



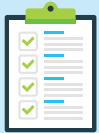


ENTERPRISE TEST DATA FEATURES

March 2019

ARCHITECTURE

The GenRocket architecture is component-based with five key components (shown below). This architecture provides complete flexibility in how complex test data challenge can be solved.

GenRocket Term	Think of it like...	Example
 Domain	A database table	A user database table
 Attribute	A column in a database table	A column in the user database table - Name, Email.
 Generator	The component that generates test data for an Attribute	A "NameGen" Generator
 Receiver	The component that formats the test data from a Generator	XML, SQL, JSON, Web Services, JDBC
 Scenario	Instructions for the GenRocket Runtime to generate test data	A user Scenario that generates user test data

SYNTHETIC DATA GENERATION

While traditional Test Data Management solutions depend on copying / virtualizing and then masking production data, GenRocket's synthetic Test Data Generation approach offers dramatic benefits including much lower cost, 100% security, and a much simpler implementation process.

Synthetically generated data has many advantages over production data in that production data is not conditioned and only contains one set of business logic - whatever the logic was in the database. Synthetic data can be conditioned and controlled so different business logic can be generated and negative as well as all kinds of positive test data can be created for testing environments.

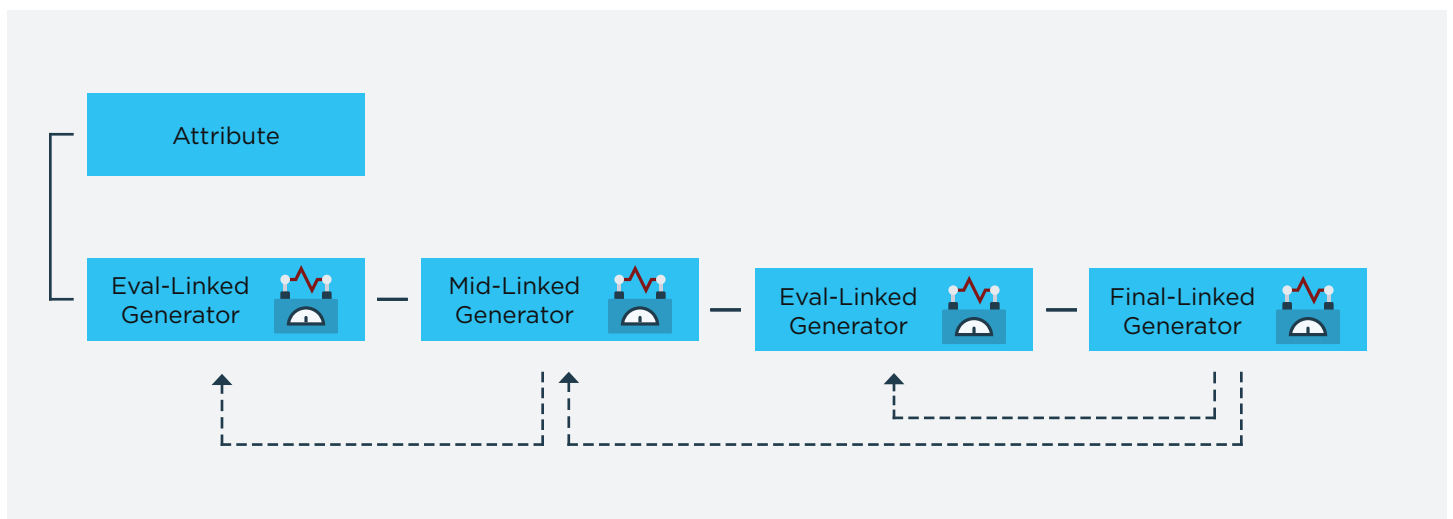
Patented Technology

The U.S. Patent was awarded in January 2017. #9,552,266 B2 for systems and methods for test data generation (test data generation with parent-child relationships).



Generators

- GenRocket currently has 222 Generators. All Generators have multiple parameters that can be configured to meet a particular test data need.
- New Generators can be added in 1 to 2 days and usually at no cost unless they are custom / proprietary in nature.
- **Unicode** (UTF-8) is supported so any natural language (Arabic, Japanese, Chinese, Cherokee, etc.) as well as many non-spoken languages (music notation, mathematical symbols, Emoticons, etc.) can be generated.
- **Linked Generators:** Generators can be combined by using the *Linked Generators* feature. As shown below, multiple Generators can be linked and their output can be queried or used so that the generated data reflects just about any business logic that is required. Linked Generators can also be saved as a "Preset" so that once a Linked Generator is configured it can be re-used.



Types of Test Data that can be generated (partial list)

1. Patterned
2. Realistic
3. Sequential
4. Random
5. Edge Case / Negative
6. Null
7. Permutations
8. Percentages
9. Calculations
10. Images
11. Dates
12. Languages
13. Emoticons
14. Credit cards
15. Identification numbers
16. Addresses (in many countries)
17. Names (in many countries)

Pattern	Realistic	Sequential	Random	Edge Case	Null
firstName1	Ms. Tereasa F. Saldana	001-01-0001	749-40-0182	749-40-0182	749-40-0182
firstName2	Mr. Everette Q. Groom II	001-01-0002	797-59-7445	797-59-7445	null
firstName3	Mr. Jules U. Hackney Jr.	001-01-0003	135-93-8060	135-93-8060	135-93-8060
firstName4	Mrs. Kristina J. Brick	001-01-0004	214-82-8447	214*82*8447	null
firstName5	Mr. Francisco M. Grimes II	001-01-0005	170-60-5224	170-60-5224	null
firstName6	Dr. Iona D. Starrett	001-01-0006	302-76-0978	302-76-0978	null
firstName7	Ms. Patricia O. Ingraham III	001-01-0007	266-20-5659	266-20-5659	266-20-5659
firstName8	Ms. Tracee M. Farah	001-01-0008	005-57-7667	005#57#7667	005-57-7667
firstName9	Mr. Alva I. Ziegler Jr.	001-01-0009	490-48-8084	490-48-8084	null
firstName10	Dr. Mike T. Youngblood II	001-01-0010	471-29-7519	471-29-7519	null

Parent-Child-Sibling relationships with Referential Integrity

- *All GenRocket Domains maintain referential integrity*

Query Generators

There are five query Generators that have different capabilities. Data can be queried from a database or a file and the data can be blended with synthetically generated data.

List Generators

List Generators allow customers to save their own custom lists in a GenRocket Generator.

DATA MASKING

GenRocket has the ability to query and mask production data. The masked data can be blended with synthetically generated data or used as is. GenRocket offers two approaches to data masking:

SDR / Synthetic Data Replacement

In the GenRocket Synthetic Data Replacement process sensitive records are identified and are replaced by fully secure synthetic data. Sensitive production data never leaves the source location so this is a fully secure process.

Obfuscation / Masking

In the obfuscation process GenRocket Generators obfuscate PII (Personally Identifiable Information) from the data set. The SDR (Synthetic Data Replacement) approach is recommended over this approach as the SDR delivers data that is higher quality test data than masked production data.

RECEIVERS

A Receiver receives generated data and morphs it into a *usable format*. There are over 44 different types of GenRocket Receivers. Receivers can be quite simple in design or more complex where multiple Receivers can be combined to format complex data; for example creating synthetic medical records in the exact format that matches the medical records in production.

Receivers can talk to just about any database, can work over Web Services and can talk to just about any kind of mainframe.

GenRocket is able to support just about any customer requirement due to the combination of Generator and Receiver components. New GenRocket Receivers can be added in 1 to 2 weeks usually at no cost unless they are custom / proprietary in nature.

Receiver Examples.

GenRocket currently offers 44 Receivers (partial listing):

1. XML (+ Nested XML)
2. JSON (+ Nested XML)
3. CSV
4. SQL
5. Delimited
6. Excel
7. Fixed File
8. Image
9. SQL
10. MySQL
11. MongoDB
12. Nested Files
13. REST
14. SOAP
15. Health Care formatted data (e.g. HL7 and EDI)
16. Financial Services formatted data (e.g. BAI2)

DATA MODEL SETUP + LIFECYCLE MANAGEMENT

GenRocket provides a logical, structured approach to setting up an organization's data model inside GenRocket. Most enterprises will have many applications /databases, will have common entities between the applications/databases (e.g. users and/or addresses in multiple systems), will have ongoing release cycles with new versions of software being released frequently and data models in many of the applications will change frequently. GenRocket is designed to support all of these needs.

- **Many applications / databases** - Each application or database will usually be reflected within GenRocket as a Project. There is no limit on the number of Projects in GenRocket.
- **Organization Variables** - Organization Variables is how GenRocket maintains consistency across Projects so that, for example, a user or an address in one Project can be consistently related to a user or an address in another Project.
- **Versioning** - GenRocket allows new Project Versions to be created as your software versions change.
- **Intelligent Automation** - GenRocket has an intelligent automation feature called "Buddy" that automates many background systems within GenRocket. Buddy allows an entire Project Version to be selected and duplicated into a new Project Version - Buddy saves a huge amount of time for this and other system tasks and greatly reduces human error.
- **Domain Refactoring** - GenRocket understands that data models are dynamic and can change frequently. When changes are made (for example - additions, deletions, changes in hierarchy) at the Domain and Attribute level those changes are automatically updated in all the Scenarios in that Project unless that Scenario is locked.
- **Data Model Setup**
 1. **XTS** - XTS stands for Extract Table Schema. In this approach GenRocket can retrieve one, many or all table schemas from a given database by connecting directly to the database via JDBC to extract the database's schema. The extracted schema is saved to an encrypted file. The encrypted file can then be imported from the GenRocket web application to automatically do the following:
 - Import the Data Schema
 - GenRocket XTS and Data Warehouse setup the Domains, Attributes and add Generators automatically
 - Use GenRocket Wizard to establish Parent-Child-Sibling relationships for the Domains
 - Swap out auto-assigned Generators for correct Generators and modify Generator parameters
 2. **DDL** - DDL or Data Definition Language is a method that GenRocket uses to import one or multiple Domains from DDL and quickly set up the data model.
 3. **CSV Import** - Domains can be created via a CSV import
 4. **ScratchPad** - Domains can quickly and easily be created through the Scratchpad feature.
 5. **Domain Presets** - Commonly used Domains can be created by using the Domain Preset feature. Domains can also be saved as a Preset within an Organization so that the Preset can be re-used in any Project.
 6. **Project Presets** - Entire GenRocket Projects can be saved as a full or partial Project so that they can be re-used on another Project within an Organization.



GENROCKET API'S

GenRocket's Runtime and REST API's allow organizations to leverage the power of GenRocket's data generation engine in an infinite number of flexible ways. The Runtime API can be particularly powerful when used in end-to-end / workflow testing applications where data needs to be dynamically changed in real time based on rules and logic. For example, end to end testing of a credit card payment system or an airline flight tracking system. The REST API is powerful when organizations want to hide the GenRocket GUI and use GenRocket functionality through their own testing tool interface.

The GenRocket Runtime API

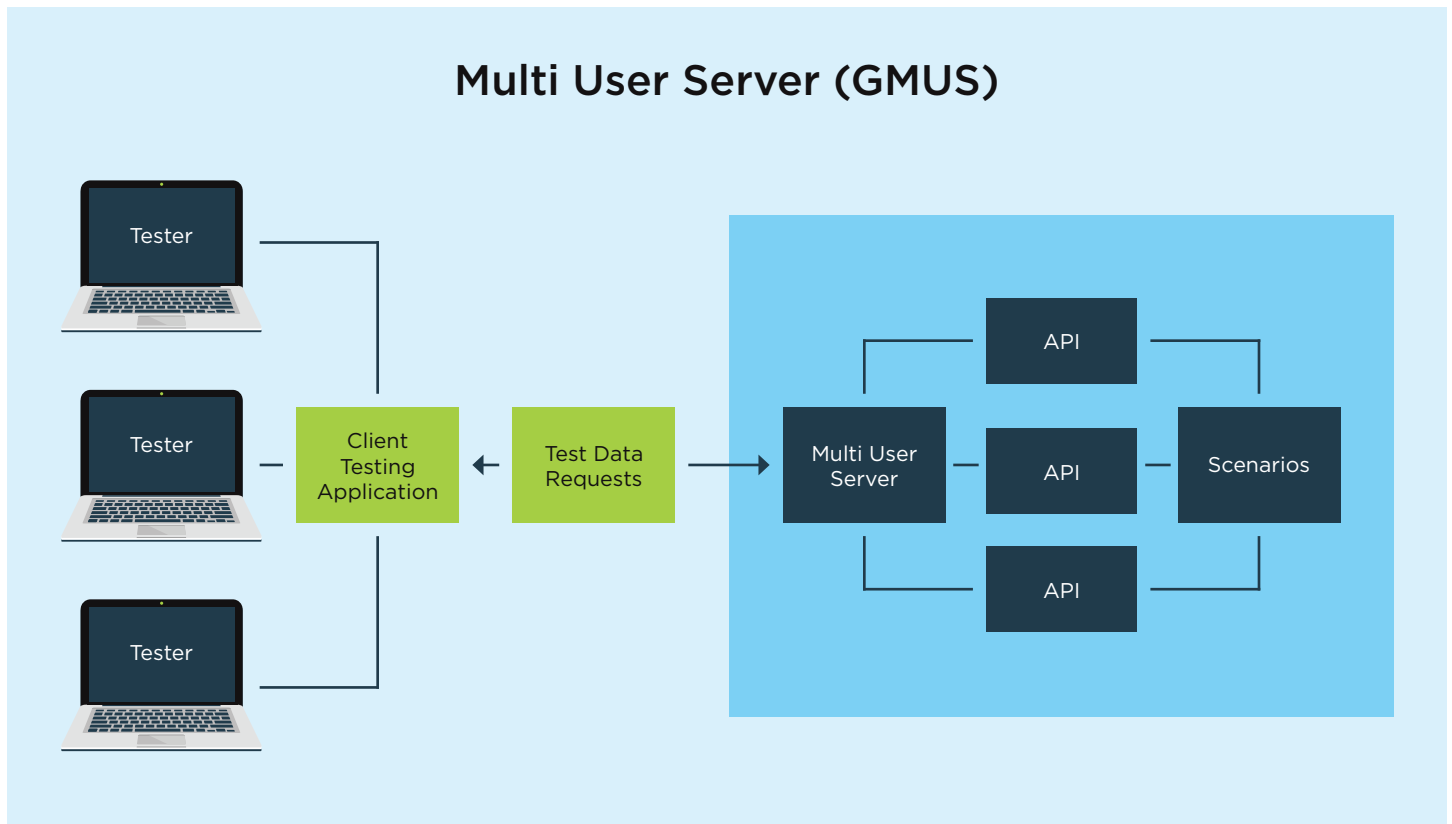
- The GenRocket Runtime application interface (API) allows programmers to access the GenRocket Runtime to directly modify and create scenarios from their own programs. Programmers will have the following control over a Scenario:
 1. Modify Domains, Attributes, Generators and Receivers
 2. Add Domains, Attributes, Generators and Receivers to an existing Scenario
- There are three interfaces for accessing the GenRocket Runtime API
 1. The GenRocket Binary Runtime
 2. The GenRocket Realtime Socket Engine
 3. The GenRocket Realtime REST Engine

The GenRocket REST API

The GenRocket REST API gives advanced, licensed GenRocket users the ability to access all major GenRocket web functions from their own custom built in-house applications. Keep your customers solely within the experience of your own custom in-house application by seamlessly and automatically managing the creation and updating of GenRocket Domains, Attributes, Receivers & Scenarios via our GenRocket Restful web services.

SELF SERVICE TEST DATA

Many organizations have their own home-grown / in-house developed testing applications and they want their testers to request test data from within that testing application. GenRocket's Multi-User Server (GMUS), used in conjunction with the GenRocket REST API, allows large volumes of simultaneous test data requests to be managed from hundreds or thousands of testers.



INTEGRATION INTO CI/CD PIPELINES AND JENKINS

Most testing organizations today are moving in the direction of Continuous Testing and many are using Jenkins to assist with the testing automation process. GenRocket Scenarios can generate small quantities of test data for functional testing in about 100 milliseconds so GenRocket is ideal for CI/CD pipelines and Jenkins.



GENROCKET PARTITION ENGINE

The GenRocket Partition Engine is used to generate hundreds of millions, to billions or even trillions of rows of test data in a short period of time. This is accomplished by partitioning the load to generate huge amounts of test data across multiple GenRocket instances running within a given server. When generating enormous amounts of test data, the load can be partitioned across multiple servers with each running multiple GenRocket instances.

On a given computer, depending on the number of CPU Cores, Memory and Operating System (OS), GenRocket may generate between 10,000 to 15,000 rows of test data per second. If we base our calculations on the idea that GenRocket is running on *one very slow computer*, then the following test data generation calculations can be approximated:

- 10,000 rows every second
- 600,000 rows per minute
- 1,000,000 rows every 1 minutes and 40 seconds

As seen from the approximations above, generating test data greater than 10 million rows takes far too much time. Thus, depending on the number of instances partitioning and generating test data across multiple servers, it is possible to drastically reduce the amount of time to generate the required test data.

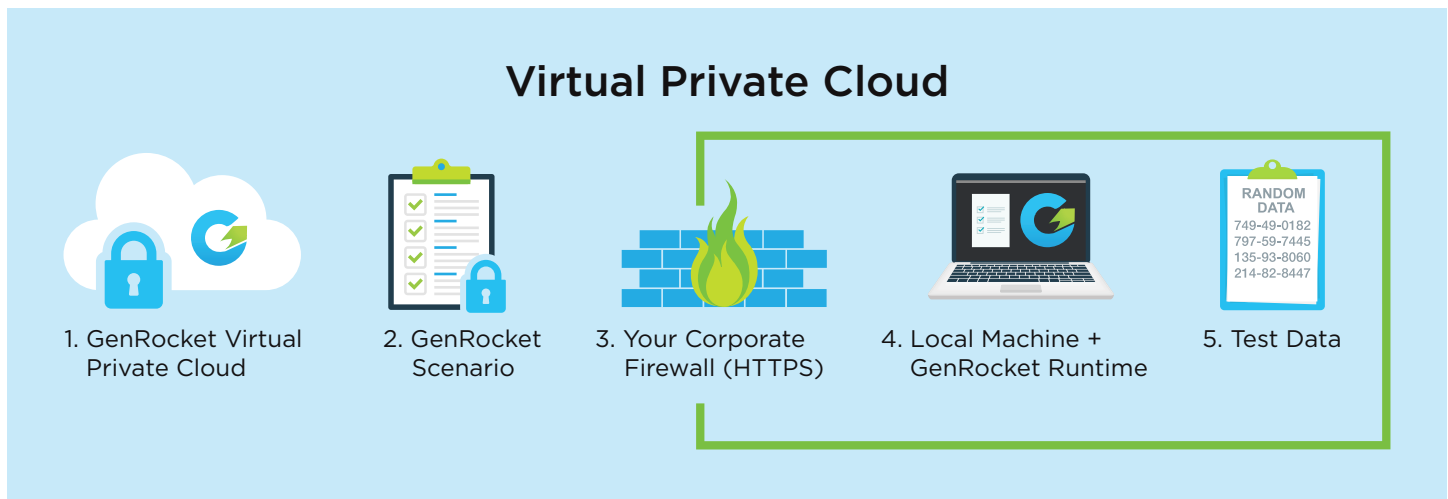
If 1,000,000,000 rows of test data were generated on 10 servers each running 10 partitioned GenRocket instances (using the Partition Engine), it would take approximately 18 to 20 minutes to generate all 1,000,000,000 rows of test data.



HOSTING

Virtual Private Cloud

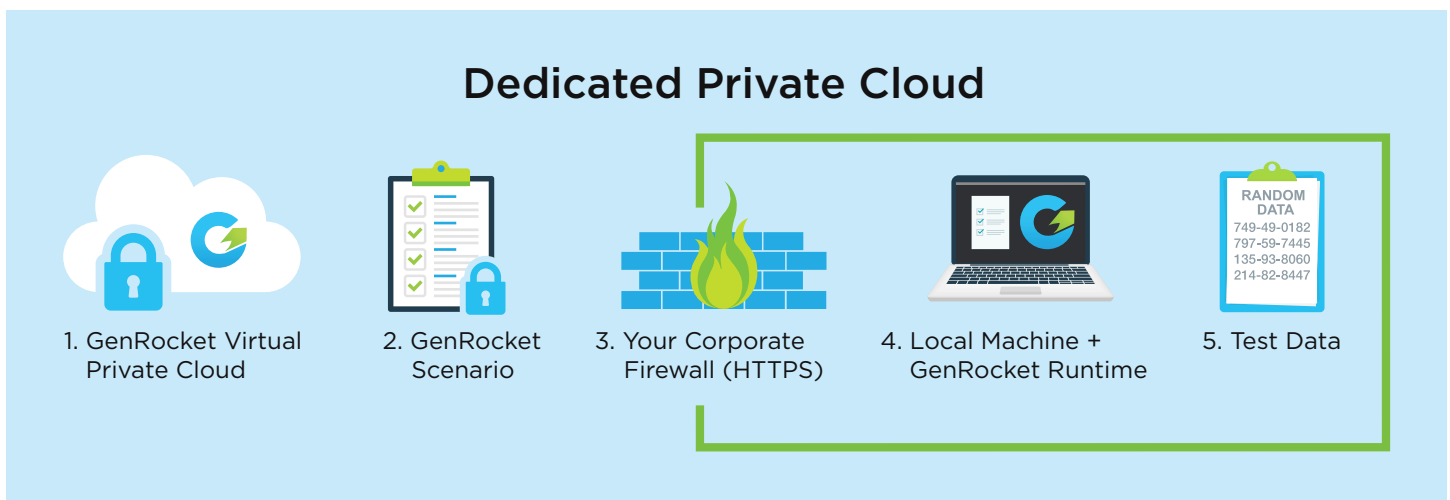
- This environment is hosted on AWS servers, is secure, redundant, backed up, managed 7 X 24 X 365 and updated daily and weekly depending on the frequency of software releases and the cost of this hosting model is included in the GenRocket software license.
- Independent security penetration tests have been run to validate the security of the hosted system.
- *This is the recommended hosting approach for all smaller GenRocket customer deployments*



- As GenRocket is a synthetic test data system it is secure by nature
- No customer data is stored in the GenRocket system
- GenRocket Scenarios are encrypted and there is no data in the Scenarios
- All data generation Scenarios run inside the secure corporate firewall
- No customer data is ever accessed from outside the corporate firewall
- User licenses are validated to ensure all users on the system are valid
- All passwords are SHA-256 one-way encrypted
- There are four levels of permissions on the system
- Only authenticated and licensed users can run GenRocket Scenarios
- The GenRocket Runtime that runs the Scenarios is a secure Java program
- All GenRocket JAR's are validated with a checksum
- SSO (Single Sign On) is available to authenticate users

Dedicated Private Cloud

- This environment is hosted on AWS servers, is secure, redundant, backed up, managed 7 X 24 X 365 and updated daily and weekly depending on the frequency of software releases; the cost of this hosting model is in addition to the GenRocket software license as the hosting and support is exclusively provided just for one client.
- Independent security penetration tests have been run to validate the security of the hosted system design.
- As GenRocket is a synthetic test data system it is secure by nature
- No customer data is stored in the GenRocket system
- *This is the recommended hosting approach for all larger more intensive GenRocket customer deployments*



- GenRocket Scenarios are encrypted and there is no data in the Scenarios
- All data generation Scenarios run inside the secure corporate firewall
- No customer data is ever accessed from outside the corporate firewall
- User licenses are validated to ensure all users on the system are valid
- All passwords are SHA-256 one-way encrypted
- There are four levels of permissions on the system
- Only authenticated and licensed users can run GenRocket Scenarios
- The GenRocket Runtime that runs the Scenarios is a secure Java program
- All GenRocket JAR's are validated with a checksum
- SSO (Single Sign On) is available to authenticate users