Documentatie AMULET-protocol

# Preambule

Dit is de volledige documentatie, beschrijving en handleiding voor het bit-**A** **M**od**ULE** 2 **T**cp Protocol, wat op donderdag 8 januari 2015 is vastgesteld in het Universiteit Twente gebouw “Carré”.

Dit protocol is opgezet om universele communicatie tussen alle verschillende server en client applicaties mogelijk te maken die ontwikkeld worden door de leden van BIT groep 2 voor het programmeren project van Module 2 van de opleiding Bedrijfs-Informatie Technologie.

Er is besloten om dit middels plain text, pre-agreed keywording over TCP te realiseren. Als standaard poort waar de server naar luistert is **2220** gekozen. Als break binnen de commando’s is gekozen voor een spatie: ␣. Alle Commando’s worden in **ALL CAPS** verstuurt.

Een beschrijving van de formatting voor verschillende exchanges en voorbeelden van de communicatie volgt. Ook zijn de functies die eventueel in java gebruikt kunnen worden opgenomen.

Aanpassingen kunnen ondervermelding van het pagina nummer en de titel verstuurd worden naar [s.m.braams@student.utwente.nl](mailto:s.m.braams@student.utwente.nl) of [p.w.beers@student.utwente.nl](mailto:p.w.beers@student.utwente.nl) .

Table of Contents

[Preambule 1](#_Toc408835600)

[Establish Connection 3](#_Toc408835601)

[Join 4](#_Toc408835602)

[Move 5](#_Toc408835603)

[Error & Debug 6](#_Toc408835604)

[Uitbreidinginen 7](#_Toc408835605)

[Leaderboard 7](#_Toc408835606)

[Challenge Login 8](#_Toc408835607)

[Security login 10](#_Toc408835608)

[Overige afspraken & Conventies 11](#_Toc408835609)

[Java functies 12](#_Toc408835610)

# Establish Connection

#### Text

1. De client initïaliseert een TCP-verbinding met de server.
2. Als de verbinding is opgezet stuurt de server een EXTENSIONS request met daarachter de extensions die hij ondersteunt in de volgorde NONE CHAT CHALLENGE LEADERBOARD.
3. De client geeft in de respons aan welke extensions de client support met EXTENSIONS NONE CHAT CHALLENGE LEADERBOARD waarbij wordt gestuurd wat wordt onderteunt door de client; als de client geen extra functionaliteiten ondersteund wordt het commando **EXTENSIONS NONE** als de client chat en leaderboard ondersteund wordt het commando: **EXTENSIONS CHAT LEADERBOARD**
4. Vervolgens weten de server en client van elkaar welke functionaliteiten ze beide ondersteunen en worden de overige functionaliteiten gedeactiveert.

#### Voorbeeld Tabel

|  |  |
| --- | --- |
| **Client** | **Server** |
| Establish Connection | Establish Connection |
|  | EXTENSIONS NONE |
| EXTENSIONS CHAT CHALLENGE LEADERBOARD |  |

# Join

#### Text

1. Nadat de extensions duidelijk zijn stuurt de client een join request met de syntax “JOINREQ Player\_Nickname” Met voor nickname de naam van de client. Stel ik wil als “Turbohenk95” spelen dan stuurt mijn client **JOINREQ Player\_Turbohenk95**.
2. In dit gedeelte van de documentatie gaan we uit van geen extensies, kijk voor het alternatief met wel CHALLENGE en wel Security verderop in dit document. Omdat deze client niet kan kiezen met wie hij speelt wacht de client na dit punt op het “GAME” commando van de server.
3. Zodra er een andere speler zich heeft aangemeld stuurt de server naar beide spelers **GAME Nickname** met i.p.v. nickname de naam van de speler waarmee de huidige client een spel aangaat.

#### Voorbeeld Tabel

|  |  |
| --- | --- |
| **Client** | **Server** |
| JOINREQ Player\_Turbohenk95 |  |
|  | GAME Player\_Gerrit |

# Move

#### Text

1. Als het spel begonnen is stuurt de server een bericht naar de client die aan de beurt is met “TURN”.
2. De client antwoord met **MOVE 0** met i.p.v de 0 de kolom van het bord welke hij geselecteerd heeft. De nummers van de 7 kolommen van het 4 op een rij bord zijn v.l.n.r. 0 1 2 3 4 5 en 6.
3. Als de move geldig is dan stuurt de Server naar beide spelers het commando **MOVEUPDATE Nickname 0** met i.p.v. Nickname de naam van de speler en i.p.v. 0 de kolom die is gekozen
   1. Als de move niet geldig is dan stuurt de server het ERROR commando en sluit de game af, dit wordt hierna behandeld.
4. Als dit de laatste move was van het spel of als de andere speler is ge disconnect stuurt de server het GAMEEND commando. Als er vervolgens niks achter het commando volgt is het spel geannuleerd en is er geen winnaar, als er DRAW achter wordt gestuurt is het gelijkspel en als er een nickname wordt gestuurt is die persoon de winnaar.

#### Voorbeeld Tabel normaal

|  |  |
| --- | --- |
| **Client** | **Server** |
|  | TURN |
| MOVE 3 |  |
|  | MOVEUPDATE Player\_Turbohenk95 3 |
|  | TURN |
| MOVE 2 |  |
|  | *GAMEEND* Player\_Turbohenk95 3 |

#### Voorbeeld Tabel ongeldigde zet of timeout

|  |  |
| --- | --- |
| **Client** | **Server** |
|  | TURN |
| MOVE 3 |  |
|  | MOVEUPDATE Player\_Turbohenk95 3 |
|  | TURN |
| Ongeldige zet/[timeout] |  |
|  | ***ERROR*** |
|  | ***GAMEEND*** |

# Error & Debug

#### Text

Als er een een Java-Time out ontstaat doordat een speler geen connectie meer heeft/niet meer reageerd word of er een andere situatie ontstaat waarbij de client iets fout doet stuurt de server het ERROR commando met een string [errormessage] waarin er aangegeven wordt wat er fout ging en sluit de connectie.

#### Tabel

|  |  |
| --- | --- |
| **Client** | **Server** |
|  | ERROR [errormessage] |
|  | [disconnect] |

#### Text

Met het DEBUG commando kun je even wat tekst naar de andere kant sturen. De [text] kan arbitrair lang zijn dus vang dit af. Hoe je dit implementeert mag je zelf weten.

#### Tabel

|  |  |
| --- | --- |
| **Client** | **Server** |
| DEBUG [text] | DEBUG [text] |
|  |  |

# Uitbreidinginen

## Leaderboard

Voor deze uitbreiding stuurt de server een leaderboard naar de client die hem opvraagt. Deze kan de client vervolgens manipuleren om verschillende zaken te laten zien zoals overall statistieken, persoonlijke records etc. Een leaderboard wordt bij gehouden op een sql server gehost op sjaars:hackerproof@null.s.pieterbos.me. Hier kan de server sql requests naar sturen om zelf een leaderboard bij te houden welke hij naar zijn clients kan sturen. Je kunt ook je eigen leaderboard bijhouden.

De sql server hanteert de volgende syntax

|  |  |  |  |
| --- | --- | --- | --- |
| Player1 (VARCHAR) | Player2 (VARCHAR) | Timestamp(TIMESTAMP) | Result (VARCHAR) |

De communicatie tussen clients en server staat hieronder. Als je van sql gebruikt maakt moet de server nu dus na het eindigen van een game ook een aanpassing aan de sql server doen en zijn eigen tabel. Je moet zelf kijken hoevaak je de server wilt laten updaten maar bedenk dat je verwantwoordelijk met de sqlserver omgaat want hij wordt vrijwillig door Pieter gehost.

#### Text

1. Voor het volledige leaderboard stuurt de Client LEADERBOARDREQ
2. Vervolgens stuurt de server het leaderboard op in de volgende syntax: LEADERBOARD [Nickname1] [Nickname2] [Timestamp] [Winner]

De Nicknames zijn de twee spelers in de volgorde wie er gestart is, de timstamp is de tijd dat het spel is geëindigt in de sql TIMESTAMP vorm en Winner is de nickname van de persoon die gewonnen heeft of DRAW als het gelijk spel is. Als er maar twee entries in het leaderboard van de server staan tussen de spelers Player\_hans en Player\_gerrit wordt het dus alsvolgt:

**LEADERBOARD Player\_hans Player\_gerrit 2015-01-12 10:16:07 DRAW Player\_gerrit Player\_hans 2015-01-12 10:40:22 Player\_hans**

**Let op de spatie binnen de timestamp je moet de datum en tijd samen opslaan.**

1. Om de dagelijkse leaderboard te krijgen stuurt de Client DAILYLEADERBOARDREQ en stuurt de server alle entries met de zelfde datum als de huidige in dezelfde format als hierboven.
2. Om de persoonlijke leaderboard te krijgen stuurt de Client PERSLEADERBOARDREQ en stuurt de server alle entries waar de nickname van de client in voorkomt in dezelfde format als hierboven.

#### Tabel

|  |  |
| --- | --- |
| **Client** | **Server** |
| LEADERBOARDREQ |  |
|  | LEADERBOARD Player\_hans Player\_gerrit 2015-01-12 10:16:07 DRAW Player\_gerrit Player\_hans 2015-01-12 10:40:22 Player\_hans |

## Chat

Met deze uitbreiding kunnen de clients berichten naar elkaar sturen via de server. Een client kan 3 dingen doen, een bericht naar een persoon sturen, een bericht broadcasten en een persoon blokken. Omdat je moet weten wie er ingelogt zijn om deze functie de gebruiken is het verstandig dit te combineren met het implementeren van de Challange extension.

#### Text

1. Als een client een bericht wil sturen naar een andere client stuurt hij **MESSAGE [nickname] [messagecontent]**.
2. De server stuurt dit bericht door naar de client die bij [nickname] hoort en vervangt de nickname van de ontvanger met die van de verzender.
3. Als een client een bericht wil broadcasten stuurt hij **BROADCAST [messagecontent].**
4. De server stuurt dit naar alle clients die de verzender niet geblokt hebben en zet de afzender er tussen: **BROADCAST [nickname][messagecontent]**
5. Als een client geen berichten meer wilt ontvangen van andere client stuurt hij **BLOCK [nickname].** Als hij de persoon weer wil unblocken stuurt hij **UNBLOCK [nickname].**

#### Tabel

|  |  |
| --- | --- |
| Client | Server |
| MESSAGE Player\_Hans Hoe gaat het |  |
|  | MESSAGE Player\_pietje Hoe gaat het |
| BROADCAST pietje kan niet proggen |  |
|  | BROADCAST Player\_Hans pietje kan niet proggen |
| BLOCK Player\_Hans |  |

## Challenge Login

Voor deze uitbreiding kunnen de spelers kiezen met wie ze spelen en worden ze niet automatisch door de server ingedeeld. Het Join Protocol wordt nu dus anders.

#### Text

1. De clients moeten weten wie er ingelogd zijn in de server om te kunnen kiezen met wie ze spelen, hiervoor moeten we sommmige processen iets aanpassen. Nadat de connectie met de server is goedgekeurd stuurt de server naar alle verbonden clients een PLAYERUPDATE waarbij de clients hun lijst van verbonden spelers aan de server kunnen updaten. De lijst die achter het PLAYERUPDATE keyword volgt is gerangschikt als elk ingelogde client mer erachter de uitbreidingen die ze hebben in de volgorde van NONE CHAT CHALLENGE LEADERBOARD SECURITY. Om verwarring te voorkomen gebruiken we voor de capabilites een comma als scheidings teken

PLAYERUPDATE Player\_Nickname1 NONE Player\_Nickname2 LEADERBOARD Player\_Nickname3 NONE Player\_Nickname4 LEADERBOARD,SECURITY Nickname5 CHAT,CHALLENGE,LEADERBOARD,SECURITY

1. Na een error op een speler of speler die disconnect stuurt de server een **DISCONNECTED Nickname** commando naar alle Challange en chat spelers zodat ze hun tables kunnen updaten.
2. Nadat de server en client van elkaar geïdentificeert hebben dat ze CHALLENGE Login ondersteunen en de PLAYERUPDATE is verstuurtkomt de client in de lobby terecht. Doordat de server de client update van wie er inloggen en weggaan kan weet de client met wie hij zou kunnen spelen. Als hij een speler heeft uitgekozen stuurt de client CHALLENGE Player\_Nickname.
3. Als de speler met wie de CHALLENGE speler wil spelen niet CHALLENGE ondersteunt start de server automatisch een game en stuurt naar beide spelers het GAME commando. Als de speler wel een CHALLENGE speler is stuurt de server naar de client het CHALLENGE commado door.
4. De Client stuurt terug CHALLANGERESP [Y/N] met dus Y als hij het accepteert en N als hij het niet accepteert. Als hij het accepteert start de server een game, als hij het niet accepteert stuurt de server het CHALLANGERESP commando door naar de originele client die dit vervolgens kan afhandelen.
5. Na het eind van een Game komen de spelers weer in de lobby terecht

#### Volledige Tabel

|  |  |
| --- | --- |
| **Client** | **Server** |
| Establish Connection | Establish Connection |
|  | EXTENSIONS CHALLENGE |
| EXTENSIONS CHAT CHALLENGE LEADERBOARD |  |
| JOINREQ ProgBaas |  |
|  | PLAYERUPDATE Player\_henk CHALLENGE,CHAT Player\_Freek CHALLENGE,LEADERBOARD ProgBaas CHALLENGE,SECURITY |
|  | [Wait for CHALLENGE] |
| CHALLENGE Player\_Henk |  |
|  | {to henk} CHALLENGE ProgBaas |
| {from henk}CHALLANGERESP N |  |
|  | {to ProgBaas} CHALLANGERESP N |
| {from ProgBaas}CHALLENGE Player\_Freek |  |
|  | {to freek} CHALLENGE ProgBaas |
| {from freek}CHALLANGERESP Y |  |
|  | {to Progbaas} GAME Player\_Freek |
|  | {to freek} GAME Progbaas |
|  | {to Progbaas} TURN |
| {freek}[i/o timeout] |  |
|  | {to freek}ERROR |
|  | {to all} DISCONNECTED Player\_Freek |
|  | {to Progbaas} ENDGAME |

## Security login

De spelers zijn met het normale inlog protocol niet beschermt tegen man in the middle attacks, sniffing en inbreuk in de server omdat er geen wachtwoorden zijn. Met deze extensie encypten we al het communicatie verkeer en slaan we de wachtwoorden verantwoordelijk op.

De server moet hiervoor wel een Certificaat moeten hebben en de client moet de public key van de server opzoeken (je kunt hem ook hardcoden). Hiervoor heeft de server een naam nodig.

Java ondersteunt SSLServerSocket, je kunt dus gewoon je ServerSocket vervangen door een SSLServerSocket om het verkeer te encrypten en in de client een SSLSocket. Vervolgens moet je de ondersteuning van het certificaat goed implementeren, kijk ook [hier](http://www.herongyang.com/JDK/SSL-Socket-Client-Example-SslSocketClient.html).

Om te herkennen dat een speler Security ondersteund wordt de Nickname **Zonder Player\_** voor de naam gebruikt.

1. Om de public key op te kunnen zoeken moet de server een naam hebben. Hier moet het Joinn protocol dus voor worden aangepast. Spelers die Security ondersteunen hebben geen Player\_ voor hun naam. Na de JoinReq stuurt de Server zijn naam alsvolgt: **HELLO [name].** Vervolgens zoekt hij de naam van de speler op in zijn registered players tabel.
2. Als de er nog niet instaat stuurt de server ERROR NOTREGISTERED naar de client
3. Als de speler al wel geregistreerd is stuurt de server **AUTHENTICATE**
4. Vervolgens stuurt de client zijn wachtwoord als **AUTHENTICATION [password]**.
5. De server vergelijkt de SHA256 hash met de opgeslagen hash. Als dit klopt stuurt de Server JOIN Y anders JOIN N
6. De client mag 7 keer een fout wachtwoord sturen, daarna wordt de speler geblokkeerd.

#### Tabel

|  |  |
| --- | --- |
| **Client** | **Server** |
| JOINREQ FourUpMaster |  |
|  | AUTHENTICATE |
| AUTHENTICATION [gehasht verkeerdwachtwoord] |  |
|  | JOIN N |
| AUTHENTICATION [gehasht wachtwoord] |  |
|  | JOIN Y |

# Overige afspraken & Conventies

**Voor een Nickname van een client die geen security ondersteunt moet er “Player\_” staan: Player\_Hans**

**Nicknames mogen geen spaties bevatten.**

**Nicknames mogen niet “EMPTY”zijn.**

**Nicknames mogen niet 1 van de commando’s zijn.**

De clients zijn zelf verantwoordelijk om hun input te sanitizen om er voor te zorgen dat het overtreden van de bovenstaande regels geen problemen opleveren

Als standaard poort waar de server naar luistert is **2220** gekozen.

Als break binnen de commando’s is gekozen voor een spatie: ␣.

Alle commandos worden in **ALL CAPS** verstuurd.

Alle commandos worden in plaintext, UTF-8 gestuurd.

Alle commandos worden voorzien van een regeleinde na de argumenten aan het commando, dat wil zeggen de java String “\r\n”.

Er staat hier alleen beschreven welke informatie moet worden uitgewisseld zodat de clients en servers kunnen functioneren; hoe ze functioneren of hoe met de informatie moet worden omgegaan is hier niet beschreven en is de verantwoordelijkheid van de groepjes zelf.

## Java functies

Van [deze](http://systembash.com/content/a-simple-java-tcp-server-and-tcp-client/) website:

*“To compile these, install Java JDK to your system. Then compile the program with javac TCPClient.java – this will create a TCPClient.class. Execute the file with java TCPClient – leave off the .class, or you will get the error: “Exception in thread “main” java.lang.NoClassDefFoundError”*

Bruikbare Java-API links:

<http://lmgtfy.com/?q=java+api>

<http://docs.oracle.com/javase/tutorial/networking/sockets/>

<http://docs.oracle.com/javase/7/docs/api/java/net/Socket.html>

<http://download.java.net/jdk7/archive/b123/docs/api/java/net/ServerSocket.html>

<http://docs.oracle.com/javase/7/docs/api/java/io/DataOutputStream.html>

<http://docs.oracle.com/javase/7/docs/api/java/io/InputStreamReader.html>

<http://docs.oracle.com/javase/7/docs/api/java/io/BufferedReader.html>

<http://docs.oracle.com/javase/7/docs/api/java/util/Formatter.html>

<http://docs.oracle.com/javase/7/docs/api/java/util/Scanner.html>

<http://docs.oracle.com/cd/B28196_01/idmanage.1014/b28171/oracle/security/crypto/core/RSA.html>

<http://docs.oracle.com/javase/7/docs/api/java/security/MessageDigest.html>

<https://docs.oracle.com/javase/6/docs/api/java/security/SecureRandom.html>