1st BPM Workshop on Workflow Security Audit and Certification (WfSAC)
at the 9th Int'l Conf. on Business Process Management (BPM 11)
Clermont-Ferrand, France, August 28th to September 2nd, 2011

Data-Privacy Assessments for Application Landscapes: A Methodology

Klaus Haller

Swisscom IT Services Finance
Testing & QA
Pflanzschulstr. 7, CH-8004 Zürich, Switzerland
klaus.haller@swisscom.com

Abstract. Data privacy is a major issue for companies today. Risks can come from external attacks or from internal users disclosing sensitive data to the public. In the latter case, restricting user access to data mitigates the risk. Thanks to role-based access models, users see only the data that they need for their work. This paper presents a methodology for assessing how effective such restrictions are. It is based on classifying data, analyzing access paths, and understanding the impact of design principles. Its special contribution is its end-to-end view. It is applicable directly to complex IT landscapes being the norm today.

Keywords: Information Systems, Privacy, Testing

1 Introduction

Data leaks are a major threat for companies in all business sectors. They can ruin the reputation and cause high costs [1]. Data leaks emerge as military spy plots, such as the Los Alamos National Lab case [2]. They can be low-key, such as non-medical person having access to patients' records [3]. Then, there are stories about bank data thefts [4,5]. But even "non-sensitive" sectors are at risk when handling customer data. Involuntary examples exist in various sectors, e.g. the airline sector (Lufthansa's leak regarding the use of frequent flyer data of German politicians [6]) or online gaming (Sony's PlayStation Network case [7]).

Outsiders can break into IT systems (Sony case). Also, internal users might disclose data (banking examples). To address the last threat, companies restrict the data access for users. They can see only the data needed for their work. The technical bases are role-based access models [8]. But they work only if set up correctly. This paper presents a methodology to assess this. It is part of a broader initiative on testing and quality assurance for database applications and information system (IS) landscapes [9,10,11]. The focus of this paper is to systemize the data privacy aspect. Various consulting projects have proved its importance. Thus, the aim of the paper is to foster discussions between consulting and academia about this topic.

The paper illustrates the methodology, using a fictive credit-rating application *CreditPlus*. It calculates how likely small and medium-sized enterprises (SME) do not pay back loans. Input data is balance sheets. CreditPlus stores them and calculates the

credit rating. A bank with branches in the US, the UK, and Switzerland (CH) uses the software. CreditPlus is adapted for each country. This reflects varying accounting standards. Besides the US, UK, and CH users, there is an auditing and risk team in Germany. It enforces the bank's risk policy and processes in all countries. Also, it calculates the distribution of the risk exposure (e.g. 5% of the loans in the retail sector, 11% in hotels etc.). The software developers work in Romania, and the testers in Singapore. This scenario covers two practical sourcing scenarios:

- Global software development and testing
- Global sourcing of business activities

W.l.o.g., the paper narrows down the data privacy aspect on whether customer-identifying data crosses borders. The set-up involves just three branches, one auditing and risk team, a software development team, and a testing team. Still, six countries are involved. This has a severe impact on the data flow (Figure 1):

- Developers in Romania and testers in Singapore need test data. The closer the data to the "real world", the more efficient the software development, the testing, and the bug fixing. This implies copying (some kind of) US, UK, and CH customer data to Romania and Singapore from time to time.[1]
- The auditing and risk team in Germany calculates the risk exposure. It checks whether the branches stick to the bank's policies. The team needs continuous online access to the US, UK, and CH data.

Many companies have faced such challenges for many years. They have solutions put in place, either in ad-hoc style or based on a comprehensive approach. But one gap remains: assessing whether a concrete solution really works. This is the focus of the paper. It starts first with a formal data privacy model (Section 2). Section 3 introduces the methodology for assessing data privacy compliance. The two following sections enhance this core finding. Section 4 looks at the challenge of software for which only limited know-how exists. Section 5 looks at how data sanitization impacts assessments. The paper concludes with a discussion of related work (Section 6) and a short summary (Section 7).

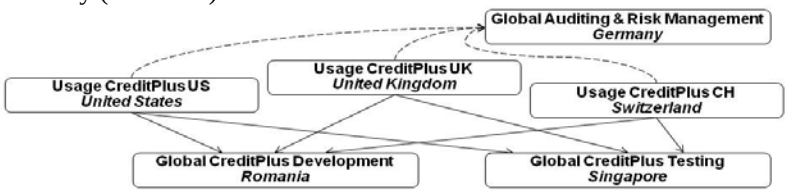

Figure 1: Sample set up for global sourcing and cross-border data flows
(continues line: on-request copy, dashed line: online access)

2 Data Privacy Model (DPM)

This section presents the data privacy model (DPM). It is a formal model for reasoning about the data privacy of IS landscapes. It has four key concepts: the usage vector, the data criticalness function, the data access diagram, and the privacy

[1] There are other options for test data, too. However, they are often less helpful for complex environments. See [10] for a detailed discussion.

compliance correctness criterion (Figure 2 compiles them and acts as a guides through this paper). The usage vector comprises (some of) the factors influencing whether a user can see data items:

- The *roles* a concrete user has: A credit officer has to know the name of his customer, not necessarily a compliance officer. Also, software developers working on future releases do not need access to customer data.
- The *responsibility segment* - UK credit officers are not allowed to see all customer data. They focus on a segment, e.g. UK corporations with starting letters A-L.
- The *country* the user *works*: UK credit officers are only allowed to see UK customers.
- The user's *current country*. A UK credit officer is allowed to see UK customers if he is accessing the system from the UK. He might not be allowed to access the data when abroad.

Certainly, one can model more components depending on the circumstances.

DEFINITION 1 (USAGE VECTOR): *A usage vector û is a 4-tuple <R, c_w, c_c, S> with R being the roles of the user, c_w the country he usually works in, c_c the country where he currently is, and S the segment he is responsible for. U denotes all possible usage vectors.*

W.l.o.g., this paper focuses on the country of work as the only component of the usage vector. This allows for a more focused discussion.

The **data criticalness function** models how sensitive data items are. It is obvious for many that they are sensitive - for example, customer names or IBAN account numbers. A data item might also be known to be non-sensitive. Account balances (without link to any customer) are an example. Then, there are data items for which it is not known (potentially sensitive data items). The DPM sums up the sensitive and potentially sensitive data items to *red* data items. The non-sensitive data items are *green* data items. Whether a data item is red or green is not a global property. It depends on the context, i.e., the usage vector (Figure 2, left).

Definition 2 (DATA CRITICALNESS FUNCTION). *Let D_{IS} be the set of data items in the IS landscape, U_{IS} the set of all usage vectors. Then, the data criticalness function*

$$C: D_{IS} \times U_{IS} \rightarrow \{red, green\}$$

states whether the data items are allowed to be seen for this usage context.

The second concept is the **data access diagram** (Figure 2, middle). A data access diagram has three layers for describing *who* can access *which data items* using which *application features*. The usage vectors U_{IS} form the top layer, and the data items D_{IS} the lower layer. Features form the middle layer. They represent the application logic with a focus on data privacy. A feature links a usage vector (e.g. London-based UK credit officers for A-L customers) with data items (e.g. all corporations in London with starting letters A-L). Arrows in Figure 2 illustrate such links.

Definition 3 (FEATURE). *A feature f_i is a pair <U_i, D_i> with U_i being a set of usage vectors getting access to a set of data items D_i by using feature f_i. F_{IS} is the set of all features in the IS landscape.*

This allows formalizing the concept of data access diagrams:

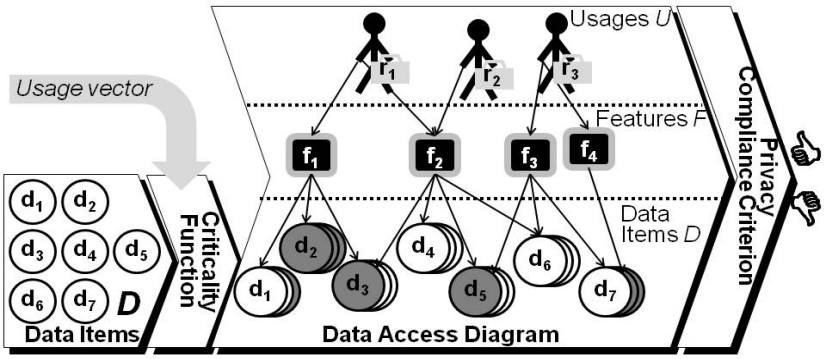

Figure 2: The Data Privacy Model (DPM)

Definition 4 (DATA ACCESS DIAGRAM). *A data access diagram A_{IS} is a triple $< U_{IS}, D_{IS}, F_{IS} >$ with U_{IS} being the set of all usage vectors, D_{IS} the set of all data items, and F_{IS} the set of all features f_i of the IS landscape. Obviously, $U_{IS} \supseteq \{U_i | \exists \langle U', D' \rangle \in F_{IS} : U' = U_i\}$ and $U_{IS} \supseteq \{D_i | \exists \langle U', D' \rangle \in D_{IS} : D' = D_i\}$ holds.*

The fourth DPM concept is the correctness criterion **data privacy compliance**. It is a formal way to say that the IS landscape respects all data privacy demands. It is based on the data access diagram. The data access diagram is data privacy compliant, if *all* usage vectors are only linked to green data items. There must be no single link to red data items.

Definition 5 (DATA PRIVACY COMPLIANCE). *Let $A_{IS} =< U_{IS}, D_{IS}, F_{IS} >$ be the data access diagram. C is the data criticalness function. The IS landscape is data privacy compliant, iff* $\forall$<U', D'>$\in F_{IS}$, u$\in$ U', d$\in$ D': C(d,u) = green.

3 Data Privacy Assessment

A data privacy assessment must state whether the IS landscape is data privacy compliant (Definition 5). A "non-compliant" alone does not help. It must come with a list of identified leaks and risks. Then, managers can decide which risks should be addressed and how. A data privacy assessment has three areas (Figure 3): classifying the data (with preparatory steps for identifying privacy rules), understanding the line of separation, and analyzing the data access paths.

Figure 3: Data Privacy Assessment for IS Landscapes

3.1 Line of Separation

Four layers of an IS landscape can separate users from the data items they must not see ("line of separation"): the zone, the application, the tenant, and the (application) feature.

- *Zone Separation.* Small companies often have a flat network model. All clients and servers form one subnet. Larger international companies structure their network into zones. A zone is a subnet with dedicated security policies. It is shielded from other zones and the internet. Only defined interactions are possible. In the CreditPlus example, there could be one zone for each country: one for the US, one for the UK, etc. Then, CreditPlus US users, for example, cannot access the UK zone. Thus, they cannot access UK applications or databases, and, therefore, no UK data items.
- *Application Separation.* Users of various countries work in the same subnet or zone. The separation takes place on the application level. Each application "belongs" to a country. CreditPlus UK, CreditPlus CH, and CreditPlus US are separate applications. Users can only log into the CreditPlus application of their country.
- *Tenant Separation.* Multi-tenant applications [12] have been on the rise for some years. They enable separating users of various countries by means of tenants: one tenant per country. Users see only the data of the tenant they belong to. So there is one CreditPlus application with three tenants (UK, US, and CH). Each user is tagged with her country of residence, e.g. CH. If she uses CreditPlus, she sees only CH data items, but no UK or US ones.
  One remark regarding local authorities: Normally, one database in one country stores the data for *all* tenants. The application blocks users (e.g. UK users) from data of other tenants (e.g. the US tenant). The application cannot prevent local authorities from enforcing access on the database level. Then, the local authorities see the data items of all tenants.
- *(Application) Feature Separation.* Here, all users of all countries work with one application. They have different access rights, e.g. depending on their country. This ensures that they see only customers of their country. So, there is one CreditPlus application. It has three wizards for finding customers: "Find customer (UK)", "Find customer (US)", and "Find customer (CH)". The UK users, for example, could access only the "Find customers (UK)" wizard. They could not access the two others. Again, the aspect of local authorities enforcing data access on the database level must be considered.

3.2 Classifying Data

In the world of theory, classifying data means applying the data criticalness function C to all data items for all usage vectors. In practice, this is not possible. First, the number of usage vectors is too high. One must choose a subset of the most relevant ones. One could look only at the country aspect as this paper does in the running example. The second challenge is that the data criticalness function *C* is normally not known. Moreover, it is (nearly) impossible to formalize the system in such a way.

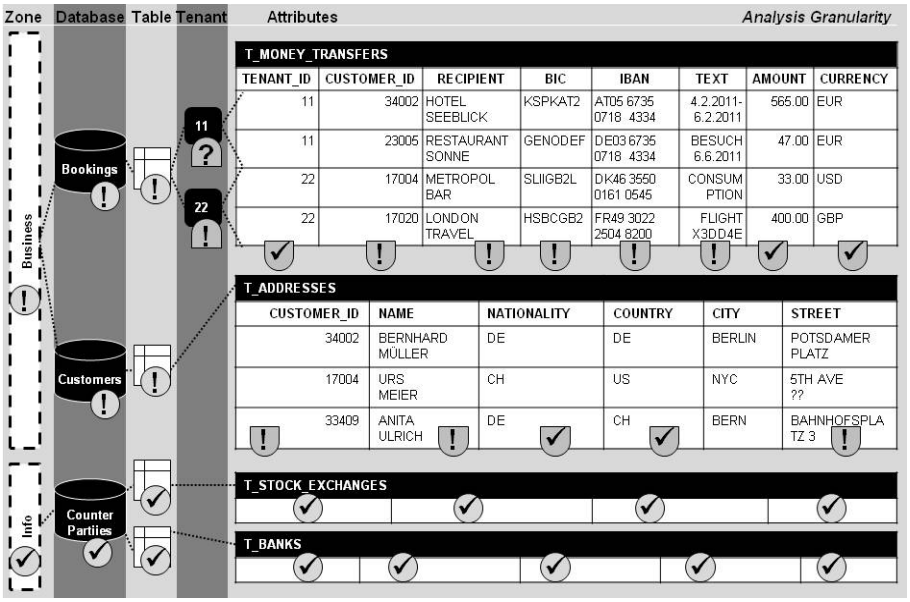

Figure 4: Example for Data Classification ("!" means red, "✓" green)

Thus, classifying data is an intellectual task. It might also need input from the legal and compliance department. Two subtasks are important:

- Collecting all laws, regulations, and company internal rules (e.g. how to protect data of politically exposed customers).
- Analyzing the organization: users and roles, business processes, and the tasks of the users. This results in a list of users needing access to which data items.

The outcome is *privacy rules*, e.g. in form of a text document They are a kind of informal function C.

Testers use them to classify all data items of the IS landscape. They are classified as red when they identify concrete customers. The classification starts on the column level.[2] As Figure 4 illustrates, the customer ID in table T_MONEY_TRANSFERS is red. So are the recipient's name, her IBAN, and the booking text. Amount, currency, the recipient's BIC and the customer's tenant ID are uncritical (green). For table T_ADRESSES, again, the customer ID is red. So are name and street. City, country, and nationality are green (this depends often on the concrete context). The tables T_STOCK_EXCHANGE and T_BANKS do not contain red columns.

The column classification can be aggregated. Tables without any red column are green. Databases without any red table are green. Zones without any red database are green. Such aggregations ease the analysis of data access paths in the next subsection. But a last remark on multi-tenancy: users must not see data items of other tenants. If the user belongs to tenant 11, all data of other tenants (e.g. 22) is red.

[2] For ease of presentation, the paper abstracts from the multi-column aspect. Columns on their own might not identify a customer, a combination of them might. An example is two columns, business sector and balance sheet sum. One column alone is not enough to identify a company. The combination of two can make it quite easy.

3.3 Data Access Analysis

The data access analysis states whether the IS landscape is data privacy compliant. It is based on the DPM data access diagram. Two methods help building up this diagram, testing and inspection. *Testing* means doing something and observing the result. One could log in as a UK credit officer and search for US and UK customers. It is OK if he sees UK customers, but he must not see US ones. In contrast, manual or (semi-)automatic *inspection*s look only at the configuration, e.g. which access rights UK credit officers have. This is faster (i.e., cheaper), but must be verified and complemented with some tests.

Access control can be *established* on three layers: the network, the application layer, and the database (Figure 5, middle). It *affects* access on five levels: zone access, application access, (application) feature access, database table access, and column access (left). The assessment table (middle) compiles the assessment needs. One dimension is authentication (Who am I?), the second is authorization (What am I allowed to do?). The second is the implementation of the technology and how it is configured. For illustration purposes, the following discussion is based on a Microsoft Server network layer combined with an Oracle database.

The data access analysis starts on the top layer, the zone. A **zone analysis** checks whether a user can login into the zone (authentication). The network layer configuration has the answer. One has to check the directory groups of the Active Directory (AD). The analysis succeeds if the user does not have access to the zone. It succeeds, too, if the user has access *and* the zone is "green". In other words, the zone must not contain red data items (see Section 3.2). If the user has access and the zone is red, an application level analysis must follow. Standard software (e.g. Microsoft Server) is the norm on the network level. It can be assumed to be implemented correctly. There is no need to assess the implementation itself (Figure 5).

An **application analysis** looks at which applications a user can access and which databases such applications connect to. First, the user must be able to start the application. This is the authorization on the network level stored in the AD. Secondly, the user must be allowed to log into the application (if requested by the application).

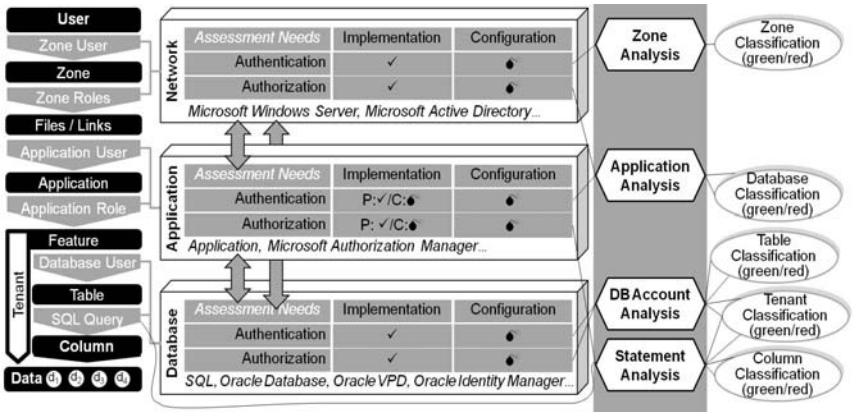

Figure 5: Enriched Data Access Model: access levels (left), access control levels (middle), and assessment needs (right)

💣/✓ assessment needed/not needed, P/C: packaged/custom software

This is the authentication on the application layer. The application can implement its own authentication service. It can also use the Windows Authorization Server [13]. In both cases, an assessment looks at the configuration. In the latter case, the AD is the place to look at, otherwise, the application-specific configuration. In the case of custom software, one might check the correctness of the implementation, i.e. whether the authorization works. This is an issue for legacy code from times when data privacy was not taken as seriously as today.

If the user can log into the application, the assessment continues with gathering all the databases that the application connects to. Sources can be the documentation, long-term application managers, or an own analysis. If there is a connection to at least one red database, a database (db) account analysis must follow.

A **db account analysis** checks if the application can access red tables and columns.[3] The starting point is a list with all the db accounts that the application uses. Again, the list can come from the documentation, from talking with long-term staff, or from own experiments. The db accounts have db roles. Db roles represent access rights to tables. SQL databases store them in the database catalogue. Looking this up is the second step. The third step is matching access rights with the red and green classification of tables and columns. If there are no access rights for red tables and columns, the assessment is successful. If not, a statement analysis must follow. Again, there is no need for checking the correctness of the implemented authentication and authorization mechanisms on the database layer. Commercial databases can be assumed as correct.

If a multi-tenant system relies on the database for tenant separation, the db account analysis also covers this aspect.[4] Up to now, the section focussed on the columns dimension. This is whether certain users can see, e.g. the customer names of T_BALANCES in Figure 6. Multi-tenant systems have a second dimension: the tenants. Table T_BALANCES stores data for three tenants: US, UK, and CH. The tenant separation is based on Oracle's virtual private database (VPD) [14]. VPD demands that tables with tenant-specific data have a tenant ID column. Oracle extends SQL statement (select, insert, update, delete) for such tables transparently. It adds a "WHERE tenant id=XX" clause. Configuring a database for VPD has two aspects. First, the table must be under a VPD policy. Secondly, database users must be associated with a tenant ID. A tenant level analysis must check this.

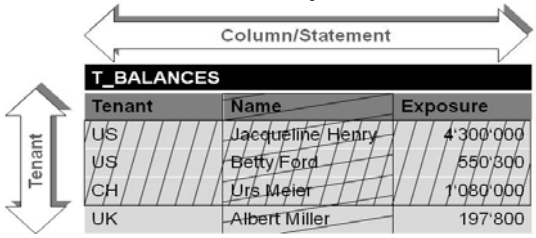

Figure 6: Tenant Level and Statement Level Analysis

[3] The idea is to look at the application and not the application user. This speeds up the analysis. A more detailed analysis is only done for a statement analysis.

[4] Using mature technologies, such as VPD, simplifies the assessment. They can be assumed to be correct. Otherwise, one must test the tenant separation implementation as part of the application analysis.

Finally, if all previous analysis methods did not prove data privacy compliance, the only option left is a **statement analysis**. It looks at all SQL statements, such as those submitted using JDBC. All statements must retrieve only green columns and green tenants. Thus, an application can be assessed as green (even if accessing red tables or columns) if the application "voluntarily" reads only green data items. Such an analysis is costly. It implies inspecting the complete source code. Thus, in practice, it is unpractical. An access path reconstruction (Section 4) might be preferable.

To conclude: There are various levels for checking if users (or usage vectors) access red data items. One starts at the top and steps only down if necessary. Certainly, one can stop at any level and assume that the IS landscape is not data privacy compliant.

4 Access Path Reconstruction

A statement level analysis requires the source code, and is expensive. This makes reconstructing the access path without looking at the implementation an option. The reconstruction treats the IS landscape as a black box. It primarily analyzes the GUI, and tries to build the data access diagram.

The first step is to list all GUIs and group them into GUIs for presenting data ("Customer Overview" in Figure 7) and GUIs for searching for data ("Customer Search" in Figure 7). The presentation GUIs are analyzed for their objects, such as customer static data and exposure, and their attributes, such as name, country, and limit. All attributes must be green (1a). The second step (1b) looks at the object instances (respectively rows) that a user can find. The "Customer Search" GUI, for example, should allow UK users only to search for UK customers.

The two initial steps 1a and 1b allow for an assessment whether a user might see red data items or not. It reflects the tenant and responsibilities (e.g. for the segment retail banking UK). If there is a need for more details, one can also match GUIs to database tables (2). The documentation or the application management team might help. This allows validating the search GUI analysis. One can compare the data items that one has found using the GUI (e.g. UK and US tenant data) with the data stored in the table (e.g. UK, US, and CH data). Thus, one can find overseen data items.

To conclude: An access path reconstruction sounds hard and expensive (and it is). Nevertheless, it allows making an assessment based on GUIs when all other approaches fail.

5 Sanitization Techniques

Sanitization makes red data items green. Two popular techniques are vertical and horizontal greening (Figure 8). Understanding them means understanding the data privacy risks they come with.

Vertical greening transforms or masks data on its way from the database up to the GUI. Figure 8 (left) provides an example. The attribute of the GUI mask "Customer overview" shows all attributes. One is red ("name"). So, it is masked. Vertical greening is an "on-the-fly" greening approach. It becomes active when data is retrieved from the database and shown on the GUI. It supports both sourcing scenarios, global software development and testing and global sourcing of business activities.

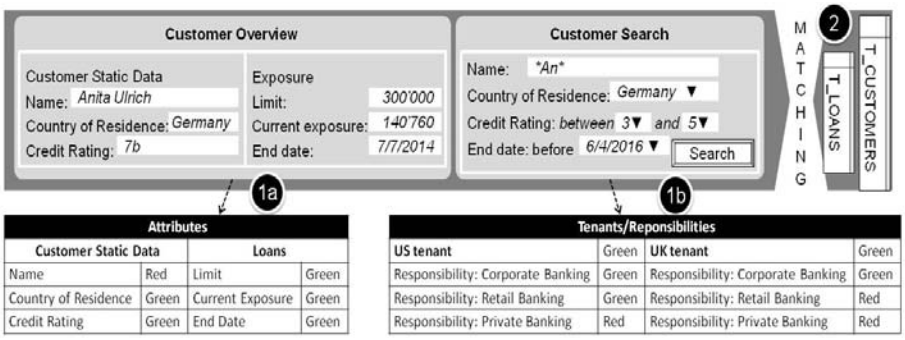

Figure 7: Access Path Reconstruction for CreditPlus sample masks

Horizontal greening replicates the data (e.g. the complete database). Then, the replica is "greened". One can drop red columns or tables, delete all data in red columns, shuffle the values, or replace them with synthetic values. Figure 8 (right) illustrates this. The data item “Anita Ulrich” becomes “Miller AG” in the replicated database. The rest of the application, i.e. the way from the database to the GUI, can remain unchanged.

Horizontal greening is a batch activity. It can be used for the global software development and testing scenario. It is suitable for the global sourcing of business activities scenario for OLAP-style applications only. OLAP-style application do not write data to the database. Combining horizontal greening and OLTP-style applications requires addressing the replication problem (and being able to understand what rows are a replica of which other rows).

Assessing the effectiveness of greening addresses completeness and greenness. *Completeness* demands finding *all* red data items. *Greenness* demands that the greened data really become uncritical. It must not happen that customer names are masked, so that they can be reconstructed (easily).

Assessing the completeness requires finding the links between the red columns of database tables and GUIs. Then, one has to check whether there is a greening mechanism in place. It can be either (a) directly on the way between the database and the GUI (vertical greening) or (b) the data item is derived as a copy from another database and during or after the copy the data item is greened.

The second aspect to be checked is greenness. Horizontal and vertical greening require different approaches. Vertical greening can only be checked by inspecting the online masking algorithm or running many tests. Both options are feasible for horizontal greening, too. However, one can simply assess the replica after it is greened. One sees immediately how good greening works for large data sets.

To conclude: Many companies use greening techniques. Checking their completeness and greenness can be implemented easily in data privacy assessments.

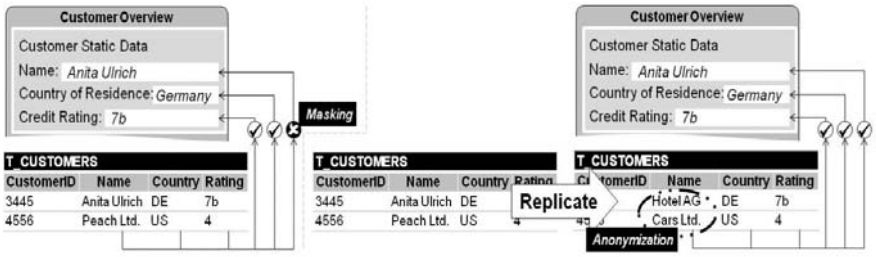

Figure 8: Greening Techniques (left: vertical greening, right: horizontal greening)

6 Related Work

When looking on related work, there is, first, pioneering work on access control concepts such as role-based access control [8] (also a base for this paper). Concrete implementations such as Bertino, et al. [15], elaborate security challenges for databases, and how systems overcome them. Similar work on the operating system layer looks on how to enforce usage control for X windows systems [16].

The logical next step is to check whether applications implement access control correctly. The work of Pretschner, et al. [17] on model-based tests for access control policies falls into this category. They discuss how to generate test cases efficiently for testing policies and which input they need. Le Traon and Baudry [18] focus on the relationship between functional tests and security policy tests and how they overlap. Besides work on testing, there are also approaches for formalizing the systems and reasoning about them, for example, whether business processes have data leaks (Accorsi and Wonnemann [19]). Stoller et al. [20] and Schaad and Moffett [21] are interested in whether a (given complex) formal access policy is compliant. This complements this paper, which provides a methodology for *extracting* a simple yet meaningful policy model from a real system. The compliance decision itself is trivial.

A different research direction focuses on the usage of data. Stufflebeam, et al. [20], for example, compare P3P and EPAL. They are newer policy specification techniques for formalizing the purpose for storing data. The manifesto for Hippocratic Databases (Agrawal, et al. [21]) demands a privacy-aware database management system. Whereas this work is more on the requirements level, later work of Byun and Li [22] discusses how to implement such a database. They associate data stored in a database with reasons why it is stored (e.g. for marketing, for research, etc.). Queries also have a purpose and return only the data stored for this purpose.

Finally, there are approaches to actively test the security of systems. Internal engineers or external consultants try to break into the IT landscape. They attack actively to identify security leaks (see Palmer [23] about “ethical hacking”).

7 Summary

This paper provides data privacy assessments based on four key concepts:

- The *usage vector* for formalizing factors influencing whether users are allowed to see certain data.
- The more theoretical *data criticalness function* which decides for a usage context if a data item is allowed to be seen. It is complemented on the practical side with privacy rules and a guideline for *classifying data* complements.
- The theoretical concept of *data access diagrams* linking usage vectors, data items and application features to see who can access which data. The practical counterpart is a *data access analysis* with the concrete examples of a zone, application, database account, and statement analysis.
- A formal *data privacy compliance correctness* criterion.

The paper also discussed briefly data access based only on GUIs, for example, for legacy applications and the impact of sanitization. In one sentence: The methodology provides a quick "health-check" for IT managers stating whether users can access only the data that they are supposed to see.

Acknowledgments: The author would like to thank Hans-Joachim Lotzer for the valuable discussions.

References

1. Five Countries: Cost of Data Breach, Ponemon Institute, 2010
2. Breach at Los Alamos: A special report, The New York Times, March 6th, 1999
3. 100,000 non-clinical NHS staff have access to confidential records, The Telegraph, March 25th, 2010
4. The Liechtenstein Connection, Spiegel Online International, February 16th, 2008
5. International Tax Evasion Scandal Spreads, Spiegel Online International, March 3rd, 2008
6. Up and away: A dogfight over frequent-flyer miles is distracting Germany's politicians , The Economist, August 8th, 2002
7. U.S. Officials Quiz Sony on Data Theft, The Wall Street Journal, April 30th, 2011
8. Ferraiolo, D., et al.: Proposed NIST Standard for Role-Based Access Control, ACM Transactions on Information System Security, Vol. 4. No. 3, p. 224-274, 2001
9. Haller, K.: White-box testing for database-driven applications: a requirements analysis, 2nd Int. Workshop on Testing Database Systems, DBTest'09, Providence, RI, June 29, 2009
10. Haller, K.: The test data challenge for database-driven applications, 3rd Int. Workshop on Testing Database Systems, DBTest 2010, Indianapolis, IN June 7, 2010
11. Haller, K.: On the implementation and correctness of information system upgrades, IEEE Int. Conference on Software Maintenance (ICSM), Sept. 12-18, 2010, Timisoara, Romania
12. Haller, K.: Web services from a service provider perspective: tenant management services for multitenant information systems, in: ACM SIGSOFT Software Engineering Notes, Vol. 36(1), pp. 1-4
13. Windows Authorization Manager, MSDN, http://msdn.microsoft.com/ (retrieved May 22nd, 2011)
14. Browder, K. and Davidson, K.A.: The virtual private database in Oracle9iR2, White Paper, Oracle, January (2002)
15. Bertino, E., et al.: Database security: research and practice, Information Systems, Vol. 20, No. 7, pp. 537-556, 1995
16. Pretschner, A., et al.: Usage Control Enforcement with Data Flow Tracking for X11., 5th Intl. Workshop on Security and Trust Management (STM), Saint Malo, France, September 24-25, 2009
17. Pretschner, A., et al.: Model-Based Tests for Access Control Policies, Int. Conference on Software, Testing and Validation, April 9-11, Lillehammer, Norway, 2008
18. Le Traon, Y., Baudry, B. : Testing security policies: going beyond functional testing, Int. Symposium on Software Reliability (ISSRE '07), Sweden, 5-9 Nov., 2007
19. Accorsi, R., Wonnemann, C.: InDico: Information Flow Analysis of Business Processes for Confidentiality Requirements, ERCIM Workshop on Security and Trust Management, 23-24 September , Athens, Greece, 2011
20. Stoller, S., et al.: Efficient Policy Analysis for Administrative Role Based Access Control, 14th Conf. on Computer and Communications Security (CCS)", Alexandria, VA, 2007
21. Schaad, A. and Moffett, J.: A lightweight approach to specification and analysis of role-based access control extensions, SACMAT, Monterey, CA, 2002
22. Stufflebeam, W., et al.: Specifying Privacy Policies with P3P and EPAL: Lessons Learned, Workshop on Privacy in the Electronic Society, WPES, Washington, DC, October 28, 2004
23. Agrawal, R., et al.: Hippocratic Databases, 28th Int. Conference on Very Large Data Bases (VLDB '02), August 20-23, 2002, Hong Kong, China
24. Byun, J.-W., Ninghui Li: Purpose based access control for privacy protection in relational database systems. VLDB Journal, Volume 17(4), pp. 603-619, 2008
25. Palmer, C.: Ethical hacking, IBM Systems Journal, Vol. 40 (3), 2001