб-қосымшаны қайта іске қосамын бірақ осы жолы оны басқа портта жүргіземін.



Сурет 4 - Өзгертілген порттан байланысты орнатудағы серверден келген жауап.

Енді WebSocket пен байланыс суреттен байқағанымыздай орындалмайды.

CSWSH-бұл WebSocket қосылымдарындағы маңызды осалдық, бұл деректерді ұрлауға, манипуляциялауға немесе басқа қауіпсіздік бұзылыстарына әкелуі мүмкін. SOP және CSRF таңбалауыштары сияқты тиісті қауіпсіздік механизмдерін енгізу арқылы әзірлеушілер өз жүйелерін CSWSH шабуылдарынан қорғауға және пайдаланушылардың деректерінің қауіпсіздігін қамтамасыз етуге көмектеседі.

Қорытындылай келе, WebSockets веб-қосымшалар немесе мобильді қосымшалар мен серверлер арасындағы қуатты және тиімді байланыс механизмін қамтамасыз етеді. Дегенмен, WebSockets-пен байланысты осалдықтарды білу және олардың алдын алу үшін тиісті қауіпсіздік

шараларын қолдану маңызды. Тиісті аутентификация және авторизация механизмдері, енгізуді тексеру және санитаризациялау және HTTPS немесе TLS пайдалану сияқты ең жақсы тәжірибелерді орындау арқылы әзірлеушілер мен қауіпсіздік мамандары өз жүйелерін ықтимал қауіптерден қорғауға көмектесе алады.

Қолданылған әдебиеттер тізімі

1. https://developer.mozilla.org/en-US/docs/Web/API/WebSockets_API/Writing_WebSocket_server
2. https://www.rfc-editor.org/rfc/rfc6455
3. “Справочник хакера веб-приложений: поиск и использование недостатков безопасности” Дафидда Статтарда и Маркуса Пинто.
4. https://web.archive.org/web/20101212010024/http://websockets.org/

УДК 004.94

СРАВНИТЕЛЬНЫЙ АНАЛИЗ ПРОГРАММНЫХ ПРОДУКТОВ ИМИТАЦИОННОГО МОДЕЛИРОВАНИЯ

Жаксыбаева Мадина Думановна
huangmadina@gmail.com
Магистрант 2 курса специальности "Информационные системы"
Евразийский национальный университет им Л.Н. Гумилева, кафедра Информационные системы, г. Астана, Казахстан
Научный руководитель – Ж.Б. Ламашева

Аннотация. Несовременные, вовремя не усовершенствованные предприятия непросто справляются с вопросами нынешнего дня. Нынешние требования экономики и индустриальных проектов неизменно усложняются и требуют исполнения новых задач. В таких критериях главной целью является повышение эффективности бизнес-процессов предприятия и оптимизации его структуры. Для первоначального анализа рентабельности работы предприятия, определения проблемных мест и поиска оптимальных методов их решения повсеместно применяются методы моделирования. Исходя из сложности моделируемых систем и их количественных характеристик, особенно результативным и актуальным подходом в настоящее время прибывает создание систем поддержки принятия решений или цифровых двойников на базе имитационного моделирования. Существует целый ряд программных продуктов, позволяющих создавать и исследовать имитационные модели.

Ключевые слова: имитационное моделирование, Actor Pilgrim, Enterprise Dynamics, Flexsim, NetLogo, AnyLogic.

Введение. В современном обществе цифровых технологий имитационное моделирование помогает исследовать и решать всевозможные задачи путем сокращения времени и упрощения самого процесса работы. Разработчики разных государств создают программные продукты, ориентированные на решение установленных перед ними задач, которые помогают в работе профессионалам различных сфер: строительной, логистической, экономической и т.п.

Программные продукты.

Система AnyLogic - это универсальная система имитационного моделирования, которая предлагает процессно-ориентированные (дискретно-событийные), системно-динамические и агентно-ориентированные технологии. Его библиотека, инструменты и графический интерфейс позволяют создавать имитационные модели для решения целого ряда задач, от бизнес-процессов и логистики до производства и развития рынка. AnyLogic широко используется в образовании и стала корпоративным стандартом для бизнес-моделирования в транснациональных компаниях [2].

Actor Pilgrim - это специализированная система для создания и отладки имитационных моделей трудоемких процессов, включая временную, пространственную и экономическую динамику. Он используется для анализа проектов в области энергетики, региональной экономики,