

BATTLE CONTRACT & BATTLE TOKEN AIRDROP

We are proud to announce the
official release of the first
Ethereum Battle Smart Contract

As a **gift** for our earlier investor we
announce the **AIRDROP** of the **Battle Token**
to all participant in our first Token battle

Battle Contract



Join the first token battle!

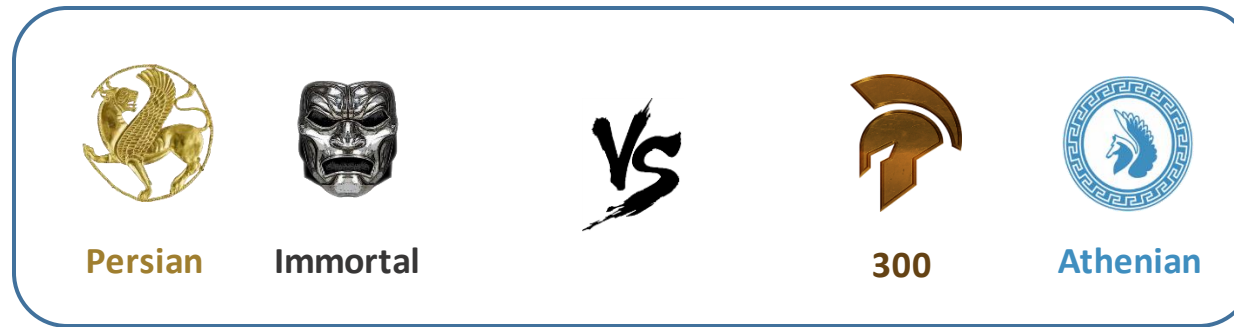
24 October 2017

31 October 2017

Battle Smart Contract Phase and rules

The Battle Contract **Battle Of Thermopylae** is a smart contract build on Ethereum Blockchain that virtually settle a challenge between token holder of Greek and Persian armies

Only Token Holder of one of this four forces can take part on the virtual battlefield

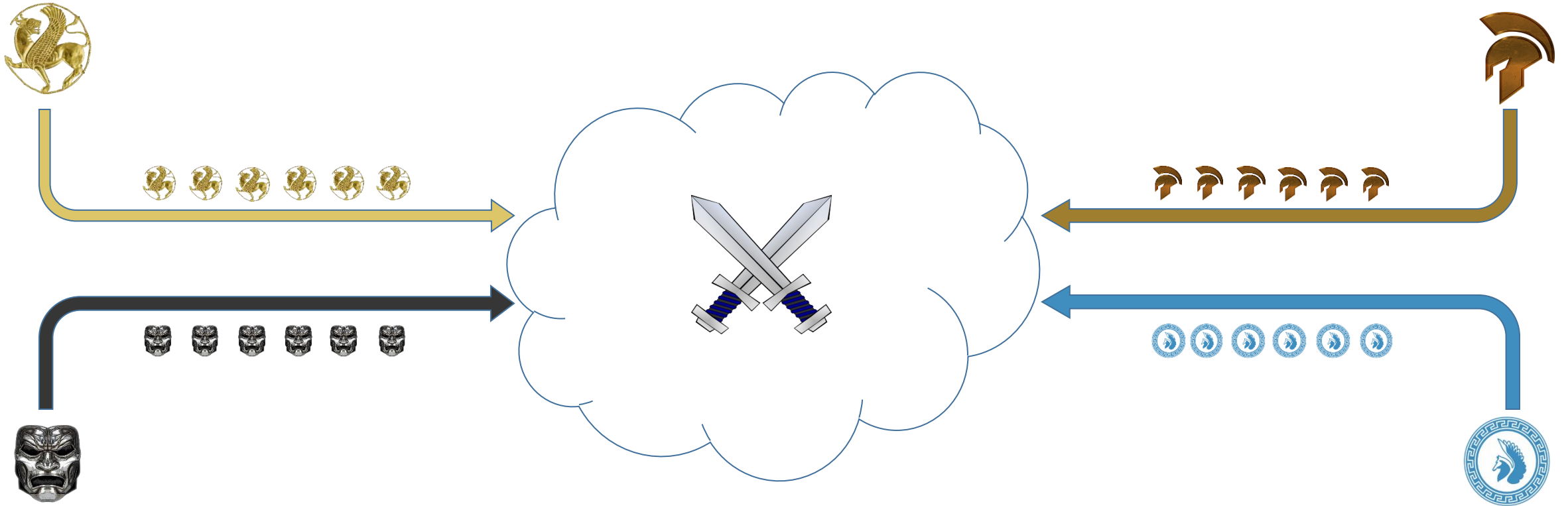


The **Battle Of Thermopylae** consist of three main phases

1. **Troops Deployment**
2. **Battle & Definition of Winners and Losers**
3. **Prize Awarding**

Phase 1 – Troops Deployment

From 24th to 31th October



Token Holder of each faction have to sent warriors token on the battlefield using battle smart contract functions (see troops deployment guide section for more info)

Phase 1 – Troops Deployment

The battle progress & status could be checked on the special **Dapp** available on the Perians token website

<http://persians.network/battle.html>



Donations are always appreciated 0xA03Ee7110FAFb324d4a931979eF4578bffB6a00

Phase 2 – Battle & Definition of Winners and Losers

By 31th October

Each soldier deployed on battlefield has a power value – **Battle Point** – according to the following rules



1 BP for each PRS Token



1000 BP for each 300 Token



100 BP for each IMT Token



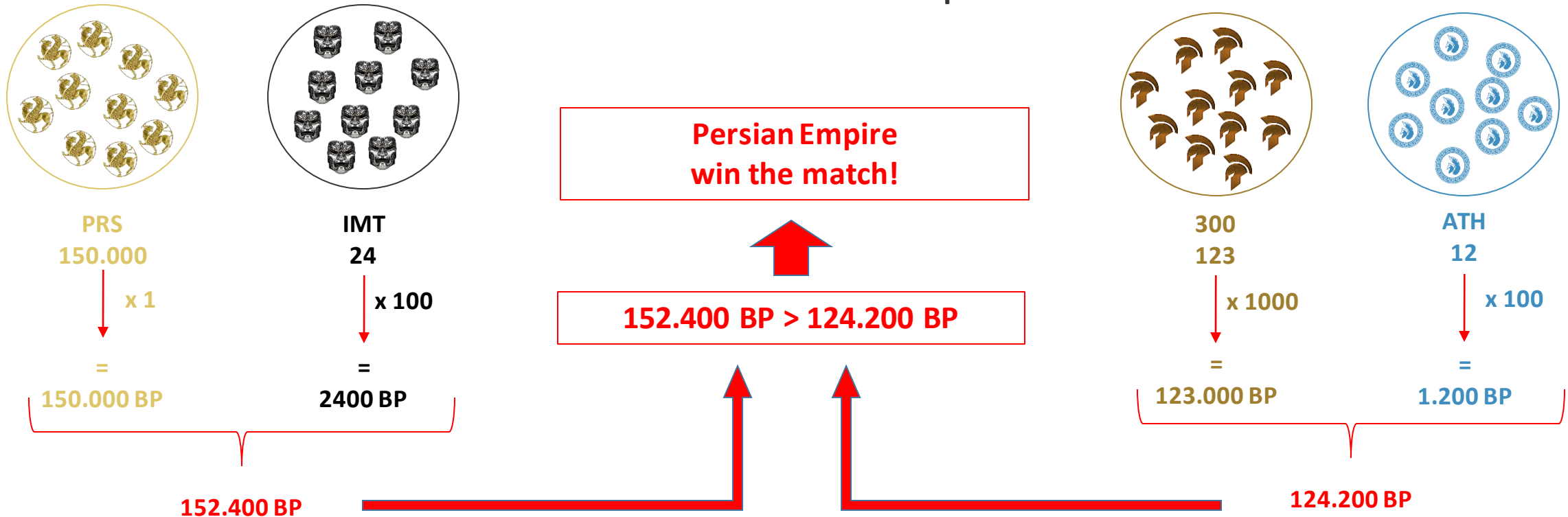
100 BP for each ATH Token

Phase 2 – Battle & Definition of Winners and Losers

By 31th October

At the end of the Battle the total amount of the **Battle Point** (BP) for each factions will be automatically counted by the smart contract functions and the outcome of the battle will be determined

Consider the follow example



Phase 3 – Prize Awarding

By 31th October

After the end of the battle:

- all IMT and ATH warriors will be sent back to each participant/token holder
- 10% of Persian and Spartan warriors are considered as casualties and will be locked forever in the smart contract as dead bodies on the battlefield

Each participant of the **winning faction**:

- will receive back 90% of his/her Persian or Spartan warriors
- will receive enemy Persian or Spartans warriors (tokens of the opposite side) as slaves in proportion to their contribution on the battlefield

Battle Token Airdrop

Join the battle and get the future token for Battle Smart Contract Platform.

This **utility** token will give special ability to the warrior token in next Battle Contracts, you have **only this chance to get BTL token**, there will be no more **AIRDROP**, BTL Token will be available only on the exchange.

For each soldier deployed on battlefield participants will receive Battle Tokens according to the following rules



1 BTL
for each PRS Token



1000 BTL
for each 300 Token



2000 BTL
for each IMT Token



2000 BTL
for each ATH Token

Battle Contract – Troops Deployment Guide

This section will show how to deploy troops on the battle contract using **MyEtherWallet** please refer to the contract names displayed below as reference for this guide

Persian Army MyEtherWallet Contract Names



Persian - Warrior for Battle Contract



Immortal - Warrior for Battle Contract

Greek Army MyEtherWallet Contract Names



300 - Warrior for Battle Contract



Athenian - Warrior for Battle Contract

Battle Smart MyEtherWallet Contract Name




Battle Of Thermopylae - Battle Contract



This guide refers to Persian Token. Generals leading other armies have to execute the same steps on contracts and functions referring to their token


Step 1 - Open MyEtherWallet
Double-check the URL (.com) before unlocking your wallet

New WalletSend Ether & Tokens SwapSend OfflineContractsENSCheck TX StatusView Wallet InfoHelp

Create New Wallet

Enter a password

Do NOT forget to save this!



Create New Wallet

This password *encrypts* your private key. This does not act as a seed to generate your keys. You will need this password + your private key to unlock your wallet.

How to Create a Wallet · Getting Started


Step 2 - Open Contracts tab

New WalletSend Ether & Tokens SwapSend OfflineContractsENSCheck TX StatusView Wallet InfoHelp

Interact with Contract or Deploy Contract

Contract Address

mewtopia.eth or 0x7cB57B5A97eAbe94205C07890BE4c1aD31E486A8



Select Existing Contract


ABI / JSON Interface

```
[{ "type": "constructor", "inputs": [{ "name": "param1", "type": "uint256", "indexed": true }], "name": "Event" }, { "type": "function", "inputs": [{ "name": "a", "type": "uint256" }], "name": "foo", "outputs": [ ] }]
```

Access

Donations are always appreciated
0xeA03Ee7110FAFb324d4a931979eF4578bffB6a00


Step 3 - Under heading Select Existing Contract is a dropdown, select **Persian - Warrior for Battle Contract** from the list and click the Access Button

New WalletSend Ether & Tokens SwapSend OfflineContractsENSCheck TX StatusView Wallet InfoHelp

Interact with Contract or Deploy Contract

Contract Address

0x163733bcc28dbf26B41a8CfA83e369b5B3af741b



Select Existing Contract

Persian - Warrior for Battle Contract0x163733bcc28dbf26B41a8CfA83e369b5B3af741b

ABI / JSON Interface

```
[{"constant":true,"inputs":[],"name":"name","outputs":[{"name":"","type":"string"}],"payable":false,"type":"function"}, {"constant":false,"inputs":[{"name":"_spender","type":"address"}, {"name":"_value","type":"uint256"}],"name":"approve","outputs":[{"name":"success","type":"bool"}],"payable":false,"type":"function"}, {"constant":true,"inputs":[],"name":"icoStartBlock","outputs":[{"name":"","type":"uint256"}],"payable":false,"type":"function"}, {"constant":true,"inputs":[],"name":"totalSupply","outputs":[{"name":"","type":"uint256"}],"payable":false,"type":"function"}, {"constant":false,"inputs":[{"name":"_from","type":"address"}, {"name":"_to","type":"address"}, {"name":"_value","type":"uint256"}],"name":"transferFrom","outputs":
```

Access

Step 4 - Under **Read / Write Contract** heading, open the dropdown **Select function** and click on **approve**

Read / Write Contract

0x163733bcc28dbf26B41a8CfA83e369b5B3af741b

approve

_spender address

0x314156...

_value uint256

151

How would you like to access your wallet?

Metamask / Mist

Ledger Wallet

TREZOR

Digital Bitbox

Keystore File (UTC / JSON)

Mnemonic Phrase

Private Key

Parity Phrase: No longer supported

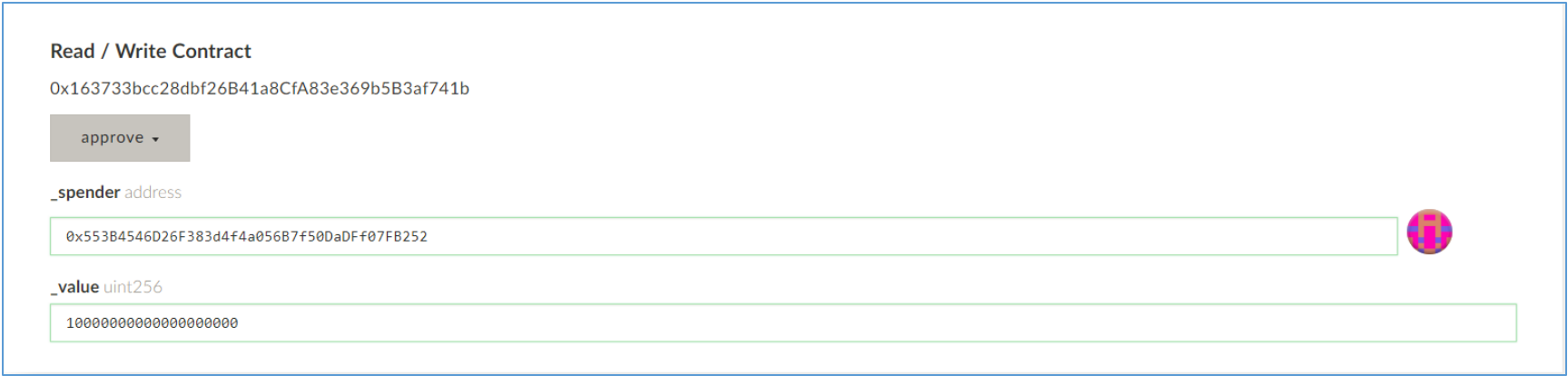
WRITE

Donations are always appreciated
0xeA03Ee7110FAFb324d4a931979eF4578bffB6a00

Step 5

Under the dropdown select a new section will open:

- In field **_spender** insert the Battle Contract Address **0x553B4546D26F383d4f4a056B7f50DaDFf07FB252**
- In field **_value** insert the amount of warrior token that you want to deploy on the battlefield



The amount to insert in **_value** field is relative to the decimal for each token, refer to table as example

Token	Decimal	_value field for 1 token	_value field for 10 token	_value field for 50 token	_value field for 100 token
Persian	18	10000000000000000000	10000000000000000000	5000000000000000000	10000000000000000000
Immortal	0	1	10	50	100
300	18	10000000000000000000	10000000000000000000	5000000000000000000	10000000000000000000
Athenian	18	10000000000000000000	10000000000000000000	5000000000000000000	10000000000000000000

So, for example, if you want to approve 123 PRS you need insert **12300000000000000000** in **_value** field

Load your wallet using any one of the methods listed, for this example we used the Keystore File:

- leave the value of **Amount to Send** to **0 ETH**
- allow the wallet to suggest a Gas Limit. If it does not autopopulate, enter 50000 or more. If your TX fails, increase Gas Limit.

Click on the **Generate Transaction** button, wait a while, when ready click on the blue button **Yes, I am sure! Make transaction.**

Raw Transaction	Signed Transaction
<pre>{"nonce": "0x0a", "gasPrice": "0x3b9aca00", "gasLimit": "0x01b459", "to": "0x553B4546D26F383d4f4a056B7f50DaDFf07"}</pre>	<pre>0xf8880a843b9aca0008301b45994553b4546d26f383d4f4a056b7f50dadff07fb25280a4fb9ec0a8000000000000000000000000</pre>
No, get me out of here!	Yes, I am sure! Make transaction.

Step 9 - Reload the MyEtherWallet Contracts tab, under heading Select Existing Contract is a dropdown, select **Battle of Thermopylae - Battle Contract** from the list and click the Access Button.

Step 10 - Under **Read / Write Contract** heading, open the dropdown **Select function** and click on **assignPersiansToBattle**

New WalletSend Ether & TokensSwapSend OfflineContractsENSCheck TX StatusView Wallet InfoHelp

Interact with Contract or Deploy Contract

Contract Address

0x553B4546D26F383d4f4a056B7f50DaDF07fB252

Select Existing Contract

Battle Of Thermopylae - Battle Contract 0x553B4546D26F383d4f4a056B7f50DaDF07fB252

ABI / JSON Interface

```
[{"name": "_faction", "type": "address"}, {"name": "getTotalSlaves", "outputs": [{"name": "slaves", "type": "uint256"}], "payable": false, "stateMutability": "view", "type": "function"}, {"constant": false, "inputs": [{"name": "_warriors", "type": "uint256"}], "name": "assignImmortalsToBattle", "outputs": [{"name": "success", "type": "bool"}], "payable": false, "stateMutability": "nonpayable", "type": "function"}, {"constant": true, "inputs": [{"name": "athenians", "outputs": [{"name": "", "type": "address"}], "payable": false, "stateMutability": "view", "type": "function"}, {"constant": true, "inputs": [{"name": "BATTLE_POINT_DECIMALS", "outputs": [{"name": "", "type": "uint8"}], "payable": false, "stateMutability": "view", "type": "function"}, {"constant": true, "inputs":
```

Access

Read / Write Contract

0x553B4546D26F383d4f4a056B7f50DaDF07fB252

assignPersiansToBattle

_warriors uint256

151

How would you like to access your wallet?

Metamask / Mist

Ledger Wallet

TREZOR

Digital Bitbox

Keystore File (UTC / JSON)

Mnemonic Phrase

Private Key

Parity Phrase: No longer supported

WRITE

Step 11

Under the dropdown select a new section will open:

- In field **_warriors** insert the amount of warrior token that you want effectively deploy on the battlefield, this amount must be minor or equal to the amount set in the approve function

Read / Write Contract

0x553B4546D26F383d4f4a056B7f50DaDFf07FB252

assignPersiansToBattle ▾

_warriors uint256

1000000000000000000

The amount to insert in **_warriors** field is relative to the decimal for each token, refer to table as example

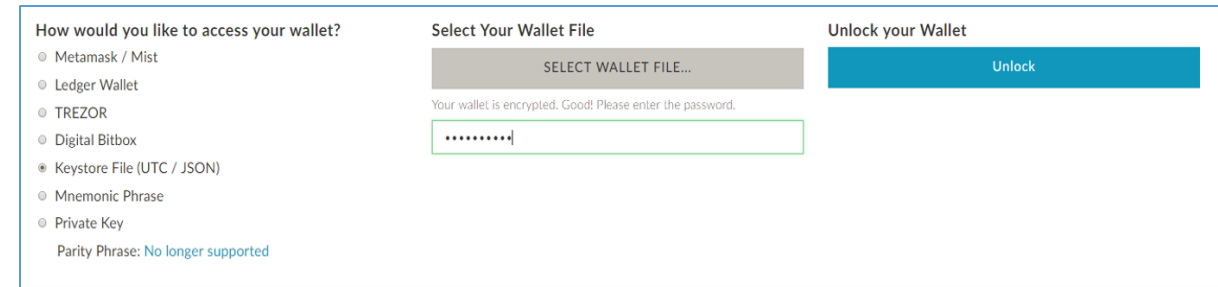
Token	Decimal	_warriors field for 1 token	_warriors field for 10 token	_warriors field for 50 token	_warriors field for 100 token
Persian	18	1000000000000000000	1000000000000000000	5000000000000000000	10000000000000000000
Immortal	0	1	10	50	100
300	18	1000000000000000000	1000000000000000000	5000000000000000000	10000000000000000000
Athenian	18	1000000000000000000	1000000000000000000	5000000000000000000	10000000000000000000

So, for example, if you want to deploy 123 PRS on battlefield you need insert 1230000000000000000 in **_warrior** field

Step 12

Load your wallet using any one of the methods listed, for this example we used the Keystore File:

- Select the wallet file from your file system with the grey **SELECT WALLET FILE...** button
- Enter your password and click the **unlock button**.



How would you like to access your wallet?

- ◉ Metamask / Mist
- ◉ Ledger Wallet
- ◉ TREZOR
- ◉ Digital Bitbox
- ◉ Keystore File (UTC / JSON)
- ◉ Mnemonic Phrase
- ◉ Private Key
- Parity Phrase: No longer supported

Select Your Wallet File

SELECT WALLET FILE...

Your wallet is encrypted. Good! Please enter the password.

.....

Unlock your Wallet

Unlock

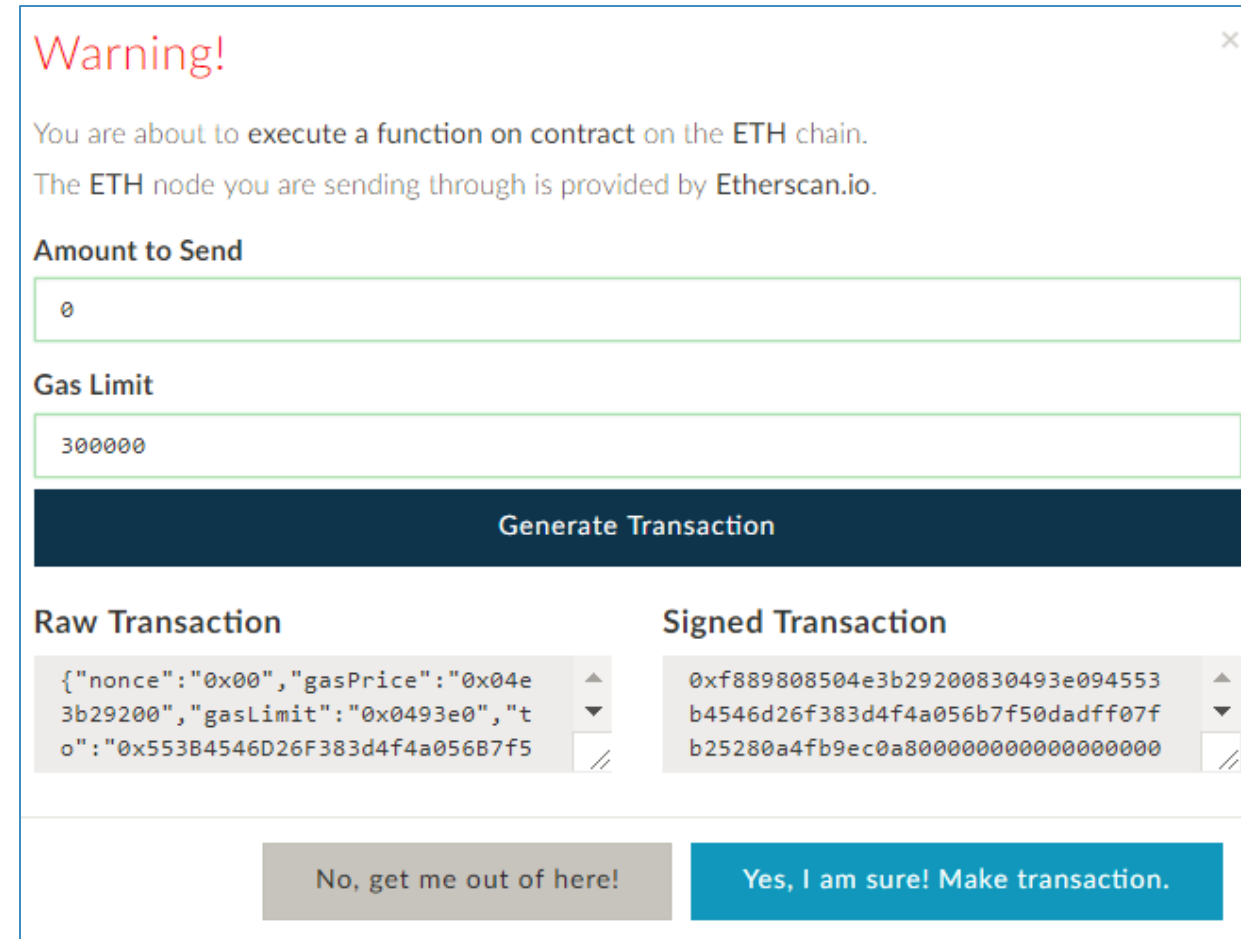
Step 13

Click on the write button, a popup will appear:

- leave the value of **Amount to Send to 0 ETH**
- allow the wallet to suggest a Gas Limit. If it does not autopopulate, enter 50000 or more. If your TX fails, increase Gas Limit.

Step 14

Click on the **Generate Transaction** button, wait a while, when ready click on the blue button **Yes, I am sure! Make transaction**.



Warning!

You are about to execute a function on contract on the ETH chain.
The ETH node you are sending through is provided by Etherscan.io.

Amount to Send

0

Gas Limit

300000

Generate Transaction

Raw Transaction

```
{"nonce": "0x00", "gasPrice": "0x04e3b29200", "gasLimit": "0x0493e0", "to": "0x553B4546D26F383d4f4a056B7f5"}
```

Signed Transaction

```
0xf889808504e3b29200830493e094553b4546d26f383d4f4a056b7f50dadff07fb25280a4fb9ec0a80000000000000000
```

No, get me out of here!

Yes, I am sure! Make transaction.