Windocks

Container based test data management

A new approach to test data management

Containers are foundational to DevOps today, and architects increasingly ask how containers are applied to test data management (1, 2). A new container based test data repo delivers secure, versioned, virtualized databases in seconds, with built-in data masking, sub setting, and synthetic data. Container based test data management (TDM) differs fundamentally from existing storage based systems (3).

Industry-standard APIs supports any enterprise or cloud block, file, and object storage, with any data masking, sub setting, and synthetic data. A container based test data repo supports existing systems, and requires no new infrastructure or TDM tooling.

A container based test data repo is piloted in hours and abstracts the complexities of containers, commands, and APIs. A single VM supports up to 100 virtualized data environments, generating an average 50 to 70% reduction in test VMs, storage, and maintenance costs (4, 5).

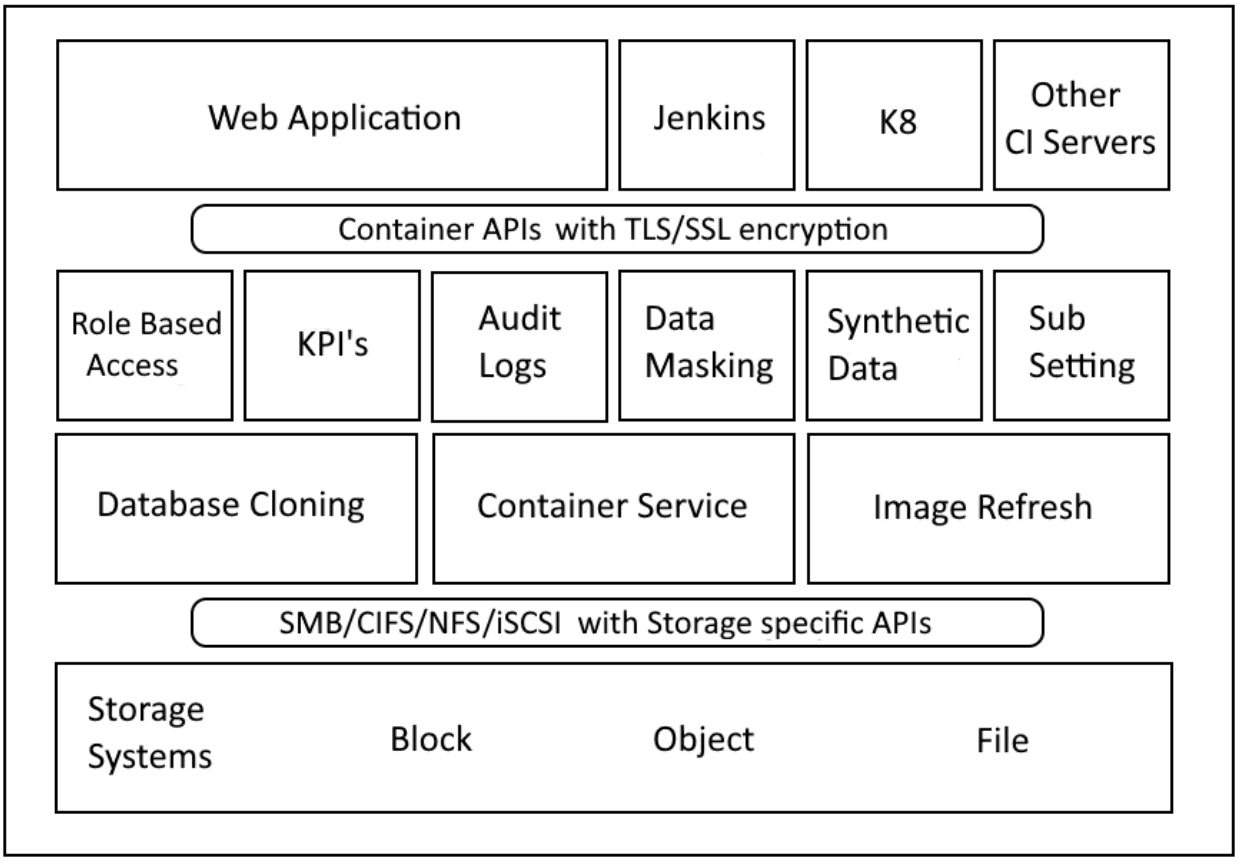

Image 1: a layered architecture of a container based versioned, virtualized test data repo.

Test data management considerations

Test data management provides virtualized masked databases in seconds, for high quality testing, release quality, and throughput. Additional goals include data governance and security, production database debug and support, and cost savings in testing infrastructure.

This paper outlines the architecture and considerations for test data management and reflects **Windocks** experience in container based test data delivery. Container based test data management introduces new capabilities and considerations that are highlighted in this paper, to help test and QA professionals evaluate and form TDM plans.

DevOps and TDM best practices seek to **minimize infrastructure dependencies**, with support for any on premise or cloud storage. **Agility** considers the time and effort to deploy and pilot a TDM solution. **Flexibility** addresses non-testing needs such as near real-time production database delivery. **Openness** includes self-curated or third party data masking, sub setting, or synthetic data tooling. **Cloud-native** support adds containers as test data environments, along with conventional instances. **Data governance, security, and collaboration** are enhanced with role based access controls, audit logs, and real-time visibility of data environments and image updates.

Building blocks and technologies

Most TDM systems limit database virtualization to a single-sourced proprietary storage system, and this paper introduces an approach that supports any enterprise or cloud block or object storage, as well as Windows Virtual Hard Drives. Virtualized databases are delivered to containers or conventional workstations and instances (VMs), and orchestrated and managed with bespoke scripts or a **Windocks** TDM server.

A versioned, virtualized test data repo delivers masked databases in seconds, with each using only 40 MB.

In addition to database virtualization, TDM systems include a range of services including container services, role based access controls, dashboards, image monitoring, and audit logs.

Building block #1: Containers and API

The original Docker open source project, subsequently renamed the Moby project transformed the technology industry and is foundational for DevOps and cloud native systems today. Containers are a defacto standard for front-end and middle tier application DevOps, and support for databases is keeping pace. **Windocks** supports containers for all releases and

editions of SQL Server 2008 to 2019, and Postgres, MySQL, and other databases. **Windocks** SQL Server containers also support Windows infrastructure, including Active Directory, Windows Authentication, encryption, and other systems.

Database containers are compatible with and provide advantages over conventional instances:

- Containers package applications with libraries and dependencies to ensure applications run reliably on varied systems. A database image will run reliably on any test server.
- Identical containers are created and delivered in seconds. Identical runtime environments support high quality testing, free of instance configuration drift. A single image is maintained, reducing support and maintenance involved with conventional instances.
- A VM can support up to 100 simultaneous containers, allowing for consolidation of VM and instance sprawl. Instances combined with storage savings yields a 50 to 70% reduction in infrastructure costs, and reduces administrative overhead.
- The Docker based API supports superior automation, enabling environments to be created, tested, and deleted as part of an automated pipeline.
- Remote teams and Work-From-Home (#WFH) benefit with environments replicated and shared in seconds.

> Should I be concerned that we lack Docker experience?
>
> No, a web application abstracts complexities of containers and commands. It takes just hours to be up and running and productive.

Building block #2: Database virtualization and cloning

Database virtualization delivers writable databases in seconds, with each consuming a small storage footprint. Virtualization is supported by block level storage systems from NetApp, Pure Storage, and others, as well as Windows Virtual Hard Drives (VHDs).

VHD virtualization begins with a full byte copy of the database environment that is copied or restored to a VHD image. The image is cloned via differencing disks, and databases are attached to instances or containers.

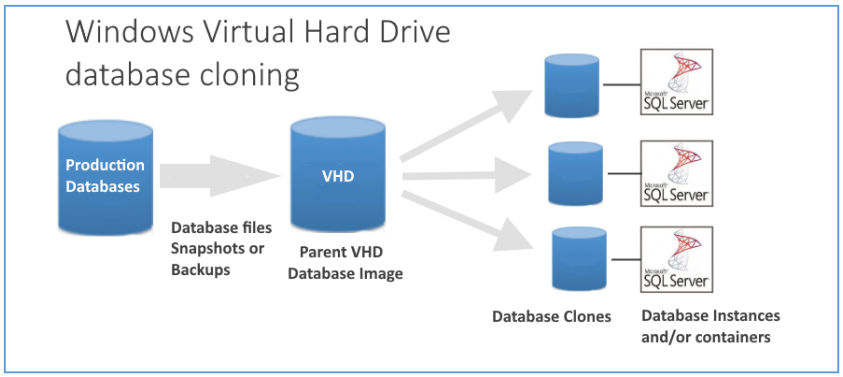

Image 2: Windows Virtual Hard Drive based database cloning.

Block storage virtualization utilizes a snapshot of the source database environment. The snapshot is "thin volume" cloned, and databases are mounted and attached to the target container or instance. Most block level storage systems include restful API support, but many existing arrays are limited to scripting for creation of snapshots and thin volume cloning.

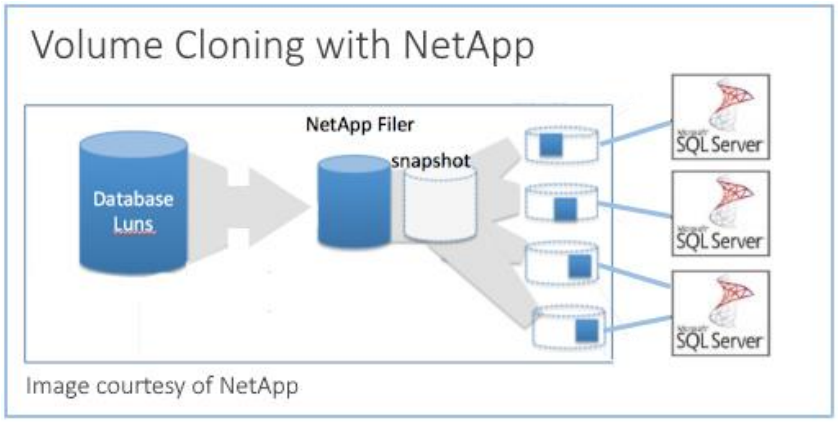

Image 3: Storage array volume cloning for delivery of database environments.

Windocks supports both block based and VHD virtualization, with database delivery to containers and conventional instances. Both technologies support delivery in seconds, with each consuming only 40 MB on delivery. The parent snapshot or VHD supports read data access, with writes supported by Copy on Write or Re-direct on Write algorithms. Storage consumption for non-production use is low, and database virtualization is proven and popular for development and testing.

Windocks VHD images are updated with transaction log backups, for near real-time delivery of production databases for support and debugging, reporting, and DR and migration readiness testing.

Database virtualization delivers a number of test management benefits:

- A 99% reduction in storage consumption for database environments.
- Testing with complete database environments can be expanded, improving test results and release quality.
- Identical databases are delivered in seconds, improving test throughput and automation.
- Database cloning supports self-service and automated pipelines on a single VM.
- Images build in data masking, sub setting, and synthetic data for complete data environments.
- Remote dev/test collaboration is enhanced with environments available in seconds.

> Tip: avoid staging servers.
> Most TDM systems require dedicated "staging" servers to deal with backups, or to create/apply masking.

Building block #3: Test data management services

Test data management requires system scalability, usability, security, and management. A secure web UI provides role based access for self-service, and automated pipelines support via restful APIs. Network traffic is protected with TLS/SSL encryption. The web UI also provides administrators a real time view of provisioned environments. Lower level VM and instance sprawl is typically reduced by 5 to 10:1 with database containers running on a central server.

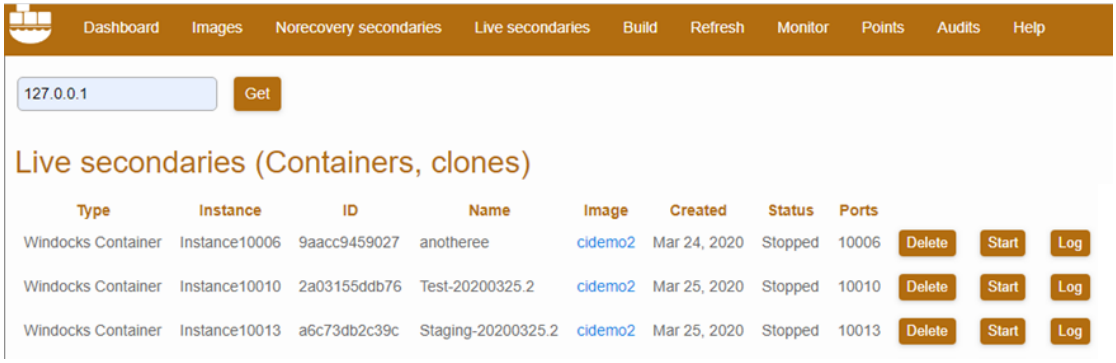

Image 4: a web UI supports self-service and a real time view of environments.

Custom images are built with plain text dockerfile. User credentials are protected with encrypted secrets, and scripts are applied to static images or applied during database clone delivery. Third party masking, sub setting, and synthetic data are also supported with batch files and executables.

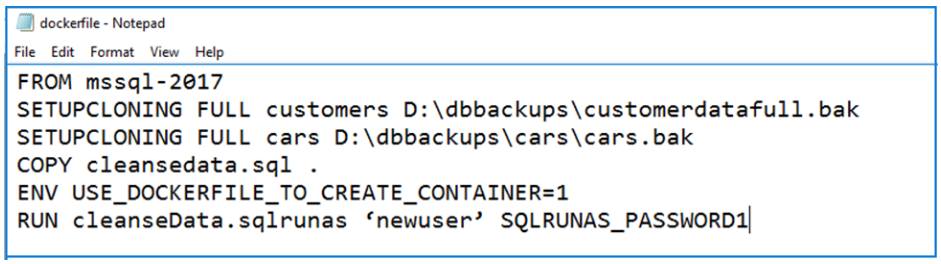

Image 5: Plain text dockerfiles define database images.

Windocks VHD images can optionally be updated with transaction log backups as needed, for delivery of near real-time production databases. Real-time update monitoring is also provided.

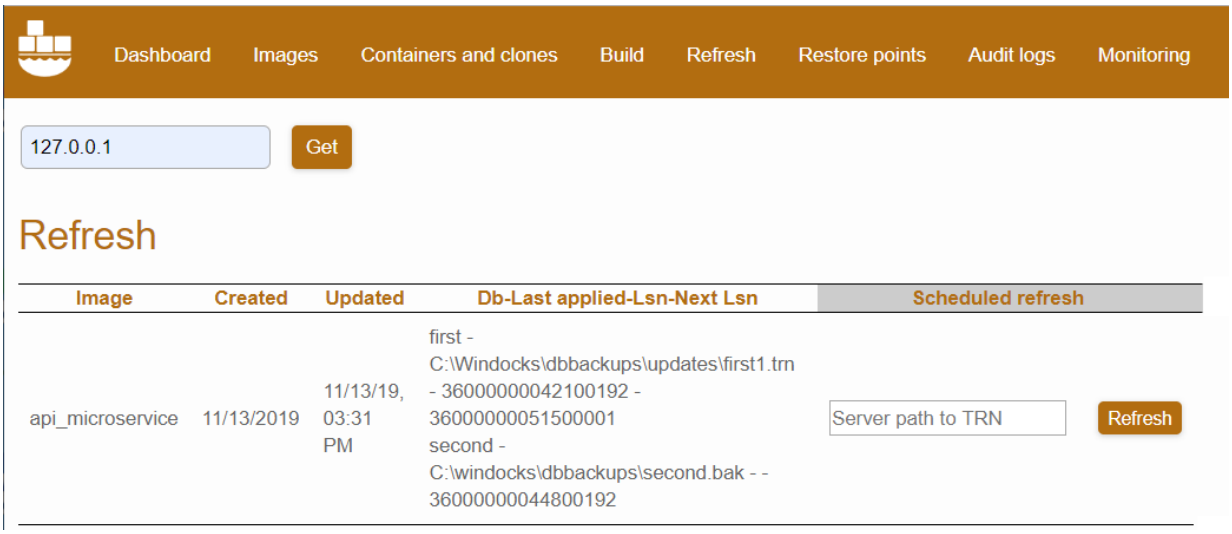

Image 7: VHD database images can be updated manually or with automated schedules.

Feedback on key performance metrics supports team collaboration.

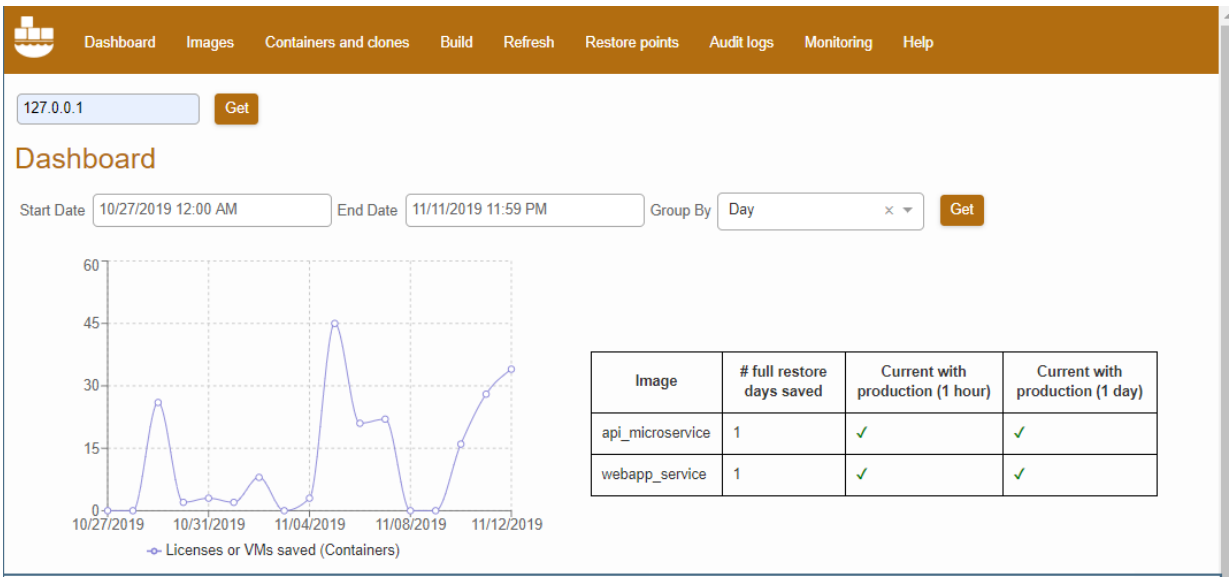

Image 8: a KPI dashboard

Compliance, security, and audits

While the technologies and security discussed earlier includes data and credential encryption, data obfuscation, Active Directory, and other security systems, the National Institutes and Standards for Technology (NIST) has identified concerns on container security (6). Security concerns include the use of public images, layered dockerfiles, namespace and isolation, credential management, and others.

The industry has addressed many of these issues, and **Windocks** database containers address these concerns with containers created by cloning a locally installed SQL Server instances, and Postgres and MySQL binaries. **Windocks** SQL Server containers are indistinguishable from a conventionally installed SQL Server instance, with each registered in the Windows Registry and WMI namespace. As **Windocks** SQL Server containers are conventional instances, they support Active Directory and Windows Authentication, and other infrastructure and applications. **Windocks** SQL Server containers also support Linked Servers, Distributed Transactions, Third Party Key Managers, and other enterprise needs (6).

The **Windocks** management server includes an Audit Log, and role based access controls (RBAC), for compliance reporting and audit support (7).

NIST container security concerns include:

Public image repositories are a concern for unvetted public contributions. **Windocks** supports enterprises with internally hosted and approved SQL Server and database configurations.

Layered dockerfile systems are a concern as they combine operating system, libraries, and application code. This complexity has led to many "container scanning" tools. **Windocks** images do not include Operating System components, and are based on previously configured, licensed, and installed instances.

Namespace and container isolation is a concern from early docker containers that allowed container users to run as admin. **Windocks** SQL Server containers rely on the proven SQL Server namespace, and security controls.

Credential management is a concern when passwords are included in dockerfiles in plain text. **Windocks** includes encrypted "secrets" to protect credentials.

Authentication is a concern as docker containers may not be authenticated on the enterprise network. **Windocks** includes support for Active Directory and Windows Authentication.

Encrypted network traffic is supported with TLS and SSL.

Malicious runtimes is a concern that .NET or other executables could support a malicious attack against the host. **Windocks** addresses this by enabling a secure "SQL sandbox" mode that precludes the use of .NET or other executables.

Pilot a container based TDM

In addition to a powerful set of capabilities, a container based test data repo can be installed and running in minutes, and is an affordable for almost any budget. If you are interested to learn more, or pilot the software, visit www.windocks.com and inquire about a free supported pilot.

End notes:

1) DORA, State of DevOps 2019, page 18.
2) DORA, State of DevOps 2019, page 19 and 25.
3) DORA, State of DevOps 2019, page 35.
4) https://www.docker.com/blog/containers-replacing-virtual-machines/
5) https://www.networkworld.com/article/3126069/which-is-cheaper-containers-or-virtual-machines.html
6) NIST SP 800-190, https://nvlpubs.nist.gov/nistpubs/SpecialPublications/NIST.SP.800-190.pdf
7) https://windocks.com/files/SQL-Server-Container-Security-NIST-SP-800-190.pdf