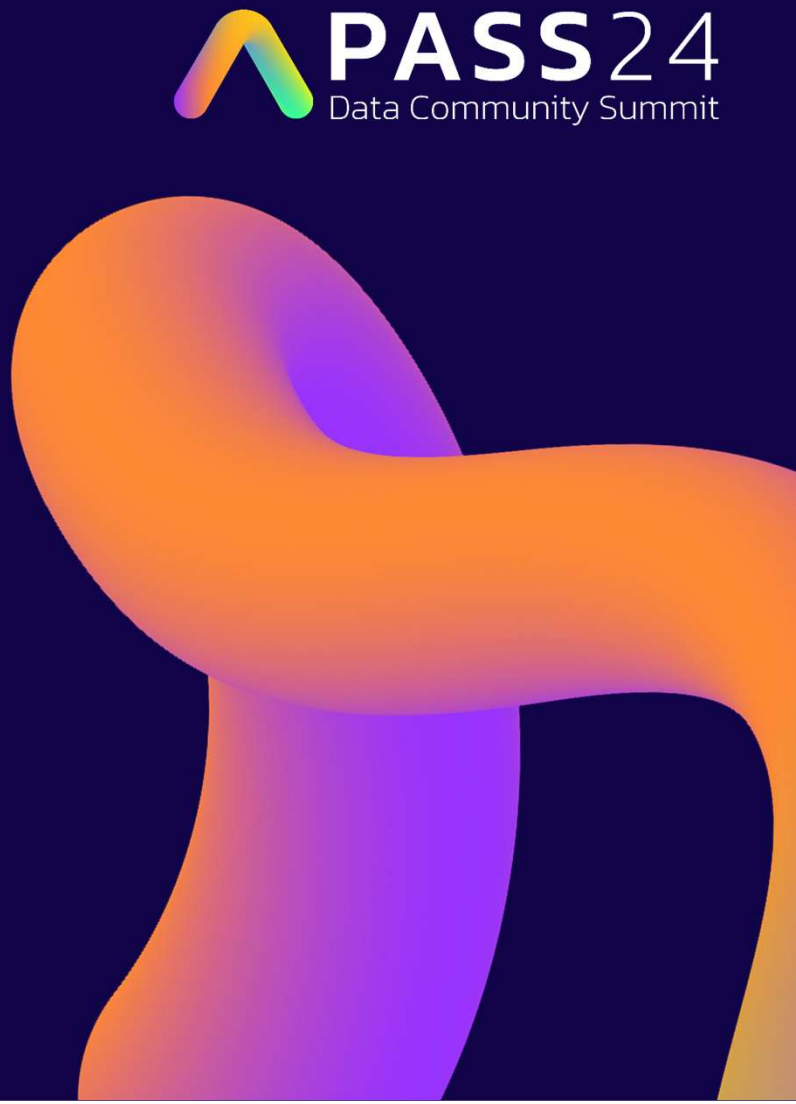
Side Channel Attacks in SQL Server



Ben Johnston

Data Architect

I-Tech Solutions, Inc.

Ben Johnston

Data Architect

I-Tech Solutions, Inc.

https://www.red-gate.com/simple-talk/author/ben-johnston/

linkedin.com/in/bensql

I've worked with SQL Server and database systems for over 25 years, building warehouses, transactional systems, reporting environments, and system integrations.

Consulting with medium and large enterprises, I specialize in database design, performance tuning, security, troubleshooting, and Azure upgrades.

The Tuple of Theseus

If each piece of data is replaced (derived) from the original set, is the dataset the same?

From a security and data perspective – this is irrelevant. The content is digital. If the data is derived – you have the data.

https://en.wikipedia.org/wiki/Ship_of_Theseus

Dynamic Data Masking
Row Level Security (RLS)

- Additional layers to your security design
- Not encryption
- Vulnerable to specially crafted attacks

Dynamic Data Masking

- Column data is obfuscated (masked)
- Depends on user security
- DBO, admins, owners, UNMASK not impacted
- All functionality supported
  - Joins
  - Functions
  - Comparisons

Masking example setup

```
ALTER TABLE sales.Customers
ALTER COLUMN CustomerName
ADD MASKED WITH (FUNCTION = 'default()')
GO

ALTER TABLE sales.Customers
ALTER COLUMN PhoneNumber
ADD MASKED WITH (FUNCTION = 'partial(6,"123-456",1)')
GO
```

CustomerID	CustomerName	PhoneNumber
74	Tailspin Toys (Indios, PR)	(787) 555-0100
84	Tailspin Toys (Aceitunas, PR)	(787) 555-0100
116	Tailspin Toys (Rafael Capó, PR)	(787) 555-0100
431	Wingtip Toys (Rosa Sánchez, PR)	(787) 555-0100
454	Wingtip Toys (Corcovado, PR)	(787) 555-0100

CustomerID	CustomerName	PhoneNumber
74	xxxx	(787) 123-4560
84	xxxx	(787) 123-4560
116	xxxx	(787) 123-4560
431	xxxx	(787) 123-4560
454	xxxx	(787) 123-4560

RLS

- Rows returned vary based on the user
- Non-prescriptive
  - Defined by you / business rules
  - Can vary by table
- Impacts all users, including admins and service accounts

RLS example setup

```
CREATE FUNCTION RLS.AccessPredicate_SupplierID_PurchasingSuppliers(@SupplierID  int)
RETURNS TABLE
WITH SCHEMABINDING
AS

RETURN

    SELECT 1 AccessResult
    WHERE IS_MEMBER('db_owner') = 1

    UNION

    SELECT
        1 AccessResult
    FROM RLS.UsersSuppliers US
        INNER JOIN Purchasing.Suppliers PS
            ON US.SupplierID    = PS.SupplierID
    WHERE US.SupplierID         = @SupplierID
        AND US.UserID           = USER_NAME()
GO
```

RLS example setup (continued)

```
CREATE SECURITY POLICY RLS.SecurityPolicy_SupplierID_PurchasingSuppliers
ADD FILTER PREDICATE RLS.AccessPredicate_SupplierID_PurchasingSuppliers(SupplierID) ON Purchasing.Suppliers
,ADD BLOCK PREDICATE RLS.AccessPredicate_SupplierID_PurchasingSuppliers(SupplierID) ON Purchasing.Suppliers AFTER UPDATE
WITH (STATE = ON, SCHEMABINDING = ON)
GO
```

RLS example (continued)

USER_NAME()	SupplierID	SupplierName	SupplierCategoryID	PrimaryContactPersonID
dbo	1	A Datum Corporation	2	21
dbo	2	Contoso, Ltd.	2	23
dbo	3	Consolidated Messenger	6	25
dbo	4	Fabrikam, Inc.	4	27
dbo	5	Graphic Design Institute	2	29

USER_NAME()	SupplierID	SupplierName	SupplierCategoryID	PrimaryContactPersonID
SideChannelUser	4	Fabrikam, Inc.	4	27

Demo – Setup

Attacks and Leaks

Attacks and Leaks Categories

- Direct attacks
- Indirect attacks
- Side channel attacks

Direct Attacks

- Disable security function (RLS)
- Change access predicate (RLS)
- Alter table definition (masking)
- Add user to authorized group (RLS / masking)
- SESSION_CONTEXT
- DBCC PAGE

Indirect Attacks

- Pre / post load
  - Load files
  - Export files
  - Source tables
  - Reports
  - Data governance tools
  - Logging

Indirect Attacks – continued

- Admin tools / elevated access
  - Backups
  - Index details
  - Trace / extended events
  - Query plans

Demo – Leaks and Direct Attacks

Side channel attacks

- Use allowed behavior in an unexpected way
- Aggregate functionality
- Derive data based on unexpected behavior
  - Carefully crafted queries
  - Metadata from error conditions

A note on scripting attacks

- My attack demonstrations are all in TSQL
  - Primarily for ease of use and demonstrations
  - Uses techniques you wouldn't use in production code
- Can be done with any language that can access SQL
  - External tools / languages may be harder to detect
  - External tools are also be easier to debug

Demo – Side Channel Attacks

Mitigations

- Organizational attitude
  - Everyone must understand the issues and protect the data
  - Data first attitude
- Training and documentation
- Lock down users
- Monitor CPU usage
- Code reviews

Mitigations

- Encrypt data that is actually sensitive
- Limit admin accounts
- Monitor / audit error #8134
  - Divide by zero in large quantities
- Leverage data governance / discovery tools
- Audit RLS / masked column rules versus expectations
- Not all attacks will trigger an error
- SESSION_CONTEXT() read_only – not CONTEXT_INFO()

Alternatives to masking / RLS

- Encryption
  - Database level
  - Application implementation
- Stored procedures
  - No direct table access
  - RLS rules still apply
- Views
  - Limit columns returned
  - Custom column logic
- Reports
- Application-level security
- Data obfuscation for non-prod

Capturing RLS attacks

- Trace / Profiler
  - On-prem only
  - Error 8134
- Extended Events
  - Error 8134

Demo – Capturing RLS Attacks

Use cases

- Layers within stored procedures
- No direct access
  - Application layer
  - Session context
  - Execute as user
- Highly trusted users
  - Already have access
  - Extra layer of protection
- Simplified report development
  - Understand potential issues
  - Actively monitor for issues

Anti-patterns

- PII
- HIPAA protected data
- Credit card / banking information
- Any data that absolutely can't be leaked
- Non-prod databases

Final Thoughts

- Security in layers
- Hire trusted admins / developers / analysts
- Careful planning is key
- Query engine behavior could change
- Direct data access is a risk
- Use the correct solution for your problem
- Monitor for breaches

Questions?

Your feedback is important to us

Evaluate this session at:

www.PASSDataCommunitySummit.com/evaluation

Thank you

I appreciate your attendance.

Ben Johnston

https://www.red-gate.com/simple-talk/author/ben-johnston/

linkedin.com/in/ben-johnston-41609011