Yellowbrick

DATA SHEET

Yellowbrick Data Warehouse Security

The data warehouse for distributed clouds

Yellowbrick is built for a world where we assume everything is “private by default.” No public buckets are enabled by default to gain access to data, no built-in guest users are present, and strict access must be granted to all data and management functionality.

Authentication

Database users can be authenticated locally or with LDAP and Active Directory. We are currently working on deep integration of OpenID Connect and SAML 2.0 multi-factor authentication (MFA) for cloud-native products (including Azure Active Directory, Okta, and Ping).

Manageability without “super users”

Yellowbrick incorporates a PostgreSQL front end for compatibility, and PostgreSQL relies on having a “super user” for administration and regular users for everything else. The super user can administer anything whatsoever, much like the “root” user on a Unix system. We deliberately split the privileges afforded to the super user into dozens of different grants to allow users to manage subsections of the database in a far more fine-grained manner—for example, the ability of a role to manage other roles, view SQL query text of other users, initiate backups, control LDAP integrations, and even the ability to grant privileges themselves all can be granted or revoked individually.

Role-based access control

Access to all database schema objects is fully role-based. Below the granularity of a table, Yellowbrick allows granting access to columns of a table.

Key features

- FIPS 140-2 compliant AES-256 encryption for data at rest, with encryption key rotation
- HIPAA and PCI-DS compliance
- SSL/TLS encryption for client communications and passwords
- No-code, on-the-fly `ENCRYPTED` column constraints
- Microsoft Active Directory and OpenLDAP integration
- Fine-grained access control on any database object
- Data masking/ tokenization for PII (through our partners)
- Threat detection (through our partners)

Encryption of data at rest

Yellowbrick stores all data fully encrypted with AES-256 using keys stored in a HashiCorp Vault. Encryption of data on cloud object stores is managed by the object store itself and ephemeral cloud storage is also encrypted and crypto-erased.

Column-level encryption and functions

Yellowbrick provides a variety of SQL functions for data encryption, decryption, and hashing using several algorithms. Individual VARCHAR columns within tables can designated as encrypted so that Yellowbrick will encrypt the data for the column as it's inserted. When a user with access to the corresponding encryption key does a query, they will see the decrypted data; however, users without access to the encryption key will see only the encrypted, scrambled data. Keys are stored in and referenced from the HashiCorp Vault.

TLS support

TLS 1.2 is supported for all traffic in and out of Yellowbrick. This is true for *DBC access and web access and all external connectivity, loading, unloading, backup, and so on. TLS mutual authentication is used for authentication of cross-database replication sessions.

PROTEGRITY

Yellowbrick has a production-quality integration with our strategic partner, Protegrity, which provides sophisticated, policy-based data protection and masking functionality that goes beyond Yellowbrick column-level encryption.