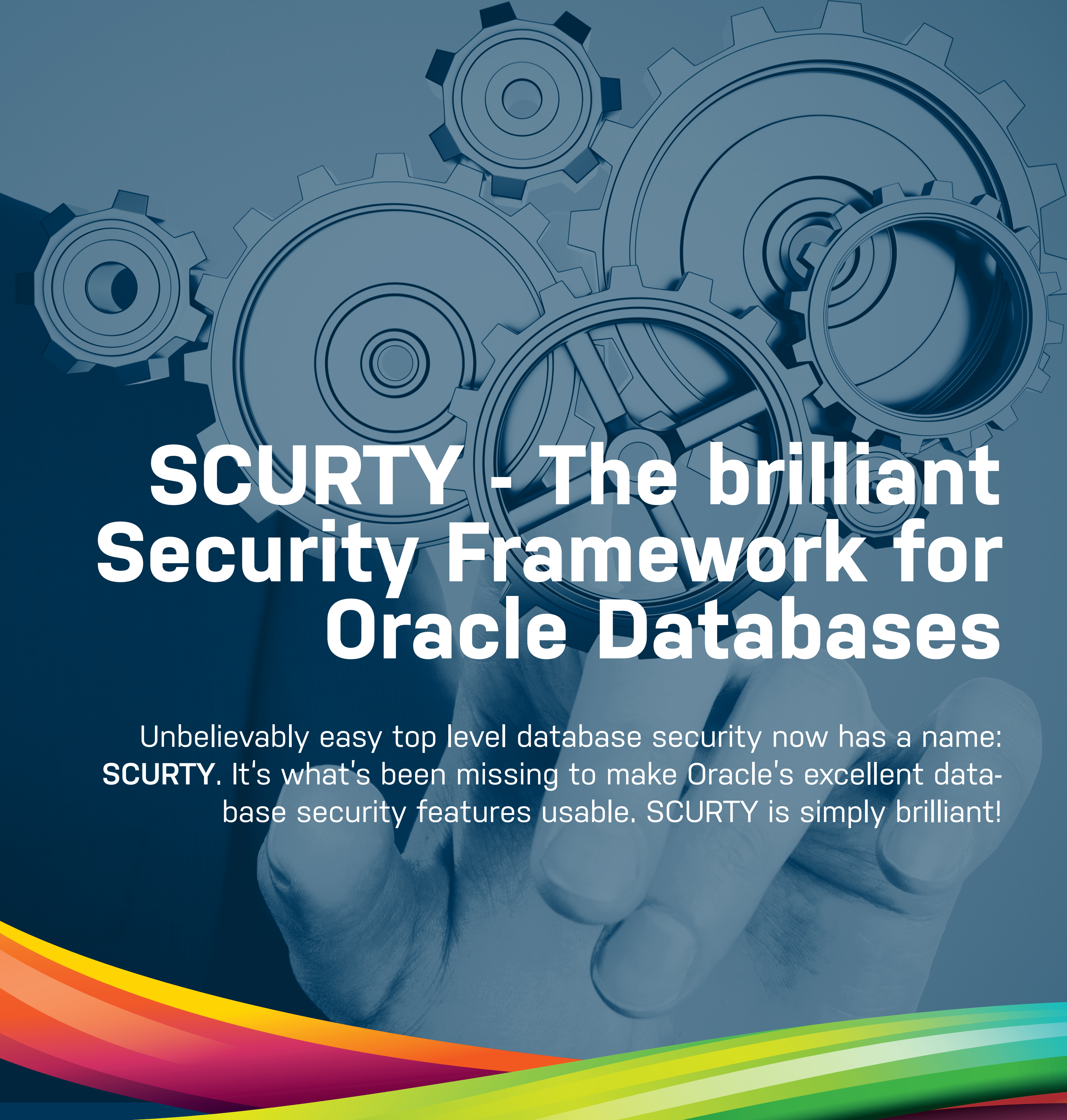


SCURTY - The brilliant Security Framework for Oracle Databases

Unbelievably easy top level database security now has a name: **SCURTY**. It's what's been missing to make Oracle's excellent database security features usable. SCURTY is simply brilliant!

sphinx

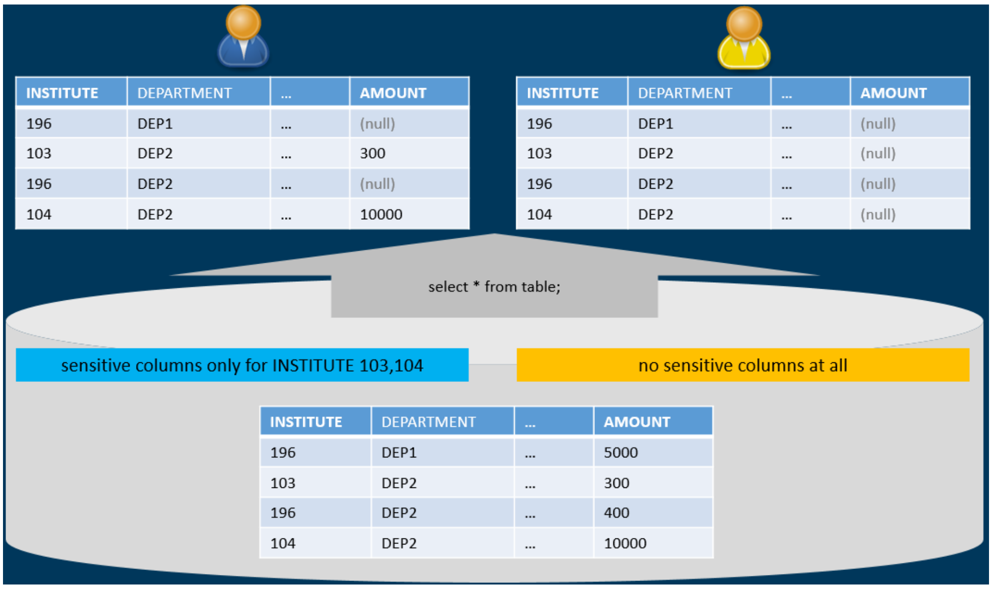
MASTERING THE GAP

SECURITY MADE USABLE

Administering database security is usually either time-consuming or incomplete. This results in excellent database security features that are rarely used due to their complex administration. SCURTY overcomes this by drastically lowering the effort needed for administration and therefore substantially raising the level of security.
SCURTY hides complexity from the administrator and delivers an easy way to provide highly sophisticated security. Behind the scenes convenient functions for administering database security act as wrappers hiding the various underlying database security features.

FINE GRAINED ACCESS RIGHTS WITHOUT EFFORT

SCURTY provides a view on Oracle database security that previously could only be dreamed about. You just have to think about what you would like to achieve, not how you would go about it and which features you would need to use. All you have to do is to tell SCURTY your needs via the convenient PL/SQL-API. SCURTY uses practically all security-relevant features of the Oracle Database Enterprise Edition, such as Logon-Triggers, Proxy-User authentication, VPD (Virtual Private Database), Secure Application roles and much more, but you don't need to know about them in detail or build anything manually. Everything is implemented automatically in the background. Usable and maintainable comprehensive database security is therefore available for the first time.



INSTITUTE	DEPARTMENT	...	AMOUNT
196	DEP1	...	(null)
103	DEP2	...	300
196	DEP2	...	(null)
104	DEP2	...	10000

INSTITUTE	DEPARTMENT	...	AMOUNT
196	DEP1	...	(null)
103	DEP2	...	(null)
196	DEP2	...	(null)
104	DEP2	...	(null)

select * from table;
sensitive columns only for INSTITUTE 103,104
no sensitive columns at all

INSTITUTE	DEPARTMENT	...	AMOUNT
196	DEP1	...	5000
103	DEP2	...	300
196	DEP2	...	400
104	DEP2	...	10000

Fig. 1: Example of column level security

DECENTRALIZED PROVISIONING

A department within an organisation knows its requirements. By hiding complexity and providing decentralized provisioning the task of provisioning access rights can be moved from the database administrator (DBA) to the departmental level. This results in faster and more appropriate provisioning and frees the DBA for more important tasks. There is no need to be a DBA or have DBA privileges to provision database access rights with SCURTY.

PERSONALIZED ACCESS

Three tier architectures usually lead to the creation of generic users, resulting in imprecise database rights and meaningless audit data. Thanks to SCURTY personalized access is easily achieved. A well thought-through approach makes it possible to provide generic users (for applications that require them) but restricting them to only work in conjunction with personalized users. A user's access rights can therefore be defined at a very fine-grained level, with audit data therefore containing valuable information. A user's access rights will always be the same, independent of the tool or the architecture, with both two- and three-tier architectures being fully supported.

TOOL-AGNOSTIC SECURITY

Security provided by SCURTY resides within the database and is therefore in place for every client, regardless of the tool or architecture used. Bypassing SCURTY is not possible. Users that are not provisioned simply can't log on or don't see any data.

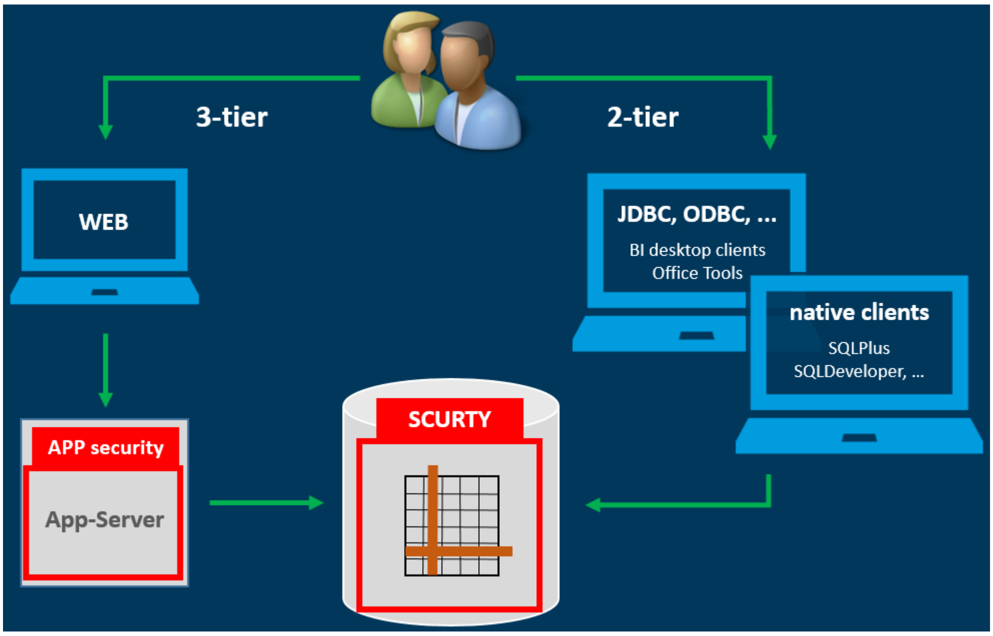


Fig.2: Access paths in 2- and 3-tier architectures

ROW- AND COLUMN-LEVEL SECURITY

Database object level grants are standard, but SCURTY goes way beyond that. Row- and column- level grants can be provided without having to know anything about the VPD being used in the background. No VPD policy has to be written and more importantly, no VPD policy has to be maintained! SCURTY administers all of that, completely hidden behind the PL/SQL API. This is the first time VPD is available without an administrative nightmare.

SANDBOXING FOR THE DEPARTMENTAL LEVEL

You wish to provide an environment for testing, developing and analytics on separate instances of data? No problem, it's easy to do! Sandboxing provides temporary schemas inside an existing database and, of course, fine grained access can be granted. Objects created in a sandbox might be available to others or marked as protected – which means that not even the sandbox master or owner is allowed to change them. Years of experience has shown that every sandbox has a lifetime, no sandbox is there forever. A sandbox schema is automatically locked at the end of the specified lifetime and dropped after an additional grace period.

DYNAMIC DATA MASKING

Column level masking of data provides security to curb the curiosity of the user. There is no way to guess a hidden value by repeatedly executing similar statements.

PERSONALIZED AUDIT

Restricting database access to personalized accounts provides meaningful audit logs. There is no need to search for the real user behind a given data access as audit shows the human user on each and every access path. A meaningful audit trail is the basis of real security.

TRANSPARENT ARCHITECTURE

All metadata definitions are well documented and available for reporting. Detailed logs of any changes made to the system provide an excellent view of what happened in the past. And, of course, the whole system is hardened against manipulation from users.

COMPONENTS OF THE DATABASE USED

The DB-Security Framework uses Oracle Database Enterprise Edition features only. No additional options are required.

HOW TO IMPLEMENT

A DBA has to run the SCURTY PL/SQL install scripts and create the first admin user. The whole procedure takes only a few minutes. After that everything can be performed by appropriately authorized users inside SCURTY. It's surprising how little work there is left to do.

SUMMARY

Security features offered by the Oracle Database Enterprise Edition are for the first time available for easy use thanks to SCURTY. Fine grained access rights, sandboxing and personalized accounts including meaningful audit can be utilized in 2- and 3-tier architectures without the usual overhead and without excessive administration. The PL/SQL API hides complexity from the admi-nistrator in an unbelievably clever way. There is no need to understand how Secure Application Roles, Virtual Private Database or Logon-Triggers work or to write code. The comprehensive SCURTY security framework uses all this and much more to raise security to an unprecedented level.

CONTACT

SPHINX IT CONSULTING GMBH | ASPERNBRÜCKENGASSE 2 | 1020 WIEN
TEL: +43 1 599 31 - 0 | office@sphinx.at

sphinx
MASTERING THE GAP

Visit us at
www.sphinx.at