ANALISA KEAMANAN DATABASE SERVER MENGGUNAKAN TEKNOLOGI VIRTUAL PRIVATE DATABASE DAN NOTIFIKASI DATABASE SERVER MENGGUNAKAN AGENT BERGERAK

Bambang Sugiantoro[1)], **Jazi Eko Istianto**[2)]
[1,2)] Program Pasca Sarjana Ilmu Komputer Fakultas Matematika dan Ilmu Pengetahuan Alam Universitas Gadjah Mada Yogyakarta
Jl Sekip Utara Bulaksumur Yogyakarta 55281
e-mail : bambang05@gmail.com , jazi@ugm.ac.id

Abstrak
Keamanan Sistem keamanan database server sangat diperlukan untuk menjaga data agar tidak digunakan oleh yang tidak mempunyai otoritas terhadap data tersebut , Virtual Private database memungkinkan kontrol akses mencapai baris yang spesifik dari database sehingga user dapat mengakses ke data yang digunakan.Metode yang dilakukan dengan cara mengekplorasi keamanan database server yang ada , kemudian akan dibandingkan arstektur database server menggunakan berdasarkan teknologi Virtual Private Database , User database dengan berbagai level privilege dapat memiliki hak untuk melihat atau mengubah baris data yang dilabeli.Menghasilkan analisa dan arsitektur Keamanan database server menggunakan berdasarkan teknologi Virtual Private Database. Teknologi agent bergerak digunakan untuk notifikasi database server.

***Keyword** : Virtual private database ,Agent bergerak ,database server*

1. PENDAHULUAN

Keperluan keamanan database timbul dari kebutuhan untuk melindungi data. Pertama, dari kehilangan dan kerusakan data. Kedua, adanya pihak yang tidak diijinkan hendak mengakses atau mengubah data. Permasalahan lainnya mencakup perlindungan data dari *delay* yang berlebihan dalam mengakses atau menggunakan data, atau mengatasi gangguan *denial of service*.

Kontrol akses terhadap terhadap informasi yang sensitif merupakan perhatian terutama oleh manajer, pekerja di bidang informasi, *application developer*, dan DBA. Kontrol akses selektif berdasarkan authorisasi keamanan dari level user dapat menjamin kerahasiaan tanpa batasan yang terlalu luas. Level dari kontrol akses ini menjamin rahasia informasi sensitif yang tidak akan tersedia untuk orang yang tidak diberi ijin (authorisasi) bahkan terhadap user umum yang memiliki akses terhadap informasi yang dibutuhkan, kadang-kadang pada tabel yang sama.

Mengijinkan informasi dapat dilihat atau digunakan oleh orang yang tidak tepat dapat menyulitkan, merusak, atau membahayakan individu, karir, organisasi, agensi, pemerintah, atau negara. Namun untuk data tertentu seringkali bercampur dengan data lainnya, informasi yang kurang sensitif yang secara legal dibutuhkan oleh berbagai user. Membatasi akses terhadap semua table atau memisahkan data sensitive ke database terpisah dapat menciptakan lingkungan kerja yang tidak nyaman yang membutuhkan biaya besar pada hardware, software, waktu user, dan administrasi.

Ketersediaan informasi terus mengalami peningkatan pesat, metoda untuk penyandian dan penyimpanan informasi juga mengalami peningkatan. Perkembangan jumlah sumber informasi ini membawa beberapa permasalahan, diantaranya tentang bagaimana mengkombinasikan tempat penyimpanan data yang terdistribusi dan berbeda. Informasi pada suatu organisasi atau perusahaan biasanya disimpan di lokasi yang terpisah dan berbeda-beda format. Ketika terjadi peningkatan kapasitas tempat penyimpanan dan besarnya biaya pencarian informasi, perusahaan dihadapkan pada masalah melimpahnya jumlah data.

Basis data terdistribusi adalah basis data dimana data ditempatkan di beberapa lokasi, tetapi menerapkan suatu mekanisme tertentu untuk membuatnya menjadi satu kesatuan basis data (Fathansyah, 2004). Sebuah sistem basis data terdistribusi hanya mungkin dibangun dalam sebuah sistem jaringan komputer. Berbeda dengan basis data terpusat yang datanya ditempatkan di beberapa lokasi tetapi tidak saling berhubungan.

Akses basis data terdistribusi merupakan proses untuk mencampur dan mencocokkan, *query*, memanipulasi, dan menggabungkan data dalam suatu basis data terdistribusi. Akses basis data akan menampilkan hasil *query* yang diinginkan pemakai. Akses basis data tidak melakukan pelacakan terjadinya perubahan pada basis data pada tiap-tiap *host*.

Agent Bergerak merupakan terobosan baru dalam perkembangan perangkat lunak. Agen merupakan entitas perangkat lunak yang didedikasikan untuk tujuan tertentu (Tri, 2001). Keunggulan agen telah menarik perhatian

banyak pihak. Salah satunya adalah IBM Jepang yang mengembangkan Aglet SDK (*Software Development Kit*) untuk mempermudah pemrogram dalam membuat Agen berbasis Java. Perpaduan antara Agen dan Java akan menghasilkan *software* jaringan yang tangguh dalam konsumsi *bandwith* rendah, sehingga dipilih teknologi ini dalam membangun akses database terdistribusi.

2. TINJAUAN PUSTAKA

Pada penelitian sebelumnya telah dilakukan penelitian tentang Oracle label security pada oracle database 10g(irawan , 2003) , paper yang akan di bahas disini dikaitkan dengan tinjauan Virtual private database dan juga agent bergerak untuk notifikasi database server.

3. METODE PENELITIAN

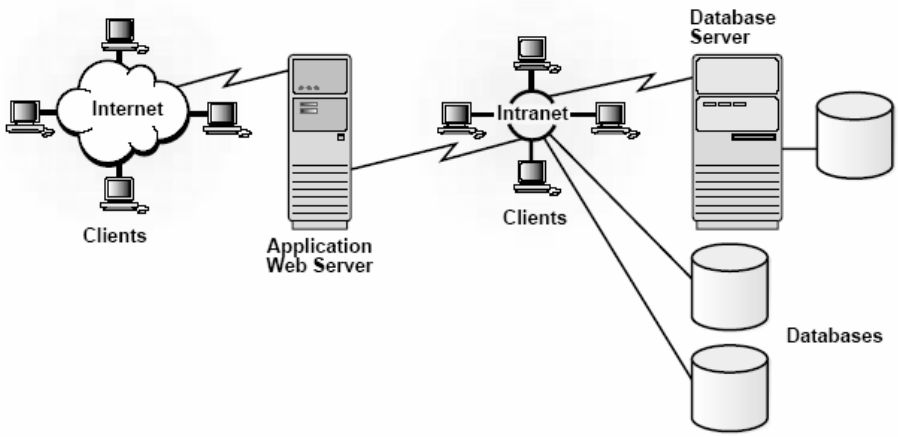
Tahap-tahap :

1. *Requirement Gathering*, mengambil informasi lengkap dari pengguna tentang sistem yang akan dibangun. Wawancara dilakukan dengan pengguna yang memiliki hubungan langsung dengan sistem. Tahap ini menyarankan untuk mewawancarai pengguna yang memiliki kemampuan teknis.
2. *Analysis*, menggali lebih dalam hasil yang diperoleh dalam tahap sebelumnya. Tahap ini mengkaji permasalahan pengguna dan menganalisis solusinya.
3. *Design,* merancang solusi yang dihasilkan pada tahap analisis. Tahap *analysis* dan *design* dapat berjalan dua arah saling menyesuaikan sampai diperoleh rancangan yang tepat.
4. *Development,* tahap ini ditangani oleh pemrogram untuk membangun kode program dan *user interface* Pengujian program dan dokumentasi juga dilakukan pada tahap ini.

Deployment, mendistribusikan produk yang dihasilkan kepada pengguna. Tahap ini mencakup instalasi dan perencanaan *backup* data bila diminta oleh pengguna sesuai dengan perjanjian sebelumnya

4. HASIL DAN PEMBAHASAN

Keamanan pada komputer mencakup perlindungan data yang terkomputerisasi dan proses modifikasi, perusakan, atau delay yang tidak diijinkan. Pada masa internet, ancaman terhadap data meningkat secara eksponensial. Gb.1 dibawah ini menunjukkan lingkungan komputasi kompleks yang harus tercakup dalam perencanaan keamanan data.



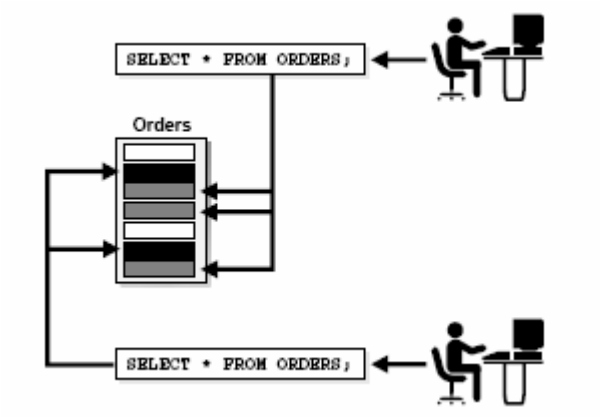
Gambar.1 Lingkungan dari Kebutuhan Keamanan Data

Staff keamanan, administrator, dan programmer aplikasi harus melindungi database dan server dimana database berada. Mereka harus mengatur dan melindungi hak user pada database internal, dan menjamin privasi *electronic commerce* sebagaimana pelanggan yang mengakses database tersebut.

Virtual Private Database

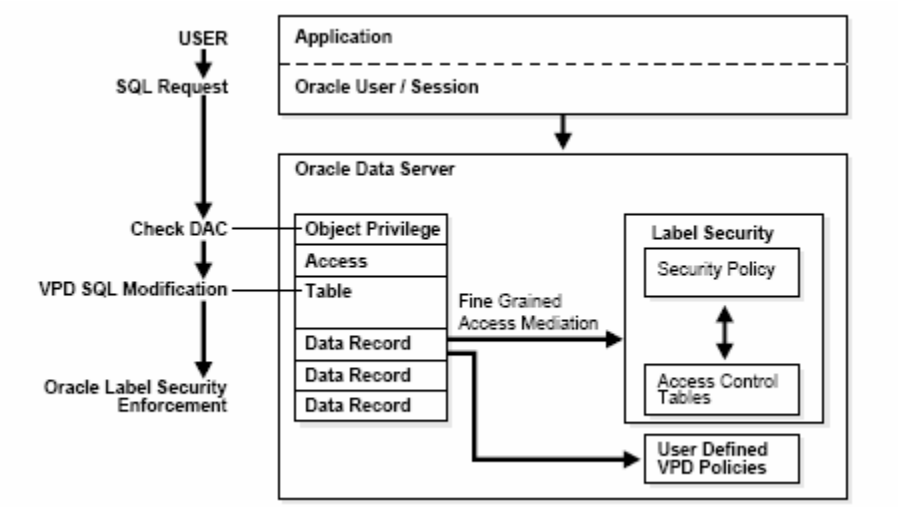
Oracle Label Security (OLS) bergantung pada konsep Virtual Private Database (VPD) untuk memperluas keamanan pada level baris. Secara esensial, ketika aturan bisnis dipersiapkan melalui OLS, VPD menambahkan kriteria seleksi tambahan yang perlu ke setiap pernyataan SQL yang dikeluarkan untuk membatasi akses user ke hanya data yang perlu. Kelebihan dari VPD ialah aplikasi aturan ditangani "dibalik layar" tanpa diketahui user. Misalnya, diterapkan aturan sehingga user SCOTT hanya dapat melihat baris pada tabel ORDERS yang ditandai

USERID-nya saja, VPD menambahkan kriteria seleksi (WHERE ORDERS.USERID = 'SCOTT') pada query. Hal ini dapat diterapkan pula pada user lainnya yang hanya dapat melihat data yang diperbolehkan seperti yang digambarkan berikut ini :



Gambar.2 Teknologi VPD Oracle

Aplikasi user dalam session Oracle menghasilkan SQL Request. Oracle mengecek *privilege* DAC, menjamin bahwa user memiliki *privilege* SELECT pada tabel. Kemudian dicek apakah aturan VPD telah diterapkan pada tabel untuk menjamin bahwa tabel tersebut diproteksi. Pernyataan SQL diubah pada proses selanjutnya. Hal tersebut digambarkan dalam arsitektur Oracle Label Security sebagai berikut :

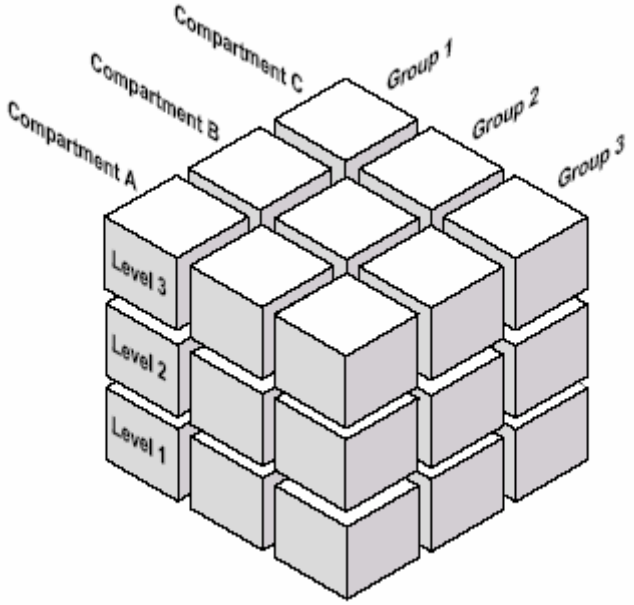


Gambar.3 Arsitektur Oracle Label Security

Keamanan dengan label menambah perlindungan data diluar DAC yang menentukan operasi yang dapat dilakukan user terhadap data dalam suatu objek, seperti tabel atau *view*. Aturan OLS mengontrol akses terhadap data dalam tiga dimensi :

- Data Label : menunjukkan level dan karakteristik sensitivitas baris dan kriteria tambahan yang harus dipenuhi user untuk mengakses baris tersebut.
- User Label : menunjukkan level sensitivitas user ditambah kompartemen dan grup yang membatasi akses user ke data yang diberi label.
- Aturan *Privilege* : user diberi hak spesifik untuk menjalankan operasi khusus atau untuk mengakses data diluar authorisasi label mereka.

OLS menggunakan tiga dimensi untuk mendefinisikan *user's permission* untuk mengakses data dalam baris, yaitu level, kompartemen dan grup. Gambar dibawah ini mengilustrasikan ketiga dimensi tersebut.



Gambar. 4 Klasifikasi Data Secara Logis

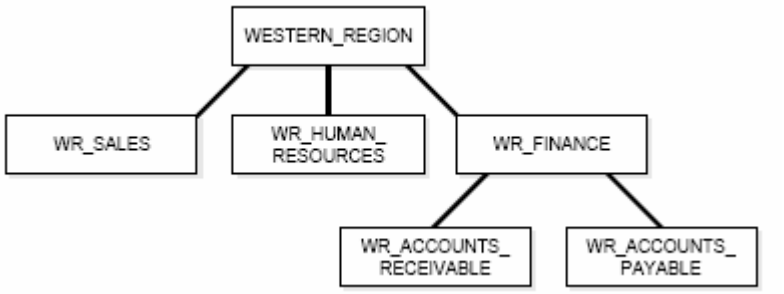
Level adalah tingkatan yang menyatakan sensitivitas informasi. Semakin sensitif informasi, semakin tinggi pula levelnya. Setiap label harus memiliki satu level. Untuk setiap level, administrator mendefinisikan bentuk numerik dan karakter, misalnya :

Tabel.1 Contoh Level

Bentuk Numerik	Bentuk Panjang	Bentuk Pendek
40	HIGHLY_SENSITIVE	HS
30	SENSITIVE	S
20	CONFIDENTIAL	C
15	PUBLIC	P

Walaupun administrator mendefinisikan kedua bentuk panjang dan pendek, hanya bentuk pendek yang terlihat oleh user ketika hendak memanipulasi label. Label lain yang umum didefinisikan ialah TOP_SECRET, SECRET, CONFIDENTIAL, dan UNCLASSIFIED, atau TRADE_SECRET, PROPRIETARY, COMPANY_CONFIDENTIAL, dan PUBLIC_DOMAIN. Jika hanya level yang digunakan, user dengan level 40 (pada contoh ini) dapat mengakses atau mengubah data apapun yang memiliki level 40 atau dibawahnya.

Grup mengidentifikasi organisasi yang memiliki atau mengakses data, seperti EASTERN_REGION, WESTERN_REGION, WR_SALES. Semua data yang berhubungan dengan departemen tertentu dapat memiliki grup departemen dalam label. Grup berguna untuk mengontrol distribusi data, dan sebagai reaksi terhadap perubahan organisasi. Grup bersifat hirarki dimana data label dibuat berdasarkan infrastruktur organisasi. Grup dapat dihubungkan dengan grup *parent*. Misalkan :



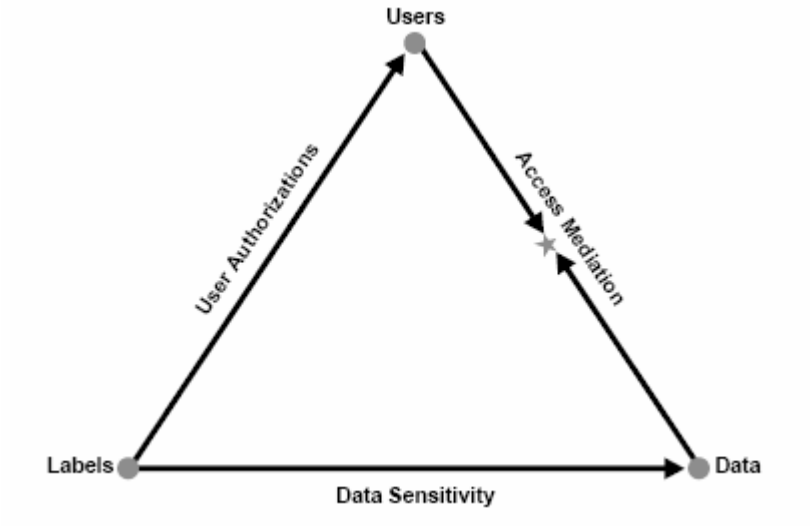
Gambar.5 Contoh Hirarki Grup

Pada Gb.3, grup WESTERN_REGION terdiri dari tiga subgrup : WR_SALES, WR_HUMAN_RESOURCES, dan WR_FINANCE. Subgrup WR_FINANCE dibagi lagi menjadi WR_ACCOUNTS_RECEIVABLE dan WR_ACCOUNTS_PAYABLE. Tabel dibawah ini menunjukkan struktur organisasi diatas dalam bentuk grup OLS.

Tabel.2 Contoh Grup

Bentuk numerik	Bentuk Panjang	Bentuk Pendek	Grup Orang Tua
1000	WESTERN_REGION	WR	
1100	WR_SALES	WR_SAL	WR
1200	WR_HUMAN_RESOURCES	WR_HR	WR
1300	WR_FINANCE	WR_FIN	WR
1310	WR_ACCOUNTS_PAYABLE	WR_AP	WR_FIN
1330	WR_ACCOUNTS_RECEIVABLE	WR_AR	WR_FIN

Untuk dapat mengakses data yang diproteksi OLS, user harus memiliki authorisasi berdasarkan label yang didefinisikan. Dibawah ini ditunjukkan hubungan antara user, data, dan label.



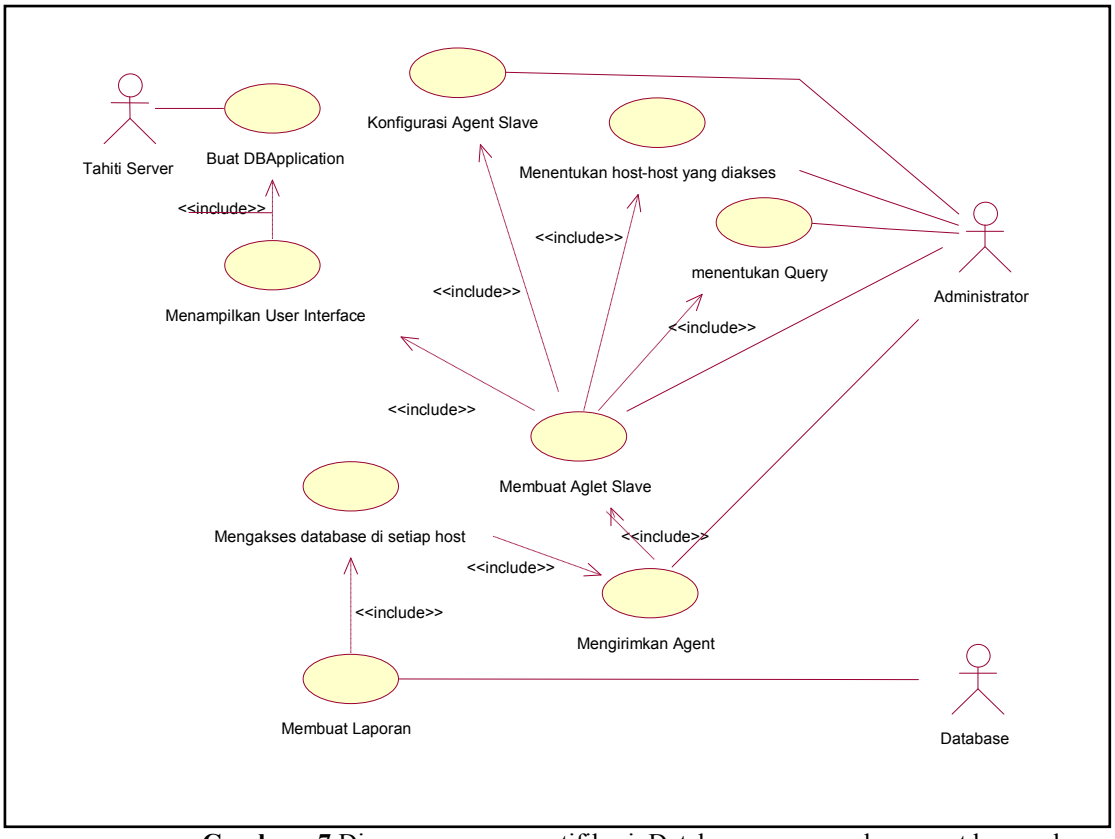
Gambar.6 Hubungan Antara User, Data, dan Label

- Data label menspesifikasi sensitivitas baris data.
- User label memberikan authorisasi ke user yang benar.
- *Access mediation* antara user dan baris data bergantung pada label.

Selama *access mediation*, OLS membandingkan nilai yang tersimpan didalam kolom label dengan label *permission* user. Jika user diberi hak yang memadai untuk mengakses baris, maka transaksi berlanjut. Untuk menjalankan perintah SELECT, user harus diberi akses *read mode*. Untuk menjalankan perintah Data Manipulation Language (INSERT, UPDATE, DELETE, atau MERGE), user harus diberi akses *write mode*.

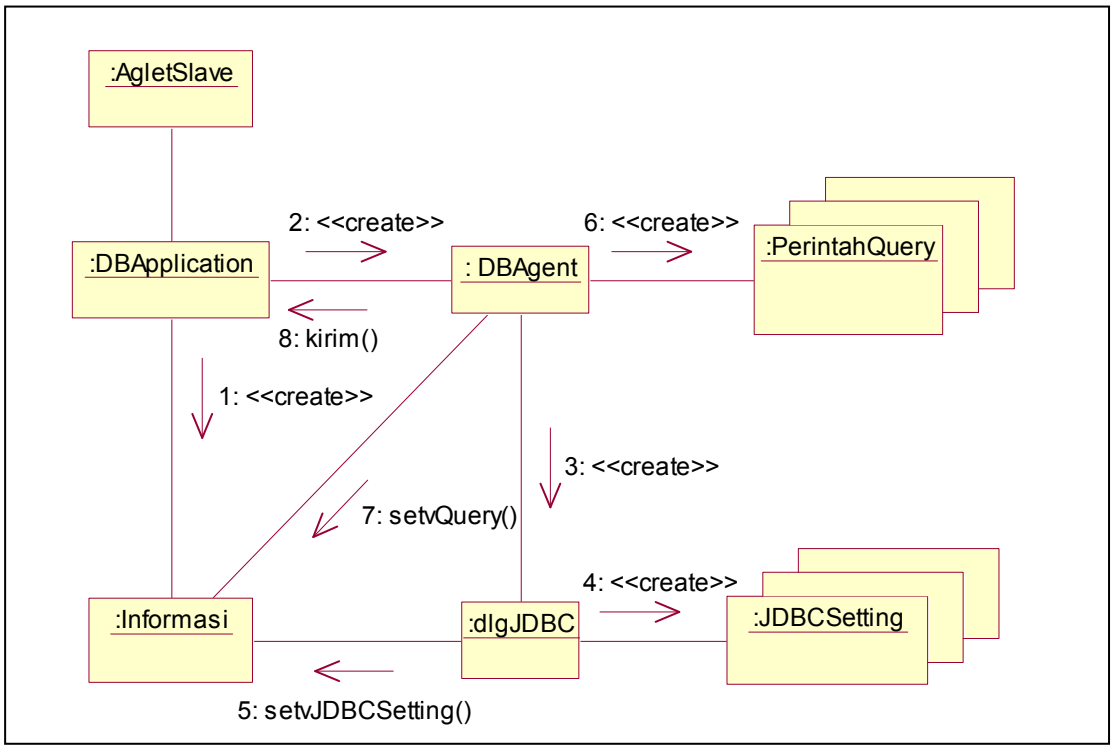
Dalam pengembangan perangkat lunak ini dapat diidentifikasi dua buah aktor yaitu berupa manusia dan perangkat lunak Tahiti *server*. Aktor adalah seseorang atau sesuatu yang berada di luar sistem namun berinteraksi dengan sistem (Quatrani, 2003).

Diagram *use case* pada gambar 7 memiliki dua buah aktor yaitu administrator dan Tahiti *server*. Pengguna dapat berinteraksi dengan lima buah *use case* dalam sistem perangkat lunak ini secara bergantian. Lima buah *use case* tersebut yaitu *use case* Konfigurasi Agent Slave, *use case* Menentukan *host-host* yang diakses, Menentukan Query, *use case* Membuat Aglet slave, *use case* Mengirimkan *agent,* dan *use case* Membuat laporan. Aktor Tahiti *Server* digunakan untuk menciptakan DBApplication dengan fungsi untuk mengirim dan menerima Aglet Slave yang dikonfigurasikan sebelumnya. Interaksi pengguna dengan program lebih banyak melalui Menu utama yang dibuat pada *use case* Menampilkan *User Interface*. Setelah terjadi pengiriman agen, setiap agen akan melakukan tugas yang dikonfigurasikan sebelumnya di setiap *host* yang dikunjungi. Proses ini digambarkan dengan *Use case* Mengakses *database* di setiap *host*.



Gambar . 7 Diagram *use case* notifikasi Database menggunakan agent bergerak

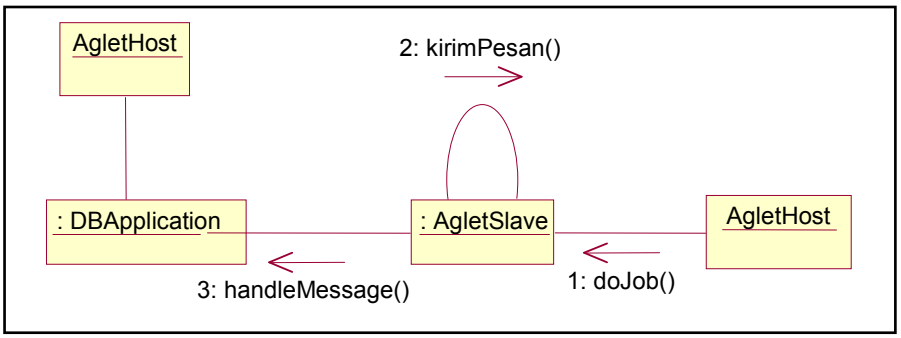
Hasil analisis pembuatan agen pada diagram *use case*, disinkronisasikan dengan *class* diagram, menghasilkan *collaboration* diagram pembuatan agen seperti pada gambar 8
Setelah objek dari AgletSlave dibuat, administrator dapat mengirimkannya ke jaringan.



Gambar. 8 *Collaboration* diagram Membuat Aglet *slave*

Objek dari *class* AgletSlave kemudian berpindah dari satu *host* ke *host* yang lain sesuai dengan rencana perjalanan yang telah ditentukan. Langkah-langkah akses *database* yang dilakukan oleh agen digambarkan dengan *collaboration* diagram pada gambar 3.4.

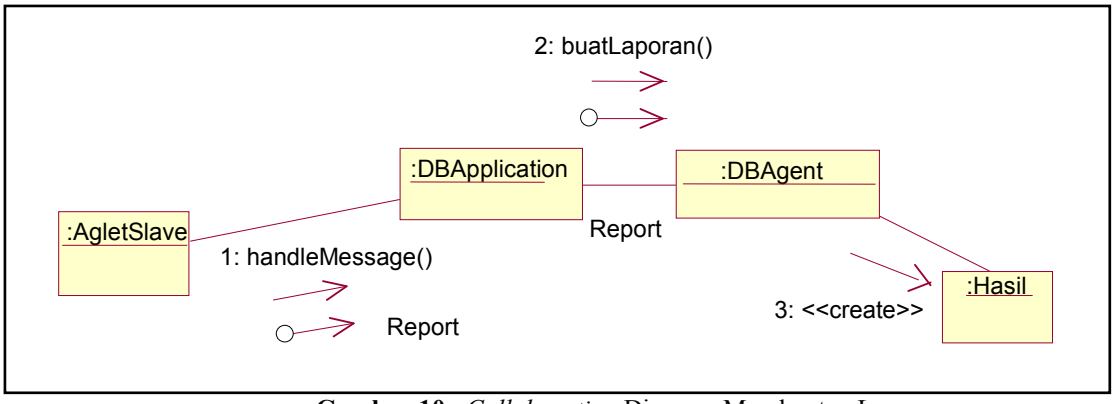
Saat agen tiba di *host* tujuan, Tahiti Server akan memanggil *method* run() dari agen yang berisi method doJob() untuk mengambil data yang diperlukan. Data-data yang diambil disimpan dalam objek dari *class* Report sebagai bahan untuk membuat laporan.



Gambar. 9 *Collaboration* diagram akses *database* oleh AgletSlave

Hal-hal diluar dugaan sering kali terjadi, misalnya saat *host* yang akan dideteksi sedang tidak aktif, *host* tersebut akan dilewati. Bila terjadi kesalahan atau deteksi selesai, maka agen akan kembali ke *host* asal dengan membawa objek Report masing-masing *host*.

Di *server*, DBApplication akan memanggil *method* buatLaporan() pada objek DBAgent. Objek Report akan dikonversi menjadi objek Hasil. Objek hasil menyimpan hasil dari akses yang dilakukan pada *database* tiap-tiap *host*. Objek ini disimpan ke dalam Vektor dan dapat disimpan ke dalam bentuk *file* teks. Laporan yang dihasilkan nanti berupa hasil eksekusi *query* umum yang dihasilkan oleh masing-masing DBMS pada masing-masing *host* yang dikunjungi. Gambar 10 menjelaskan tentang pembuatan laporan oleh DBApplication.



Gambar 10 *Collaboration* Diagram Membuatan Laporan

5. KESIMPULAN

Telah berhasil dibuat analisa keamanan database server menggunakan pendekatan priavte virtual database dan perancangan notifikasi database server berbasis agent bergerak.

6. DAFTAR PUSTAKA

Dharwiyanti, Sri, 2003, Pengantar Unified Modeling(UML), http://www.ilmukomputer.com, (2003 accessed 10-02-2005)

Irawan W, 2003, Oracle Label Security pada Oracle 10g, ITB

Jogiyanto, 2000, Pengenalan Komputer, Penerbit Andi, Yogyakarta

Lange, D.B, 1997, *Java Aglet Application Programming Interface* (J-AAPI) *White Paper – Draft 2,* <http://www.trl.ibm.co.jp/aglets/api/Package-com.ibm.aglets.html>, (19-2-1997, accessed 10-02-2005)

Oshima, Mitsuru, 1998, Aglet Spesification 1.1 Draft, <http://www.trl.ibm.co.jp/aglets/spec11.htm>, (8-9-1998, accessed 10-02-2005)

Schmuller, Joseph, 1999, *Teach Yourself UML in 24 Hours*, Sams Publishing, Indianapolis.

Suhendar dkk, 2002, Visual Modelling Menggunakan UML dan Rational Rose, Informatika Bandung, Bandung

Leonardo, Ian, 2003, Pemrograman Database dengan Java, Elex Media Komputindo, Jakarta

http://download-west.oracle.com/docs/cd/B13789_01/network.101/b10774.pdf

http://download-west.oracle.com/docs/cd/B13789_01/network.101/b10777.pdf

Jim Czuprynski, Oracle Label Security Part 1: Overview, online at

http://www.databasejournal.com/features/oracle/article.php/3065431

http://download-west.oracle.com/docs/cd/B13789_01/network.101/b10773.pdf

http://otn.oracle.com/deploy/security/pdf/ds_security_db_labelsecurity_10r1_0104.pdf