

KAJIAN APLIKASI MOBILE AGENT UNTUK DETEKSI PENYUSUPAN PADA JARINGAN KOMPUTER

Oleh Bambang Sugiantoro email : bambang@upnyk.ac.id

NIM:23202008

1. Pendahuluan

1.1 Latar Belakang

Jaringan komputer terus mengalami perkembangan, baik dari skalabilitas, jumlah node dan teknologi yang digunakan. Hal ini memerlukan pengelolaan jaringan yang baik agar ketersediaan jaringan selalu tinggi. Tugas pengelolaan jaringan yang dilakukan administrator jaringan memiliki banyak permasalahan, diantaranya yang berkaitan dengan keamanan jaringan komputer.

Penyusupan (*intrusion*) adalah seseorang yang berusaha merusak atau menyalahgunakan sistem, atau setiap usaha yang melakukan *compromise* integritas,kepercayaan atau ketersediaan suatu sumberdaya komputer [2]. Definisi ini tidak bergantung pada sukses atau gagalnya aksi tersebut, sehingga berkaitan dengan suatu serangan pada sistem komputer.

Intrusion detection (ID) singkatnya adalah usaha mengidentifikasi adanya penyusup yang memasuki sistem tanpa otorisasi (misal *cracker*) atau seorang user yang sah tetapi menyalahgunakan (*abuse*) privelege sumberdaya sistem (misal *insider threath*) [24]. Intrusion Detection System (IDS) atau Sistem Deteksi Penyusupan adalah sistem komputer (bisa merupakan kombinasi software dan hardware) yang berusaha melakukan deteksi penyusupan [23}. IDS akan melakukan pemberitahuan saat mendeteksi sesuatu yang dianggap sebagai mencurigakan atau tindakan ilegal. IDS tidak melakukan pencegahan terjadinya penyusupan. Pengamatan untuk melakukan pemberitahuan itu bergantung pada bagaimana baik melakukan konfigurasi IDS.

Software Agent (selanjutnya disebut agent saja) adalah entitas perangkat lunak yang didedikasikan untuk tujuan tertentu [21]. Agen bisa memiliki ide sendiri mengenai bagaimana menyelesaikan suatu pekerjaan tertentu. Sejumlah riset tentang agentelah

membuat bermacam aplikasi, misal untuk *distributed meeting scheduler*, *network mapping*, *auction*, dan *searching database*.

1.2 Identifikasi masalah

Saat ini jenis serangan baru pada jaringan komputer tumbuh dengan cepat melebihi kemampuan update dari sistem. Suatu jaringan komputer yang kompleks harus selalu dijaga keamanannya oleh administrator jaringan. Namun demikian, harus juga diperhatikan agar jangan sampai usaha untuk mengelola keamanan justru akan menurunkan ketersediaan jaringan karena banyak *bandwith* jaringan yang dipakai untuk pengelolaan jaringan tersebut.

Di samping itu, seorang administrator jaringan tidak mungkin sepanjang waktu selalu berada di dekat jaringan komputer yang dikelolanya. Apabila diperlukan, seorang administrator masih dapat melakukan aktivitas pengelolaan keamanan jaringan komputer. Meskipun ia berada jauh dari jaringan komputer yang dikelolanya.

Pada prakteknya terdapat kesulitan untuk memungkinkan perangkat lunak yang dapat melakukan pemantauan kebijaksanaan keamanan dalam lingkungan yang terdistribusi dan heterogen. Solusi tersebut harus dapat menjamin penetapan kebijakan keamanan yang konsisten diseluruh sistem, dan memantau kemungkinan terjadinya kebocoran dan inkonsistensi, khususnya bila terjadi penambahan atau pengurangan sumberdaya sistem.

Untuk itu diperlukan suatu *Agent* yang dapat melakukan tugas-tugas pengelolaan keamanan jaringan komputer, dan dapat beroperasi melalui jaringan *internet* atau *intranet*. Salah satu pendekatan yang mungkin adalah dengan menerapkan *mobile agent Java*.

IDS pada umumnya melakukan dua pekerjaan, yaitu pengumpulan data dan analisis data. Penggolongan IDS bisa dilakukan berdasar banyak karakteristik, diantaranya adalah [23]:

1. *host based – network based collection*
2. *direct – indirect monitoring*
3. *internal – external sensor*

Penerapan IDS telah mengalami peningkatan pesat pada tahun-tahun belakangan ini. Salah satu alasannya adalah perkembangan dari internet dan jumlah jaringan yang cukup

besar pada setiap organisasi. Peningkatan jumlah mesin pada jaringan ini memunculkan aktivitas yang tidak diinginkan, tidak hanya dari serangan luar, tetapi juga dari dalam seperti *disgruntled employes* dan orang yang menyalahgunakan *privelege* untuk keperluan pribadi.

1.3 Tujuan

Mobile agent adalah agen yang aktif dan dapat bergerak menuju ke komputer lain, atau menjelajahi jaringan untuk menjalankan tugasnya. Penelitian ini menghasilkan rancangan aplikasi mobile agent untuk membentuk suatu prototype sistem deteksi penyusupan terdistribusi (*Distributed Intrusion Detection System*) .

IDS yang ada , diharap bisa melakukan pekerjaannya mendekati realtime, tahan terhadap serangan, dan overhead yang cukup kecil pada host, baik untuk penggunaan memori dan CPU. Dalam report ini mencoba mendeskripsikan konsep dari penggunaan IDS berbasis mobile agent, karakteristik dan kemampuannya. Pendekatan ini diusahakan mampu menangani masalah skalabilitas, reliabilitas, dan konfigurabilitas.

Tujuannya mempelajari pendekatan ini dan kemampuan deteksinya. Dengan ini akan diperoleh kemampuan dan keterbatasan pendekatan berbasis agen saat diterapkan pada dunia nyata.

1.4 Batasan Masalah

Sebelumnya akan dijelaskan klasifikasi sumber data untuk deteksi penyusupan dan mekanisme pengaksesannya. IDS yang ada kebanyakan mempergunakan analisis data terpusat atau pengumpulan data per host dan komponen analisis yang diimplementasikan sebagai proses terpisah pada satu atau lebih komputer pada jaringan. Permasalahan skalabilitas, konfigurabilitas, dan efisiensi dari IDS umumnya berasal dari rancangannya yang berupa suatu entitas monolithic tunggal [3]. Pada arsitektur tersebut data dikumpulkandan diproses oleh host tunggal.

Dalam report ini diajukan suatu arsitektur terdistribusi yang mempergunakan entitas-entitas disebut agen otonom (atau disebut agen saja), untuk mendeteksi perilaku anomali atau penyalahgunaan. Rancangan ini diharapkan memiliki keunggulan dari arsitektur lainnya dalam skalabilitas, efisiensi, *fault tolerance* , dan konfigurabilitas.

Di sini mobile agent adalah program komputer yang bisa berjalan-jalan pada jaringan. Mobile agent akan melakukan pengambilan informasi pada suatu host tujuan, meliputi :

ketersediaan memori, ketersediaan space disk, informasi log dari login user, keaktifan port dan proses tertentu. Kemudian hasil pengambilan tersebut akan dianalisa berdasarkan konfigurasi kebijakan jaringan yang ditetapkan administrator (disini dengan memakai suatu file), sehingga akan didapat informasi penting yang perlu diperhatikan dari setiap host. Informasi tersebut akan berguna bagi administrator untuk memperhatikan kemungkinan adanya potensi penyusupan. Tidak semua kemungkinan serangan atau usaha penyusupan ditangani oleh perangkat lunak ini. Perangkat lunak ini bisa di implementasikan dengan bahasa pemrograman java, memanfaatkan tool untuk pengembangan software agents yaitu Aglets Software Development Kit, serta diharapkan bisa berjalan pada platform sistem operasi Windows dan Linux.

2. SISTEM DETEKSI PENYUSUPAN (IDS)

2.1 Keamanan Jaringan Komputer

Tujuan utama dari keamanan sistem adalah memberikan jalur yang aman antara entitas yang saling bertukar informasi dan untuk menyediakan perlindungan data [12]. Insiden keamanan jaringan komputer adalah suatu aktivitas yang berkaitan dengan jaringan komputer, dimana aktivitas tersebut memberikan implikasi terhadap keamanan. Secara garis besar insiden dapat diklasifikasikan menjadi [2,25]:

1. Probe/scan: Usaha-usaha yang tidak lazim untuk memperoleh akses ke dalam suatu sistem, atau untuk menemukan informasi tentang sistem tersebut. Kegiatan probe dalam jumlah besar dengan menggunakan tool secara otomatis biasa disebut *Scan*. Berbagai macam tool yang dipergunakan untuk keperluan ini seperti : *network mapper, port mapper network scanner, port scanner, atau vulnerability scanner*. Informasi yang diperoleh misalkan :
 - a. Topologi dari jaringan target
 - b. Tipe traffic yang melewati firewall
 - c. Hosts yang aktif
 - d. Sistem operasi pada host
 - e. Software yang berjalan pada server dan versinya.

2. *Account Compromisse* : Penggunaan account sebuah komputer secara ilegal oleh seseorang yang bukan pemilik account, dimana account tersebut tidak mempunyai privilege sebagai administrator sistem.
3. *Root Compromisse* : Mirip *account compromisse* tetapi mempunyai privilege sebagai administrator sistem.
4. *Packet sniffer* :Perangkat lunak/keras yang digunakan untuk memperoleh informasi yang melewati jaringan komputer, biasanya dengan NIC bermode promiscuous.
5. *Denial of service (DOS)* : Membuat sumberdaya jaringan maupun komputer tidak bekerja, sehingga tidak mampu memberikan layanan kepada user. Misalkan saja dengan membanjiri sumber daya komputer, misal CPU,memori,ruang disk, bandwidth jaringan. Serangan dapat dilakukan dari satu komputer atau beberapa komputer (*Distributed DOS*).
6. Eksploitasi perintah : Menyalahgunakan perintah yang bisa dieksekusi.
7. *Malicious code* : Program yang bila dieksekusi akan menyebabkan sesuatu yang tidak diinginkan didalam sistem. Misal trojan horse, virus dan worm.
8. *Penetration* : Pengubahan data, privilege, atau sumber daya. Beberapa jenisnya:
 - a. *User to Root* : User lokal pada suatu host memperoleh hak admin
 - b. *Remote to user* : Pengakses luar memperoleh account lokal di host target
 - c. *Remote to Root* : Pengakses luar memperoleh account admin di host target
 - d. *Remote to Disk Read* : Pengakses luar bisa membaca file di host target
 - e. *Remote Disk write* : Pengakses luar bisa menulis file di host target
9. *Privilege Escalation* : User Publik bisa memperoleh akses sebagai user lokal, yang nantinya bisa dilanjutkan ke hak akses sebagai admin.

2.2 Tujuan Penggunaan IDS

IDS merupakan software atau hardware yang melakukan otomatisasi proses monitoring kejadian yang muncul di sistem komputer atau jaringan, menganalisanya untuk menemukan permasalahan keamanan [2]. IDS adalah pemberi sinyal pertama jika seorang penyusup mencoba membobol sistem keamanan komputer kita. Secara umum penyusupan bisa berarti serangan atau ancaman terhadap keamanan dan integritas data,

serta tindakan atau percobaan untuk melewati sebuah sistem keamanan yang dilakukan oleh seseorang dari internet maupun dari dalam sistem.

IDS tidak dibuat untuk menggantikan fungsi firewall karena kegunaanya berbeda. Sebuah sistem firewall tidak bisa mengetahui apakah sebuah serangan sedang terjadi atau tidak. IDS mengetahuinya. Dengan meningkatnya jumlah serangan pada jaringan, IDS merupakan sesuatu yang diperlukan pada infrastruktur keamanan di kebanyakan organisasi.

Secara singkat, fungsi IDS adalah pemberi peringatan kepada administrator atas serangan yang terjadi pada sistem kita. Alasan mempergunakan IDS [2,3]:

1. Untuk mencegah resiko timbulnya masalah.
2. Untuk mendeteksi serangan dan pelanggaran keamanan lainnya yang tidak dicegah oleh perangkat keamanan lainnya. Biasanya penyusupan berlangsung dalam tahapan yang bisa diprediksi. Tahapan pertama adalah probing, atau eksploitasi pencarian titik masuk. Pada sistem tanpa IDS, penyusup memiliki kebebasan melakukannya dengan resiko kepergok lebih kecil. IDS yang mendapati *probing*, bisa melakukan blok akses, dan memberitahukan tenaga keamanan yang selanjutnya mengambil tindakan lebih lanjut.
3. Untuk mendeteksi usaha yang berkaitan dengan serangan misal *probing* dan aktivitas *dorknob rattling*
4. Untuk mendokumentasikan ancaman yang ada ke dalam suatu organisasi. IDS akan mampu menggolongkan ancaman baik dari dalam maupun dari luar organisasi. Sehingga membantu pembuatan keputusan untuk alokasi sumber daya keamanan jaringan.
5. Untuk bertindak sebagai pengendali kualitas pada administrasi dan perancangan keamanan, khususnya pada organisasi yang besar dan kompleks. Saat ini IDS dijalankan dalam waktu tertentu, pola dari pemakaian sistem dan masalah yang ditemui bisa nampak. Sehingga akan membantu pengelolaan keamanan dan memperbaiki kekurangan sebelum menyebabkan insiden.

6. Untuk memberikan informasi yang berguna mengenai penyusupan yang terjadi, peningkatan diagnosa, *recovery*, dan perbaikan dari faktor penyebab. Meski jika IDS tidak melakukan block serangan, tetapi masih bisa mengumpulkan informasi yang relevan mengenai serangan, sehingga membantu penanganan insiden dan *recovery*. Hal itu akan membantu konfigurasi atau kebijakan organisasi.

Meskipun vendor dan administrator berusaha meminimalkan vulnerabilitas yang memungkinkan serangan, ada banyak situasi yang tidak memungkinkan hal ini [2]:

1. Pada banyak sistem, tidak bisa dilakukan patch atau update sistem operasi
2. Administrator tidak memiliki waktu atau sumber daya yang mencukupi untuk melacak dan menginstall semua patch yang diperlukan. Umumnya ini terjadi dalam lingkungan yang terdiri dari sejumlah besar host dengan hardware dan software yang berbeda.
3. User mempergunakan layanan protokol yang merupakan sumber vulnerabilitas
4. Baik user maupun administrator bisa membuat kesalahan dalam melakukan konfigurasi dan penggunaan sistem
5. Terjadi disparitas policy yang memungkinkan user melakukan tindakan yang melebihi kewenangannya.

Salah satu tujuan dari pengelolaan keamanan komputer adalah mempengaruhi perilaku dari user dengan suatu jalan yang akan melindungi sistem informasi dari permasalahan keamanan. IDS membantu organisasi mencapai tujuan ini dengan meningkatkan kemampuan penemuan resiko.

2.3 Arsitektur IDS

Banyak model IDS bisa dinyatakan dalam tiga komponen mendasar [2] :

1. Sumber Informasi : network, host , application
2. Analisis :
 - a. *misuse detection* : mengamati aktivitas yang memiliki pola sesuai dengan suatu teknik penyusupan yang sudah diketahui (*signature*) atau vulnerabilitas sistem.

b. *Anomaly* : mengamati aktivitas yang berbeda dari perilaku user atau sistem yang normal

3. Response : Apakah melakukan aksi saat mendeteksi penyusupan (active/passive)

Komponen fungsional di dalam IDS pengaturannya satu sama lain bisa dilakukan secara [2]:

1. Host – Target Co location : IDS berjalan pada sistem yang dilindunginya. Mengurangi biaya tetapi rentan.

2. Host – Target Separation: IDS terpisah untuk kontrol dan analisa sistem. Ini meningkatkan keamanan karena menyembunyikan keberadaan IDS dari penyusup.

IDS umumnya berdasar pada arsitektur multi-tier dari[2]:

1. Teknologi deteksi, yang bergantung pada:

a. Sensor : biasanya disebut *engine/probe*, merupakan teknologi yang memungkinkan IDS untuk memantau sejumlah besar traffic.

b. Agents : Software yang di install pada suatu PC untuk memantau file atau fungsi tertentu. Dan melakukan pelaporan jika terjadi sesuatu.

c. Collector : seperti agent , tetapi lebih kecil, dan tidak membuat keputusan, tetapi hanya menyampaikan ke manager pusat.

2. Analisis data : Proses analisis data dan *data mining* sejumlah besar data dilakukan oleh lapisan, kadang diletakan pada pusat data/server.

3. Manajemen konfigurasi /GUI : Biasa disebut juga console merupakan antarmuka operator dengan IDS.

2.4 Pengumpulan Data IDS

Mekanisme pengumpulan data pada IDS bisa dilakukan secara[27]:

1. Langsung (*direct*) : pengukuran dilakukan pada obyek

2. Tidak langsung (*indirect*) : pengukuran dilakukan pada efek dari obyek (misal file log)

Berkaitan dengan tempatnya, pengumpulan data bisa dilakukan [10]:

1. *Host based* : akuisisi data dari sumber di host, seperti file log atau keadaan memori.

2. *Network based* : akuisisi data dari jaringan, misal menangkap paket yang lewat.

Monitoring secara langsung pada suatu host bisa dilakukan dengan[27]:

1. *External sensor* : suatu program yang mengamati komponen hardware atau software pada suatu host, dan diimplementasi terpisah dari komponen tersebut.
2. *Internal sensor* : suatu program yang mengamati komponen hardware atau software pada suatu hosts, dan diimplementasikan sebagai bagian dalam komponen tersebut

Pada bagian selanjutnya akan dijelaskan berkaitan dengan perbandingan *host based* dan *network based* IDS.

2.5 Network Based IDS (NIDS)

Network based IDS (NIDS) mempergunakan paket jaringan sebagai sumber data. Umumnya dengan adapter jaringan yang dijalankan dalam mode *promiscuous* untuk memonitor dan menganalisa semua traffic secara real time. Pengenalan serangan dilakukan dengan cara [10]:

1. Pencocokan pola, atau bytecode
2. Penyilangan frekuensi atau threshold
3. Korelasi kejadian
4. Deteksi Anomali statistik

Sekali serangan telah dideteksi modul IDS bisa memberikan respon pemberitahuan, peringatan (*alert*) dan tindakan pemutusan hubungan, pencatatan session untuk analisis forensik dan pengumpulan barang bukti.

Keunggulan NIDS [2,10]:

1. Biaya lebih rendah: Titik deteksi lebih sedikit dan memudahkan deployment dengan jangkauan lebih luas
2. Deteksi serangan yang dilewatkan oleh HIDS : misal, HIDS tidak terlihat header packet, serangan seperti denial Of Service dan TearDrop.
3. Penyusup lebih sulit menghilangkan barang bukti
4. Deteksi dan respon realtime
5. Deteksi penyusupan gagal
6. Tidak bergantung pada sistem operasi
7. Bisa dibuat lebih aman dari serangan bahkan tidak kelihatan oleh penyerang
8. Bila berkemampuan respon, lebih efisien karena bisa memblokir serangan

9. Bisa berinteraksi dengan teknologi perimeter lainnya misal router dan firewall untuk melakukan respon ancaman.

Kekurangan NIDS [2]:

1. Kesulitan untuk memproses semua paket pada jaringan yang besar atau dengan traffic yang sibuk
2. Jaringan dengan switch akan membatasi kemampuannya
3. tidak bisa menganalisa informasi terenkripsi
4. Tidak bisa menentukan suatu serangan sukses atau berhasil
5. Kesulitan menghadapi network-based attack yang mengikutkan paket terfragmentasi yang bisa menyebabkan IDS crash.

2.6 Host Based IDS (HIDS)

HIDS saat ini umumnya merupakan tool memonitor sistem, event dan log keamanan di Windows NT, dan syslog pada lingkungan Unix. Saat ada perubahan file tersebut maka IDS akan Membandingkan entry terbaru dengan tanda-tanda serangan (*attack signature*) . IDS ini melakukan deteksi pada aktivitas port dan memberitahukan pada administrator bila port tertentu di akses.

Keunggulan HIDS [10]:

1. Menentukan suatu penyusupan gagal atau berhasil
2. Memonitor aktivitas yang spesifik
3. Deteksi serangan yang dilewatkan oleh NIDS, misal serangan yang tidak melewati antar jaringan
4. cocok untuk lingkungan terenkripsi dan dengan switch
5. Respon mendekati real-time bila diimplementasikan secara tepat
6. Tidak memerlukan hardware tambahan

Kekurangan HIDS [2]:

1. Lebih sulit dikelola, karena harus dikonfigurasi di setiap host
2. Karena IDS bertempat pada host yang mendapatkan serangan, maka IDS didalam host bisa ikut down
3. Kurang bagus untuk melakukan scan pada keseluruhan jaringan
4. Kinerja pada host yang dimonitor bisa berkurang

2.7 Respon IDS

Respon yang diberikan oleh suatu IDS biasanya dikelompokkan dalam tiga kategori : pemberitahuan (*notification*), *storage*, dan *active response*. Contoh respon yang ada [10]:

Tabel 2.2 Respon IDS

	NIDS	HIDS
<i>Notification</i>	Alarm ke konsole	Alarm ke konsole
	E-mail	E-mail
	SNMP trap	SNMP trap
	Melihat session yang aktif	
<i>Storage</i>	Laporan log	Laporan log
	Data log mentah	
Aktif	Memutuskan koneksi (TCP reset)	Menghentikan login user
	Konfigurasi ulang firewall	Melakukan disable Account user
	Aksi terdefinisi oleh user	Aksi terdefinisi oleh user

2.8 Kemampuan analisis IDS

IDS adalah sistem yang mempunyai kemampuan untuk mengumpulkan informasi-informasi yang bisa digunakan untuk menganalisis kemungkinan serangan dari luar. Sebuah sistem IDS mampu mengetahui pola-pola tertentu yang bisa berarti serangan dari luar sedang terjadi, serta mampu melakukan monitoring terhadap aktivitas dalam sebuah jaringan.

Sebuah IDS dapat ditentukan jenisnya berdasarkan kemampuan analisis yang dimilikinya[2,8], yaitu *misuse detection* dan *anomaly detection* :

1. *Misuse Detection (Signature/Pattern Analysis)*

Jenis IDS ini menganalisa aktivitas sistem, mengamati event atau sekumpulan event yang cocok dengan suatu pola tertentu yang mendeskripsikan suatu serangan. Pola tersebut biasanya disebut *signature*, sehingga *misuse detection* sering disebut dengan “*signature-based detection*”. IDS ini mengetahui beberapa vulnerabilitas. Ia akan melakukan monitoring pada jaringan terhadap pola-pola yang diketahuinya.

Selain itu terdapat pula pendekatan yang lebih baik dengan mempergunakan analisa berbasis state untuk mendeteksi sekumpulan serangan.

Keuntungan :

- a. Cukup efektif mendeteksi serangan dan meminimalkan *false alarm*.
- b. Bisa secara cepat dan reliable mendiagnosa penggunaan teknik atau tool serangan tertentu. Ini akan membantu administrator menentukan prioritas perbaikan.
- c. Memungkinkan administrator melacak permasalahan keamanan di sistem, sesuai prosedur penanganan insiden.

Kekurangan :

- a. Hanya bisa mendeteksi serangan yang sudah diketahui, sehingga harus terusdi – update *signature*-nya.
- b. Kurang bisa mendeteksi serangan yang sudah dimodifikasi polanya (ini memang bisa diatasi oleh analisa berbasis state, tetapi masih kurang umum dipakai).
- c. Cenderung tergantung pada suatu lingkungan tertentu.

2. *Anomaly Detection*

IDS dengan kemampuan ini bekerja berdasarkan sebuah acuan, bagaimana sebuah sistem dikatakan normal. Jika IDS beranggapan bahwa aktivitas sistem sudah tidak normal (perilaku anomali), maka IDS akan mengambil keputusan serangan sedang terjadi. Mereka bekerja dengan asumsi serangan adalah sesuatu yang berbeda dari hal “normal” (legitimate) sehingga bisa dideteksi oleh sistem yang mengidentifikasi perbedaan ini.

Detektor anomalai membuat suatu profile yang merepresentasikan perilaku normal user, host, atau koneksi jaringan. Profil ini dibuat dari data historis yang dikumpulkan selama suatu periode operasi normal. Detektor kemudian akan

mengumpulkan data kejadian dan mempergunakan sejumlah pengukuran untuk menentukan bilamana aktivitas yang dimonitor menyimpang dari normal.

Pengukuran dan teknik yang dipergunakan dalam deteksi anomali termasuk :

- a. *Threshold detection*, di mana sejumlah atribut tertentu dari user dan perilaku sistem dinyatakan dalam jumlah, dengan beberapa tingkatan yang di ijinakan. Misalkan jumlah file yang diakses oleh user dalam suatu periode, jumlah usaha login yang gagal ke dalam sistem, atau jumlah proses pada CPU. Tingkatan ini bisa ditentukan secara statis atau heuristik.
- b. Pengukuran statistik, baik parametrik maupun non-parametrik
- c. Pengukuran rule-based, mirip dengan statistik non-parametrik, tapi berbeda penggunaan *pattern* yang dinyatakan sebagai aturan bukan kuantitas.
- d. Pengukuran lainnya, termasuk jaringan syaraf tiruan atau algoritma genetik.

Saat ini IDS komersial hanya mempergunakan no 1 dan 2 saja. Sayangnya IDS dengan anomali sering menghasilkan banyak false alarm meski pada pola normal user dan perilaku sistem. Sejumlah IDS komersial menggunakannya dalam deteksi *network scanning*.

Keuntungan:

- a. Bisa diketahui pola serangan yang belum pernah ditemukan sebelumnya, tanpa administrator harus mengetahui bagaimana pola sebuah sistem yang normal dan bagaimana yang tidak.
- b. Menghasilkan informasi yang bisa dipergunakan untuk menentukan signature untuk misuse detector.

Kekurangan:

- a. Tingginya kesalahan saat pengambilan keputusan karena perilaku tak terprediksi
- b. Biasanya memerlukan training set tambahan pada rekaman kejadian untuk menggolongkan pola normal.

2.9 Karakteristik IDS

Berikut adalah beberapa kriteria yang diinginkan untuk suatu IDS yang ideal[3,27]:

1. Meminimalkan overhead sistem untuk tidak mengganggu operasi normal.
2. Mudah dikonfigurasi untuk disesuaikan dengan kebijakan keamanan sistem
3. Mudah diinstalasi (deploy)
4. Mudah beradaptasi dengan perubahan sistem dan perilaku user, misal aplikasi atau resource baru.
5. Mampu memonitor sejumlah host dengan tetap memberikan hasil yang cepat dan tepat.
6. Dampak negatif yang minimal
7. Memungkinkan konfigurasi dinamis, khususnya bila pemantauan dilakukan pada sejumlah besar host.
8. Berjalan secara kontinu dengan supervisi minimal dari manusia
9. Mampu mendeteksi serangan:
 - a. Tidak salah menandai aktivitas yang legitimate (*false positive*)
 - b. Tidak gagal mendeteksi serangan sesungguhnya (*false negative*)
 - c. Segera melakukan pelaporan penyusupan yang terjadi.
 - d. Cukup *general* untuk berbagai tipe serangan
10. Mampu fault tolerant dalam arti:
 - a. Bisa melakukan recover dari sistem yang crash baik secara insidental atau karena aktivitas tertentu.
 - b. Setelah itu bisa melanjutkan state sebelumnya tanpa mempengaruhi operasinya.
11. Mampu menolak usaha perubahan:
 - a. Adanya kesulitan yang tinggi bila penyerang mencoba memodifikasinya.
 - b. Mampu memonitor dirinya sendiri dan mendeteksi bila dirinya telah dirubah oleh penyerang.

Kebanyakan IDS memiliki permasalahan sebagai berikut[3.23]:

1. Tingkat sentralisasi. Kebanyakan deteksi dilakukan secara terpusat.

2. Konsumsi sumberdaya. Karena sentralisasi tersebut maka terjadi kebutuhan sumberdaya pemrosesan yang besar.
3. Batasan skalabilitas
4. Masalah keamanan, misalnya single point failure.
5. Kesulitan untuk melakukan konfigurasi ulang atau penambahan kemampuan

2.10 Pemilihan IDS

IDS paling baik diimplementasikan dengan mengkombinasikan penggunaan solusi berbasis host dan network. Tahapan evaluasi umumnya terdiri dari tiga fase [8]:

1. Fase 1: Penentuan kebutuhan IDS: untuk mencakup aset penting dan kelengkapan dengan kebijakan keamanan. Tingkatan keamanan ini bisa mencakup perlindungan perimeter, aplikasi, , e-business, server kunci, kebijakan dan perlindungan hukum. Hal ini harus diurutkan sesuai prioritas.
2. Fase 2: Evaluasi solusi IDS:
 - a. Memilih produk yang tepat untuk memenuhi kebutuhan
 - b. Pemahaman bagaimana tiap IDS mendeteksi penyusupan dalam jaringan
 - c. Pemahaman bagaimana tiap produk menempatkan prioritas dan menjelaskan penyusupan pada jaringan, termasuk false positif yang dihasilkan.
 - d. Pemahaman kemampuan pelaporan dari tiap produk, kelengkapan, fleksibilitas dan penerapan teknisnya.
3. Fase 3: Deployment IDS : penempatan solusi dalam organisasi secara efektif

Fleksibilitas IDS sendiri bisa didasarkan pada [2]:

1. Kustomisasi : adaptasi IDS pada kebijakan tertentu dari organisasi
2. Deployment: penempatan pada jaringan yang heterogen
3. Skalabilitas manajemen

3. AGEN dan AGLETS

3.1 Agen

Software Agent (selanjutnya di sebut Agen saja) adalah entitas perangkat lunak yang didedikasikan untuk tujuan tertentu [2,1]. Agen bisa memiliki ide sendiri mengenai

bagaimana menyelesaikan suatu pekerjaan tertentu atau agenda tersendiri. Karakteristik dari Agen[15]:

1. *Autonomy*: Komputer umumnya hanya berespon pada manipulasi langsung. Kontras dengan agen perangkat lunak yang mengamati lingkungannya dan bisa melakukan tindakan otonom.
2. Kontinuitas temporal: *Software Agents* adalah suatu program yang mana user menentukan suatu tujuan atau task. Disini idenya sekali task atau tujuan tertentu telah didelegasikan, maka terserah pada agen untuk melakukan pencapaian tujuan.
3. Reaktif: Suatu *software agent* berespon dalam waktu yang bermacam-macam untuk merubah dalam lingkungannya
4. *Goal driven*: Suatu agen bisa menerima *request* tingkat tinggi yang menentukan tujuan dari user manusia (atau agen lainnya) dan memutuskan dimana dan bagaimana untuk memuaskan *request* tersebut.

Definisi agen[21] yang lebih rinci, ditinjau dari sudut pandang sistem, adalah obyek perangkat lunak yang:

1. Diletakan dalam lingkungan eksekusi
2. Memiliki sifat sebagai berikut :
 - a. Reaktif, dapat merasakan perubahan dalam lingkungannya dan bertindak sesuai perubahan tersebut.
 - b. *Autonomous*, mampu mengendalikan tindakannya sendiri
 - c. Proaktif, mempunyai dorongan untuk mencapai tujuan
 - d. Bekerja terus menerus sampai waktu tertentu
3. Dapat mempunyai sifat ortogonal sebagai berikut :
 - a. Komunikatif, dapat berkomunikasi dengan agen yang lain.
 - b. *Mobile* , dapat berpindah dari satu host ke host yang lain
 - c. *Learning*, mampu menyesuaikan diri berdasarkan pengalaman sebelumnya
 - d. Dapat dipercaya sehingga menimbulkan kepercayaan kepada *end user*.

Agen yang tidak berpindah ke host lain disebut *stationary agent*.

3.2 Mobile Agent

Mobile agent adalah agen yang aktif dan dapat bergerak menuju komputer lain, atau menjelajahi jaringan untuk menjalankan tugasnya [15]. *Mobile agent* sering digunakan untuk mengumpulkan data, informasi atau suatu perubahan.

Mobile agent tidak terikat pada sistem dimana ia mulai dieksekusi. *Mobile agent* mempunyai kemampuan unik untuk memindahkan dirinya sendiri dari satu sistem ke sistem yang lain dalam suatu jaringan. Kemampuan untuk berkeliling memungkinkan *Mobile agent* untuk berpindah ke sistem yang mengandung obyek yang akan berinteraksi dengan agen dan kemudian mengambil manfaat selama berada dalam *host* dan jaringan yang sama dengan obyek [12].

Secara praktis dapat dikatakan bahwa *Mobile agent* adalah sebuah program yang dapat menghentikan eksekusi, berjalan melalui jaringan dengan membawa kode dan *state*-nya, dan kemudian melanjutkan eksekusinya pada *host* yang lain [21]. Aglets merupakan contoh perangkat lunak yang memungkinkan pengembangan dan penerapan mobile agents ini [15].

Beberapa penerapan dari *Mobile agent*: Pengumpulan data, pencarian dan penyaringan, pemantauan asinkron, dan pemrosesan paralel.

3.3 Keuntungan *Mobile agent*

Ada beberapa keuntungan yang dapat dari penerapan *Mobile agent* bila dibandingkan dengan teknologi agen yang lain, misalnya RMI. Meskipun kenyataannya hampir semua masalah dalam komputasi terdistribusi dapat diselesaikantapa menggunakan *Mobile agent*, namun dengan menerapkan *Mobile agent* akan dapat memudahkan pengembangan aplikasi dan dapat meningkatkan keandalan dan efisiensi. Sedikitnya ada beberapa keuntungan yang didapatkan dari *Mobile agent* [12,14,15,16]:

1. Mengurangi beban jaringan: Sistem terdistribusi sangat tergantung pada protokol komunikasi yang melibatkan banyak interaksi untuk menyelesaikan tugas yang diberikan. Akibatnya, traffic jaringan tinggi. Sebaliknya, *Mobile agent* memungkinkan untuk mengemas suatu aplikasi, mengirimkannya ke *host* tujuan dan kemudian interaksi dapat terjadi secara lokal. *Mobile agent* juga bermanfaat untuk mengurangi aliran data pada jaringan. Jika terdapat data dengan jumlah yang sangat besar

tersimpan di *remote host*, maka lebih baik memproses data tersebut secara lokal ditempatnya, dari pada mengirimkannya melalui jaringan. Moto *Mobile agent* adalah: lebih baik memindahkan komputasi ke tempat data, dari pada memindahkan data ke tempat komputasi.

2. Efisiensi sumber daya: Konsumsi sumber daya (CPU dan memori) dapat dihemat, sebab *Mobile agent* menetap dan bekerja hanya pada satu *node* pada satu waktu. *Node* yang lain tidak menjalankan agen sampai *node* tersebut memerlukannya.
3. Menanggulangi *latency* jaringan: Sistem *real-time* yang kritis perlu tanggap terhadap perubahan lingkungannya secara *real-time*. Keterlambatan tanggapan yang diakibatkan oleh masalah jaringan harus dihindari. *Mobile agent* menawarkan suatu pemecahan dengan mengirimkan agen ke tujuan dan dieksekusi secara lokal.
4. Encapsulate protocol : Ketika terjadi pertukaran data pada sistem terdistribusi, setiap *host* memiliki kode sebagai implementasi dari protokol yang diperlukan untuk mengkode data yang keluar dan menerjemahkan data yang masuk secara tepat. Untuk menjamin kompatibilitas, kode-kode protokol tersebut sudah dibakukan. Tetapi untuk mengakomodasi keperluan efisiensi dan keamanan yang lebih baru, tidak praktis bila harus meng-upgrade kode protokol tersebut, meskipun hal tersebut memungkinkan. *Mobile agent* dapat memecahkan masalah tersebut, karena *Mobile agent* dapat dikirim ke *remote host* dengan tujuan untuk membentuk “*channel*” yang berdasarkan protokol yang sudah baku tersebut.
5. Eksekusi secara *asynchronous* dan *autonomous*: Perangkat *mobile* sering kali harus menggunakan koneksi jaringan yang mahal dan mudah putus. Pekerjaan dapat digabungkan ke dalam *Mobile agent* yang kemudian dapat dikirimkan ke tujuan. Setelah dikirimkan dan sampai ke tujuan, koneksi antara perangkat *mobile* dan jaringan dapat diputuskan. *Mobile agent* menjadi berdiri sendiri dan dapat beroperasi secara *asynchronous* dan *autonomous*. Kemudian, perangkat *Mobile* dapat dikoneksikan ke

jaringan untuk mengambil dan mengumpulkan kembali agen yang telah dikirimkan. Dengan cara *disconnected operation* ini *agent dispatched* tidak memerlukan sistem asal untuk tetap aktif sementara mereka dieksekusi. Mereka juga bisa menunggu (*work while you sleep*)

6. Beradaptasi secara dinamis: *Mobile agent* mempunyai kemampuan untuk mengetahui perubahan di lingkungan eksekusinya dan dapat bereaksi secara *autonomous* melakukan perubahan. *Multiple Mobile agent* mempunyai kemampuan yang khas untuk berdistribusi sendiri di antara host-host dalam jaringan sedemikian rupa untuk memelihara konfigurasi yang optimal untuk penyelesaian masalah khusus.
7. Andal dan toleran terhadap kesalahan : Kemampuan *mobile agent* untuk berinteraksi secara dinamis pada situasi dan keadaan yang tak menguntungkan menjadikan *Mobile agent* mudah untuk membuat sistem terdistribusi yang andal dan toleran terhadap kesalahan. Jika sebuah *host* akan di-*shutdown* , semua agen yang sedang eksekusi di mesin tersebut akan diberi peringatan dan diberikan waktu untuk berpindah dan melanjutkan operasinya di *host* yang lain dalam jaringan tersebut.
8. Mendukung lingkungan yang heterogen: Komputasi jaringan pada dasarnya sangat heterogen. Baik pada sisi perangkat keras maupun perangkat lunak. *Mobile agent* tidak tergantung pada komputer dan jaringan, tetapi hanya bergantung pada lingkungan eksekusinya. Sebagai contoh, *Mobile agent* java dapat ditujukan ke segala sistem yang mempunyai JVM (Java Virtual Machine).
9. Real Time Notification: Agen yang ditempatkan pada remote site bisa memberitahukan perubahan isi secara cepat (misal upgrade software, pasar saham). Tanpa harus diminta oleh user. Ini juga terkait dengan hasil pencarian yang up to date, meningkatkan kemampuan penyimpanan data terpusat yang telah ada.
10. Ekeekusi paralel: Komputasi masif bisa dibagi ke dalam sejumlah agen, dilakukan dispatch ke node yang paling sesuai untuk eksekusi dari tiap

komponen, dan merakitnya di home. Sumber daya hardware bukan lagi batasan.

11. Paradigma Komputasi yang adaptif: *Mobile agent* bisa di-retract,dispatch, clone atau deactivate sesuai dengan perubahan kondisi.
12. Skalabilitas: Bila jumlah elemen komputasi di jaringan meningkat, agen baru bisa dikirim atau clone dan dispatch ke mesin baru di jaringan.

3.4 Aglets

IBM's Aglets Workbench(Aglets) dikembangkan oleh Danny B. Langedan Mitsuru Oshima dari IBM tokyo Research laboratory pada 1996. Aglets adalah obyek java yang dapat bergerak dari satu host ke host lain dalam suatu jaringan [13]. Aglet yang sedang bekerja disuatu host dapat menghentikan eksekusinya, pergi ke host yang lain dan kemudian memulai eksekusinya kembali[18]. Ketika aglet bergerak , aglet membawa kode program dan juga state dari semua obyek yang membawanya. Sebuah mekanisme keamanan yang built-in akan mengamankan host dari untrusted aglet.

Istilah aglet sesungguhnya adalah kombinasi dari kata agen dan applet. Perbedaan dengan applet adalah aglet juga membawa serta state, dan memiliki itinerary (rencana perjalanan)

ASDK(Aglets Software Development Kit) adalah paket perangkat lunak yang digunakan untuk menulis aplikasi mobile agent. ASDK menggunakan bahasa java dan bisa didapatkan secara gratis dari internet. Karena kemampuannya membuat lightweight agent , maka aglet API ini sering disebut sebagai RISC mobile agent.

Tahiti adalah suatu program aplikasi yang bekerja sebagai server aglet. Aplikasi tahiti merupakan suatu paket dengan ASDK. Beberapa server (tahiti) dapat dijalankan dalam satu komputer dengan cara memberikan nomor port yang berbeda. Tahiti menyediakan user interface untuk pemantauan, penciptaan, pengiriman, dan pemusnahan suatu aglet serta untuk menetapkan hak akses (privelege access) untuk server agent.

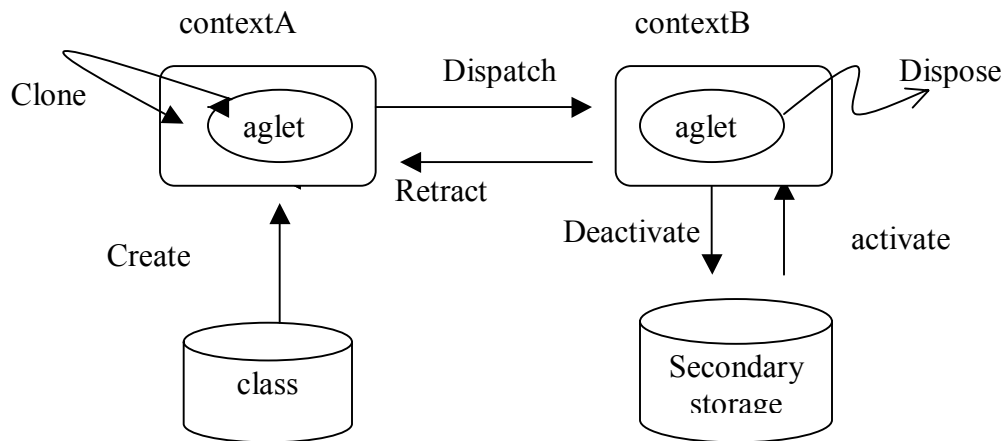
Karena aglet memungkinkan otomatisasi banyak proses yang sebelumnya dilakukan secara manual oleh user, teknologi ini bisa mentransformasi cara user berinteraksi dengan internet. Struktur yang mendasari aglets sebagai berikut:

1. Aglet: suatu obyek java yang mobile dan dapat berkunjung ke host lainnya, mampu berespon terhadap pesan.

2. Proxy: representasi dari aglet yang melindunginya dari akses langsung ke method publik. Menyediakan transparansi lokasi.
3. Context: aglets workplace, stationary object yang memungkinkan pengelolaan dan pengaturan eksekusi aglets.
4. Message: pertukaran objek antar aglets.
5. Future Reply: pengiriman message asinkron
6. Identifier: melekat pada tiap aglet dan unik secara global.

3.5 Siklus Hidup Aglets

Setelah diciptakan, aglet dapat melakukan operasi-operasi dasar seperti yang dijelaskan dibawah ini. Siklus hidup suatu aglet seperti digambarkan pada gambar 3.1:



Gambar 3.1 Model siklus hidup aglet [21]

1. *Creation*: penciptaan sebuah *aglet*. *Creation* terjadi di dalam *context*. *Aglet* yang baru diberi sebuah *identifier*, dimasukkan ke dalam *context* dan diinisialisasi. *Aglet* mulai eksekusi segera setelah inisialisasi sukses.
2. *Cloning*: proses penggandaan sebuah *aglet*. *Cloning* menghasilkan turunan (*copy*) yang hampir identik dengan *aglet* yang asli didalam *context* yang sama. Perbedaannya hanya terletak pada *identifier* yang diberikan dan eksekusi *aglet* baru hasil cloning dimulai dari awal (*restart*). Catatan bahwa thread eksekusi tidak di *-clone*.

3. *Dispatching*: pemindahan sebuah *aglet* dari satu *context* ke *context* yang lain. *Dispatching* akan memindahkan *aglet* dari *context* yang sedang berlangsung, masuk ke *context* tujuan dan kemudian memulai awal eksekusinya.
4. *Retraction*: proses untuk “menarik” *aglet* dari *context* yang sedang berlangsung dan masuk ke *context* yang melakukan permintaan *retraction*.
5. *Activation*: kemampuan untuk mengembalikan *aglet* ke dalam *context*.
6. *Deactivation*: kemampuan untuk menghentikan sementara jalannya eksekusi *aglet* dan menyimpan state *aglet* dalam penyimpanan sekunder.
7. *Disposal*: proses untuk menghentikan jalannya eksekusi *aglet* yang sedang berlangsung dan mengeluarkan *aglet* dari *context* yang sedang berlangsung.
8. *Messaging* : antar *aglet* meliputi pengiriman, penerimaan dan penanganan message baik *synchronous* maupun *asynchronous*.

3.6 Model Event Aglet

Model pemrograman *aglet* berdasar pada event. Model ini memungkinkan pemrogram untuk menempatkan listener yang customized ke dalam *aglet*. Listener ini akan menangkap event tertentu dalam siklus hidup *aglet*, lalu akan menjalankan suatu method. Sejumlah event dan method yang berkaitan dengannya pada *aglets*, sebagai berikut:

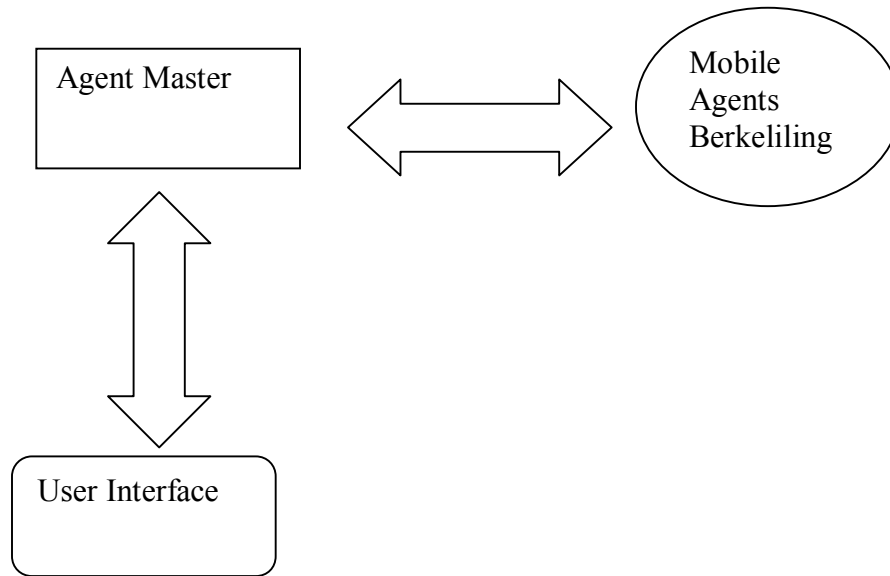
Event	Method	
	Sebelum Event terjadi	Setelah Event terjadi
Creation		onCreation ()
Cloning	OnCloning ()	onClone ()
Dispatching	OnDispatching ()	OnArrival ()
Retraction	onReverting ()	OnArrival ()
Disposal	onDisposing ()	
Deactivation	onDeactivating ()	
Activation		onActivation ()

3.7 Relasi Mobile Agent dan Sistem Deteksi penyusupan

Mengimplementasikan sistem deteksi penyusupan mempergunakan mobile agent adalah salahsatu paradigma baru untuk mendeteksi penyusupan. Penggunaan arsitektur mobile agent untuk menyediakan kemampuan keamanan masih relatif sedikit. Beberapa riset yang pernah dilakukan untuk penerapan tersebut pada beberapa lab: *Autonomous Agents for Intrusion Detection (AAFID)* di purdue University, *Hummingbird* di University of Idaho, *Java Agents for Meta-Learning (JAM)* di Columbia University. *Intrusion Detection Agent (IDA)* dari information Technology Promotion Agency (IPA) di jepang.

Mobile agent memiliki sejumlah karakteristik yang memungkinkan untuk meningkatkan kemampuan deteksi pemyusupan. Mobility adalah salah satunya. Mobile agent memiliki sifat otonom, kolaboratif, *self-organizing dan mobile*. Feature ini tidak terdapat pada program tradisional, dan memungkinkan deteksi penyusupan diimplementasikan dengan pendekatan baru untuk melakukan deteksi penyusupan. *Multipoint detection* bisa dilakukan pada sejumlah lokasi, misalnya untuk serangan terdistribusi atau pada jaringan, sehingga berfungsi seperti *mobile sensor*.

Karena feature yang dimilikinya, mobile agent merupakan suatu jawaban yang menjanjikan untuk keamanan dan komunikasi sistem. Mereka bisa dikembangkan dan diluncurkan dari suatu lokasi , misalkan saja perusahaan yang khusus menangani otomatisasi deteksi penyusupan. Mereka bisa dikirim ke sejumlah site, memantau sistem dan memberitahukan peringatan kepada user. Dalam beberapa kasus bisa mengambil tindakan koerksi, misal dengan menginstall perangkat lunak IDS yang relevan. Gambaran sederhana dari komponen-komponennya bisa dilihat pada gambar 3.2



Gambar 3.2 Gambaran sederhana Sistem

Diharapkan mobile agent ini :

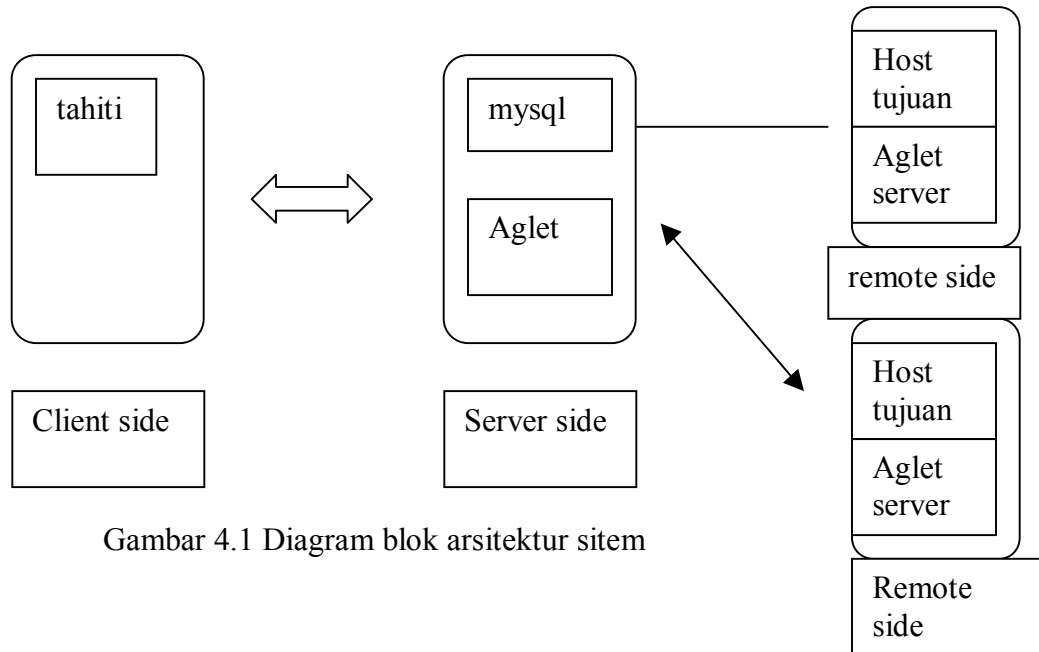
1. Cukup adaptif pada perubahan topologi dan konfigurasi jaringan karena penambahan atau pengurangan elemen komputasi pada jaringan
2. Memberikan informasi yang cukup untuk perbaikan dan penentuan kerusakan akibat penyusupan.
3. Kemampuan deteksi pada penyusupan terdistribusi
4. Konsumsi sumber daya yang rendah.

4. DESAIN dan IMPLEMENTASI

4.1 Proses Perancangan Sistem

Proses desain sistem deteksi penyusupan jaringan komputer ini ditempuh dalam beberapa tahap. Tahap pertama adalah menentukan pertimbangan sistem deteksi penyusupan jaringan komputer yang akan di buat. Tahap kedua adalah menentukan sumber data dan informasi apa saja yang akan dikumpulkan, dianalisa, dan kemudian dalam bentuk apa hasilnya akan ditampilkan. Tahap ketiga adalah menentukan syarat batas minimal yang perlu dipenuhi agar proses deteksi penyusupan berjalan sukses. Tahap keempat adalah mendesain skema deteksi penyusupan atau urutan langkah-langkah yang akan ditempuh dalam proses deteksi penyusupan suatu jaringan komputer.

Secara blok diagram, perancangan sistem terdiri dari 3 blok utama, yaitu client side, server side, dan remote site. Client side berupa Tahiti sebagai antarmuka dengan user. Server side terdiri dari aglet server dan database server di host asal. Sedang remote site berupa aglet server pada host tujuan yang menjadi tujuan deteksi penyusupan serta informasi kondisi host tersebut yang akan diambil.



Gambar 4.1 Diagram blok arsitektur sitem

4.2 Pertimbangan Sistem

Mobile agent merupakan teknologi yang menjanjikan untuk implementasi tool yang beroperasi pada sumber data yang terdistribusi. Misalkan saja administrator ingin melihat kondisi di setiap host, maka mobile agent yang akan dikirimkan ke setiap host yang akan melakukannya.

Di sini IDS dirancang sebagai external sensor, dengan pengumpulan data dari sejumlah host (multi host based), dan sumber data diambil baik secara langsung maupun tidak. Kelebihan yang diinginkan :

1. Keperluan instalasi yang minimal, karena cukup menginstall Java Virtual Machine(JRE/JDK) dan server aglet saja, selanjutnya perangkat lunak tersebut bisa juga dimanfaatkan untuk operasional agen bagi keperluan lainnya.
2. Lebih sederhana dan lebih mudah dipelihara daripada harus melakukan setting konfigurasi di tiap-tiap komputer pada jaringan, lebih-lebih saat menghadapi timbulnya jenis penyusupan baru atau memperbarui sistem.

3. Respon lebih cepat untuk mendeteksi penyusupan terdistribusi pada beberapa host
4. Akuisisi data dari sejumlah host (multi-host based)
5. Mampu menangani perubahan skalabilitas jaringan dan kesibukan traffic
6. Tetap memberikan laporan meski host mendapat serangan.

Mobile agent disini merupakan HIDS yang terdistribusi (bisa dianggap sebagai multi HIDS) dilakukan karena:

1. Kebanyakan penyusupan bisa dideteksi pada host, misal eksekusi perintah, akses ke service, dan lain-lain. Serangan akan berakhir pada host meski melalui jaringan, misalkan saja *flooding* jaringan juga akan terdeteksi di host.
2. Memungkinkan pengumpulan data yang merefleksikan secara akurat apa yang terjadi, ketimbang menebak paket yang lewat jaringan
3. Dalam traffic jaringan yang tinggi, paket bisa terlewat oleh network monitor (NIDS)
4. Bisa menentukan tingkatan dan jenis aktivitas tertentu secara spesifik yang ingin dimonitor

Direct monitoring dilakukan karena:

1. Sumber data tidak langsung, potensial telah diubah oleh penyusup
2. Tidak semua event terekam
3. Sumber data tidak langsung menyebabkan volume data yang besar, sehingga pemrosesannya membutuhkan lebih banyak waktu dan sumber daya
4. Direct monitoring hanya mengambil data yang diperlukan saja
5. Indirect monitoring mengalami permasalahan skalabilitas

4.3 Sumber data

Pada dasarnya mobile agent melakukan pengumpulan kondisi pada host di jaringan. Sumber data diambil dengan menggunakan fungsi-fungsi sistem dan jaringan yang disediakan oleh Java. Serta perintah yang disediakan oleh sistem operasi windows atau Linux:

1. ping
2. dir
3. free

4. df
5. du
6. lastlog
7. uptime
8. w

sejumlah informasi yang diambil:

1. Nama host
2. Sistem operasi
3. Ketersediaan space disk
4. Ketersediaan memori JVM
5. port tertentu yang aktif

Untuk host yang menjalankan sistem operasi Linux, akan diambil juga informasi berikut:

1. Ketersediaan memori
2. Proses tertentu yang aktif
3. Waktu uptime
4. Ukuran direktory temporary
5. Mesin asal user melakukan login

4.4 Informasi Deteksi penyusupan

sejumlah kegiatan yang akan berusaha dideteksi dan dicurigai sebagai penyusupan bergantung pada kebijakan jaringan yang ditentukan administrator:

1. N user berbeda yang melakukan login ke satu host berasal dari satu mesin
2. Login berasal dari satu mesin ke N host berbeda (kemungkinan penyusupan terdistribusi)
3. Serangan terhadap memori
4. Serangan terhadap space disk
5. Serangan terhadap direktory temporary (/tmp)
6. Mendeteksi port aktif dari trojan yang sudah diketahui atau mendeteksi port aneh (weird connection)
7. Mendeteksi proses tertentu yang tidak diperkenankan/ dicurigai
8. Kemungkinan mesin pernah mengalami rebooting, berdasar waktu uptime

Konfigurasi aturan untuk penentuan nilai parameter tersebut akan disimpan dalam sebuah file yang bisa diedit oleh administrator secara manual.

Informasi deteksi penyusupan yang dikumpulkan diambil dari host-host yang sedang aktif pada saat proses pengumpulan sedang berlangsung. Informasi tersebut adalah:

1. Host-host yang aktif dan kondisinya
2. Event yang terjadi dan perlu mendapat perhatian

Informasi ini akan ditampilkan, dan juga disimpan ke database. Aplikasi ini menggunakan Mysql sebagai database server untuk menyimpan informasi hasil deteksi penyusupan. Driver MM.MySQL dipergunakan sebagai JDBC (JAVA Database Connectivity) untuk berhubungan ke program java. Struktur tabel untuk menampung informasi, sebagai berikut :

1. no (int(11) , PRIMARY KEY, NOT NULL , auto_increment): untuk primary key tabel.
2. pengambilan (varchar (50)): saat pengambilan informasi deteksi dilakukan
3. laporan (text): berisi laporan hasil deteksi

Laporan hasil deteksi akan memuat informasi penting yang perlu diperhatikan, serta informasi keseluruhan yang diambil dari host tujuan.

4.5 Penentuan batasan Sistem

Sebelum melangkah ke proses desain selanjutnya, perlu ditentukan dahulu beberapa syarat batas minimal yang harus dipenuhi agar proses deteksi penyusupan berjalan dengan sukses.

Deteksi penyusupan dilakukan dengan mengirimkan agen melalui jaringan dari satu host ke host tujuan yang berada dalam jaringan komputer. Agar proses pengiriman agen tidak menghabiskan banyak waktu dan bandwidth jaringan, agen tersebut harus berukuran cukup kecil, hanya beberapa puluh kilobyte saja. Bahasa pemrograman yang cocok untuk keperluan tersebut adalah Java karena hasil kompilasi Java cukup kecil dan Java sangat handal untuk aplikasi jaringan.

Syarat host yang akan menjadi tujuan pengiriman mobile agent adalah :

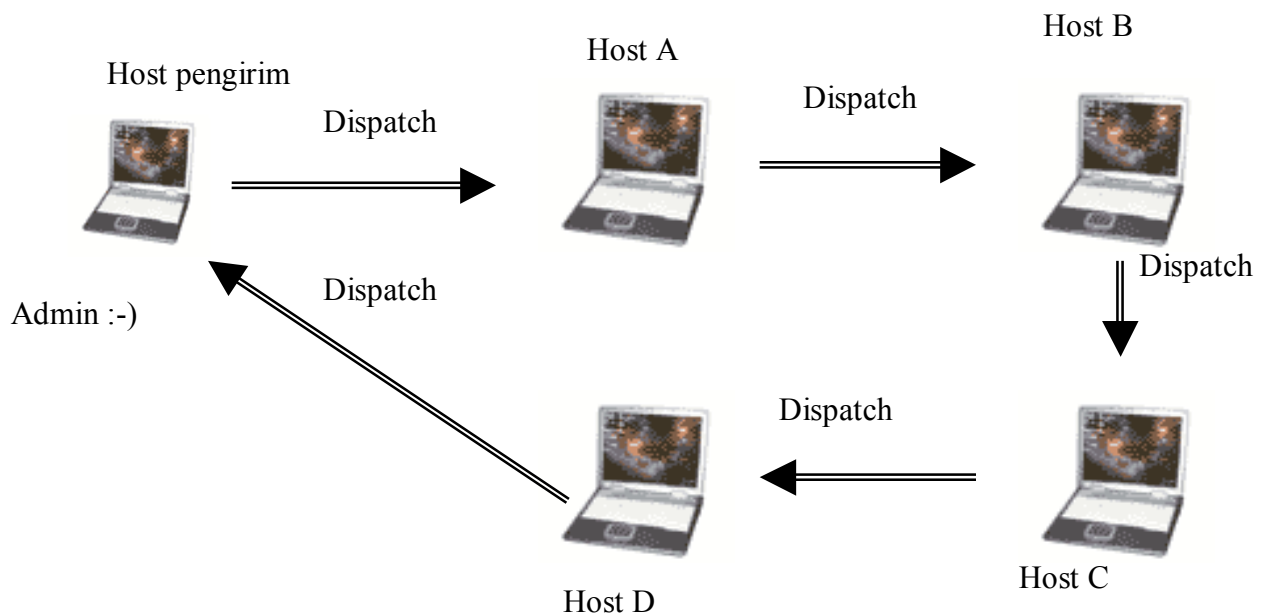
1. Telah di kenal alamat IP-nya dan alamat IP tersebut dapat diakses melalui jaringan

2. Telah dipasang JDK (Java Development Kit) versi 1.1.8 serta aglet viewer tahiti berjalan dengan nomor port yang telah ditentukan
3. Sistem operasi menggunakan NT/2000/XP atau Linux

Untuk syarat terakhir, yaitu mengenai sistem operasi yang digunakan oleh host yang akan menjadi tujuan pengiriman mobile agent, perlu sedikit modifikasi pada salah satu class dari mobile agent bila menggunakan sistem operasi selain tersebut di atas.

4.6 Skema Dasar Proses Deteksi Penyusupan

Urutan langkah deteksi penyusupan suatu jaringan komputer ditunjukkan dengan gambar skema deteksi penyusupan seperti pada gambar 4.2. di bawah ini akan dijelaskan urutan langkah proses deteksi penyusupan dari suatu jaringan komputer tersebut. Dengan asumsi setiap host sudah dijalankan aglet server yang siap menerima aglet yang masuk pada port yang sudah ditentukan. Sementara tahiti dijalankan pada komputer asal dimana administrator akan melihat hasil kerja agen tersebut.



Gambar 4.2 Skema dasar deteksi

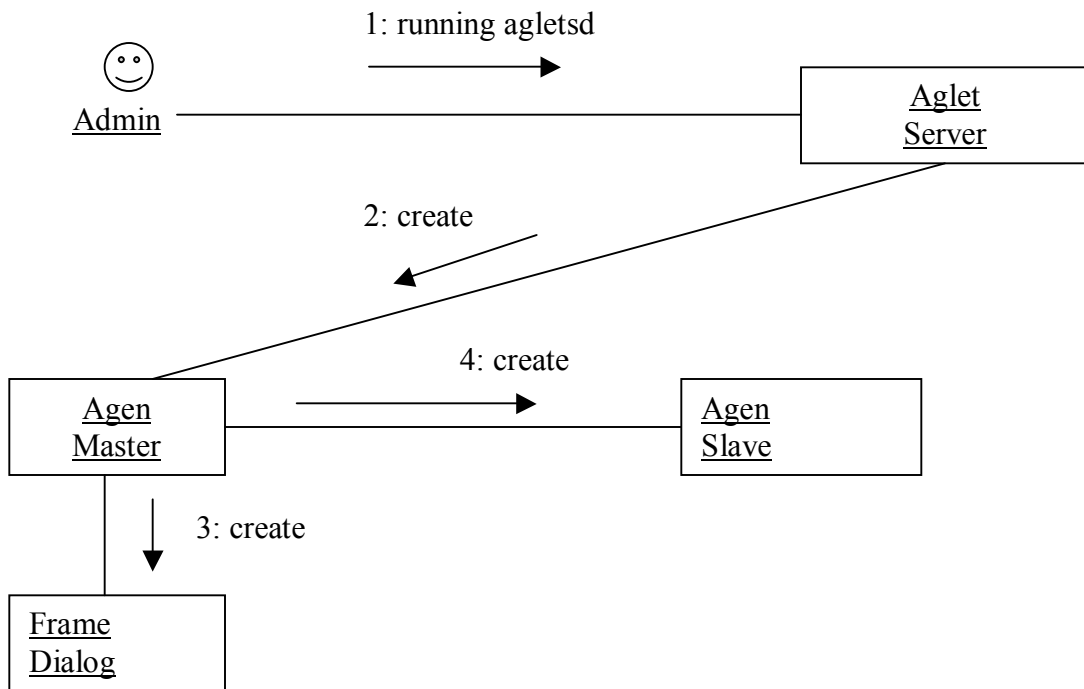
Langkahnya sebagai berikut :

1. Langkah pertama adalah Admin menentukan host-host mana yang akan dideteksi

2. Melakukan scanning nama host yang aktif. Untuk mempercepat proses, terlebih dahulu dilakukan pengiriman paket ping ke masing-masing host untuk memeriksa apakah host yang bersangkutan sedang aktif atau tidak. Bila sampai batas waktu tertentu tidak diperoleh balasan (reply) dari host yang bersangkutan. Maka dapat disimpulkan bahwa host tersebut sedang tidak aktif atau sedang tidak terhubung ke jaringan sehingga tidak dilakukan proses pengiriman mobile agent. Hanya terhadap host-host yang sedang aktif saja proses tersebut dilakukan.
3. Mengirimkan agen ke host tersebut. Agar agen dapat sampai ke tujuan, sebelumnya harus diketahui terlebih dahulu alamat IP atau nama host dari host tujuan.
4. Di tiap host yang disinggahi, agen akan melakukan pengambilan informasi yang diperlukan untuk deteksi penyusupan.
5. Mobile agent membawa rencana perjalanannya (itinerary) dan akan berpindah ke host berikutnya dalam rencana perjalanan tersebut.
6. Setelah semua host dalam rencana perjalanan dikunjungi, agen kembali ke host pengirim dengan membawa serta informasi yang telah dikumpulkan tersebut.
7. Untuk kemudian dilakukan analisa dan ditampilkan hasilnya di komputer pengirim agen. Informasi yang ditampilkan dalam bentuk informasi setiap host, maupun informasi penting seluruh host pada jaringan.

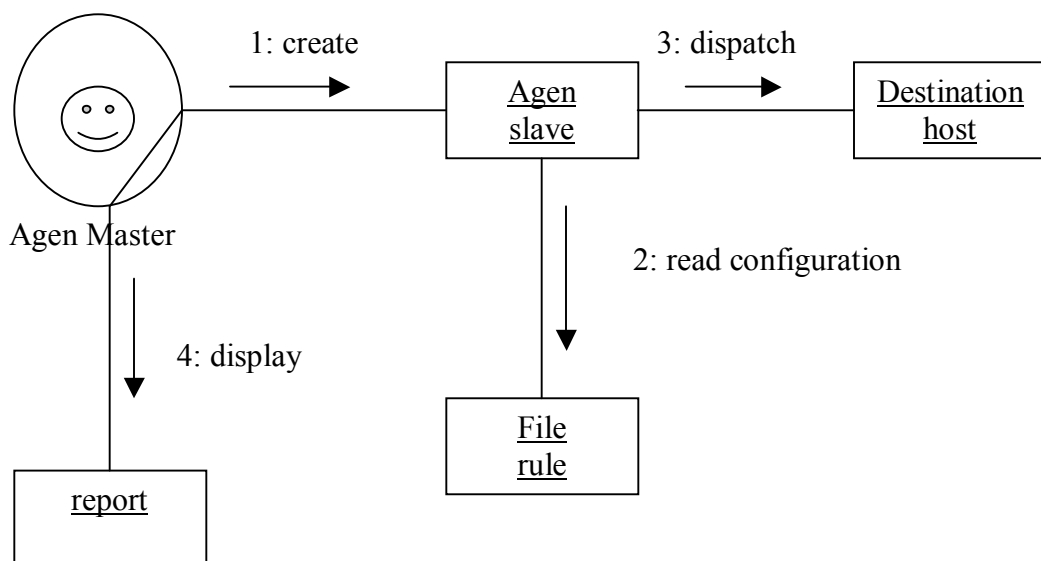
4.7 Perancangan Agen

Dalam aplikasi ini dipergunakan dua buah agen: yaitu agen Master (MasterIDS) dan agen Slave (SlaveIDS). Agen Master sebagai stationary agent, akan dijalankan di komputer asal oleh admin dengan mempergunakan Tahiti. Setelah agen Master berhasil dipanggil, maka agen tersebut akan membuat frame untuk dialog interaksi dengan user, serta membuat agen SlaveIDS.



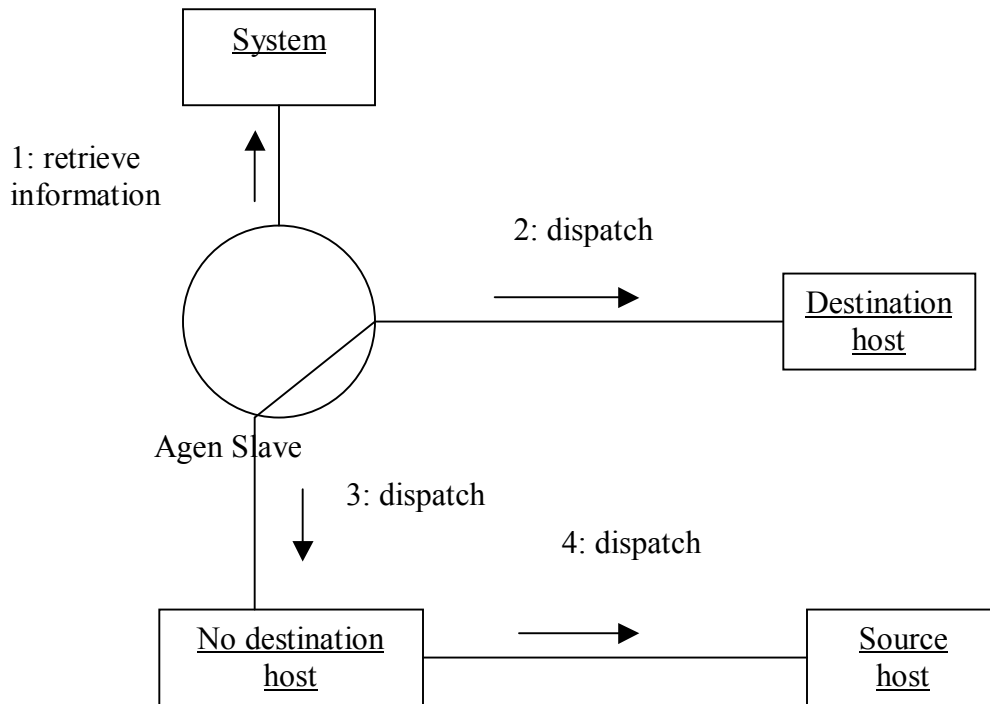
Gambar 4.3 Collaboration diagram create agent

Selanjutnya agen Slave membaca file konfigurasi yang diperlukan untuk menentukan pengambilan informasi dan kebijakan jaringan. Lalu dispatch agen Slave lewat jaringan ke host tujuan. Langkahnya dapat dilihat pada diagram kolaborasi di gambar 4.3.



Gambar 4.4 Collaboration diagram dispatching agent

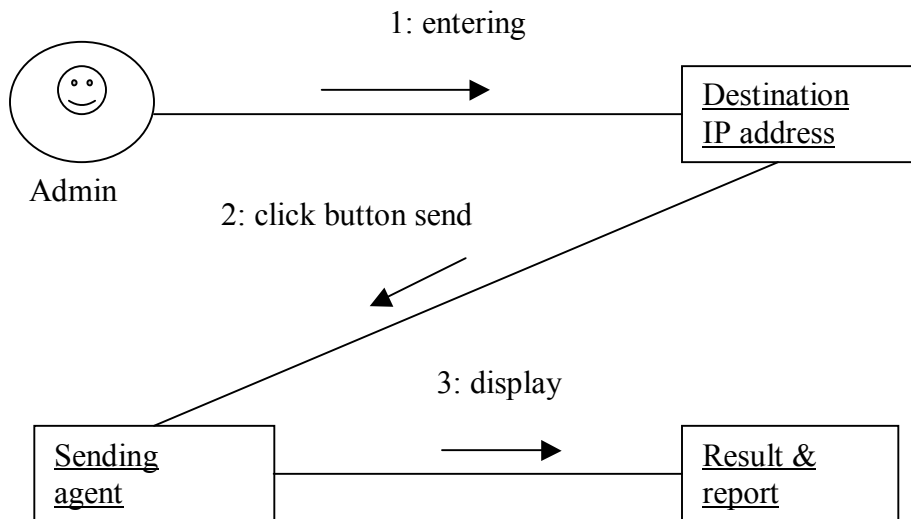
Sesampai di host tujuan, agen Slave akan mengumpulkan informasi deteksi penyusupan yang diperlukan. Bila tidak ada lagi host tujuan maka agen Slave akan kembali ke host asal pengiriman.



Gambar 4.5 Collaboration diagram detection agent

4.8 Antarmuka User

Antarmuka user di sini selain sudah disediakan oleh Tahiti, juga dilakukan oleh agen Master dengan membuat frame dialog. Ini memungkinkan user untuk memasukan alamat dari host-host yang menjadi tujuan deteksi penyusupan. Kemudian di akhir perjalanan, agen master akan menampilkan hasil dari pengumpulan dan analisa informasi deteksi penyusupan yang diperoleh. Seperti terlihat pada gambar 4.6



Gambar 4.6 Collaboration diagram user

4.9 Lingkungan Implementasi

Program diimplementasikan dengan memanfaatkan:

1. Aglet API Development Kit versi 1.1.0. Karena kemampuannya membuat lightweight agent, maka Aglet API sering disebut RISC mobile agent.
2. Bahasa pemrograman yang dipergunakan adalah Java dengan tool pengembangan JDK versi 1.1.8 for linux/windows
3. Semua perangkat lunak dipergunakan pada sistem operasi windows dan Linux RedHat 7.0
4. MySQL merupakan server database yang memiliki kemampuan multiuser, berukuran kecil, memiliki kecepatan dan kinerja yang baik. Selain itu juga merupakan proyek open source yang bebas dipergunakan secara gratis.

4.10 Pembuatan Class

Aplikasi mobile agent ini terdiri dari tiga buah class utama, yang bertugas untuk melakukan deteksi dan interaksi masukan/keluaran dengan user.

Class tersebut adalah:

1. MasterIDS
2. FrameIDS
3. SlaveIDS

4.10.1 Class MasterIDS

Class ini merupakan class utama yang akan dijalankan pertama kali oleh user dari antarmuka Tahiti. Class MasterIDS merupakan stationary agent yang mengontrol masukan dari class FrameIDS, serta bertugas membentuk, mengirimkan /menerima informasi dari class SlaveIDS. Class ini dimodifikasi dari class CirculateAglet.

Karena ada kemungkinan tidak semua host tujuan sedang aktif, maka class ini memiliki fungsi ping untuk memeriksa host yang aktif. Selanjutnya pengiriman mobile agent hanya akan ditujukan pada host yang aktif saja. Hal ini bertujuan untuk mempercepat proses deteksi. Sejumlah fungsi yang ada:

1. boolean handleMessage(Message msg): menangani pesan dari agen Slave
2. boolean ping (String pingAddress): menemukan host tujuan yang aktif

4.10.2 Class FrameIDS

Class ini merupakan class yang bertujuan membuat tampilan dialog untuk interaksi dengan user. Sejumlah pekerjaan yang dilakukannya:

1. Menerima masukan penambahan dan pengurangan host-host yang akan menjadi tujuan perjalanan mobile agent (itinerary), serta perintah pengiriman agent
2. Menampilkan informasi deteksi yang diperoleh oleh mobile agent, atau menampilkan informasi deteksi yang sudah tersimpan pada database.
3. Menampilkan status keberangkatan / kepulangan suatu mobile agent

Class ini dimodifikasi dari class CirculateFrame pada aplikasi CirculateAglet yang terdapat pada Aglets. Fungsi yang ada disini adalah :

void actionPerformed (ActionEvent) untuk menangani event.

4.10.3 Class Slave IDS

Ini adalah class mobile agent yang akan berkeliling jaringan. Class ini dibuat oleh MasterIDS, dikirimkan ke host tujuan, mengambil informasi dari host, dan melanjutkan perjalanan sesuai rencana perjalanan (itinerary). Class ini dimodifikasi dari class

FingerSlave yang terdapat pada aplikasi Finger di Aglets. Sejumlah pekerjaan yang dilakukan oleh class SlaveIDS:

1. Melakukan inisialisasi yang diperlukan untuk memulai perjalanan, seperti rencana perjalanan
2. Melakukan perjalanan ke host-host tujuan
3. Mengambil informasi pada host tersebut
4. Kembali ke host asal pengiriman, memberikan pesan kepulangan ke class MasterIDS
5. Melakukan analisa dan penyimpanan hasil ke database

Class Slave IDS mempergunakan sejumlah fungsi:

1. void onCreate (Object ini): dijalankan saat agen ini tercipta, lalu akan memanggil fungsi void readConfig () untuk membaca file konfigurasi (aturan.txt) untuk menentukan inisialisasi sejumlah variabel sesuai kebijakan jaringan.
2. boolean handle Message(Message msg): penanganan pesan dari agen master.
3. void start () : diaktifkan untuk memulai perjalanan berkeliling jaringan
4. void doJob () : diaktifkan pengambilan informasi yang dilakukan di tiap host.

Untuk keperluan itu mempergunakan fungsi-fungsi:

- a. void ambilport () : mengambil port tertentu yang aktif
 - b. void ambilDIR () : mengambil free disk space di windows
 - c. void ambildf () : mengambil free disk space di linux
 - d. void ambilfree () : mengambil free memori di linux
 - e. void ambiluptime () : mengambil waktu uptime di linux
 - f. void ambiltmp () : mengambil ukuran directory /tmp di linux
 - g. void ambillog () : mengambil informasi login di linux
 - h. void ambilproses () : mengambil perintah tertentu yang aktif
5. void doResult () : menangani dan analisa informasi deteksi, penyimpanan ke database

Kesimpulan

Dalam perkembangannya, sistem deteksi penyusupan telah dirancang dengan bermacam-macam algoritma dan struktur deteksi. Diawali dengan host-based, lalu berubah ke sistem network-based, dan dalam beberapa tahun kemudian memiliki

kecenderungan kombinasi terdistribusi dari keduanya. Bagaimanapun, selama perubahan itu, sumber informasi yang dipergunakan tidak berubah. Diharapkan hasil kerja ini akan memberikan suatu panduan untuk integrasi pada perancangan sistem selanjutnya yang akan berdampak pada peningkatan kinerja. Beberapa kontribusi utama :

1. Menyatakan properti dari mobile agent berkaitan dengan penggunaannya untuk deteksi penyusupan, dan arsitektur untuk pembuatan IDS berdasar pada mobile agent.
2. Merancang suatu prototype dari IDS mempergunakan arsitektur tersebut, sehingga menunjukkan kemungkinan mempergunakan mobile agent untuk melakukan deteksi penyusupan yang dapat dijalankan di platform sistem operasi berbeda.
3. Mobile agent bisa meningkatkan kemampuan deteksi sistem, mengeksplorasi kemampuan deteksi serangan tertentu yang susah bila diimplementasikan dengan IDS tradisional. Dengan sekumpulan agen ini menembus batasan tradisional dari IDS dengan *host based* atau *network based*.

Referensi

- [1] R.Kresno Aji, “Kejahatan Internet, Trik Aplikasi & tip Penanggulangannya”, Elexmedia Komputindo,2002
- [2] Rebecca Bace and Petter Mell, “Intrusion Detection System”, NIST Special Publication on IDS,2002.
- [3] Jai Sundar Balasubramaniyan, Jose Omar, David Isacoff, dan Diedo Samboni,” An Architecture for Intrusion Detection Using Autonomous Agents”, Center for Education and Research in Information Assurance and Security, Departemen of Computer Sciences Purdue University, 11 juni 1998
- [4] Mike Bursell, “Aglets Puppies Workshop”, December 1997
- [5] Dan Farmer and Wietse Venema,”Improving The Security of Your Site by Breaking Into It”, Sun Microsystem, 1994
- [6] Sidiq , “Notifikasi dan Akses Database Terdistribusi menggunakan Agen:, Tesis, 2002
- [7] Craig Hunt, “TCP/IP Network Administration”, O’Reilly & Associates, Inc,1992
- [8] InfoLinux,”Sistem Pendeteksiian Intrusi”,www.infolinux.web.id, juni 2002
- [9] Internet Security Systems,”A Strategy For A Succesfull IDS Evaluation”,www.iss.net,2002
- [10] Internet Security Systems, “Network VS Host-based Intrusion Detection: A Guide to Intrusion Detection Technology”,www.iss.net.net,2002
- [11] Jerry R Jackson,”Java by Example Edisi Indonesia”, Andi offset,1996
- [12] Admir Kulin.” A Distributed Security Managment System Based on Mobile Agents”, Universitas Teknik Wina,2001
- [13] Danny B Lange, “ Mobile Object and Mobile Agent: The Future of Distributed Computing?”, General Magic Inc, California,
www.moe-lange.com/danny/ecoop98.pdf,2000
- [14] Danny B Lange,”Mobile Agents with Java: The Aglets API”, www.moe-lange.com/danny/wwwj.pdf,2000
- [15] Danny B Lange, “ Programming and Deploying Java Mobile Agents with aglets”, Addison-Wesley,1998
- [16] Qusay H. Mahmoud, “Distributed Programming with Java”, Manning Publications

Co,2000

- [17] Microsoft Corporation, “Microsoft Windows NT Server Instalation Guide”,Microsoft Cop,1995
- [18] Mitsuru Oshima, Guenter, “Aglets Specification 1.1 Draft”, IBM Corp,
www.trl.ibm.co.jp/aglets/spec11.html
- [19] Onno W. Purbo,”Trik Pemrograman Java untuk Jaringan dan Internet”,
Elexmedia,2000
- [20] Rahmat Rafiudin, “Menguasai Securiti Unix”, Elexmedia,2002
- [21] Agus Tri,” Desain dan Implementasi aplikasi Agent untuk Pemetaan Jaringan”,Tesis
Teknik Elektro,2001
- [22] Joel Scambray,Stuart and G. kurtz” Hacking Exposed: Network Security Secrets and
Solutions”, McGraw-Hill,2001
- [23] Eugene Spafford, “ A Framework and Prototype for Distributed Intrusion Detection
System”, Departement od Computer Sciences Purdue University,1998
- [24] Eugene Spafford, “Data Collection Mechanism For Intrusion Detection Systems”, ”,
Departement od Computer Sciences Purdue University,2000
- [25] Tony Wiharjito,”Keamanan Jaringan Internet”, Elex Media,2000
- [26] Hendra Jaya,” Belajar Sendiri Windows 2000 Server”, Elex Media,2002

