DEFESA EM PROFUNDIDADE (DEFENSE IN-DEPTH) E A FALÁCIA DA SEGURANÇA

24/01/2014

Continuando a série de posts que estou fazendo sobre segurança da informação (mais uma vez um grande obrigado a 4Linux por me ceder o espaço e garantir a audiência) e algumas das inter-relações com a comunidade de software livre e de pesquisas, decidi escrever um pouco sobre a defesa em profundidade, pra mim um dos diversos pontos amplamente de modo errôneo.

Apenas para esclarecer, eu mesmo no passado acreditei na existência de métodos para implementação da defesa em profundidade usando tecnologias atuais (recém lançadas na época, tais quais IDSs – intrusion detection systems e antivírus em diversos locais da rede com tecnologias diferentes de detecção). Também visando a clareza, acredito e apoio a necessidade de defesa em profundidade. Este artigo apenas destaca de que realmente se trata a defesa em profundidade.
O grande motivador para a escrita veio de uma recente discussão via Twitter (me sigam ou vejam a discussão: @bsdaemon) com um pesquisador aclamado pelo qual tenho profundo respeito (@grsecurity). Ele mencionou a questão de defesa em profundidade em relação a diferentes formas (4 diferentes formas na verdade) que o sistema desenvolvido por ele (por sinal Open Source e a melhor opção de segurança para sistemas operacionais – e aplicações se contarmos os plugins de GCC apresentados no H2HC [1]) conseguia bloquear uma vulnerabilidade recém lançada que afetava o Kernel do Linux [2].
Se pensarmos nas origens militares para o tema defesa em profundidade e percebermos como ele é aplicado em segurança da informação, teríamos a seguinte situação (Obs.: O exemplo militar demonstrado não condiz nem tentou condizer com o real uso de forças militares e disposição em campos de batalha, apenas demonstrar o que seria segurança em profundidade):

- Imagine que você possui uma divisão de seu exército disposta da seguinte forma:
  - Infantaria
  - Artilharia
  - Cavalaria
- Imagine que tal divisão protege uma cidade importante pois possui o ponto central do sistema de comunicação de seu exército (obviamente já um erro estratégico)
- Um ataque inimigo iria considerar as fortificações de sua cidade antes de um ataque:

- Homens na infantaria, armas da artilharia, capacidade da cavalaria e assim por diante -> Isso é feito pela inteligência militar

- Ao iniciar o ataque, o inimigo teria de passar por todas estas camadas de defesa, incluindo uma não mencionada, que seria a contrainteligência militar (que tentaria impedir que o inimigo conhecesse com exatidão as capacidades defensivas)

- Mesmo que no cenário acima o inimigo saiba das suas capacidades, o ataque dele ainda assim seria prolongado por elas, ou seja, ele teria de destruir cada camada sua de defesa, enfrentar milícias da cidade e outras camadas adicionais, propiciando assim a possibilidade de você enviar reforços (ou seja, a inteligência militar de seu inimigo teria também de considerar onde estão outros batalhões de seu exército, capacidade de contingenciamento, armamentos e logística, para ter certeza de que você não iria conseguir proteger a cidade antes do ataque destruir os sistemas.

Agora se convertermos este cenário para o ambiente computacional, temos:

- A inteligência inimiga seria a fase de reconhecimento do ataque. Ela visa levantar informações sobre as capacidades defensivas do ambiente e o que o mesmo executa.

- O ataque inimigo é desferido com exatidão, ou seja, se você possui um IPS, um antivírus no gateway, um firewall, um antivírus na estação, diferentes recursos de proteção no sistema operacional e todos os outros elementos que formam sua defesa (que deveria ser em profundidade conforme as empresas de segurança tentam lhe convencer), ainda assim o tempo DURANTE o ataque não é prolongado. Estes recursos todos são completamente inutilizados AO MESMO tempo.

- Alguns argumentarão: Sim, mas aumentará a complexidade para o inimigo. Vejam que a complexidade aumentada aqui é a da fase de inteligência/preparação do ataque. Não do ataque em si.

Meu argumento aqui é de que todos entendemos a importância da segurança em profundidade, mas infelizmente, o que implementamos é na GRANDE maioria das vezes, apenas artifícios de contrainteligência. Todos eles fazem parte apenas de uma única camada.

Não importa que o inimigo gastará mais para preparar um exploit contra sua infraestrutura, que terá de possuir múltiplas vulnerabilidades combinadas para o ataque ser efetivo. Devemos lembrar que a segurança da informação é assimétrica, ou seja, o inimigo precisa de múltiplas vezes menos esforços para atacar seus pontos vulneráveis do que você para defendê-los, portanto aumentar o custo da contrainteligência não irá efetivamente bloquear os ataques (ou evitar que aconteçam) – isto pode efetivamente ser visto com os números crescentes de ocorrência de explorações.

Usando a analogia do cenário hipotético criado, em segurança da informação criamos diversas "camadas" que deveriam nos proteger, mas o inimigo encontra um túnel de acesso direto a nossa sala de equipamentos de

comunicação. Ou seja, os dados podem ser disponibilizados em um equipamento onde todas as "camadas" não existem pois, por exemplo, o mesmo se conecta fora da rede de computadores.
Um dos poucos cenários que consigo imaginar onde segurança em camadas de fato existe em segurança da informação:

- O adversário (inimigo?) deseja acessar uma informação que está contida em apenas um local (imaginemos um desktop que não permite conexões externas de dispositivos e limita os programas que podem rodar no equipamento, não permitindo portanto a cópia de tal arquivo)
- Uma vulnerabilidade existe em um aplicativo deste desktop, que permitiria ao atacante ganhar acesso ao mesmo. Tal desktop possui randomização de memória, páginas não executáveis (e diversos outros hardenings que discutirei em um artigo futuro).
- Ao ganhar acesso a tal aplicativo, o atacante ainda não possui acesso à informação desejada, afinal, a mesma não pode ser enviada para fora da máquina e portanto a aplicação vulnerável que se comunica com outras máquinas não possui acesso à informação (isso pode ser implementado facilmente via MAC – mandatory access control).
- O atacante terá de explorar outra vulnerabilidade no equipamento, mas agora, já no desktop da vítima, todas as informações do sistema em funcionamento são escondidas do processo em que o atacante está
- Mesmo sabendo que o Kernel do sistema operacional do desktop comprometido é um Linux, e mesmo possuindo alguns detalhes de tal Kernel e mesmo já tendo uma vulnerabilidade para tal Kernel, o atacante precisará migrar para outra aplicação, pois a que ele está possui um acesso limitado a interfaces do Kernel (o Sandbox do Chrome, por exemplo, não permite aos processos dentro do SVG acessarem todas as APIs exportadas pelo Kernel para aplicativos).
- O atacante já sabia disso e já possuía uma falha em uma aplicação mais privilegiada, mas dado todos os limites que o sistema impõe, ele terá de fazer um brute-force para conseguir bypassar a randomização.
- O sistema possui um limitador de tentativas falhas na execução do software, bloqueando o respawn do mesmo e alertando que algo está errado
- O alerta é enviado para um servidor remoto, que possui um sistema que automaticamente ao recuperar tal alerta, desliga o equipamento que enviou tal notificação, bloqueando totalmente o atacante antes dos dados serem acessados.

O exemplo que coloquei acima é interessante e pode ser implementado (apesar de ser irreal para diversas realidades). O mesmo inclusive ser implementado exclusivamente com tecnologias Open Source. Ainda assim conta com o fato do atacante não ter tido uma vulnerabilidade melhor para o segundo ataque, ou não ter obtido a informação do limitador de falhas. Ele também ignora o fato de que em algum momento o sistema alvo terá de ser

iniciado novamente, e mesmo que você corrija as duas falhas obtidas, agora o atacante possui mais informações (inteligência) sobre o seu sistema, e que ele não foi prejudicado realmente (ou seja, ele poderá atacar novamente). Tal exemplo também nos indica a necessidade de pensarmos segurança de uma forma diferente, discutirmos sobre segurança mais abertamente e esquecermos conceitos que concebemos como verdade, mas que pouco se aplicam na realidade.
Com uma mentalidade diferente, pode-se implementar segurança de forma criativa e efetiva, sem os tremendos gastos em software propostos pela indústria, e que pouco adicionam de fato a proteção das informações.

[1] – PaX Team. "PaX: The untold history (part II)". Hackers 2 Hackers Conference 10a Edição, 2013.
[2] – Krause, Mathias. "IPC DoS Fix". Site: https://lkml.org/lkml/2013/11/2/81. Acessado em: 11/11/2013.

Sobre o autor

Rodrigo Rubira Branco (BSDaemon), atua como pesquisador sênior no centro de excelência em segurança da Intel . Foi fundador do projeto Dissect || PE de análise de malware e palestrante em diversas conferências nacionais e internacionais, tais como Blackhat, Defcon, Hack in The Box, XCon e Hackito.

Membro do comitê técnico de diversas conferências (Blackhat Brasil, Hackito e Nosuchcon, por exemplo) também foi palestrante principal (keynote) em eventos fora do Brasil e em território nacional. É organizador do evento Hackers to Hackers (H2HC), maior e mais antigo evento de pesquisas em segurança da informação na América Latina. Atuou em diversas empresas, tais como Check Point (como Chief Security Research) e Qualys (como Diretor de Pesquisas de Vulnerabilidades e Malware). Também é conselheiro do Instituto Coaliza.

Em 2011 foi homenageado pela Adobe como um dos contribuidores principais em vulnerabilidades nos produtos da empresa. Brasileiro convicto (apesar de ter morado em Israel, Dubai e atualmente nos Estados Unidos), é membro do Comitê Técnico da RENASIC, ligada ao Centro de Defesa Cibernética (CDCiber) do Departamento de Defesa Brasileiro.