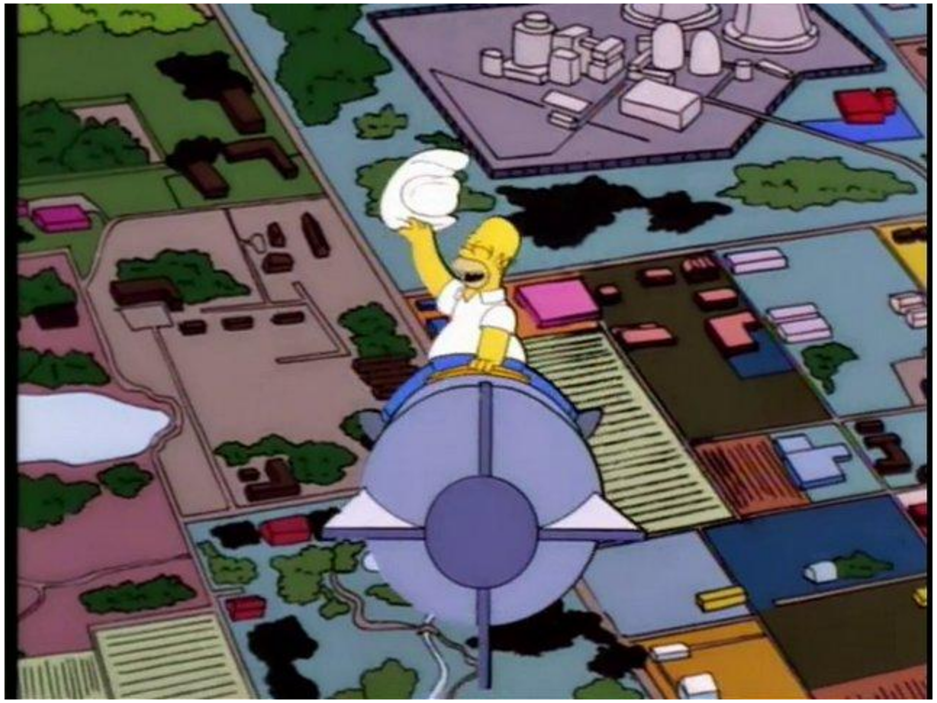
Module Signing

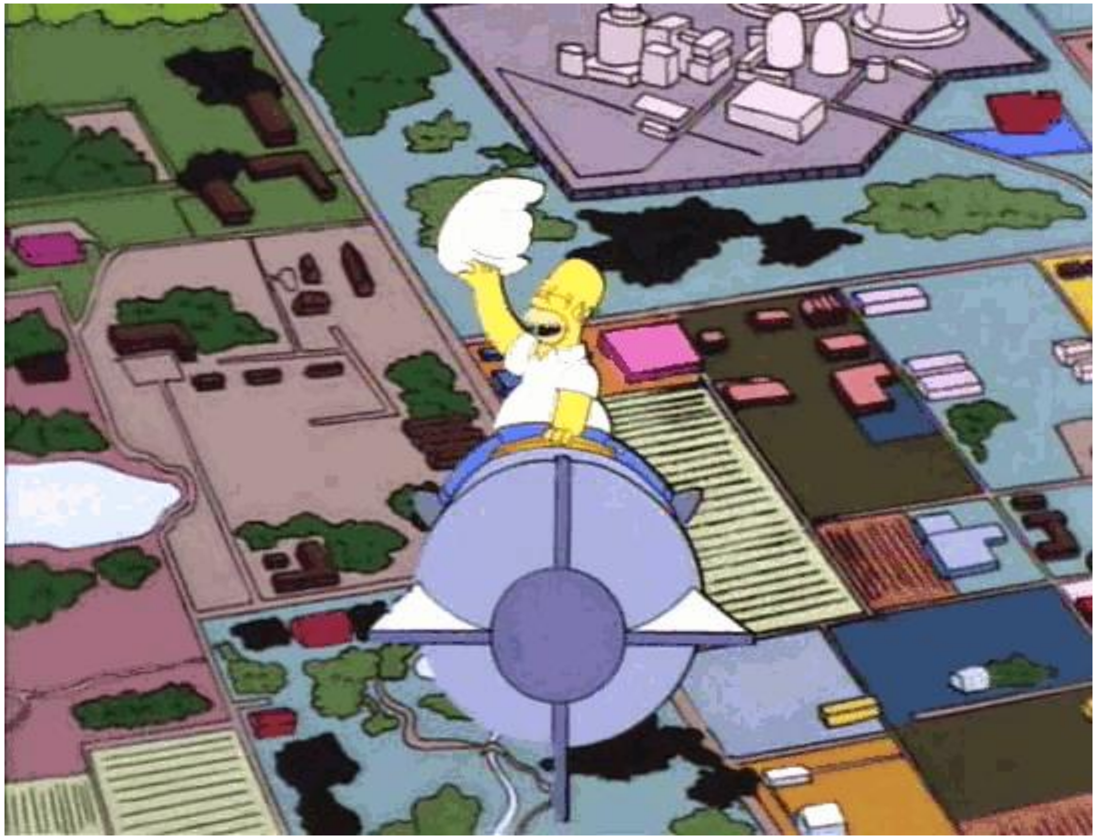
or: How I Learned to Stop Using TRUSTWORTHY / EXECUTE AS and Love Certificates



Version: 3.4-B (20191211)

Sql Quantum Lift

Reference:
Simpsons spoof of “Dr. Strangelove or: How I Learned to Stop Worrying and Love the Bomb” (1964) http://www.imdb.com/title/tt0057012/



Solomon Rutzky
Email: SRutzky@SqlQuantumLift.com
Company: https://SqlQuantumLift.com/
Blog: https://SqlQuantumLeap.com
Twitter: @SqlQuantumLeap
https://ModuleSigning.info/

C:\> whoami

- Founder of Sql Quantum Lift:
  - ➢ **SQL#** (SQLsharp) : SQLCLR library of functions
  - ➢ OmniExec : Multi-threaded, multi-server & DB query tool
- Blog: Sql Quantum Leap
- Areas of interest / concentration:
  - ➢ Module Signing, Collations & Encodings, SQLCLR
- Articles:
  - ➢ SQL Server Central (incl. Stairway to SQLCLR series)
  - ➢ Simple-Talk
- Working in IT and with databases since 1996:
  - ➢ SQL Server (since 2002), SQLCLR (since 2006), specializing in Collation & Module Signing (since 2014)
- Variety of Roles, OSes, Languages, and DBs

Solomon Rutzky
Email: SRutzky@SqlQuantumLift.com
Company: https://SqlQuantumLift.com/
Blog: https://SqlQuantumLeap.com
Twitter: @SqlQuantumLeap
https://ModuleSigning.info/

Agenda

- GOAL
  - LURN !! ;-)
  - Understand concepts and mechanisms, not how to copy/paste
- AGENDA
  - Typical Problems
  - Security Basics
  - Typical Solutions
  - Problems with Typical Solutions
  - Module Signing
    - What it is, What it can do, and Why use it
    - Asymmetric Keys & Certificates
  - Examples
  - Wrap-up / Q & A

Solomon Rutzky
Email: SRutzky@SqlQuantumLift.com
Company: https://SqlQuantumLift.com/
Blog: https://SqlQuantumLeap.com
Twitter: @SqlQuantumLeap
https://ModuleSigning.info/

You Gotta Problem?

- Common “Problem” Scenarios:
  - 😱 Need Elevated Permission that is not Grantable
  - 😱 Need Elevated Permission that is not Granular
  - 😱 Dynamic SQL
  - 😱 Cross-Database Operations
  - 😱 Allow Access to a Restricted Database
  - 😱 Loading SQLCLR Assemblies

    (especially starting in SQL Server 2017)

Solomon Rutzky
Email: SRutzky@SqlQuantumLift.com
Company: https://SqlQuantumLift.com/
Blog: https://SqlQuantumLeap.com
Twitter: @SqlQuantumLeap
https://ModuleSigning.info/

What Are Ya Gonna Do About It?

- Common Solutions

😈 Impersonation (EXECUTE AS)

😈 Cross-Database Ownership Chaining

😈 TRUSTWORTHY ON

Solomon Rutzky
Email: SRutzky@SqlQuantumLift.com
Company: https://SqlQuantumLift.com/
Blog: https://SqlQuantumLeap.com
Twitter: @SqlQuantumLeap
https://ModuleSigning.info/

Security Basics: Logins and Users

- Logins:
  - Server / Instance –level
  - sys.server_principals & sys.server_permissions
  - SUSER_NAME(), SUSER_ID()
  - "sa" always principal_id = 1 and sid = 0x01

- Users:
  - Database-level
  - sys.database_principals & sys.database_permissions
  - USER_NAME(), DATABASE_PRINCIPAL_ID()
  - SID matches Login's SID, but Name can be different
  - "guest" if no User entry (and enabled)
  - "dbo"
    - always principal_id = 1
    - SID changes to Login of owner

Solomon Rutzky
Email: SRutzky@SqlQuantumLift.com
Company: https://SqlQuantumLift.com/
Blog: https://SqlQuantumLeap.com
Twitter: @SqlQuantumLeap
https://ModuleSigning.info/

Security Basics: PRINCIPAL_IDs & SIDs

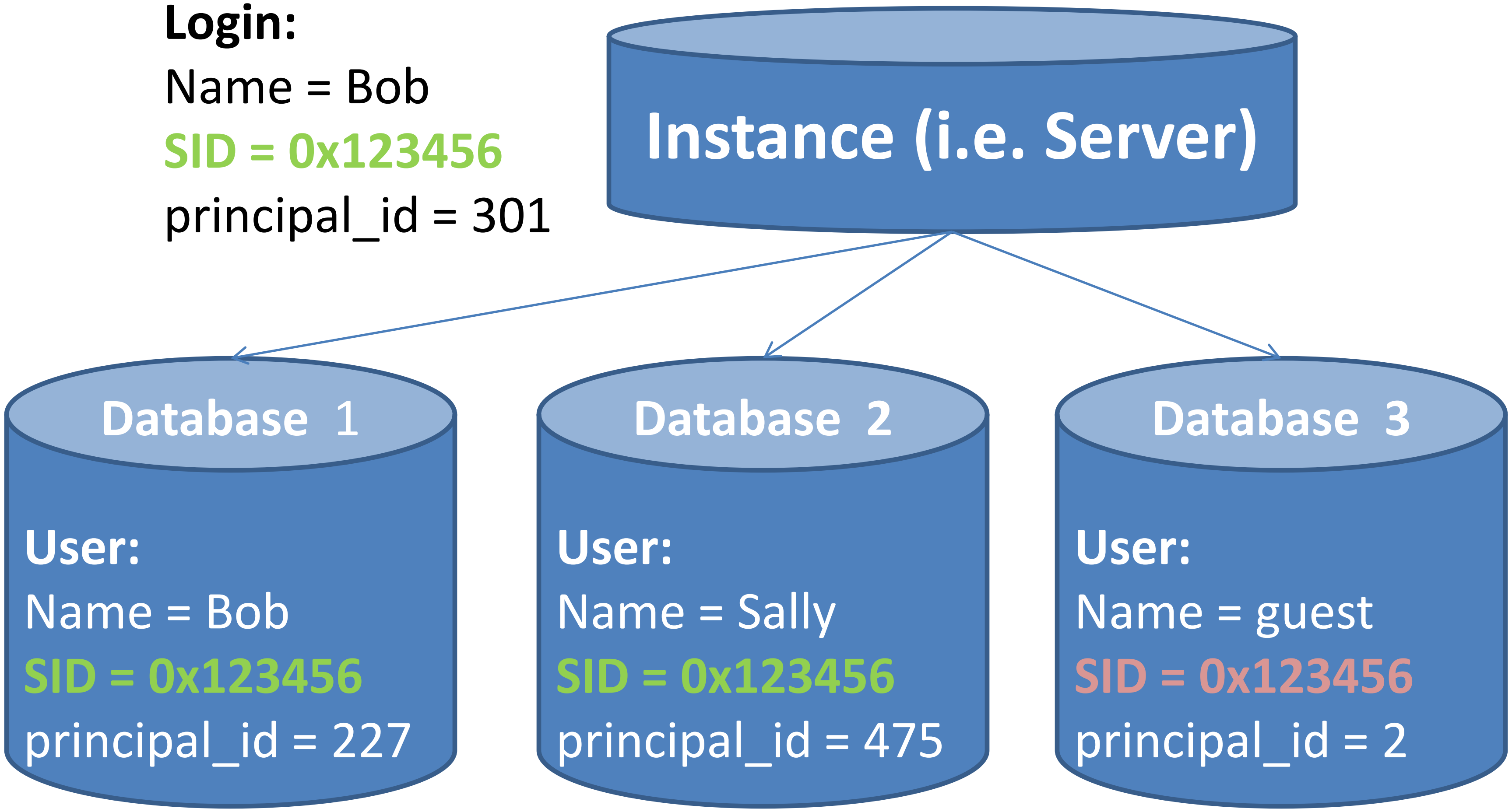
- principal_id
  - `INT`
  - Exists only in SQL Server
  - Used to FK to other system tables in same security context
  - No relationship between security contexts for same account
  - Always arbitrary

- Security Identifier (SID)
  - `VARBINARY(85)`
  - Might exist at the OS level (Windows Logins, Windows Groups)
  - Used to associate between DBs, and between DBs and Server
  - Sometimes meaningful (except for Server Roles and SQL Server Logins)
  - Binary: `0x01060000000000090100000006A91F17B3F6334F2C782536D9D66E88A07624983`
  - String: `S-1-9-1-2079428970-4063519551-1834189511-2330486429-2202624519`

Solomon Rutzky
Email: SRutzky@SqlQuantumLift.com
Company: https://SqlQuantumLift.com/
Blog: https://SqlQuantumLeap.com
Twitter: @SqlQuantumLeap
https://ModuleSigning.info/

Security Basics: Logins and Users

Login:
Name = Bob
SID = 0x123456
principal_id = 301



Solomon Rutzky
Email: SRutzky@SqlQuantumLift.com
Company: https://SqlQuantumLift.com/
Blog: https://SqlQuantumLeap.com
Twitter: @SqlQuantumLeap
https://ModuleSigning.info/

Security Basics: Ownership Chains

- Inherently how permissions work
- Permissions check skipped if sub-object is same owner
- DML, `SELECT`, and `EXEC` only
- Slight performance benefit (but can also skip a `DENY`)
- Within single DB by default
- Dynamic SQL breaks chain
- Can enable Cross-Database Ownership Chaining

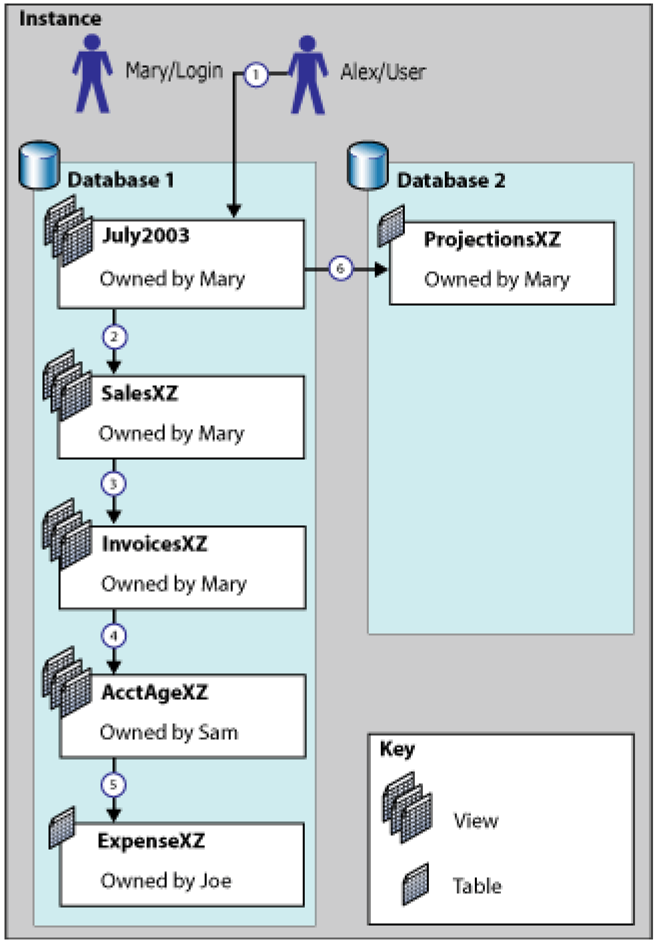
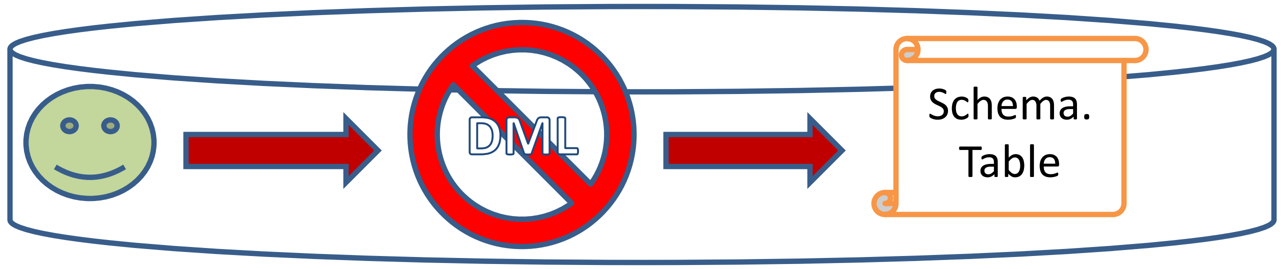


Image taken from:
https://technet.microsoft.com/en-us/library/ms188676.aspx

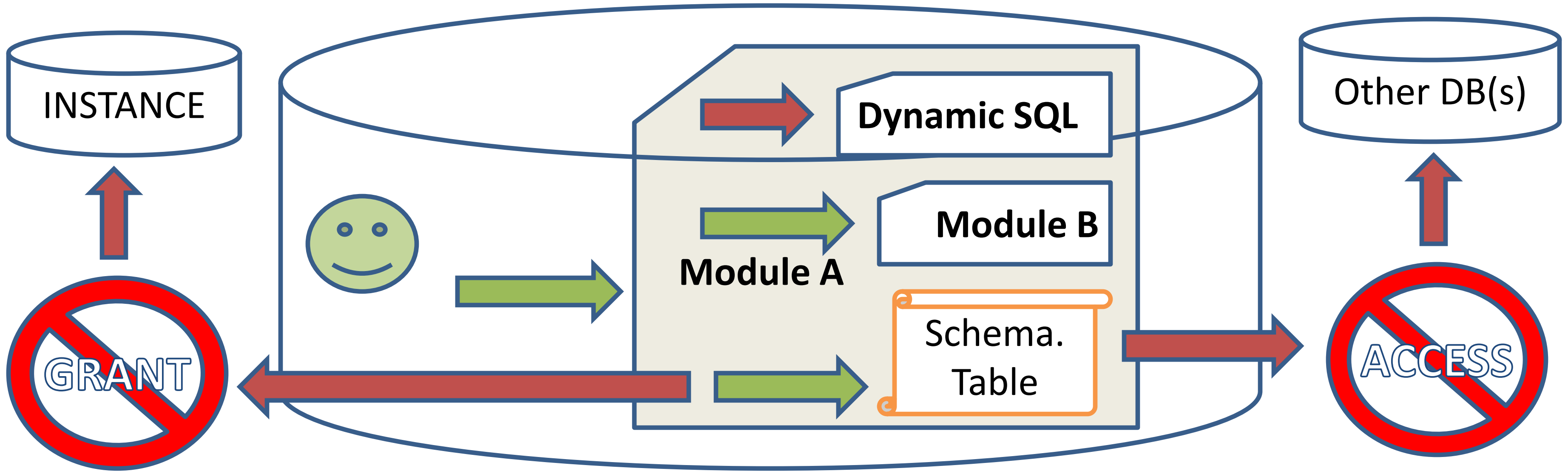
Solomon Rutzky
Email: SRutzky@SqlQuantumLift.com
Company: https://SqlQuantumLift.com/
Blog: https://SqlQuantumLeap.com
Twitter: @SqlQuantumLeap
https://ModuleSigning.info/

Graphimical Overmaview of Default Behavior and Benefit of Modules

DEFAULT:



OWNERSHIP CHAINING:



Solomon Rutzky
Email: SRutzky@SqlQuantumLift.com
Company: https://SqlQuantumLift.com/
Blog: https://SqlQuantumLeap.com
Twitter: @SqlQuantumLeap
https://ModuleSigning.info/

Impersonation

Solomon Rutzky
Email: SRutzky@SqlQuantumLift.com
Company: https://SqlQuantumLift.com/
Blog: https://SqlQuantumLeap.com
Twitter: @SqlQuantumLeap
https://ModuleSigning.info/

Sql Quantum Lift

Impersonation

- “Instead-of” Permissions
- Account-based security
- Requires a Login and/or User with elevated permissions
- Security Context (SYSTEM_USER and SESSION_USER) changes to this “impersonated” principal
- Accomplished via EXECUTE AS

Solomon Rutzky
Email: SRutzky@SqlQuantumLift.com
Company: https://SqlQuantumLift.com/
Blog: https://SqlQuantumLeap.com
Twitter: @SqlQuantumLeap
https://ModuleSigning.info/

EXECUTE AS

- Clause
  - Part of “CREATE OBJECT” statement
  - Impersonated Principals are always DB level (i.e. Users)
  - No IMPERSONATE permission needed
- Statement
  - Can do Server-level Logins and DB-level Users
  - Requires IMPERSONATE permission

Solomon Rutzky
Email: SRutzky@SqlQuantumLift.com
Company: https://SqlQuantumLift.com/
Blog: https://SqlQuantumLeap.com
Twitter: @SqlQuantumLeap
https://ModuleSigning.info/

Cross-Database Ownership Chaining

- Ownership chaining activation
  - Instance-level
    - “cross db ownership chaining”
    - When enabled, enables *all* Databases
  - Database-level
    - DB_CHAINING
    - Only used for enabling when server-level is disabled
- Extends ownership chain between DBs
  - Object Owner’s SID *and* Caller’s SID must exist in both DBs
  - Can’t elevate permissions
  - Dynamic SQL breaks!! Fix with either:
    - Impersonation *and* TRUSTWORTHY ON (bad ☹)
    - Module Signing (good ☺)

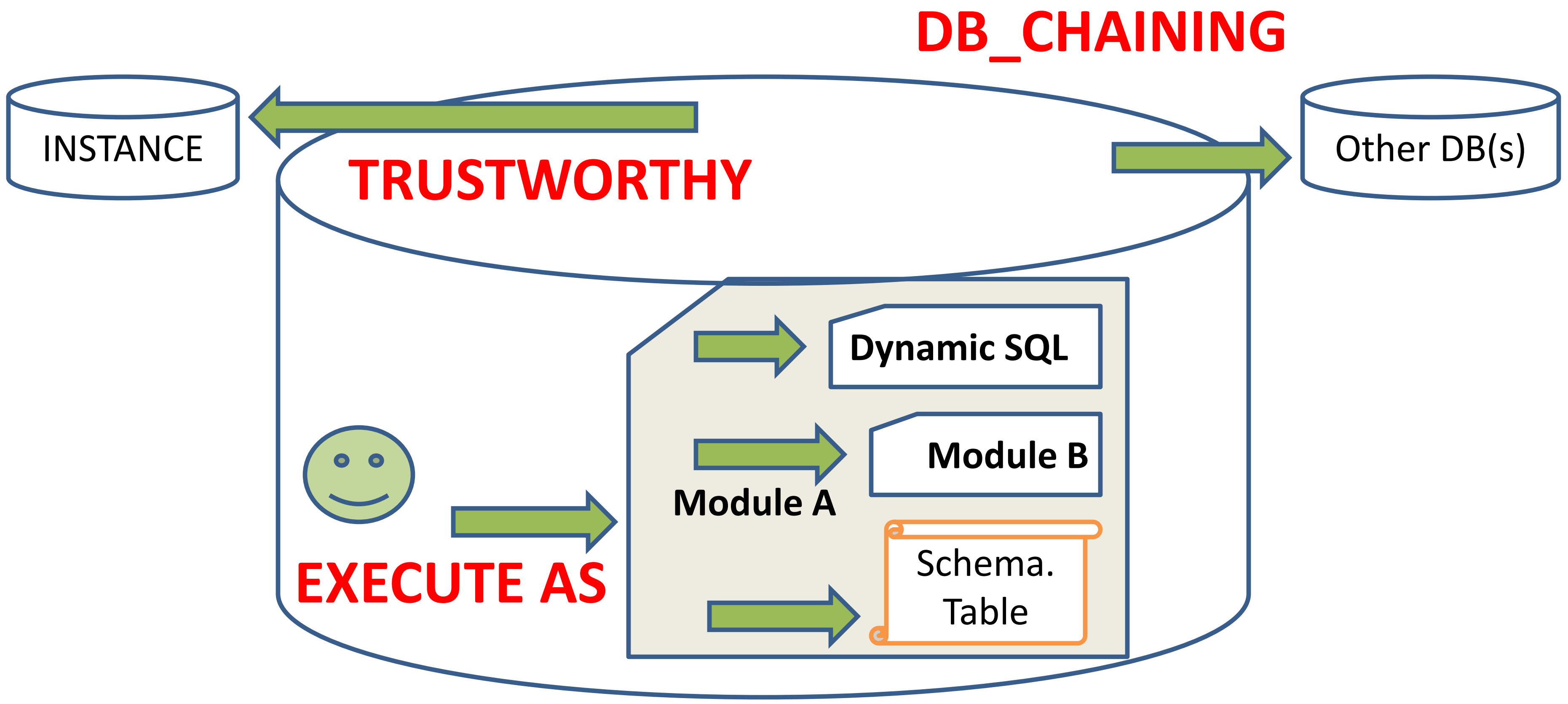
Solomon Rutzky
Email: SRutzky@SqlQuantumLift.com
Company: https://SqlQuantumLift.com/
Blog: https://SqlQuantumLeap.com
Twitter: @SqlQuantumLeap
https://ModuleSigning.info/

TRUSTWORTHY

- OFF by default
- Tells Instance to trust User SIDs from the DB:
  - Doesn’t quarantine process to “current” DB
  - Process can go up to instance-level or to another database (if same SID exists there)
- ALTER DATABASE { Name | CURRENT } SET TRUSTWORTHY { ON | OFF } ;
- Often used to:
  - Gain Instance-level permissions
  - Make loading SQLCLR Assemblies easier
- Might be *easier*, but never *necessary*

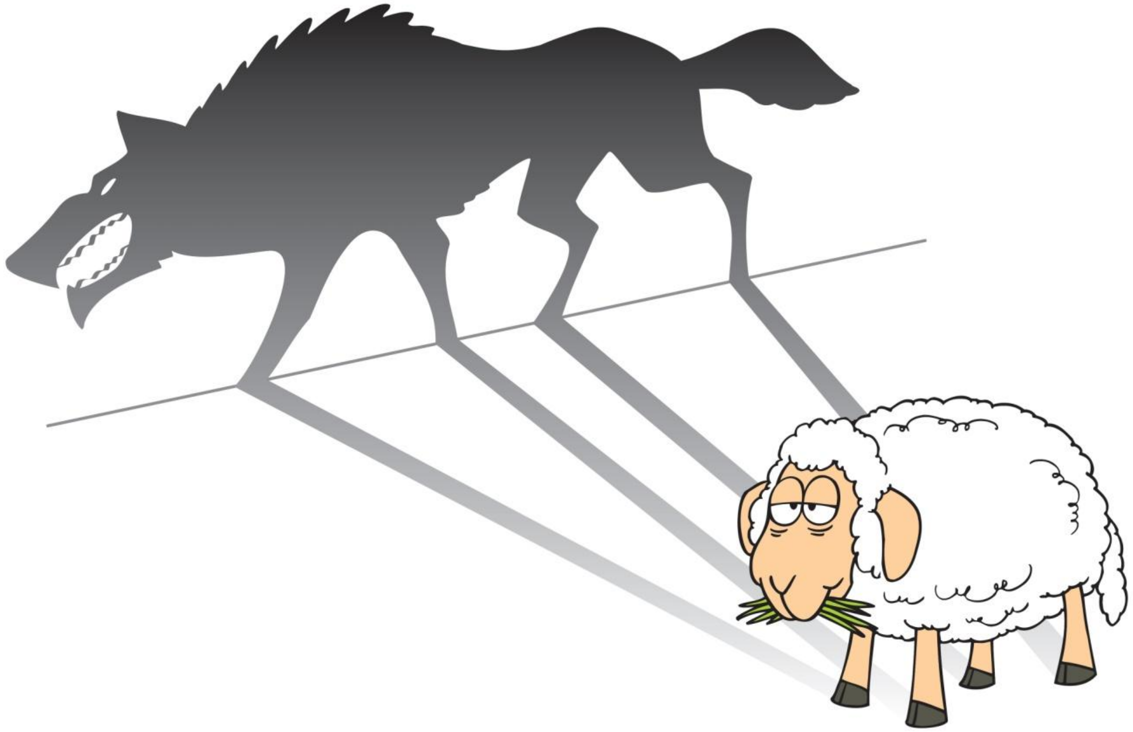
Solomon Rutzky
Email: SRutzky@SqlQuantumLift.com
Company: https://SqlQuantumLift.com/
Blog: https://SqlQuantumLeap.com
Twitter: @SqlQuantumLeap
https://ModuleSigning.info/

Graphimical Overmaview of Problems and Common Solutions



Solomon Rutzky
Email: SRutzky@SqlQuantumLift.com
Company: https://SqlQuantumLift.com/
Blog: https://SqlQuantumLeap.com
Twitter: [@SqlQuantumLeap](https://twitter.com/SqlQuantumLeap)
https://ModuleSigning.info/

Problems with Impersonation, TRUSTWORTHY, and Cross-Database Ownership Chaining



Solomon Rutzky
Email: SRutzky@SqlQuantumLift.com
Company: https://SqlQuantumLift.com/
Blog: https://SqlQuantumLeap.com
Twitter: @SqlQuantumLeap
https://ModuleSigning.info/

Problems with Impersonation, TRUSTWORTHY, and Cross-Database Ownership Chaining

- Cross-DB Ownership Chaining:
  - security risk (can spoof User / DB-level)
  - db_ddladmin & db_owner users can create objects for other owners
  - Users with CREATE DATABASE permission can create new databases and attach existing databases
- Impersonation:
  - If IMPERSONATE is required:
    - can be used any time
    - No granular control over permissions
  - Cross-DB operations need TRUSTWORTHY ON
  - Need to use ORIGINAL_LOGIN() for Auditing
  - Elevated permissions last until process / sub-process ends or REVERT
- TRUSTWORTHY:
  - Bigger security risk (can also spoof Logins, such as “sa” !)

Solomon Rutzky
Email: SRutzky@SqlQuantumLift.com
Company: https://SqlQuantumLift.com/
Blog: https://SqlQuantumLeap.com
Twitter: @SqlQuantumLeap
https://ModuleSigning.info/

And the Preferred Solution is...

(click picture to go to YouTube for video)

Solomon Rutzky
Email: SRutzky@SqlQuantumLift.com
Company: https://SqlQuantumLift.com/
Blog: https://SqlQuantumLeap.com
Twitter: [@SqlQuantumLeap](https://twitter.com/SqlQuantumLeap)
https://ModuleSigning.info/

Module Signing

- "In Addition To" Permissions
- Code-based security
- Signatures = authenticity *and* change detection
  - Hash only provides change detection
- Security Context (SYSTEM_USER and SESSION_USER) does NOT change to this "privileged" principal

Solomon Rutzky
Email: SRutzky@SqlQuantumLift.com
Company: https://SqlQuantumLift.com/
Blog: https://SqlQuantumLeap.com
Twitter: @SqlQuantumLeap
https://ModuleSigning.info/

Module Signing (cont.)

- Also requires a Login and/or User with elevated permissions
- Accomplished using ADD SIGNATURE
  - Regular vs. COUNTER SIGNATURE
- Can sign modules:
  - Multi-statement Table-Valued Functions
  - Stored Procedures
  - Scalar Functions
  - Triggers
  - [SQLCLR Assemblies](#)

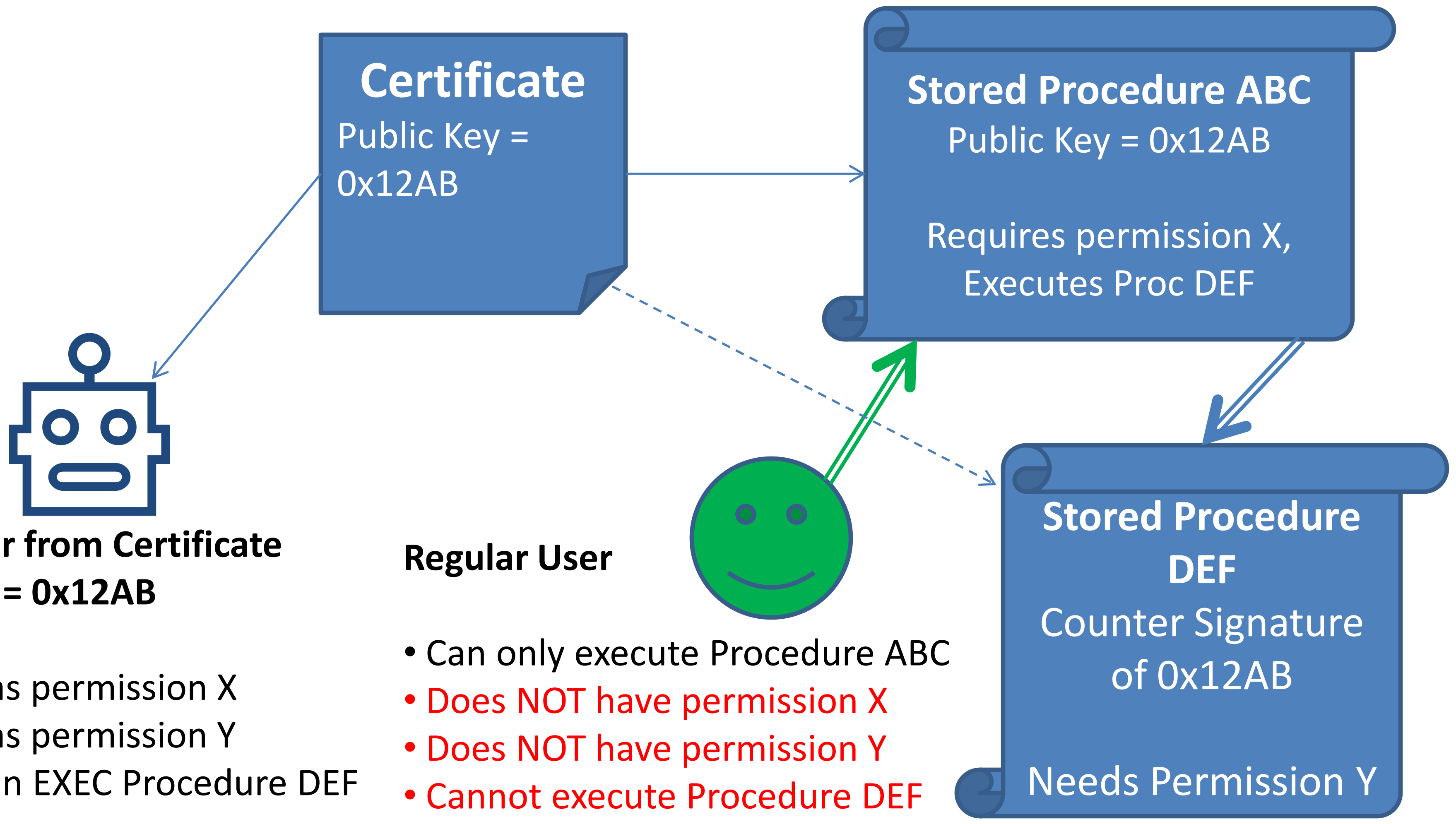
Solomon Rutzky
Email: SRutzky@SqlQuantumLift.com
Company: https://SqlQuantumLift.com/
Blog: https://SqlQuantumLeap.com
Twitter: @SqlQuantumLeap
https://ModuleSigning.info/

Benefits

- Privileged principal *cannot* be impersonated
- *Very* Granular permissions
- No security holes (e.g. TRUSTWORTHY, etc.)
- Signature is dropped if code is changed !!
- Elevated permissions confined to signed code
- Multiple Signatures can be used to combine permission "sets"

Solomon Rutzky
Email: SRutzky@SqlQuantumLift.com
Company: https://SqlQuantumLift.com/
Blog: https://SqlQuantumLeap.com
Twitter: @SqlQuantumLeap
https://ModuleSigning.info/

Signatures and Counter Signatures



User from Certificate SID = 0x12AB

- Has permission X
- Has permission Y
- Can EXEC Procedure DEF

Regular User

- Can only execute Procedure ABC
- Does NOT have permission X
- Does NOT have permission Y
- Cannot execute Procedure DEF

Solomon Rutzky
Email: SRutzky@SqlQuantumLift.com
Company: https://SqlQuantumLift.com/
Blog: https://SqlQuantumLeap.com
Twitter: @SqlQuantumLeap
https://ModuleSigning.info/

Asymmetric Keys & Certificates

Common Aspects

- Consist of a Private Key and Public Key pair
- Can have the Private Key removed
- Common Properties:
  - Thumbprint (hash of Public Key, sys.crypt_properties)
  - SID
  - principal_id
  - name
- Create from File (.snk / .cer, or .dll) or Assembly
- Provide password or use Database Master Key (DMK)

Solomon Rutzky
Email: SRutzky@SqlQuantumLift.com
Company: https://SqlQuantumLift.com/
Blog: https://SqlQuantumLeap.com
Twitter: @SqlQuantumLeap
https://ModuleSigning.info/

Asymmetric Keys

- Where: `SELECT * FROM [sys].[asymmetric_keys];`
- Properties:
  - public_key
- Can create from Key Store / EKM
  - BUT, EKM created Keys not supported for Module Signing
- Can specify Algorithm:
  - RSA_512, RSA_1024, RSA_2048, RSA_3072, or RSA_4096
- Cannot backup 😾
  Add function to extract Asymmetric Key similar to CERTENCODED for Certificates
- Cannot restore Private Key 😾

Solomon Rutzky
Email: SRutzky@SqlQuantumLift.com
Company: https://SqlQuantumLift.com/
Blog: https://SqlQuantumLeap.com
Twitter: @SqlQuantumLeap
https://ModuleSigning.info/

Certificates

- Where: `SELECT * FROM [sys].[certificates];`
- Asymmetric Key + extra properties
- Properties:
  - Serial Number: unique ID of the Certificate
  - Subject: essentially a description
  - Start Date: UTC; default = GETUTCDATE();
  - Expiration Date: UTC; default = 1 year from Start
- Module Signing ignores Expiration Date
- Can backup !!
- Can restore Private Key !!

Solomon Rutzky
Email: SRutzky@SqlQuantumLift.com
Company: https://SqlQuantumLift.com/
Blog: https://SqlQuantumLeap.com
Twitter: @SqlQuantumLeap
https://ModuleSigning.info/

Certificates & Asymmetric Keys: Basic Usage

- Encryption
  - Message + Public Key → 0x... (encrypted binary)
  - 0x... (encrypted binary) + Private Key → Message
- Signing
  - Message / Code + Private Key → Signature
  - Message / Code + Signature + Public Key → SAME vs NOT Same

Solomon Rutzky
Email: SRutzky@SqlQuantumLift.com
Company: https://SqlQuantumLift.com/
Blog: https://SqlQuantumLeap.com
Twitter: @SqlQuantumLeap
https://ModuleSigning.info/

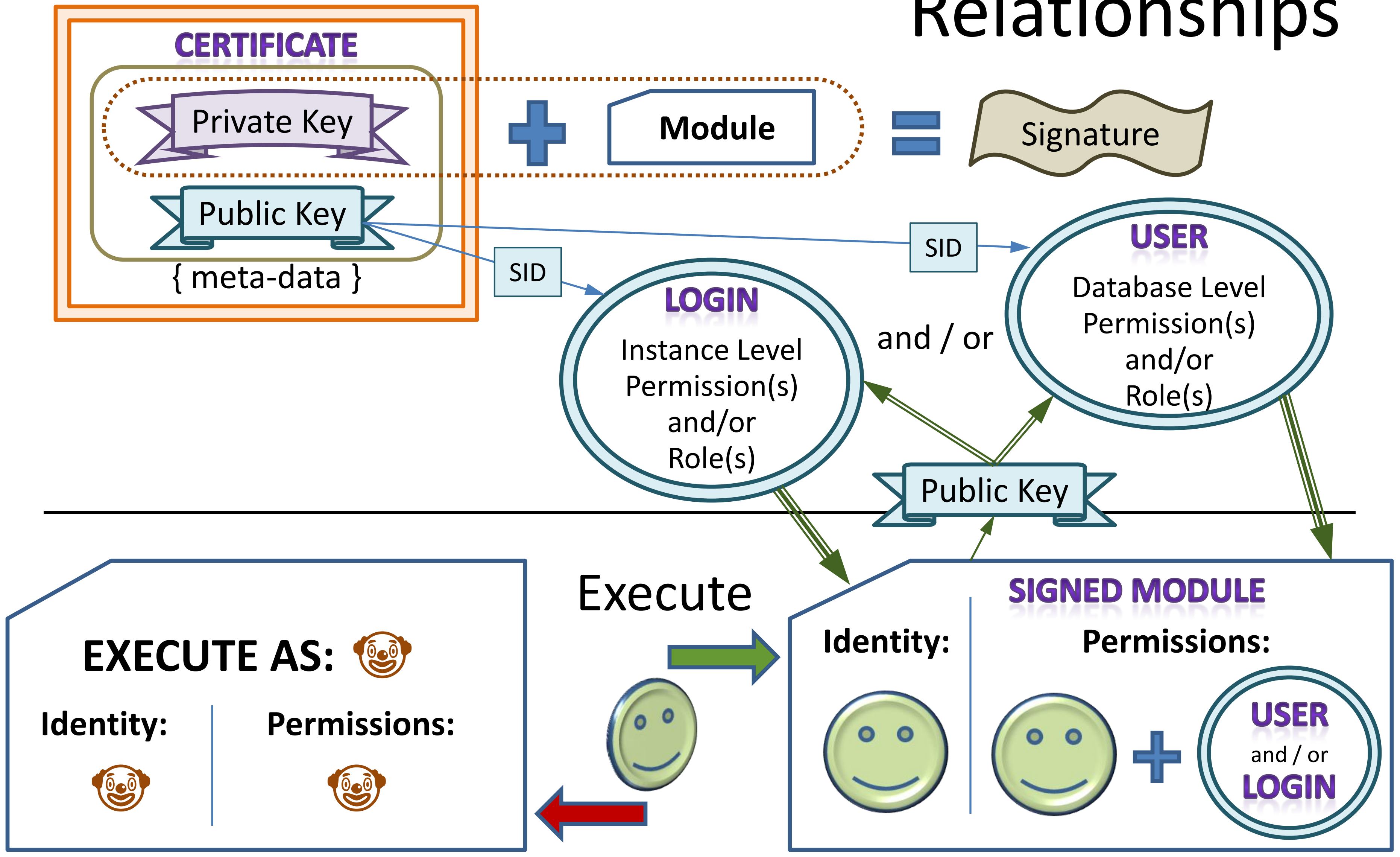
Certificates & Asymmetric Keys: Use in Module Signing

Execute signed module:

1. Get [thumbprint] and [crypt_property] (signature) of signed module from `sys.crypt_properties`
2. Get public key and [sid] from `sys.certificates` based on [thumbprint] (from step 1)
3. Use [crypt_property] (from step 1), public key (from step 2), and source code of current module to verify that source code has not changed:
   a. If source code has changed, do not apply any additional permissions.
   b. Else, add instance/database -level permissions, if any, of associated Login and/or User, based on [sid] (from step 2)

Solomon Rutzky
Email: SRutzky@SqlQuantumLift.com
Company: https://SqlQuantumLift.com/
Blog: https://SqlQuantumLeap.com
Twitter: @SqlQuantumLeap
https://ModuleSigning.info/

Relationships



Solomon Rutzky
Email: SRutzky@SqlQuantumLift.com
Company: https://SqlQuantumLift.com/
Blog: https://SqlQuantumLeap.com
Twitter: @SqlQuantumLeap
https://ModuleSigning.info/

Examples

- Safely and Easily Use High-Level Permissions Without Granting Them to Anyone: Server-level
  https://sqlquantumleap.com/2018/02/15/safely-and-easily-use-high-level-permissions-without-granting-them-to-anyone-server-level/

- Safely and Easily Use High-Level Permissions Without Granting Them to Anyone: Database-level
  https://sqlquantumleap.com/2018/03/05/safely-and-easily-use-high-level-permissions-without-granting-them-to-anyone-database-level/

- Proc Inserts via Dynamic SQL into Table with Trigger that Inserts into Other Table
  https://pastebin.com/ALgLuZAP

- Can't use msdb.dbo.sp_send_dbmail when in service broker - executes as guest?
  https://dba.stackexchange.com/a/166280/30859

Solomon Rutzky
Email: SRutzky@SqlQuantumLift.com
Company: https://SqlQuantumLift.com/
Blog: https://SqlQuantumLeap.com
Twitter: @SqlQuantumLeap
https://ModuleSigning.info/

Conclusionarium

- Cross-Database Ownership Chaining
- Impersonation / EXECUTE AS
- TRUSTWORTHY ON

S.U.C.K.S. 💩 😠 😦 😭 🚽

- Certificates and Module Signing

AWESOME !!! 😺 😹 😇 🙌

- Details of this presentation: **PLEASE, Please, please Stop Using Impersonation, TRUSTWORTHY, and Cross-DB Ownership Chaining** (https://SqlQuantumLeap.com/2017/12/30/please-please-please-stop-using-impersonation-execute-as/)
- Module Signing Info (https://ModuleSigning.Info/)

Solomon Rutzky
Email: SRutzky@SqlQuantumLift.com
Company: https://SqlQuantumLift.com/
Blog: https://SqlQuantumLeap.com
Twitter: @SqlQuantumLeap
https://ModuleSigning.info/

Hiding in Plain Sight

- Module Signing Resources:
  - https://ModuleSigning.Info/
- Blog:
  - https://SqlQuantumLeap.com/
- Articles:
  - https://www.SqlServerCentral.com/author/solomon-rutzky
  - https://www.SqlServerCentral.com/stairways/stairway-to-sqlclr (Stairway to SQLCLR)
  - https://www.simple-talk.com/author/solomon-rutzky/
- SQLsharp.com
  - https://SQLsharp.com/
- StackOverflow.com & DBA.StackExchange.com
  - https://StackExchange.com/users/281451/solomon-rutzky
- LinkedIn
  - http://www.LinkedIn.com/in/srutzky/
- Email:
  - SRutzky@SqlQuantumLift.com

Solomon Rutzky
Email: SRutzky@SqlQuantumLift.com
Company: https://SqlQuantumLift.com/
Blog: https://SqlQuantumLeap.com
Twitter: @SqlQuantumLeap
https://ModuleSigning.info/