

# RGi Token

## System Overview

### Service Overview

The RGi.Token.WebAPI is a .NET Core Web API application, deployed as linux docker container, on the environment's linux application servers. It acts as a micro-service for supplying and validating tokens generated by SharpGaming, and allows for security and session control.

The gateway runs on port 5092 (blue)/5093(green)

Swagger contains detailed information on the available operations, data-types and payloads <https://prod01-rgi-token.gib.prod.sharpgaming.ltd/swagger>

Information on docker commands can be found [here](#).

### Production Details

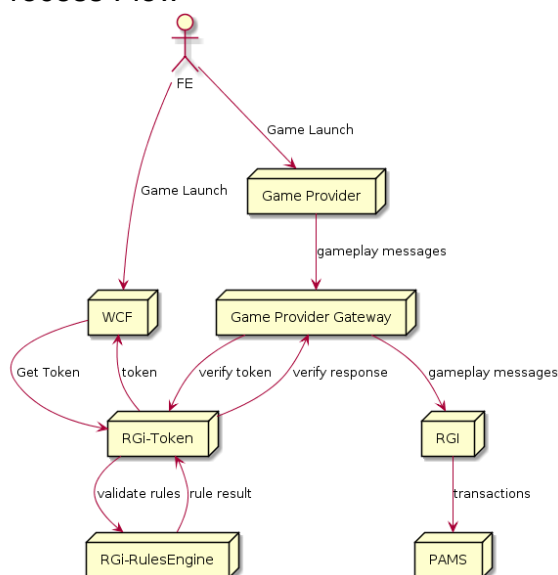
See [Production Details](#)

### Contributing Applications and OS services

The service requires access to a rgitokendb on the database server. tokens are registered and validated in the token table.

Rgi-token also relies on decisions from rgi-rulesengine, using the RTS14C.json rules. Decisions are formed based upon the transaction type, whether the token is deleted, whether it exists and whether it has expired through age. The outcome of the decision for Stake (withdrawal) requests, is to invalid all other player tokens. NonStake (deposit) requests are permitted to be processed by the rest of rgi, to prevent blocking a player's payout.

### Process Flow



### Environmental Variances

RGi-Token is deployed as the same docker service across all environments. The only variance is between game providers. Netent, RGS, DWG and Inspired are currently the only providers using this service, until the other game providers are migrated to this way of working.

### Hours of Operation (scheduled or 24/7)

24/7. The RGi-Token will need to have an SLO of 99.999 uptime (6 second per week downtime). It's critical that if this service is disrupted, either during deployments or when downstream services are impacted, we can automatically manage and report on the performance before we incur downtime.

### Expected traffic & load

#### Contents

- [System Overview](#)
  - [Service Overview](#)
  - [Production Details](#)
  - [Contributing Applications and OS services](#)
  - [Process Flow](#)
  - [Environmental Variances](#)
- [Hours of Operation \(scheduled or 24/7\)](#)
- [Expected traffic & load](#)
- [Resilience / High Availability](#)
- [Infrastructure](#)
- [System Configuration](#)
  - [Logging](#)
  - [TokenConfigurations](#)
- [Monitoring & Alerting](#)
  - [Splunk](#)
  - [New Relic](#)
  - [Health Checks](#)
- [Operational Tasks](#)
  - [Repositories](#)
  - [Deployment](#)
  - [Troubleshooting](#)
- [Maintenance Tasks](#)
- [Failover & Recovery procedures](#)
- [Contact Details](#)
- [Document Control](#)
  - [Document Information](#)
  - [Version History](#)

Request per second:  
Weekday: 120  
Weekend: 156  
Royal Ascot, Cheltenham, Grand National: 250

See [Games Performance Testing Requirements](#)

## Resilience / High Availability

**Availability:** 24/7

**Resilience:** The component is clustered across the *RGiApps* linux boxes.

## Infrastructure

See [RGI Components - Production Details](#) and [Gaming Components Runbooks](#)

## System Configuration

### Logging

Uses Serilog. See here <https://github.com/serilog/serilog-settings-configuration>

Key items are

Setting	Description
SplunkHost	Splunk URL to log to
EventCollectorToken	Http Collector token value, used for identifying requests and indexing in splunk
MinimumLevel.Default	The current log level

### TokenConfigurations

Array of settings, one per company/franchise, since the settings can/will be different per franchise.

Name	Friendly name for this group of settings - i.e. OddsKing
Company	Company Id
Franchise	Franchise Id
ProviderConfigurations.Provider	The Provider name for these settings i.e NetEnt
ProviderConfigurations:SessionExpiryMinutes	How long the token is lasts before being rejected
ProviderConfigurations:SessionMaxAgeMinutes	How long the token will last before being "soft" deleted

## Monitoring & Alerting

### Splunk

In Splunk data from rgi-token can be found using the query term

Splunk query term
<code>index = "rgi-token"</code>

### New Relic

The service can be configured to log to New Relic for performance monitoring. This is enabled via a container environment variable, which is set from cmdb. Setting `rgi-token.newrelic.enabled = 1` will enable new relic output.

## Health Checks

The service has asp dot net core health check ability, performing a GET <https://prod01-rgi-token.gib.prod.sharpgaming.ltd/health/all> will return information and the status of memory, version info, the current configuration settings and timing (cpu) information.

## Operational Tasks

### Repositories

Repositories
<a href="https://bitbucket.org/sharpgaming/rgi-token/">https://bitbucket.org/sharpgaming/rgi-token/</a>

### Deployment

Component	Pipeline	Notes
Deployment	DevOps/CMDB/Backend-Deploy/Rgi-Token-Docker-Deploy master * prod01-blue	Blue/green possible
Rgi Token DbMigrations	DevOps/CMDB/Backend-Deploy/Rgi-Token-DbMigrations-Deploy master * prod01	

### Troubleshooting

Each game provider may differ slightly, but in general a token is issued during game launch, within the WCF. This sends a URL back to the game provider. The game provider then sends game play messages to the game provider gateway. i.e. rgi-netent which then interrogates rgi-token to determine if the token is still valid.

DWG, RGS and Inspired send an initialisation (init) message, that is used to initialise the token, which marks it as being used. If someone tries to launch a game using the same url, the fact it has been initialised already prevents and secures the game launch process.

Netent, don't have a unique initialisation message, so the token is initialised at the same time it is created.

To view the container logs use docker logs <container id> which you would have got from docker ps. You can use the -n switch to view the last n lines. I.e. docker log <container id> -n 100.

## Maintenance Tasks

No maintenance tasks are required.

## Failover & Recovery procedures

No recovery process is required.

## Contact Details

- Games Team
- Dominic Torrisi, Martin Brindle (BE), Russell Cooksley (BE)




## Document Control

### Document Information

Owner	<a href="#">Unknown User (russell.cooksley)</a>
Team	Games
Status	Publish
Last Modified	22 Jun 2023

### Version History

Version	Published	Changed By	Comment
---------	-----------	------------	---------

<b>CURRENT (v. 25)</b>	<b>22 Jun, 2023 07:18</b>	 <b>Rennick Clark</b>
<b>v. 24</b>	<b>20 Jun, 2023 12:58</b>	 <b>Unknown User (russell.cooksley)</b>
<b>v. 23</b>	<b>20 Jun, 2023 12:42</b>	 <b>Unknown User (russell.cooksley)</b>

[Go to Page History](#)