APTIKOM
ASOSIASI PERGURUAN TINGGI INFORMATIKA DAN KOMPUTER

MODUL PJJ

Mata Ajar

MANAJEMEN KEAMANAN INFORMASI DAN INTERNET

Topik Bahasan

EMPAT DOMAIN KERAWANAN SISTEM I

Versi

2013/1.0

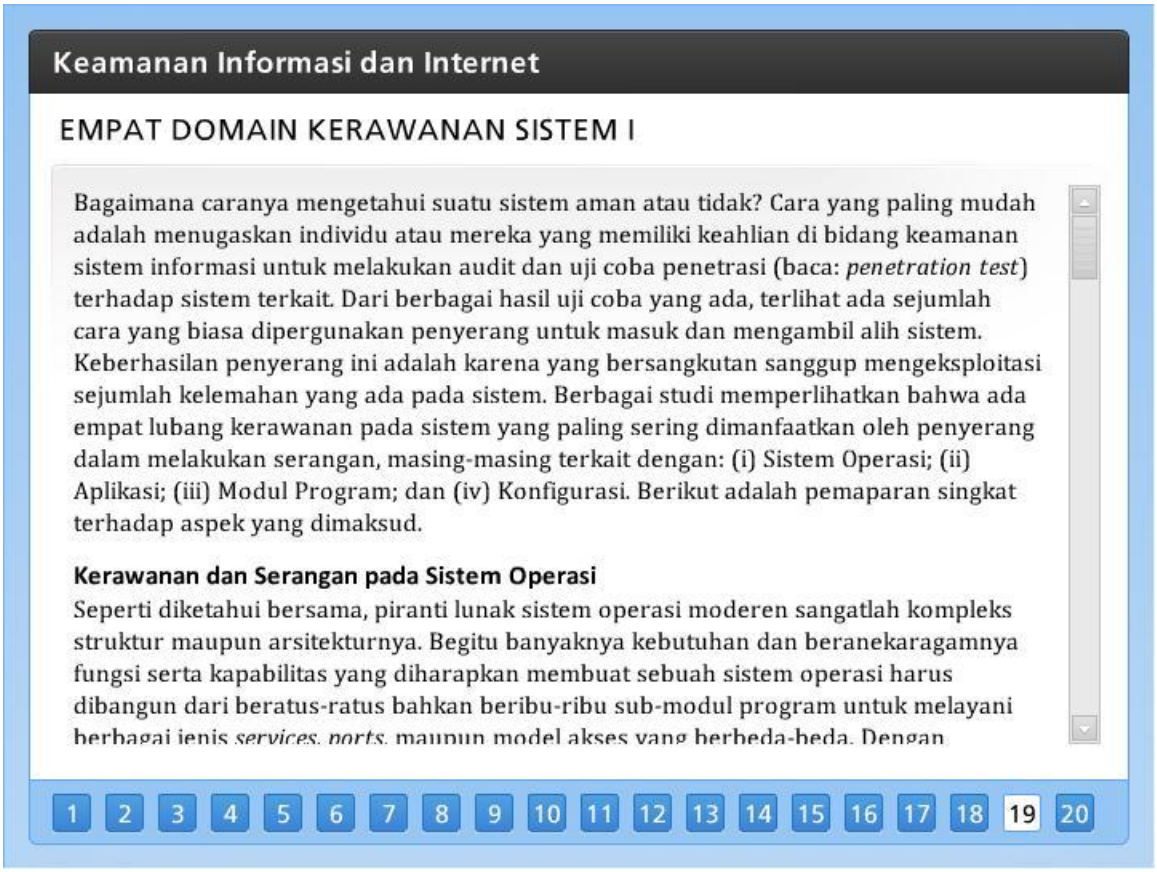
Nama File

MKIDI-4B-EmpatDomain1.pdf

Referensi Pembelajaran

4-B

EMPAT DOMAIN KERAWANAN SISTEM I



Bagaimana caranya mengetahui suatu sistem aman atau tidak? Cara yang paling mudah adalah menugaskan individu atau mereka yang memiliki keahlian di bidang keamanan sistem informasi untuk melakukan audit dan uji coba penetrasi (baca: *penetration test*) terhadap sistem terkait. Dari berbagai hasil uji coba yang ada, terlihat ada sejumlah cara yang biasa dipergunakan penyerang untuk masuk dan mengambil alih sistem. Keberhasilan penyerang ini adalah karena yang bersangkutan sanggup mengeksploitasi sejumlah kelemahan yang ada pada sistem. Berbagai studi memperlihatkan bahwa ada empat lubang kerawanan pada sistem yang paling sering dimanfaatkan oleh penyerang dalam melakukan serangan, masing-masing terkait dengan: (i) Sistem Operasi; (ii) Aplikasi; (iii) Modul Program; dan (iv) Konfigurasi. Berikut adalah pemaparan singkat terhadap aspek yang dimaksud.

Kerawanan dan Serangan pada Sistem Operasi

Seperti diketahui bersama, piranti lunak sistem operasi moderen sangatlah kompleks struktur maupun arsitekturnya. Begitu banyaknya kebutuhan dan beranekaragamnya fungsi serta kapabilitas yang diharapkan membuat sebuah sistem operasi harus dibangun dari beratus-ratus bahkan beribu-

ribu sub-modul program untuk melayani berbagai jenis *services, ports,* maupun model akses yang berbeda-beda. Dengan demikian maka diharapkan pengguna dapat melakukan instalasi sistem operasi sesuai dengan spesifikasi keinginannya, melalui penyesuaian terhadap berbagai parameter yang tersedia. Namun di sini pulalah letak kerawanannya. Seorang pengguna atau *user* misalnya, sering sekali dalam melakukan instalasi sistem memilih mode "standard", alias tanpa melakukan kustomisasi terhadap sejumlah parameter terkait, yang diantaranya menyangkut dengan masalah tingkat keamanan. Tentu saja hal ini berakibat tidak terkonfigurasinya proteksi keamanan terhadap sejumlah *services* maupun *ports* terkait, sehingga memudahkan penyerang untuk melakukan penyusupan dengan memanfaatkan lubang-lubang kerawanan tersebut.

Alasan berikutnya mengapa banyak terdapat kerawanan pada sistem operasi adalah karena begitu banyaknya modul-modul serta sub-sub modul pembentuknya, mengakibatkan sering kali sebuah perusahaan pembuat sistem informasi "tidak sempat" melakukan uji coba keamanan terhadap seluruh kombinasi dan/atau permutasi dari keseluruhan modul dan sub-modul pembentuk sistem operasi yang dimaksud.

Namun terdapat pula lubang kerawanan yang sifatnya "tak terhindarkan" karena merupakan bagian dari *trade-off* kinerja yang diharapkan dari sebuah sistem operasi - sehingga harus dilakukan sebuah desain atau rancangan piranti lunak yang sedemikian rupa. Lihatlah fenomena *buffer overflow* yang sebenarnya merupakan dampak dari mekanisme *memory swap* yang pada dasarnya merupakan solusi dari permasalahan sistem operasi klasik. Atau masalah penentuan tingkat keamanan yang dapat diubah-ubah parameternya oleh pengguna sesuai dengan profil resiko yang ingin diadopsi.

Hal lainnya yang juga mengemuka adalah begitu banyaknya aplikasi yang berfungsi sebagai "tambal sulam" (baca: *patches*) terhadap lubang-lubang sistem operasi yang belum mengalami tes uji coba secara holistik. Sifatnya yang *ad-hoc* dan lebih bersifat reaktif dan jangka pendek terkadang menimbulkan sebuah kerawanan baru yang tanpa disadari tertanam dalam sistem operasi terkait.

Kerawanan dan Kualitas Aplikasi

Dalam kenyataan sehari-hari, ada tiga jenis piranti lunak aplikasi yang biasa dipergunakan. Jenis pertama adalah aplikasi siap pakai yang dibeli di pasar *software* dan langsung diterapkan, jenis kedua merupakan aplikasi yang dibangun sendiri oleh perusahaan yang bersangkutan, dan jenis ketiga yaitu aplikasi yang merupakan kombinasi dari keduanya. Terlepas dari perbedaan ketiga jenis tersebut, keseluruhannya merupakan sebuah karya intelektual dari seorang atau sekelompok orang

yang memiliki kompetensi terkait dengan pengembangan atau rekayasa sebuah piranti lunak (baca: *software engineering*). Dari sinilah cerita terciptanya kerawanan bermula.

Pertama, pekerjaan pembuatan sebuah piranti lunak aplikasi biasanya memiliki target dan durasi penyelesaian tertentu, karena pengembangannya dilakukan melalui sebuah proses aktivitas berbasis proyek. Hal ini berarti bahwa setiap praktisi pengembang aplikasi, memiliki waktu yang sangat terbatas. Dalam kondisi ini sangat wajar jika terjadi sejumlah "kecerobohan" atau "kekurang-hati-hatian" karena dikejar atau diburu-buru target "waktu tayang" alias penyelesaian. Jadwal yang ketat ini secara teknis dan psikologis sangat berpengaruh terhadap terciptanya sebuah piranti aplikasi yang terbebas dari berbagai lubang-lubang kerawanan yang ada.

Kedua, mengingat begitu banyaknya modul dan objek program pembentuk sebuah aplikasi, dimana pada saatnya nanti *software* tersebut akan diinstalasi di berbagai jenis dan ragam lingkungan piranti keras serta jejaring komputer yang berbeda, akan teramat sulit untuk melakukan uji coba aplikasi yang mencakup seluruh kemungkinan konfigurasi sistem yang ada. Artinya adalah bahwa sang pengembang tidak memiliki data atau informasi yang lengkap dan utuh mengenai kinerja sistem secara keseluruhan dalam berbagai kemungkinan konfigurasi sistem.

Ketiga, masih begitu banyaknya *programmer* jaman sekarang yang memikirkan aspek keamanan sebagai sesuatu yang *additional* atau bersifat *afterthought consideration* - alias dipikirkan kemudian sebagai sebuah "pertimbangan tambahan" setelah sebuah piranti aplikasi dibangun. Padahal sifat dan karakteristik keamanan yang holistik haruslah dipikirkan sejalan dengan kode program dibuat dan dikembangkan. Belakangan ini mulai terlihat marak diperkenalkan teknologi yang terkait dengan *secured programming* untuk mengatasi berbagai jenis kerawanan akibat aktivitas pembuatan program konvensional yang tidak memperhatikan aspek penting ini.

Keempat, pengetahuan "pas-pasan" dari pengembang piranti lunak tidak jarang membuat kualitas keamanan dari sebuah aplikasi sedemikian buruk dan rendahnya. Hal ini disebabkan karena kebanyakan pemilik dan pengguna aplikasi hanya menilai efektivitas serta kinerja sebuah program dari segi kelengkapan fungsionalitas dan *user interface* saja, tanpa memikirkan mengenai kebutuhan berbagai jenis pengamanan yang diperlukan.

Kerawanan pada Modul Program

Seperti diketahui bersama, metodologi dan konsep pengembangan aplikasi moderen adalah dengan menggunakan pendekatan objek. Artinya adalah kebanyakan pengembang piranti lunak tidak selalu membuat fungsi, prosedur, modul, atau sub-program dari nol atau "from scratch", tetapi terlebih dahulu mencari apakah telah ada objek program yang telah dibuat orang lain dan dapat dipergunakan (baca: *reusable*). Kebiasaan ini mendatangkan sejumlah keuntungan, terutama terkait dengan faktor kecepatan proses dan penghematan biaya pengembangan aplikasi. Namun segi negatifnya adalah begitu banyaknya pengembang yang "pasrah" percaya saja menggunakan sebuah objek tanpa tahu kualitas keamanan dari "penggalan" program tersebut. Contohnya adalah penggunaan modul-modul semacam *libraries, scripts, drivers,* dan lain sebagainya. Pada kenyataannya, begitu banyak ditemukan modul atau objek program yang sangat rawan karena tidak dibangun dengan memperhatikan faktor keamanan - karena sebagian besar dari objek tersebut dibangun hanya dengan memperhatikan unsur fungsionalitasnya semata.

Kerawanan Akibat Konfigurasi Standar

Sebuah sistem informasi terdiri dari sejumlah piranti lunak dan piranti keras. Agar bekerja sesuai dengan kebutuhan, perlu diperhatikan sungguh-sungguh proses instalasi dan konfigurasi keseluruhan piranti yang dimaksud. Namun yang terjadi pada kenyataannya, tidak semua organisasi memiliki sumber daya manusia yang berpengetahuan dan berkompetensi memadai untuk melakukan hal tersebut. Akibatnya adalah tidak jarang dari mereka hanya mencari mudahnya saja, alias melakukan instalasi dan konfigurasi sistem secara standar, tanpa memperhatikan kebutuhan khusus dari organisasi yang bersangkutan. Misalnya adalah dalam hal melakukan konfigurasi *firewalls* dimana *port-port* yang seharusnya ditutup karena alasan keamanan menjadi terbuka karena sesuai dengan *set up* standar pabrik. Atau pada saat menginstalasi *software anti virus* dimana tingkat keamanan yang "dipasang" adalah *low* sehingga tidak berfungsi maksimal dalam melindungi sistem. Hal-hal semacam inilah yang menyebabkan "terciptanya" lubang-lubang kerawanan tanpa disadari.

Berdasarkan keempat jenis aspek kerawanan tersebut, ada baiknya sebuah organisasi menjalankan strategi khusus untuk menghindari diri dari kemungkinan dieskploitasi oleh pihak-pihak tidak bertanggung jawab, misalnya adalah dengan cara:

- Sebelum membeli atau mengadakan sebuah modul objek atau aplikasi, dilakukan penelitian terlebih dahulu mengenai kinerja program yang dimaksud, terutama dilihat dari segi atau aspek keamanannya;
- Pada saat mengembangkan sistem, dilibatkan *programmer* atau pihak-pihak yang paham benar dan memiliki kompetensi dalam ilmu *secured programming*, sehingga produk modul maupun aplikasi yang dibangun telah mempertimbangkan aspek keamanan yang dimaksud;
- Jika tidak memiliki sumber daya yang memiliki kometensi dan keahlian dalam mengkonfigurasi sebuah sistem, bekerjasamalah dengan konsultan atau ahli di bidang piranti tersebut agar dapat melakukan instalasi parameter keamanan dalam sistem yang dimaksud; dan lain sebagainya.