

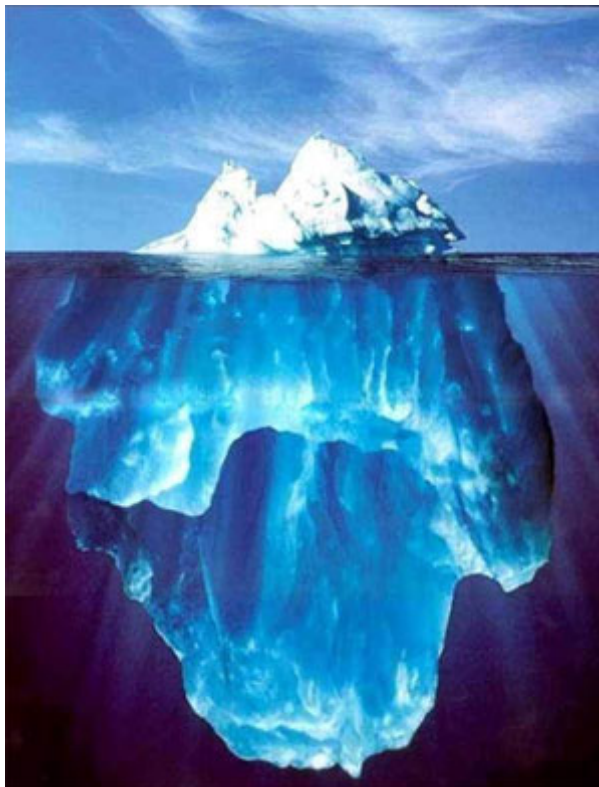
Segurança das Aplicações Web (PCI Standards)

“Most applications will never be secure enough to meet evolving threats. Companies must install a new line of defense in front of web applications.”
- Gartner, 2004

Roubo, Chantagem, Fraude

Em 9 de janeiro de 2000, a Diretoria da empresa CD Universe, varejista de CD's pela Internet, muito conhecida nos EUA, recebeu um e-mail de um certo Maxus, dizendo que havia roubado o cadastro de Cartões de Crédito da empresa, com mais de 300 mil nomes.

O cadastro incluía o nome do cliente, número do cartão, data de validade, endereço e outros dados confidenciais. Como prova desse roubo, divulgava em um site na Internet, 25 mil cartões utilizáveis.



A maior parte das vulnerabilidades da sua aplicação não está visível.

O crescimento do cybercrime nestes últimos anos deveu-se à crescente profissionalização dos criminosos. De simples e românticos hackers, para quadrilhas que procuram sistematicamente vulnerabilidades em sites de e-commerce.

Os ataques tem se concentrado nas portas públicas (80 e 443) que não devem ser bloqueadas pelos firewalls comuns, pois os visitantes não acessariam o seu site, nem as aplicações.

Mesmo os firewalls mais modernos da família “Stateful Multilayer Inspection” não conseguem analisar e filtrar a camada de Aplicação (Layer 7 do modelo OSI).

As aplicações específicas de uma empresa não tem Patches de atualização e nem foram pensadas em termos de segurança. Sobre este e outros riscos leia

As principais bandeiras de cartões de crédito, Visa, Mastercard e Amex se uniram em torno da organização sem fins lucrativos PCI Standards Council (www.pcisecuritystandards.org) definindo os padrões mínimos de segurança para os players da indústria de cartões de crédito.

Exigia certa quantia em dinheiro para não divulgar os outros quase 300 mil cartões.

A empresa chamou o FBI, para constatar que o hacker se encontrava na Rússia, ao abrigo das leis americanas contra hacker e fraude e que nada poderia ser feito.

Foi um ataque nas Portas Públicas (Portas 80 e 443) e aparentemente o mecanismo usado foi o SQL Injection, que possibilitou ao hacker conseguir as informações diretamente do banco de dados.

Complexidade

A segurança das aplicações, principalmente aquelas conectadas a uma rede aberta e perigosa como é a Internet é bastante complexa. Essa complexidade advém do fato que as aplicações web, e-commerce, Internet bank, na realidade são agrupamentos bastante heterogêneos de plataformas, bancos de dados, servidores de aplicação, etc..

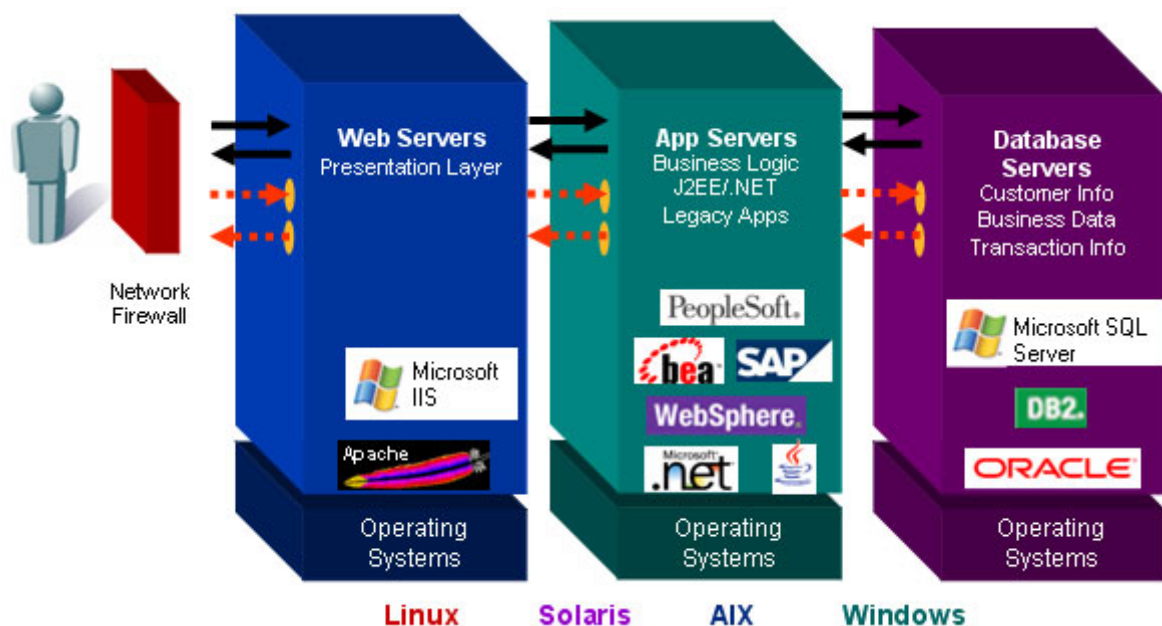
Uma aplicação típica, geralmente, está distribuída em vários servidores, rodando diversos aplicativos e para funcionar na velocidade adequada, a aplicação precisa que as interfaces entre os diversos sistemas sejam construídas com a premissa que os dados passados através da mesma são confiáveis e não hostis. Não há tempo hábil para duplas verificações.

O “calcanhar de Aquiles” destas aplicações é a necessidade de haver “confiança” entre os diversos subsistemas e é disso que os hackers e outros cibercriminosos se aproveitam.

Nestes sistemas complexos, a segurança dos produtos disponíveis no mercado é assegurada pelos fabricantes, que fornecem periodicamente *patches* que os atualizam.

Para o sistema aplicativo, freqüentemente desenvolvido *in house* ou por terceiros, especificamente para a empresa, não existem *patches* de segurança.

Segundo o Gartner, 75% dos ataques são concentrados nos aplicativos específicos de cada empresa, pois os atacantes sabem das suas fragilidades.



Desafio

Segundo a revista americana Software Magazine, uma aplicação web típica tem de 150 mil a 250 mil linhas de código. São grandes e complexas. Lidam com vários subsistemas, bancos de dados e sistemas operacionais.

Na sua especificação foi pedido rapidez de processamento (*os visitantes não gostam de esperar*), beleza e usabilidade (*a idéia é transformar visitantes em clientes*), prazo de entrega e custo baixo.

Normalmente segurança não faz parte dos requisitos do sistema, a não ser vagamente, “... assegurar que os visitantes e clientes se identifiquem com login e password...”, “... as passwords serão armazenadas criptografadas no sistema...” ou se faz, não se prevê nenhum mecanismo periódico de atualização dos sistemas em função de novas ameaças.

Normalmente segurança é subentendida como um problema do pessoal da segurança, da infraestrutura e não do pessoal de desenvolvimento ou produto, a não ser em termos de generalidades.

Segundo estudos do Ministério da Defesa Americano (DoD ou Pentágono), “ *cada 1.000 linhas de código embutem 15 defeitos de segurança!*”

Numa aplicação típica de 200 mil linhas, existirão, portanto, 3.000 defeitos de segurança, que precisarão ser identificados e corrigidos.

Softwares que *scaneiam* aplicações, como o WebInspect da SpiDynamics são de grande ajuda para garantir a qualidade do que é entregue pelo pessoal de desenvolvimento em termos de segurança.

O que esses softwares não fazem é corrigir a parte defeituosa identificada no código. Isto tem que ser feito por analistas e programadores que conheçam o sistema, tenham sido treinados em segurança e sigam uma metodologia que garanta o desenvolvimento de aplicações seguras.

Um estudo de cinco anos, realizado pelo próprio Pentágono, estima que se gastem 75 minutos em media para identificar um defeito de segurança e de 2 a 9 horas para corrigi-lo.

Pode-se estimar facilmente o numero de horas necessárias para se corrigir todos os defeitos de segurança de uma aplicação típica, supondo-se 5 horas de analista/programador para identificar e corrigir um defeito de segurança:

200.000 linhas / 1.000 linhas por defeito x 15 defeitos

x 5 horas = 15.000 horas

Cada aplicação demandaria 15 mil homens/hora de analista/programador para ser corrigida. Supondo que cada analista/programador trabalhe realmente 6 horas por dia, vezes 22 dias úteis, teríamos 132 horas de trabalho por mês,

A aplicação demandaria 114 homens/mês, e se fossem alocados 4 pessoas na equipe de correção, o sistema estaria pronto e corrigido em modestos 29 meses!

Alem do tempo necessário para corrigir os erros e defeitos, o custo, muitas vezes escondido na forma de custos fixos, passa a ser um desafio importante.

Tudo isso supondo que a aplicação não sofra melhorias e manutenções, que certamente introduzirão novos defeitos de segurança. O processo todo fica parecido com “enxugar gelo”, mal os erros são detectados e corrigidos, uma nova versão da aplicação é liberada por pressão dos usuários, normalmente as áreas de marketing e comercial, que conseguem toda a atenção administração da empresa.

Ataques

Os ataques que hoje conhecemos são baseados em vulnerabilidades típicas de aplicações web complexas.

Mesmo os sistemas operacionais (Windows, AIX, Solaris) que são mantidos por grandes empresas, empregando milhares de profissionais, têm vulnerabilidades que são periodicamente descobertas por *hackers* e só se transformam em *patches* depois que os *hackers* já se atacaram algumas vezes, que o problema foi comunicado ao fabricante e devidamente corrigido.

A Internet agregou outros componentes de risco, sendo muito importante o “efeito comunidade” em que os *hackers* e outros criminosos se julgam fazendo parte de uma “comunidade” e obrigados a compartilhar rapidamente suas descobertas.

Isto significa que qualquer vulnerabilidade descoberta nas suas aplicações será rapidamente divulgada, com as ferramentas necessárias para atacá-la, e outros *hackers* e cibercriminosos aproveitarão as vulnerabilidades da sua aplicação.

Os ataques podem causar uma série de problemas, entre os quais se pode citar:

- Perdas Financeiras;
- Transações Fraudulentas;
- Acesso não autorizado a dados, inclusive confidenciais;
- Roubo ou modificação de Dados;
- Roubo de Informações de Clientes;
- Interrupção do Serviço;
- Perda da confiança e lealdade dos clientes;
- Dano à imagem da marca.

Os tipos mais comuns de ataques são:

1. [Cross-Site Scripting](#)
2. [SQL Injection](#)
3. [Command Injection](#)
4. [Cookie/Session Poisoning](#)
5. [Parameter/Form Tampering](#)
6. [Buffer Overflow](#)
7. [Directory Traversal/Forceful Browsing](#)
8. [Cryptographic Interception](#)
9. [Cookie Snooping](#)
10. [Authentication Hijacking](#)

Uma explicação mais detalhada desses e outros tipos de ataque encontram-se no site www.owasp.org

Application Firewalls

A maioria dos *firewalls de rede*, por se concentrar nas camadas mais baixas, não protege as aplicações da maior parte desses ataques, protege sim o acesso aos recursos de rede.

Uma nova geração de *appliances* está surgindo para resolver este e outros problemas, os **Application Firewalls**. Fazem parte de um novo conceito, que é a defesa na camada de aplicação.

Defesa das aplicações dos clientes, não padronizadas, heterogêneas, distribuídas em vários *sistemas operacionais*, usando diversos *servidores de aplicação* e de *bancos de dados*.

Surgiram só agora, por duas razões, primeiro a necessidade de combater ataques cada vez mais inteligentes e segundo a disponibilidade da tecnologia necessária para a criação desses *appliances* que necessitam monstruosa capacidade de computação.

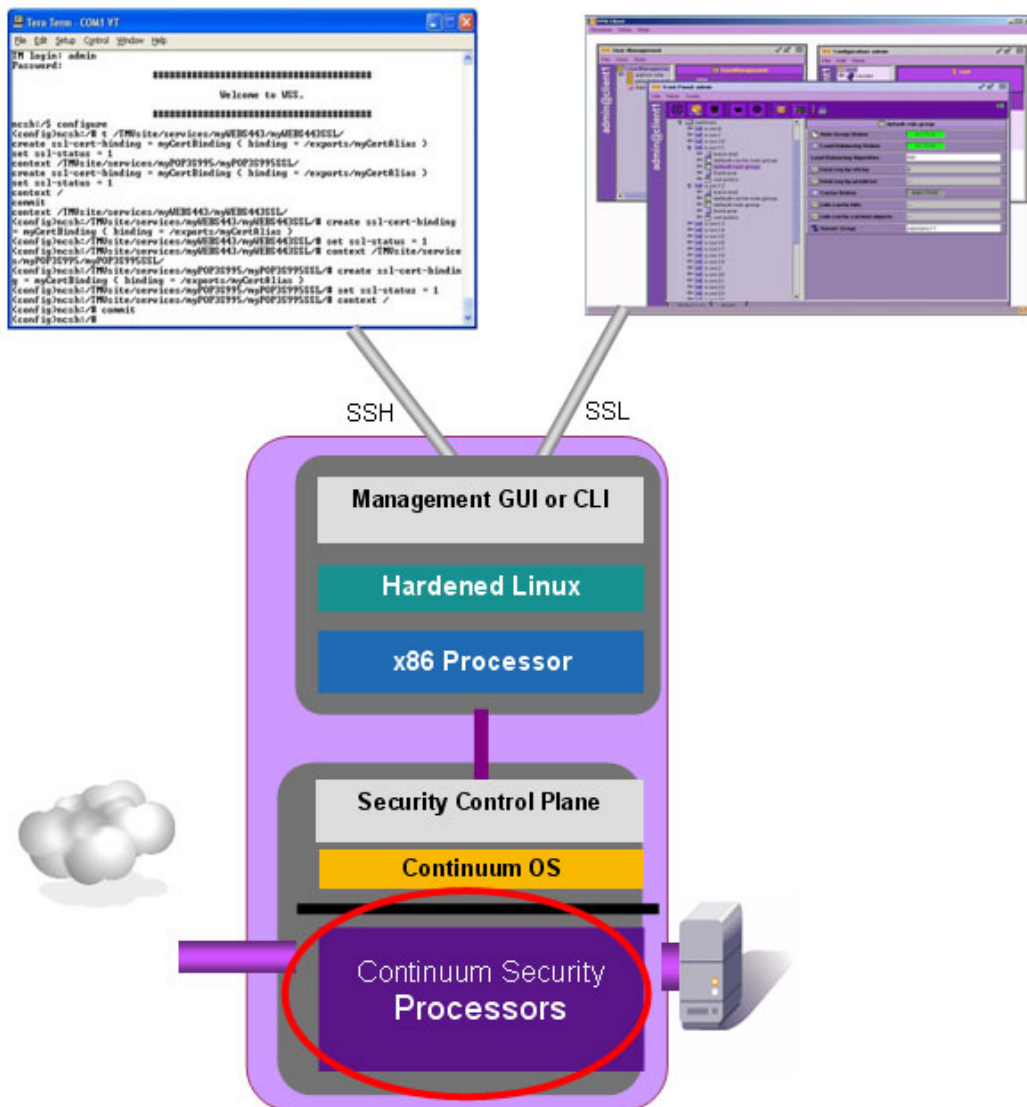
Tecnologia

O desenvolvimento dos Firewall da Aplicação exigiu além de um conhecimento profundo de como funcionam os protocolos Internet, como se processam os ataques, principalmente na camada de aplicação, o desenvolvimento da tecnologia necessária para suprir as gigantescas necessidades de computação de um Application Firewall.

Essas necessidades são oriundas dos algoritmos e regras do software de detecção e da velocidade em que os pacotes transitam pela rede.

Além disso existe a necessidade de decriptografar e criptografar os pacotes que passam por ele, o que é uma atividade que consome enormes recursos computacionais.

Só como um exemplo da necessidade e complexidade da arquitetura de um firewall de aplicação, mostrados a seguir o que é do Barracuda Web Application Firewall, que usa a tecnologia da Netcontinuum, um dos líderes da categoria.



Para mais informações, acesse: www.clm.com.br/pci