ufpelogo

Universidade Federal de Pernambuco

Centro de Informática

Graduação em Ciência da Computação

**Um Esquema de Integração de Informações sobre Vulnerabilidades**

Luis Otávio Cavalcante Borba

Trabalho de Graduação

Recife, 22 de junho de 2009

Universidade Federal de Pernambuco

Centro de Informática

**Um Esquema de Integração de Informações sobre Vulnerabilidades**

Luis Otávio Cavalcante Borba

*Monografia apresentada ao Centro de Informática da Universidade Federal de Pernambuco, como requisito parcial para obtenção do Grau de Bacharel em Ciência da Computação.*

*Orientador: Djamel Fawzi Hadj Sadok*

*Co-orientador: Eduardo Luzeiro Feitosa*

Recife, 22 de junho de 2009

*“To live is the rarest thing in the world. Most people exist, that is all”*

**Oscar Wilde**

À minha estimada mãe, Jovanete.

**Agradecimentos**

Como é de costume, começo agradecendo a Deus por estar concluindo mais esta fase da minha vida. Deus é a força que nos move em direção aos bons objetivos. Por isso, agradeço a ele mais esta conquista.

Em segundo lugar, como não poderia ser diferente, agradeço à minha adorável mãe. Mãe solteira, batalhadora, criou-me ensinando princípios, ética, benevolência. Não tenho a menor dúvida que, de todos os meus professores, ela foi a mais sábia, a melhor!

Aos amigos que fiz na faculdade também agradeço. Com eles compartilhei experiências, horas e horas de projetos, conversas, risos etc.

Faço um agradecimento todo especial à minha namorada, Monaliza. Ela é a responsável por me acalmar em momentos de desespero, por arrancar um sorriso do meu rosto mesmo nas horas mais difíceis. Por essas e outras tenho certeza que ela é a mulher da minha vida.

Outra pessoa importante neste último período foi o Eduardo Feitosa. Sem sua co-orientação, este trabalho talvez não fosse possível. Sua ajuda foi muito além do que eu esperava. Portanto, obrigado, Eduardo.

Agradeço aos professores do centro de informática pelas aulas e projetos passados. Sem eles não seriamos nada.

Por fim, agradeço ao meu pai pelos gens, e só.

**Resumo**

Nos dias de hoje, problemas envolvendo tráfego não desejado gerado por tendências tecnológicas maliciosas em redes bem como novas aplicações e serviços têm afetado uma porção cada vez maior da Internet. Contudo, os problemas relacionados a violações de segurança causadas principalmente por vulnerabilidades em software e sistemas tem se mostrado maior a cada ano e com potencial para causar efeitos a nível global.

Uma vez que garantir a inexistência de vulnerabilidades em software e sistemas é praticamente impossível e que a quantidade de vulnerabilidades conhecidas aumenta diariamente, a melhor forma de evitar que tais vulnerabilidades venham a ter conseqüências catastróficas é manter seus utilizadores (administradores, gerentes e até mesmo usuários finais) informados sobre os acontecimentos. É neste contexto que o uso de bases de descoberta de vulnerabilidades tem surgido como principal solução para disseminação de informações sobre vulnerabilidades.

Apesar da importância das informações contidas nessas bases, uma característica facilmente percebida em consultas realizadas é a questão da incompletude. Nem sempre os resultados obtidos apresentam todas as informações necessárias. Por isso, este trabalho propõe um esquema de atualização automática de uma base de vulnerabilidades, capaz de identificar informações ainda desconhecidas ou não informadas na base, procurá-las na Internet e aproveitá-las para suprir as faltas encontradas.

Sumário

[1. Introdução 10](#_Toc233669157)

[1.1. Motivação 11](#_Toc233669158)

[1.2. Objetivos 11](#_Toc233669159)

[1.3. Organização do Trabalho 12](#_Toc233669160)

[2. Background 13](#_Toc233669161)

[2.1. Padrões 13](#_Toc233669162)

[2.1.1. Common Vulnerabilities and Exposures (CVE) 13](#_Toc233669163)

[2.1.2. Bugtraq 14](#_Toc233669164)

[2.2. Bancos de Dados de Vulnerabilidades (BDV) 14](#_Toc233669165)

[2.2.1. National Vulnerability Database (NVD) 15](#_Toc233669166)

[2.2.2. Secunia Advisories 16](#_Toc233669167)

[2.2.3. Open-Source Vulnerability DataBase (OSVDB) 17](#_Toc233669168)

[2.2.4. Comparações 18](#_Toc233669169)

[2.3. Sites sobre Vulnerabilidades 19](#_Toc233669170)

[2.3.1. Shadowserver 19](#_Toc233669171)

[2.3.2. Cyclops 19](#_Toc233669172)

[2.3.3. ATLAS 20](#_Toc233669173)

[2.4. Justificativa do Trabalho 21](#_Toc233669174)

[3. Proposta 23](#_Toc233669175)

[3.1. Problemática 23](#_Toc233669176)

[3.2. Solução Proposta 24](#_Toc233669177)

[4. Implementação 26](#_Toc233669178)

[4.1. Ferramentas utilizadas 26](#_Toc233669179)

[4.2. Questões preliminares 26](#_Toc233669180)

[4.2.1. Esquema do Banco de Dados 26](#_Toc233669181)

[4.2.2. Método de Atualização 28](#_Toc233669182)

[4.3. Solução encontrada 28](#_Toc233669183)

[5. Avaliações e Resultados 31](#_Toc233669184)

[5.1. Ambiente de Teste 31](#_Toc233669185)

[5.2. Avaliação inicial do OSVDB 31](#_Toc233669186)

[5.3. Rastreamento de Vulnerabilidades Incompletas 32](#_Toc233669187)

[5.4. Tipos de ataques 33](#_Toc233669188)

[5.5. Resultados 34](#_Toc233669189)

[6. Conclusões 38](#_Toc233669190)

[6.1. Dificuldades Encontradas 38](#_Toc233669191)

[6.2. Trabalhos Futuros 38](#_Toc233669192)

[7. Referências 39](#_Toc233669193)

**Índice de figuras**

[Figura 1. Vulnerabilidades reportadas ao CERT de 1995 a 2007 10](#_Toc233669194)

[Figura 2. Fluxo de atividades do Shadowserver 19](#_Toc233669195)

[Figura 3. Alerta do Cyclops 20](#_Toc233669196)

[Figura 4. Ranking de ataques da Internet 21](#_Toc233669197)

[Figura 5. OSVDB exibindo informações incompletas 23](#_Toc233669198)

[Figura 6. Secunia exibindo informações faltantes no OSVDB 24](#_Toc233669199)

[Figura 7. Etapas da solução 25](#_Toc233669200)

[Figura 8. Esquema do OSVDB 27](#_Toc233669201)

[Figura 9. Organização dos bancos de dados 28](#_Toc233669202)

[Figura 10. Esquema do banco de dados que armazena dados das referências externas 29](#_Toc233669203)

[Figura 11. Gráfico mostrando a reincidência das vulnerabilidades. 33](#_Toc233669204)

[Figura 12. Gráfico de pizza exibindo os tipos de ataques. 33](#_Toc233669205)

[Figura 13. Vulnerabilidade 55148 sem informações externas. 34](#_Toc233669206)

[Figura 14. Vulnerabilidade 55148 com informações externas. 35](#_Toc233669207)

[Figura 15. Vulnerabilidade 55229 sem informações externas 36](#_Toc233669208)

[Figura 16. Vulnerabilidade 55229 com informações externas 37](#_Toc233669209)

# Introdução

Nos dias de hoje, problemas envolvendo tráfego não desejado gerado por tendências tecnológicas maliciosas em redes bem como novas aplicações e serviços têm afetado uma porção cada vez maior da Internet. Contudo, os problemas relacionados a violações de segurança causadas principalmente por vulnerabilidades em software e sistemas tem se mostrado maior a cada ano e com potencial para causar efeitos a nível global.

Exemplos de perdas financeiras podem ser encontrados em todo o mundo. Em 2006, os prejuízos ocasionados por vulnerabilidades foram de aproximadamente US$ 245 milhões, somente entre provedores de acesso norte-americanos (Morin, 2006). O instituto de segurança americano CSI (*Computer Security Institute*) (Richardson, 2007), depois de entrevistar 194 empresas nos Estados Unidos, contabilizou perdas superiores a US$ 66 milhões ocasionadas pelo tráfego não produtivo gerado principalmente por vulnerabilidades em 2007.

Por definição, uma vulnerabilidade é um potencial ponto de ataque ou acesso dentro de um sistema computacional, podendo ocasionar perdas consideráveis e graves incidentes de segurança (MA et al., 2001). Estatísticas fornecidas pelo CERT (*Computer Emergency Readiness Team*) mostram que até o terceiro trimestre de 2008 haviam sido registrados 6058 casos de vulnerabilidades contra 262 casos em 1998, aumento de aproximadamente 2800% em 10 anos (CERT, 2009).

Figura 1. Vulnerabilidades reportadas ao CERT de 1995 a 2007

Uma vez que garantir a inexistência de vulnerabilidades em software e sistemas é praticamente impossível e que a quantidade de vulnerabilidades conhecidas aumenta diariamente, a melhor forma de evitar que tais vulnerabilidades venham a ter conseqüências catastróficas é manter seus utilizadores (administradores, gerentes e até mesmo usuários finais) informados sobre os acontecimentos. É neste contexto que o uso de bases de descoberta de vulnerabilidades têm surgido como principal solução para disseminação de informações sobre vulnerabilidades. Segundo (MEUNIER; SPAFFORD, 1999), o compartilhamento de informações sobre vulnerabilidades pode auxiliar na construção de softwares de detecção de intrusão e ferramentas de rastreamento de vulnerabilidades mais precisas.

Atualmente existem dezenas de bancos de dados de vulnerabilidades mantidos de forma privada como, por exemplo, o VulDa (IBM, 2009) e *Cisco Security Center* (CISCO, 2009) e bancos públicos (gratuitos e de uso geral) como o *National Vulnerability Database* (NVD, 2009), Secunia (Secunia, 2009) e o *Open Source Vulnerability DataBase* (OSVDB, 2009). As informações nessas bases são colhidas a partir de relatos de vulnerabilidades feitos pelos fabricantes e/ou pelos próprios usuários.

## Motivação

Apesar da importância das informações contidas nessas bases, uma característica facilmente percebida em consultas realizadas é a questão da incompletude. Nem sempre os resultados obtidos apresentam todas as informações necessárias. Normalmente, tal fato acontece por três motivos. O primeiro está relacionado com o processo de aceitação de uma vulnerabilidade. Basicamente, para que um relato de vulnerabilidade seja aceito realmente como uma vulnerabilidade, uma série de estágios deve ser completada antes de ser considerada uma vulnerabilidade. Em outras palavras, uma vulnerabilidade só é aceita quando sua veracidade é comprovada.

O segundo motivo é a o processo de atualização. Uma vez que tipicamente a atualização das informações sobre vulnerabilidades depende dos responsáveis pela administração da base (moderadores), é possível que uma vulnerabilidade comprovada e até possivelmente resolvida continue a ser exibida com incompleta.

Por fim, o último motivo refere-se ao fato de que apesar de existirem padrões para divulgação de vulnerabilidades como, por exemplo, o CVE (*Common Vulnerabities and Exposures*) (CVE, 2009), as atuais bases de vulnerabilidade não possuem mecanismos efetivos de interação. De modo geral, não existe qualquer mecanismo automático de troca de informações entre essas bases. Normalmente, a ligação entre uma vulnerabilidade de uma base com outra é estabelecida por um link web.

Uma vez que estes motivos contribuem para a incompletude das informações sobre vulnerabilidades, prejudicando os usuários, percebe-se a existência de um interessante tema de pesquisa, um sistema com acesso direto a uma de base de vulnerabilidades capaz de identificar a inexistência de alguma informação e de preenchê-la, diminuindo, assim, a falta de informações sobre tal vulnerabilidade. Tal sistema desempenharia essa função ou buscando em referências externas a base ou empregando técnicas de mineração na Internet.

## Objetivos

Este trabalho propõe um esquema de atualização automática de uma base de vulnerabilidades, capaz de identificar informações ainda desconhecidas ou não informadas, procurá-las na Internet e aproveitá-las para suprir as “falhas” encontradas.

Especificamente, pretende-se:

* Avaliar e escolher uma das diversas bases de vulnerabilidade gratuitas como base padrão;
* Desenvolver uma ferramenta tipo um *crawler* para procurar na Internet informações que possam complementar as existentes na base escolhida, estendendo-a se for necessário para acomodar outras informações úteis;
* Disponibilizar acesso a essa base “atualizada” para a comunidade.

## Organização do Trabalho

O trabalho está organizado em sete capítulos. No primeiro capítulo há uma breve introdução sobre o problema e a motivação para o trabalho proposto. O segundo capítulo apresenta trabalhos relacionados com vulnerabilidades e bancos de dados, fazendo comparações entre sistemas e justificando a necessidade do trabalho desenvolvido. O terceiro capítulo apresenta em detalhes o problema tratado mostra como será resolvido. O capítulo quarto capítulo mostra como a solução foi implementada e como os problemas foram tratados. O capítulo seguinte faz algumas avaliações da implementação e exibe os resultados encontrados. O sexto capítulo apresenta as conclusões do trabalho, as dificuldades encontradas e os trabalhos futuros. No último capítulo, estão elencadas as referências do trabalho.

# Background

Atualmente, existem diversos iniciativas na área de segurança que visam lidar com vulnerabilidades, abrangendo desde padrões para definição e especificação de informações sobre vulnerabilidades, bases de armazenamento dessas informações até sites que disponibilizam diferentes tipos de dados sobre ataques, invasões e vulnerabilidades.

Este capítulo descreve alguns destes projetos, objetivando apresentar um visão mais abrangente do tema abordado e também justificando a importância do sistema proposto nesta monografia.

## Padrões

Um padrão é um documento, estabelecido por consenso e aprovado por uma reconhecida organização, que provê, para uso comum e repetitivo, regras e orientações ou características para atividades ou seus resultados, objetivando a realização de um grau ótimo de ordem num dado contexto (ISO, 2009). Assim, um padrão provê interoperabilidade, permitindo que informações sejam trocadas facilmente e de forma exata (GORDON, 2003).

### Common Vulnerabilities and Exposures (CVE)

O CVE, proposto em 1999 por (MEUNIER; SPAFFORD, 1999), surgiu da necessidade de padronizar os relatos de vulnerabilidades como forma de facilitar o compartilhamento de dados e a comparação entre os diversos bancos de informação sobre vulnerabilidades existentes. Basicamente, o problema encontrado era o uso do nome da vulnerabilidade como índice de pesquisa nessas bases. Assim, era muito difícil determinar quando diferentes ferramentas se referiam a mesma vulnerabilidade e, portanto, a troca de informações ficava impossibilitada.

O uso do CVE se revela importante porque pode aumentar a efetividade dos sistemas de segurança, já que permite a busca de informações em várias fontes sem a necessidade de se saber o nome dado à vulnerabilidade em cada base de dados, facilitando o intercâmbio de informações, além de aumentar o grau de padronização dado às vulnerabilidades.

No padrão CVE, um relato de vulnerabilidade passa por um processo para verificar a veracidade das informações e recolher mais fatos sobre a vulnerabilidade. Esse processo tem as seguintes fases:

* **Descobrimento:** o potencial ponto de exposição é descoberto.
* **Publicação:** a informação é anunciada publicamente como uma candidata a vulnerabilidade.
* **Identificação:** a candidata é verifica para se saber se já não é conhecida, evitando duplicações. Caso não seja, é numerada com um identificador chamado “CAN”.
* **Proposta:** a candidata é discutida por um grupo formado por representantes de importantes organizações como CISCO, Symantec, SUN, Microsoft e pesquisadores, entre outros. A decisão do grupo pode ser pela aceitação, rejeição, modificação, além de poder não opinar ou pedir mais tempo para tomar a decisão.

Após esse conjunto de estágios, há mais discussões e decisões até que se chegue a um consenso e à publicação das decisões (ROHSE, 2003). Após um relato ser aceito, o resultado é uma entrada que consiste um nome, descrição, referências e o identificador padrão do CVE.

O identificador padrão do CVE tem o formato “CVE-aaaa-nnnn”, onde as quatro letras ‘a’ representam o ano de publicação da vulnerabilidade e as letras ‘n’ são usadas para numerar as vulnerabilidades ordenadamente. Por exemplo, a vulnerabilidade identificada como:

* **CVE-2009-0001:** foi a primeira vulnerabilidade adicionada à lista de vulnerabilidades do CVE em 2009.
* **CVE-2003-1272:** foi a milésima ducentésima septuagésima segunda vulnerabilidade adicionada à lista do CVE em 2003.

### Bugtraq

O Bugtraq é uma fonte de informações sobre vulnerabilidades composta por listas de discussão e bases de dados, cujo objetivo é promover a discussão e catalogação de vulnerabilidades de segurança. Adota uma política de “revelação total”, onde os mecanismos de exploração das vulnerabilidades são revelados em detalhes, o que possibilita que qualquer um possa verificar o que foi relatado. Essa filosofia gera debates, pois da mesma forma que essas informações detalhadas servem para os bem intencionados, serve para aqueles que querem explorar as falhas em sistemas (ROHSE, 2003).

O Bugtraq tem seu funcionamento bastante parecido com o CVE. Possui seu próprio padrão de nomenclatura e disponibiliza um mapeamento do seu padrão para o CVE. Além disso, assim como o CVE, também disponibiliza informações sobre as vulnerabilidades de forma sucinta.

Entretanto, o processo de relato de vulnerabilidade é mais simples do que no CVE. A pessoa que reporta a vulnerabilidade é orientada a comunicar ao responsável pelo software ou sistema e esperar um determinado período de tempo para que uma solução seja definida ou um parecer seja relatado. Se após este tempo, o responsável não se expressar, as informações são publicadas e discutidas pelos membros do Bugtraq. A corretude das informações é verificada pela *Security Focus*, companhia que mantém e modera o conteúdo do Bugtraq (SECURITY FOCUS, 2009).

Apesar de ser um padrão para identificação de vulnerabilidades, o Bugtraq não conseguiu tornar-se um padrão de fato.

## Bancos de Dados de Vulnerabilidades (BDV)

Os bancos de dados de vulnerabilidades começaram ser discutidos a partir de 2000 através da preocupação da comunidade acadêmica em compartilhar informações sobre vulnerabilidades de softwares (Schumacher, 2000), uma vez que até aquele momento as vulnerabilidades só eram discutidas em grupos fechados e listas de e-mail.

Esta seção apresenta alguns bancos de dados de destaque no atual cenário, informando suas características, informações armazenadas e, por fim, comparando-os.

### National Vulnerability Database (NVD)

O NVD (*National Vulnerability Database*) é o repositório de vulnerabilidades do governo americano para gerenciamento de vulnerabilidades e medição de segurança. É gerenciado pelo *National Institute of Standards and Technology* (NIST) e patrocinado pelo *Department of Homeland Security* (DHS) (NVD, 2009).

O NVD permite o gerenciamento automático de vulnerabilidades, integra todas as informações publicas do governo e provê referências para fontes externas. É baseado e sincronizado com o CVE e utiliza o mesmo padrão de nomes adotado por ele. Além disso, fornece sumários para todas as vulnerabilidades contidas no CVE, ligações (links) para outros sistemas, corretivos (*patches*) e outras fontes de informações relacionadas a vulnerabilidades. O NVD também provê um escore da periculosidade de cada vulnerabilidade chamado de *Common Vulnerability Scoring System* (CVSS). Os dados presentes em seus registros provêem mecanismos de análise quantitativa de vulnerabilidades com baixo nível de risco (NVD, 2009).

Atualmente, o NVD possui mais de trinta e sete mil vulnerabilidades registradas, com média diária de vinte novas vulnerabilidades inseridas, além de conter milhares de vendedores de produtos cadastrados em seu banco.

A identificação das vulnerabilidades no NVD é feita através de identificadores (nomes) atribuídos pelo CVE a ela. A tabela 1 apresenta as informações de vulnerabilidade fornecidas pelo NVD.

|  |  |
| --- | --- |
| Indicador | Descrição |
| Overview | Apresenta uma visão geral da vulnerabilidade. Normalmente informa como explorar a brecha de segurança |
| CVSS Base Score | Representa um escore muito abrangente calculado a partir dos valores de um vetor de outros escores. Permite mensurar quantitativamente uma vulnerabilidade |
| Impact Subscore | Medição quantitativa do grau de impacto pela exploração da vulnerabilidade |
| Exploitability Subscore | Mede o potencial de exploração da vulnerabilidade |
| Access Vector | Indica o vetor de acesso à vulnerabilidade. Pode ser através de acesso remoto, local e ambos |
| Access Complexity | Mensura o nível de dificuldade que o atacante tem para efetuar a exploração da falha de segurança |
| Authentication | Indica se é necessário estar autenticado no sistema vulnerável para explorá-lo |
| Impact Type | Elenca os tipos de impactos causados pela exploração |
| External Source | Exibe referências externas que disponibilizam informações sobre a mesma vulnerabilidade. Entre as informações está o nome da referência externa (nome do sistema ou site) |
| Vulnerable Software and Versions | Elenca os softwares e suas versões afetadas pela vulnerabilidade |
| Vulnerability Type | Indica o tipo de vulnerabilidade. Alguns valores possíveis são: *authentication assues, credentials management, access control, code injection*, entre outros |

O NVD é um dos maiores bancos de dados da atualidade, possui um conjunto de informações relevantes e destaca-se pela procedência das informações armazenadas. Seu sistema de escore é bastante completo e diferenciado, pois leva um grande conjunto de variáveis em consideração enquanto outros sistemas, apesar de apresentarem escores para o potencial de risco trazido pela vulnerabilidade, não deixam claro quais variáveis são levadas em conta nessa medição.

### Secunia Advisories

Secunia é uma empresa de serviços de segurança da informação que disponibiliza um banco de dados de informações sobre vulnerabilidades (SECUNIA, 2009). O *Secunia Advisories* armazena vulnerabilidades de todos os tipos. Para identificar as entradas, o Secunia utiliza seu próprio padrão de identificação, diferentemente do NVD, que usa o padrão CVE. Juntamente com as informações, fornece um serviço de alerta de vulnerabilidades relevantes.

A tabela 2 apresenta as informações constantes na base do Secunia.

|  |  |
| --- | --- |
| Indicador | Descrição |
| Description | Descrição textual da vulnerabilidade indicando o produto afetado, de que forma a falha pode ser explorada, as conseqüências e, por vezes, algumas informações detalhadas de como a exploração é feita |
| Critical | Informa o quão critica é a vulnerabilidade através de uma escala de risco. Difere do escore informado pelo NVD por se tratar de uma informação qualitativa e não quantitativa. Assim, uma vulnerabilidade pode ser classificada como extremamente crítica, muito crítica, moderadamente crítica, pouco crítica ou não crítica |
| Impact | Refere às conseqüências causadas pela exploração da vulnerabilidade. As conseqüências podem ser: *cross-site scripting, denial of Service, exposure of sensitive information, hijacking*,entre outras |
| Popularity | Indica a quantidade de visualizações da vulnerabilidade, ou seja, quantas vezes a vulnerabilidade foi visualizada pelos usuários |
| Where | Indica se a exploração da vulnerabilidade é feita remotamente, localmente ou se das duas formas é possível explorá-la |
| Solution Status | Indica se a solução para a falha já foi disponibilizada pelo vendedor do produto |
| Softwares | Lista os produtos e as versões afetadas pela vulnerabilidade |
| OS | Lista os sistemas operacionais em que a vulnerabilidade pode ser explorada. Esta informação pode ser omitida em várias vulnerabilidades quando o sistema operacional não influencia na exploração |

O Secunia Advisories possui um conjunto pequeno, porém relevante de informações sobre vulnerabilidades. Apresenta continua e rápida expansão. Por dia, são adicionadas dezenas de vulnerabilidades, além de atualizações feitas nas que já estão catalogadas. Para popular seu banco, o Secunia além de receber relatos, também faz pesquisas em softwares utilizando engenharia reversa para descobrir vulnerabilidades (SECUNIA, 2009).

A identificação das vulnerabilidades é feita por um padrão próprio. O padrão consiste nas letras “SA” mais uma numeração. À medida que novas vulnerabilidades vão sendo adicionadas à base esse número vai sendo incrementado de uma unidade. O Secunia possui cerca de trinta e cinco mil entradas.

### Open-Source Vulnerability DataBase (OSVDB)

O OSVDB (*Open-Source Vulnerability Database*) é um projeto que mantém uma lista de vulnerabilidades de segurança de computadores disponível gratuitamente para uso de profissionais de segurança, e qualquer tipo de produto de segurança que necessite de um banco de dados de vulnerabilidades. Foi criado em 2002 com a idéia de manter um banco de dados de vulnerabilidades totalmente independente de interesses particulares e operado pela própria comunidade.

A tabela 3 apresenta as informações constantes na base do OSVDB.

|  |  |
| --- | --- |
| Indicador | Descrição |
| Description | Descrição textual da vulnerabilidade indicando em que componente do produto ela está localizada, como pode ser explorada e o que pode ser afetado pela exploração. |
| Solution | Descreve de que forma a vulnerabilidade pode ser solucionada. |
| Location | Indica de que localização é possível explorar a vulnerabilidade. Por exemplo, uma vulnerabilidade pode ser explorável via celular, através de acesso físico, remotamente, etc. |
| Attack Type | Indica os tipos de ataques possíveis pela exploração da vulnerabilidade. |
| Impact | Indica o tipo de impacto que a exploração da vulnerabilidade causa. Pode ser a perda da confidencialidade, integridade, disponibilidade etc. |
| Exploit | Informa se existem explorações conhecidas para a vulnerabilidade |
| External References | Exibe uma lista de referências externas que podem ser consultadas para se conseguir mais informações sobre a vulnerabilidade |
| Products | Lista os produtos reconhecidamente afetados pela vulnerabilidade. |

O OSVDB é um banco com grande volume de vulnerabilidades cadastradas. Atualmente, cerca de cinqüenta e três mil entradas fazem parte de seus registros. Esse número de vulnerabilidades cadastradas coloca o OSVDB entre um dos maiores bancos de vulnerabilidades do mundo, se não o maior. Para identificar as vulnerabilidades, o OSVDB usa seu próprio padrão. O padrão é chamado OSVDB ID. Consiste apenas em números que vão sendo incrementados de uma unidade à medida que novas vulnerabilidades são adicionadas à base.

### Comparações

A tabela abaixo sumariza algumas características sobre os bancos analisados.

|  |  |  |  |
| --- | --- | --- | --- |
| Característica | NVD | Secunia | OSVDB |
| Quantidade armazenada | + 37000 | + 35000 | + 54000 |
| Completude das informações | Alta | Alta | Média |
| Quantidade de informações por vulnerabilidade | Média | Média | Alta |
| Agilidade na publicação | Baixa | Desconhecida | Alta |
| Aberto à comunidade | Não | Não | Sim |

A primeira característica, quantidade armazenada, indica a quantidade de vulnerabilidades presente em cada banco. Como se pode ver, o OSVDB é o banco com maior número de vulnerabilidades registradas, indicando maior abrangência.

A completude das informações indica se todas as informações sobre uma vulnerabilidade estão presentes no banco ou se há informações ainda não preenchidas sobre uma vulnerabilidade. O Secunia e o NVD possuem alta taxa de completude de seus registros. Já no OSVDB é não é raro encontrar campos vazios.

A quantidade de informações por vulnerabilidade indica a quantidade de campos que os registros de vulnerabilidades possuem. O Secunia e NVD armazenam apenas as informações mais importantes da vulnerabilidade. Já o OSVDB, possui um conjunto mais abrangente de informações sobre cada vulnerabilidade. Informações como quem descobriu a falha, quando foi, análises sobre a falha, listas de discussão discutindo a falha etc. Assim, o OSVDB disponibiliza mais informações sobre as vulnerabilidades.

A agilidade na publicação trata do tempo decorrido entre o momento que a organização que mantém esses bancos toma conhecimento da vulnerabilidade e o momento que publica este fato. No NVD este processo demora em média seis meses, o OSVDB disponibiliza rapidamente o relato, mesmo que não haja confirmação da vulnerabilidade. No Secunia não foi possível descobrir essa informação.

Dos bancos analisados, apenas o OSVDB é aberto à participação da comunidade. O Secunia é mantido por capital privado e o NVD pelo governo americano através do NIST.

## Sites sobre Vulnerabilidades

### Shadowserver

A Shadowserver é uma organização formada por profissionais da área de segurança que, voluntariamente, reúnem, rastreiam e reportam informações da atividade de *botnets* e *malwares* em geral, além de qualquer espécie de fraude eletrônica. Sua missão é aumentar a segurança da Internet. As atividades desenvolvidas pelo grupo envolvem a captura, análise, monitoramento e disseminação de softwares maliciosos, ataques, *botnets[[1]](#footnote-2)*, vírus[[2]](#footnote-3), falhas de sistemas e malwares[[3]](#footnote-4) em geral (Figura 2).

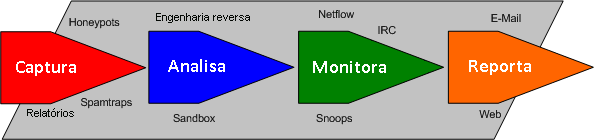
****

Figura 2. Fluxo de atividades do Shadowserver

De modo geral, as atividades começam com o recebimento de relatos sobre softwares maliciosos ou de dispositivos comprometidos. A análise do relato ocorre na próxima fase. É utilizada uma técnica de engenharia reversa chamada *sandbox[[4]](#footnote-5)* para analisar o código do software comprometido. O terceiro estágio é o de monitoramento (se o caso for de ataques maliciosos e brechas de segurança) ou de rastreamento (se o caso for de atividade de *botnets*). Após essas etapas, as informações são relatadas via web e e-mail. O Shadowserver trabalha com outras entidades de segurança, com as quais desenvolve estratégias contra ameaças e planeja formas de mitigar suas conseqüências.

O Shadowserver mantém uma base de conhecimento sobre *botnets*, *honeypots[[5]](#footnote-6)*, fraudes eletrônicas e *malwares*. Nessas bases, qualquer usuário da Internet pode se informar sobre a definição desses elementos, sua forma de propagação, os mecanismos utilizados pelas pessoas mal intencionadas para controlá-los, os tipos de ataques que podem ser feitos, técnicas usadas para detectar e remover tais elementos e formas de mitigar os estragos causados.

### Cyclops

O Cyclops é uma ferramenta de auditoria permite detectar várias formas de ataques do tipo *route hijack[[6]](#footnote-7)*, além de outras anomalias no roteamento de pacotes das redes. Basicamente, o Cyclops é uma ferramenta que utiliza um banco de dados de rotas da internet para fins de segurança, detectando ataques e anomalias das rotas para redes.

O Cyclops integra dados recolhidos de sistemas que armazenam rotas para diversas redes em todo o mundo. As rotas armazenadas em sistemas como o *RouteViews[[7]](#footnote-8)*, *RIPE-RIS[[8]](#footnote-9)*, *Abilene[[9]](#footnote-10)*, *Packet Clearing House[[10]](#footnote-11)* e *Bgpmon[[11]](#footnote-12)* são colhidas e processadas para posterior uso da ferramenta. Além dos propósitos de segurança, o Cyclops utiliza as informações das rotas para exibir os vários caminhos possíveis para se chegar a uma dada rede (Oliveira, 2009). Quando detecta algum ataque ou anomalia, o sistema gera alertas para seus usuários fornecendo sobre o evento.

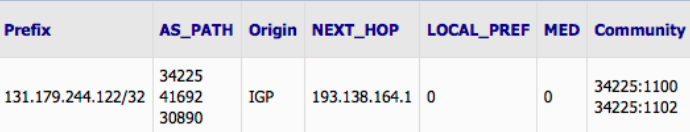


Figura 3. Alerta do Cyclops

A figura acima mostra um alerta gerado pelo Cyclops ao detectar que o prefixo 131.179.244.122/32 repentinamente começou a ser anunciado por um domínio diferente do normal. Na ocasião, um provedor de Internet da Romênia estava sendo atacado.

### ATLAS

ATLAS (*Active Threat Level Analysis Network System*) é um sistema de análise de ameaças em redes de escopo global, desenvolvido e mantido pela empresa *Arbor Networks* (Arbor Networks, 2009). O ATLAS disponibiliza informações sobre uma ampla gama de ataques, atividades de port/host scanning, vulnerabilidades em estado “*zero-day”* (desconhecidas), propagação de exploits e worms e outras variantes de eventos de segurança. Além disso, o ATLAS também é capaz de rastrear *bots* e *botnets*. O acesso a essa variada gama de informações é feita por um portal público.

O portal ATLAS disponibiliza informações de segurança que incluem:

* **Mapa global de ameaças:** visualização em tempo real da propagação global de ameaças.
* **Resumos de ameaças:** uma síntese dos mais significantes eventos de segurança que ocorreram nas últimas 24 horas.
* **Ranking de fontes de ameaças:** visualização multidimensional das origens dos ataques.
* **Índice de ameaças:** resumo das atividades maliciosas na Internet com detalhada classificação de ameaças.
* **Ranking de ataques da Internet:** visualização dos ataques mais usados nas últimas 24 horas.
* **Índice de riscos de vulnerabilidades:** índice das vulnerabilidades mais perigosas que estão sendo exploradas atualmente.

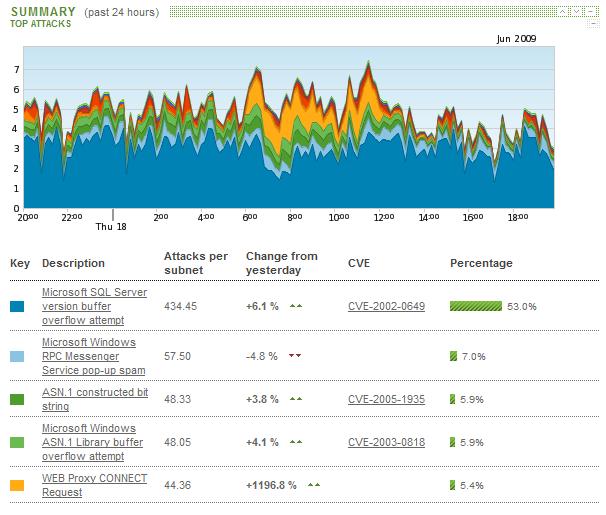


Figura 4. Ranking de ataques da Internet

Como exemplo, a figura acima exibe um gráfico gerado pelo ATLAS com os cinco ataques mais utilizados nas últimas vinte quatro horas. Junto com o gráfico, vem uma tabela com a descrição da vulnerabilidade, a quantidade de ataques por sub-rede a porcentagem de mudança no número de ataques ocorrida do dia anterior para o atual, o identificador CVE, se houver, e a porcentagem do ataque em relação a todos os ataques detectados pelo sistema.

O ATLAS tem como fonte de dados sensores *Arbor Peakflow*, produtos da própria empresa. Atualmente, esta plataforma é usada pela maioria dos ISPs (*Internet Service Provider*), cobrindo cerca de 70% de toda a Internet.

## Justificativa do Trabalho

Olhando para os trabalhos apresentados aqui, pode-se perceber o interesse que as várias empresas e organizações do ramo segurança têm por manter grandes conjuntos de informações e compartilhá-las. Focando na área de vulnerabilidades, o uso de bancos de dados de vulnerabilidades é justificado pela necessidade de se gerenciar, armazenar e disponibilizar de forma fácil informações sobre essas brechas de segurança.

Uma característica comum a todos os sistemas que fazem rastreamento, análise e descoberta de eventos de segurança é a necessidade de buscar informações em várias fontes para reuni-las e disponibilizar informações consistentes. Nesse contexto, o propósito de criar um banco de dados de vulnerabilidades abrangente (no sentido de armazenar todo tipo de vulnerabilidade) e completo (no sentido de possuir o máximo de informações possíveis sobre as vulnerabilidades) vai ao encontro dos anseios da comunidade de segurança. Tal banco contribuiria muito efetivamente para a segurança, pois, reunindo informações de várias fontes e disponibilizando-as, facilitaria a vida dos pesquisadores, que poderiam encontrar grande parte do que precisam em um só lugar.

# Proposta

Este capítulo tem como objetivo prover informações detalhadas sobre o OSVDB, o problema discutido neste trabalho e a solução proposta.

## Problemática

Como mencionado e apresentado nos capítulos anteriores, um fato comum a maioria dos BDV é a disponibilização de informações sobre possíveis vulnerabilidades o quanto antes. Contudo, tal processo tipicamente é caracterizado pela inserção de um conjunto pequeno de dados.

A Figura 5 ilustra o problema de incompletude da informação para o OSVDB.

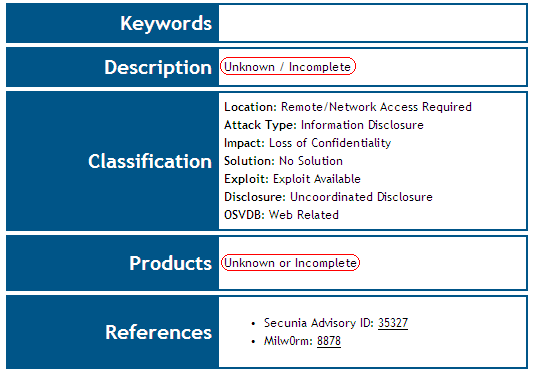


Figura 5. OSVDB exibindo informações incompletas

Uma vez que dispor de informações úteis é fundamental para detecção e correção de vulnerabilidades, a questão é como suprir a falta de informações dos BDV através de um processo automatizado?

A Figura 6 mostra o resultado de uma busca no banco de dados Secunia usando como índice a referência 35327 apresentada na Figura 5.



Figura 6. Secunia exibindo informações faltantes no OSVDB

Percebe-se rapidamente que as informações não apresentadas no OSVDB (software afetado e descrição) já constam na base do Secunia. Essa constatação pode ser considerada um bom ponto de partida para o desenvolvimento de uma solução que utiliza referências externas para extrair informações e complementar a base de vulnerabilidades.

## Solução Proposta

Desta forma, a solução proposta por este trabalho envolve seis etapas:

1. A identificação de informações incompletas ou não informadas de vulnerabilidades em um BVD (Figura 6 acima);
2. Para cada vulnerabilidade identificada nessa situação, buscar a lista de referências externas, ou seja, as ligações para as páginas web de outros sistemas;
3. O passo seguinte é fazer o *download* das páginas e guardá-las em sistema de arquivos;
4. Extração, através de um componente especialista, das informações relevantes das páginas e armazená-las em *templates* predefinidos;
5. Efetuar a união das informações dos *templates* num único objeto;
6. Por fim, armazenar as informações contidas nesse objeto no banco de dados.

A Figura 7 ilustra as etapas da solução proposta.

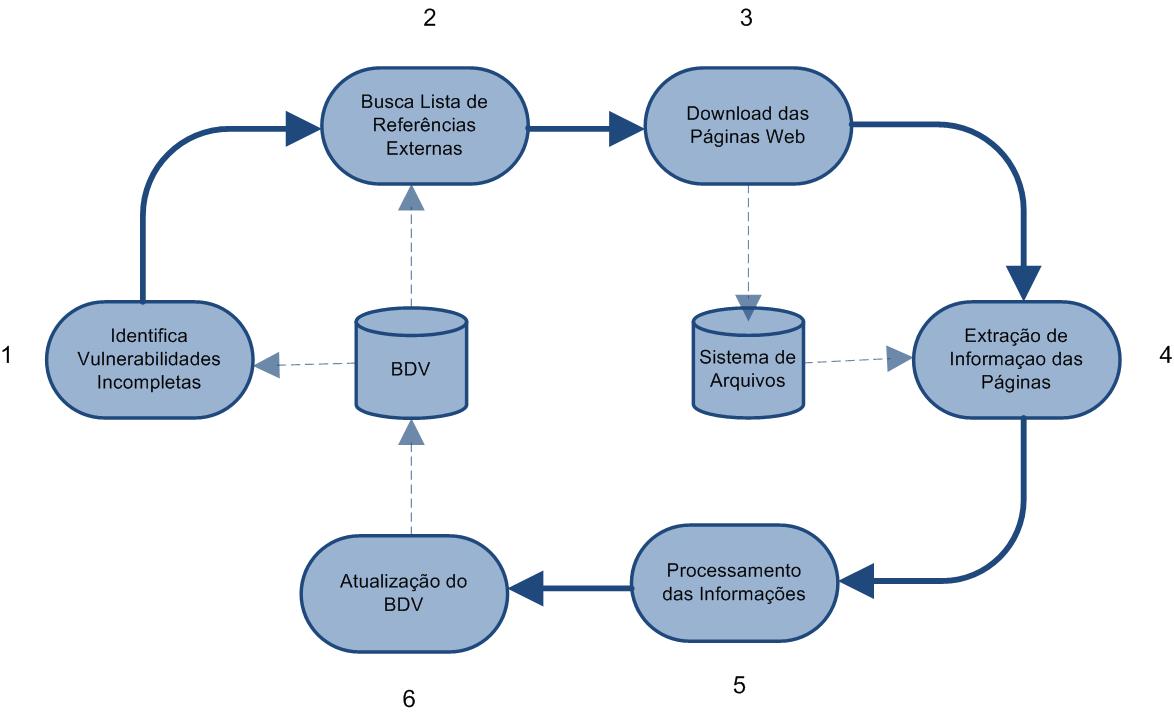


Figura 7. Etapas da solução

# Implementação

Este capítulo explica como a implementação do sistema foi feita, mostra o porquê de várias decisões de projeto e tece alguns comentários mais específicos sobre características do OSVDB que tiveram impacto na implementação. O capítulo apresenta a implementação do sistema dividindo-o em partes segundo o que foi ilustrado na figura 7.

## Ferramentas utilizadas

Para iniciar a implementação da solução proposta, um dos pontos principais da proposta é a definição de qual BDV utilizar como base de vulnerabilidade padrão. Usando a tabela apresentada na seção 2.2.4 como padrão para análise, o OSVDB foi escolhido devido a: (i) ser uma base de acesso fácil e gratuito; (2) utilizar fontes externas confiáveis como Secunia, NVD, ISS X-Force[[12]](#footnote-13) e *Security Focus[[13]](#footnote-14)*; (3) disponibilizar seu esquema de banco de dados livremente e em diversos formatos; (4) apresentar uma grande quantidade de vulnerabilidades. Conseqüentemente, para dar suporte ao OSVDB, o MySQL foi escolhido como SGBD.

Em relação à linguagem de programação, Java foi escolhida por apresentar um bom suporte ao banco MySQL e boa manipulação de arquivos. Adicionalmente, foi utilizado um *parser* HTML chamado de *htmlCleaner* - escrito em Java e com código aberto.

## Questões preliminares

Para a implementação da solução ilustrada na Figura 7, dois problemas iniciais precisam ser resolvidos, ambos relacionados ao OSVDB. O primeiro é a estrutura do banco de dados e o segundo é o método de atualização dos dados.

### Esquema do Banco de Dados

O esquema do OSVDB é organizado como mostra a Figura 8, onde *Vulnerabilities* é a tabela principal.

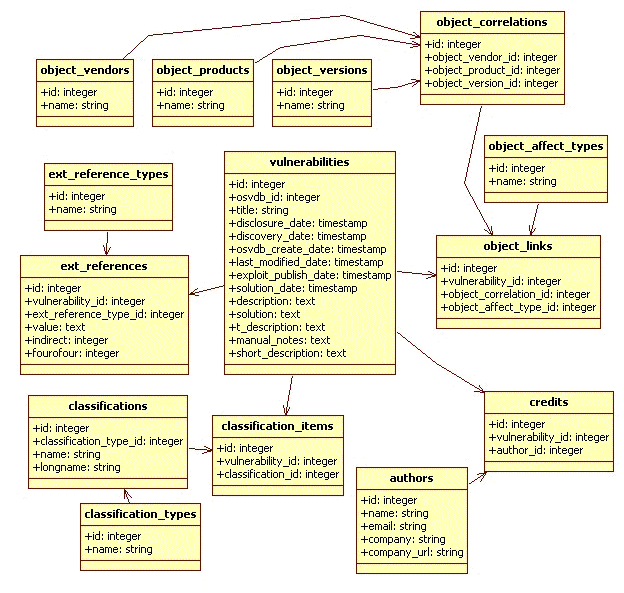


Figura 8. Esquema do OSVDB

Para melhorar o entendimento do problema, o esquema será dividido em quatro (4) segmentos. O primeiro representa as referências externas e engloba as tabelas *ext\_reference\_types* e *ext\_references*.

O segundo representa os produtos envolvidos com vulnerabilidades e envolve seis (6) tabelas que se relacionam para armazenar de forma bem estruturada os produtos afetados. Contudo, para um produto ser relacionado a uma vulnerabilidade é preciso: nome do produto (armazenado na tabela *object\_products*), versão do produto (tabela *object\_versions*) e vendedor do produto (tabela *object\_vendors*).

O problema dessa organização das tabelas é que, tipicamente, nem todas as referências externas disponibilizam todas as três informações. Algumas apenas colocam o nome do produto, outras, nome e versão. Como fator complicador, alguns descrevem individualmente as versões afetadas (por exemplo, Windows XP, versão SP1, SP2 e SP3). Outros se referem a famílias inteiras de versões com uma só numeração (por exemplo, Windows XP SPx).

O terceiro segmento representa o descobridor da vulnerabilidade através das tabelas *authors* e *credits*. Esta organização é extremamente funcional para o OSVDB, uma vez que evita a duplicação de nomes. O problema é que fontes diferentes podem nomear a mesma pessoa de formas diferentes. Por exemplo, Jonh Wendell Smith não é igual à Jonh W. Smith.

O quarto e último segmento representa a função de classificação e é formado pelas tabelas *classification\_types*, *classifications* e *classification\_items*. A primeira peculiaridade deste segmento é que as classificações do OSVDB já são armazenadas em suas tabelas. A tabela *classification\_types* possui os tipos de classificações que o OSVDB faz. A tabela *classifications* possui os valores possíveis para cada tipo de classificação. Por exemplo, para o tipo de classificação *Attack Type*, que está em *classification\_types,* há os valores *Denial of Service, Hijacking* e *Input Manipulation*, na tabela *classifications*.

É óbvio que esta predefinição de valores é inviável para atualizações externas. Por exemplo, no OSVDB, uma vulnerabilidade explorada remotamente é “*Remote/Network Access Required*” enquanto que no Secunia é “*From Remote*”.

### Método de Atualização

O outro problema encontrado não está diretamente no banco de dados do OSVDB, mas sim na forma utilizada para disponibilizar suas atualizações diárias. Ao invés de disponibilizar somente as informações inseridas e/ou modificadas no último dia, o banco completo é disponibilizado todos os dias.

Tal fato torna o processo de atualização mais demorado a cada dia e faz com que qualquer alteração feita no banco (através da busca de informações nas referências externas, por exemplo) seja apagada.

## Solução encontrada

Visto que os dois problemas discutidos acima impossibilitam a construção de mapeamentos e a manutenção das informações atualizadas, a solução encontrada e sem dúvida mais viável é a criação de uma base auxiliar para armazenar informações extras ao OSVDB.

A Figura 9 apresenta o esquema proposto para o armazenamento de informações.

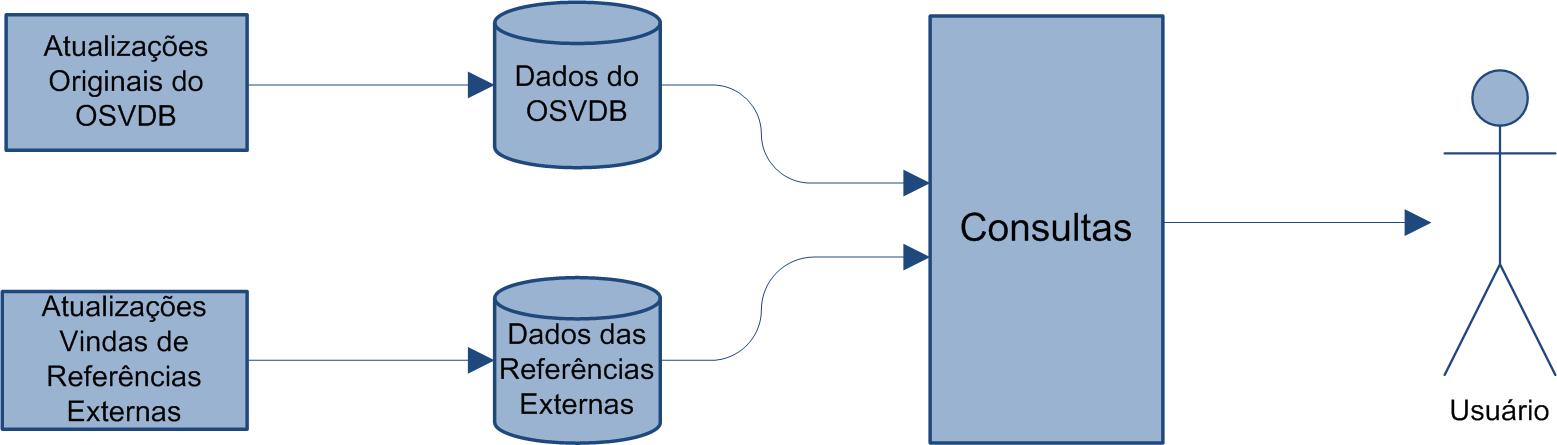


Figura 9. Organização dos bancos de dados

Este esquema prevê a utilização de dois bancos de dados, um contendo o esquema original do OSVDB (Figura 8), outro com um esquema para armazenar os dados extraídos das referências externas. A Figura 10 ilustra o esquema desse banco de dados proposto.

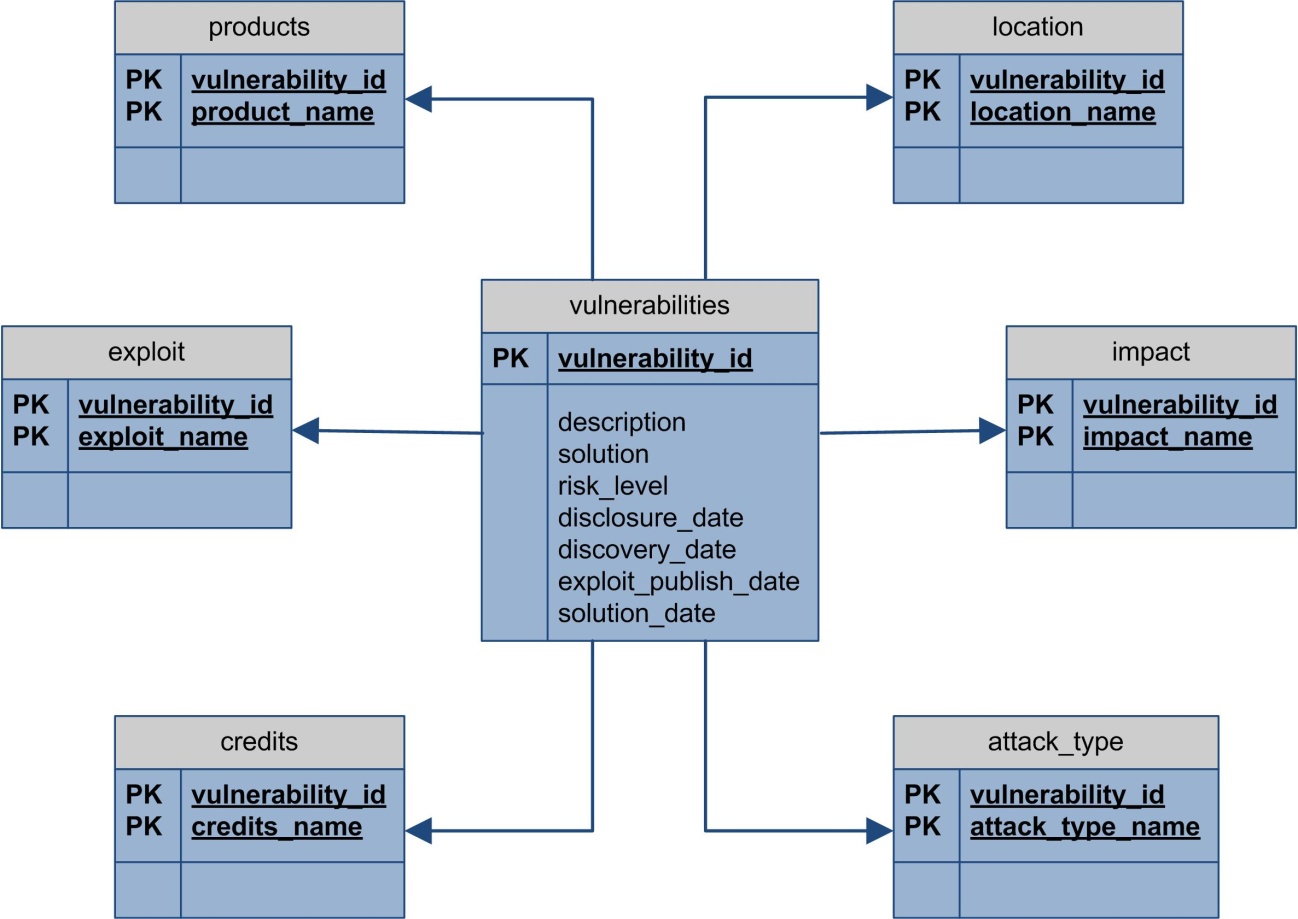


Figura 10. Esquema do banco de dados que armazena dados das referências externas

Comparando os dois esquemas, OSVDB e o proposto, percebe-se que a tabela *vulnerabilities* possui uma quantidade bem menor de campos, bem como as outras tabelas relacionadas a ela.

A tabela *vulnerabilities* é menor porque algumas das informações que o OSVDB armazena em seu esquema original dizem respeito ao próprio projeto e outras informações só são encontradas nesse banco de dados. Qualquer informação que se enquadre nesse perfil não poderá ser encontrada nas fontes externas, portanto não há necessidade do esquema proposto possuir tais campos. É o caso do campo *osvdb\_create\_date*. Entretanto, o campo *risk\_level*, que representa o grau de periculosidade da vulnerabilidade, foi inserido, uma vez que é encontrada em diversas fontes, mas não está presente no OSVDB.

Já os ramos créditos e objetos foram remodelados para conter apenas uma tabela e armazenar valores diretamente. Assim, se a vulnerabilidade *v* atinge os produtos *x* e *y*, haverá duas entradas na tabela *products* com as tuplas (v, x) e (v, y). No esquema do OSVDB há uma complexidade muito maior nesse ramo. Nele, existe uma tabela exclusiva para nome de produtos, outra para vendedores, outra para versões e uma que faz o relacionamento entre essas três. Esta última (que relaciona as três primeiras) ainda se liga a uma quinta tabela que faz o relacionamento com a vulnerabilidade e com outra tabela.

O ramo classificações, que fazia vários tipos de classificações foi desmembrado em cada um desses tipos, sendo aproveitado apenas os tipos de classificações relativos a *impact, attack\_type, exploit* e *location.* Os outros (*OSVDB, Solution,* e *disclosure*) não foram aproveitados, pois nenhuma das referências externas possui as classificações.

O uso de um banco de dados paralelo ao OSVDB é uma abordagem que resolve os dois problemas discutidos. Por todas as informações não serem armazenadas diretamente no banco do OSVDB, o problema das atualizações originadas do grupo é resolvido, pois elas serão carregadas normalmente em sua base sem apagar as informações extraídas das fontes externas, já que estas estão no outro banco. Os problemas da estrutura do OSVDB que leva à necessidade de se ter as informações bastante quebradas e organizadas, bem como o fato das classificações do OSVDB terem valores padrões já armazenados no banco também são resolvidos com o uso do banco paralelo, pois este contém um esquema mais simples e que permite armazenar diretamente as informações extraídas da fonte.

# Avaliações e Resultados

Este capítulo mostra como foi o ambiente em que o esquema construído foi testado, faz algumas avaliações do mesmo e exibe os resultados conseguidos.

## Ambiente de Teste

Para realizar os testes do esquema foi utilizado um computador portátil (*laptop*) do autor desta monografia. A máquina possui processador Intel Core 2 Duo T7500, com 2Gbytes de RAM e 160Gbytes de HD. O sistema operacional utilizado foi o Windows Vista Ultimate juntamente com o ambiente de desenvolvimento JDK 1.6.0\_10 para compilar o código Java implementado. A conexão com a Internet é feita via banda larga a uma velocidade não compartilhada de 3,3Mbits para download e 640Kbits para upload, em média.

## Avaliação inicial do OSVDB

Com o objetivo de conhecer melhor a rotina de inserção de novas vulnerabilidades no banco, a seguinte tabela sumariza dados recolhidos do banco entre os dias 1 e 10 de junho de 2009.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Item | Data | Número de registros | Crescimento diário | Tamanho da base (MB) |
| 1 | 01/06 | 53930 | 0 | 54,2 |
| 2 | 02/06 | 53934 | 4 | 54,2 |
| 3 | 03/06 | 53958 | 24 | 54,2 |
| 4 | 04/06 | 53986 | 28 | 54,3 |
| 5 | 05/06 | 54004 | 18 | 54,3 |
| 6 | 06/06 | 54012 | 8 | 54,3 |
| 7 | 07/06 | 54029 | 17 | 54,3 |
| 8 | 08/06 | 54034 | 5 | 54,3 |
| 9 | 09/06 | 54036 | 2 | 54,3 |
| 10 | 10/06 | 54061 | 25 | 54,3 |

Pela tabela acima, pode-se perceber o crescimento do banco ao longo dos 10 dias em que foram recolhidos esses dados. A média diária de atualizações foi aproximadamente 13 atualizações/dia.

A próxima tabela sumariza informações sobre o mesmo conjunto de dados da tabela acima, porém agora o foco é a completude dos registros. Neste contexto, está se considerando um registro de vulnerabilidade como completo apenas aquele que possui todos os campos que o esquema proposto neste trabalho é capaz de encontrar nas referências externas preenchidos.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Item | Data | Número de registros | Completos | Incompletos |
| 1 | 01/06 | 53930 | 12388 | 41542 |
| 2 | 02/06 | 53934 | 12390 | 41544 |
| 3 | 03/06 | 53958 | 12391 | 41567 |
| 4 | 04/06 | 53986 | 12407 | 41579 |
| 5 | 05/06 | 54004 | 12407 | 41597 |
| 6 | 06/06 | 54012 | 12408 | 41604 |
| 7 | 07/06 | 54029 | 12412 | 41617 |
| 8 | 08/06 | 54034 | 12416 | 41618 |
| 9 | 09/06 | 54036 | 12416 | 41620 |
| 10 | 10/06 | 54061 | 12419 | 41642 |

Analisando os números exibidos na tabela, fica claro como o número de registros completos e incompletos evolui. Nos dez dias, a base cresceu 131 registros, sendo que apenas 31 registros foram completados (de 12388 para 12419), já o número de registros incompletos aumentou 100 unidades (de 41542 para 41642).

Pode-se tirar uma importante conclusão da última tabela. Pelo altíssimo número de vulnerabilidades incompletas e a clara incapacidade de manter as informações sempre atualizadas, o OSVDB caminha para ter uma alta, e indesejada, taxa de incompletude em seus registros. Esse é mais um motivo que justifica a necessidade de haver um processo automatizado de atualização dos dados. Afinal, pelo presente número de incompletudes (41642 registros incompletos em 10/06/2009), a quantidade de trabalho necessária para recolher as informações faltantes é enorme.

## Rastreamento de Vulnerabilidades Incompletas

O experimento discutido nesta seção visa ir um pouco mais fundo no processo de atualização de vulnerabilidades do OSVDB e dar uma idéia de quais vulnerabilidades estão sendo completadas no banco. A tabela acima, apesar de exibir a evolução do número de vulnerabilidades completas dia após dia, não nos mostra quais são as vulnerabilidades que estão sendo completadas. Logo, também não é possível saber quanto tempo leva pra que uma vulnerabilidade possua todas as informações completas.

O primeiro problema para efetuar este teste é que a massa de vulnerabilidades incompletas é muito grande. Em uma execução do componente que identifica vulnerabilidades incompletas com a base do dia 20 de abril de 2009, foram encontradas mais de 40 mil vulnerabilidades com algum tipo de pendência de informação. Não é desejável, pois, executar o teste proposto com todas as vulnerabilidades, pois seria difícil identificar quando cada vulnerabilidade completada foi inserida.

Portanto, iremos utilizar como conjunto inicial de dados uma amostragem com as 100 últimas vulnerabilidades incompletas inseridas no banco. A idéia do experimento é começar com as 100 vulnerabilidades e ver quais delas continuam incompletas no segundo dia, bem como quantas vulnerabilidades incompletas foram inseridas no segundo dia. No terceiro dia, iremos ver quantas vulnerabilidades do primeiro e segundo dias permanecem incompletas e quantas novas vulnerabilidades incompletas foram adicionadas ao banco no terceiro dia e assim sucessivamente.

O experimento fez o rastreamento de quais vulnerabilidades estavam sendo completadas por cinco dias. Com os dados recolhidos foi construído o seguinte gráfico.

Figura 11. Gráfico mostrando a reincidência das vulnerabilidades.

Olhando para a coluna que representa as vulnerabilidades do primeiro dia do experimento (100 vulnerabilidades), podemos ver que a coluna praticamente não tem seu tamanho alterado. Isso significa que a quantidade de vulnerabilidades incompletas não diminuiu ao longo dos dias. O mesmo acontece com as outras colunas. Através dessa constatação, podemos concluir que o processo de atualização das vulnerabilidades é demorado e muitas vezes nem acontece.

## Tipos de ataques

Uma das informações mais importantes sobre uma vulnerabilidade é o tipo de ataque que elas permitem que seja feito. O gráfico abaixo exibe a porcentagem das variantes de ataques possíveis de serem feitos pela exploração de vulnerabilidades.

Figura 12. Gráfico de pizza exibindo os tipos de ataques.

Este gráfico foi construído a partir dos dados colhidos pelo esquema construído neste trabalho. Na ocasião, foi possível encontrar os tipos de ataques de 22901 vulnerabilidades do OSVDB. Como pode ser visto, 43% das vulnerabilidades permitem exploração de erros de validação, 10% permitem que ataques de buffer overflow sejam feitos, entre outros tipos. Este gráfico é importante também para se ter uma idéia de quais os tipos de ataques que as vulnerabilidades mais permitem que sejam feitos.

## Resultados

Para exibir os resultados conseguidos pelo esquema, foi implementado um aplicativo. Ele permite que o usuário digite o OSVDB ID da vulnerabilidade e veja tanto as informações que o OSVDB possui quanto aquelas que foram extraídas pelo esquema. Também é possível que o usuário só visualize o que o OSVDB possui.

As figuras a seguir mostram os resultados exibidos pelo aplicativo sem as informações externas e depois com elas. Para simplificar, o aplicativo só exibe as informações que o esquema é capaz de extrair. Portanto, ele omite algumas informações contidas no OSVDB.

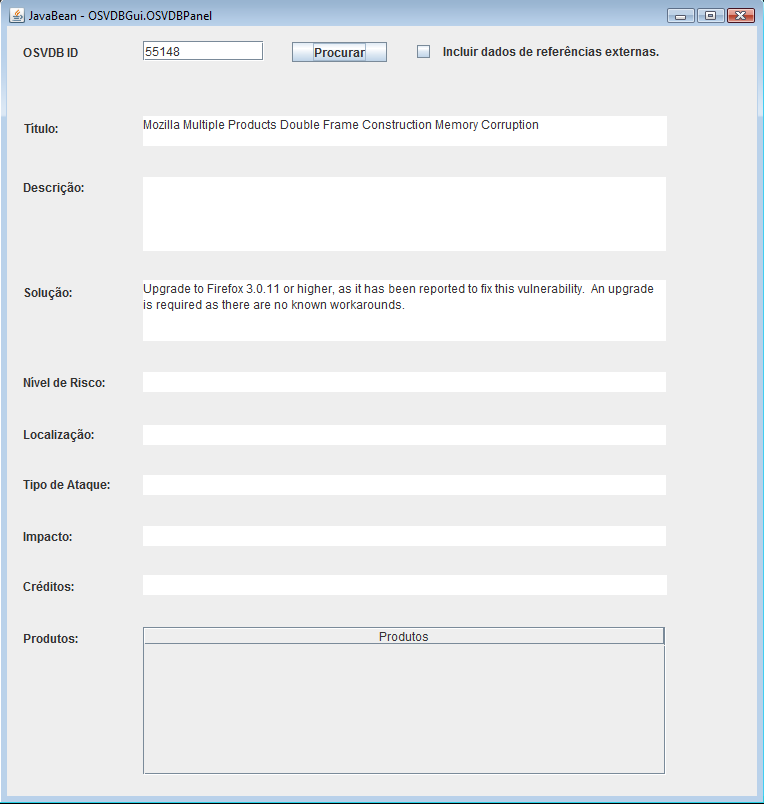


Figura 13. Vulnerabilidade 55148 sem informações externas.

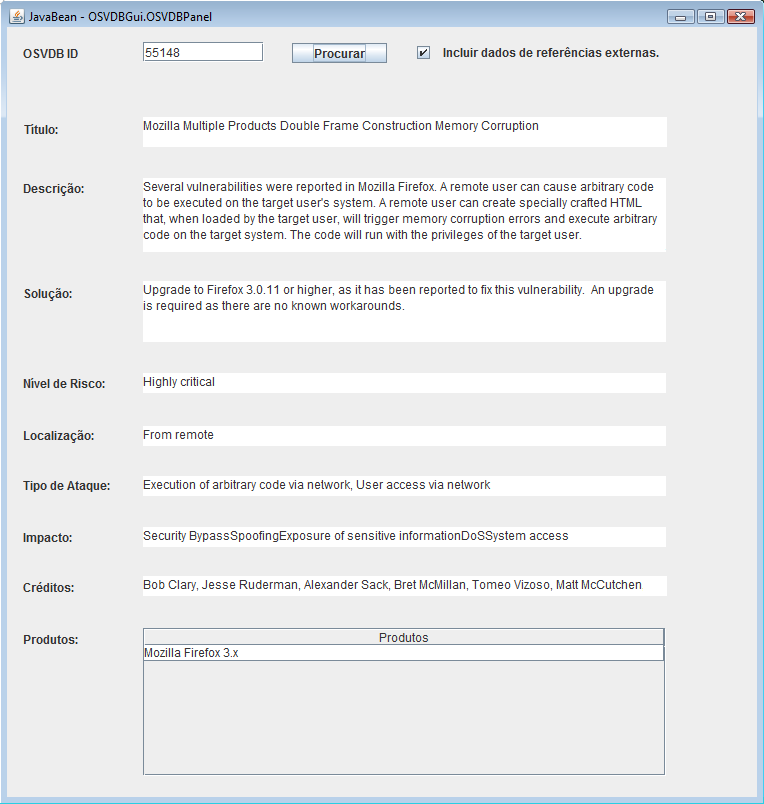


Figura 14. Vulnerabilidade 55148 com informações externas.

Na figura 13, o aplicativo exibe as informações que o OSVDB possui sobre a vulnerabilidade de ID 55148. O banco só possui informações sobre o título da vulnerabilidade e sua solução. Na figura 14, são exibidas as informações que o esquema construído foi capaz extrair de outros bancos de vulnerabilidades.

A seguir, mais um resultado mostrado pelo aplicativo.

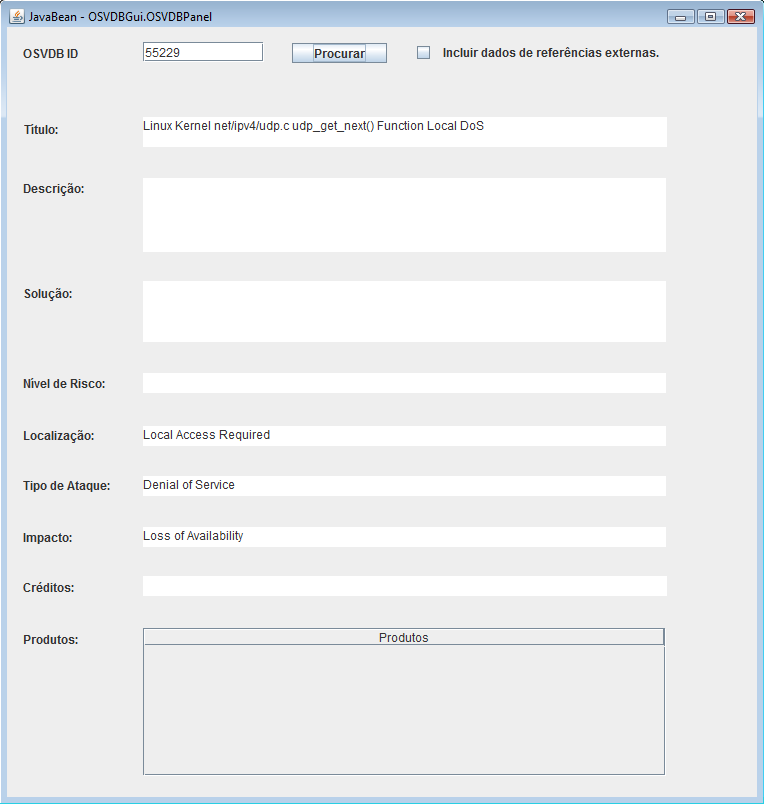


Figura 15. Vulnerabilidade 55229 sem informações externas

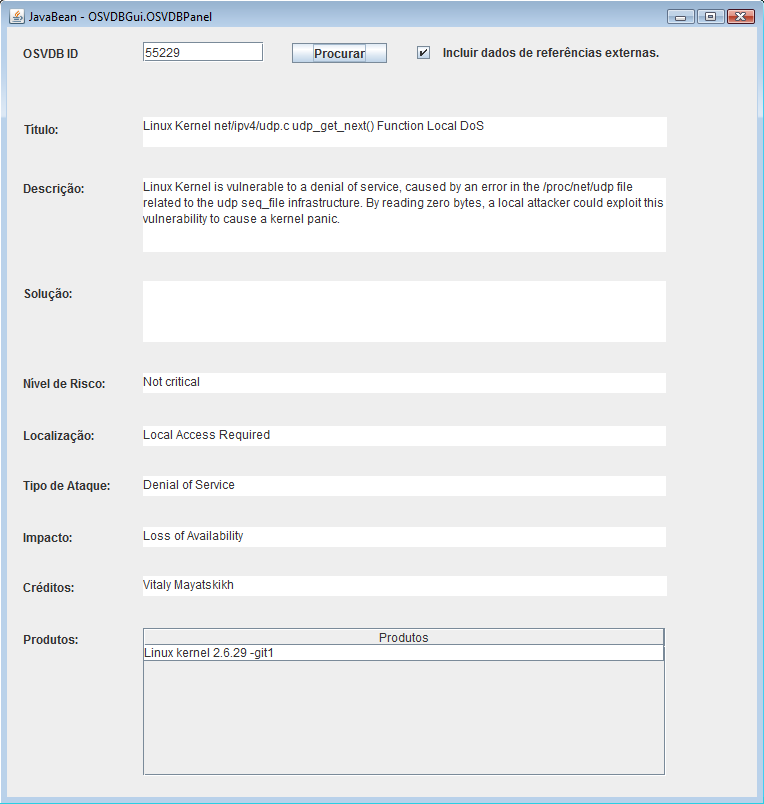


Figura 16. Vulnerabilidade 55229 com informações externas

Outro exemplo mostrando a vulnerabilidade de OSVDB ID 55229. Porém, nessa vulnerabilidade não foi possível encontrar nas referências externas a solução para a vulnerabilidade.

# Conclusões

Este trabalho apresentou um esquema de atualização automática de bancos de dados de vulnerabilidades baseado em extração de informação de referências externas. Tal esquema é capaz de diminuir o tempo para que informações sobre vulnerabilidades sejam adicionadas ao banco através da pesquisa em outros bancos. O processo de extração é feito em páginas web que exibem informações contidas em outros bancos de vulnerabilidades. Após o download das páginas, um componente, faz a extração, preenchendo *templates* predefinidos para depois as informações serem armazenadas em um banco de dados projetado para guardar estas informações.

O trabalho incluiu um detalhado estudo sobre o OSVDB para conhecer suas características e adequar o esquema proposto a elas. Para concluir pela real necessidade do que foi proposto, também foram feitas avaliações quantitativas do número de vulnerabilidades, quantas são incompletas e como é a rotina de atualizações diárias do banco.

Adicionalmente, foi desenvolvido um aplicativo capaz de permitir a visualização das informações contidas no OSVDB. O aplicativo permite visualizar apenas as informações disponibilizadas pelo grupo que mantém o OSVDB ou vê-las juntamente com o que foi extraído de outros bancos pelo esquema.

A principal contribuição do trabalho é a possibilidade de integrar informações de vários bancos de dados e concentrá-las em um só lugar. Assim, é possível que a comunidade encontre tudo que precisa em um só lugar, de forma simples, rápida e compreensível, assim, podendo tomar as devidas precauções quanto à defesa contra explorações.

## Dificuldades Encontradas

Uma das dificuldades encontradas no desenvolvimento do trabalho está na fase de extração de informações das páginas HTML. Primeiramente, documentos HTML são semi-estruturados, o que dificulta a extração exata de informações. Em segundo lugar, cada banco de dados consultado tem a estrutura da sua página Web diferente, logo, é preciso adequar o componente para cada uma das páginas. Essas características tornam o processo de extração trabalhoso.

## Trabalhos Futuros

Um interessante trabalho futuro seria estender o sistema que foi implementado neste trabalho para mapear os OSVDB IDs em identificadores CVE e, assim, ser capaz de extrair mais informações sobre a vulnerabilidade.

Outro interessante trabalho futuro é utilizar um processo de extração de informação bastante robusto capaz de extrair informações de textos escritos em linguagem natural para alimentar o banco. Um sistema desses seria capaz de alimentar a banco com informações extraídas de listas de discussões e fóruns espalhados por toda a Internet.

# Referências

UCLA. **Survey Glossary**. Disponível em: <http://www.sfs.finance.ucla.edu/survey/glossary.htm>. Acesso em: 01 jun. 2009.

WILSON, Clay. **Botnets, Cybercrime, and Cyberterrorism**: Vulnerabilities and Policy Issues for Congress. Recife: Crs, 2008.

Morin, M. (2006) **The Financial Impact of Attack Traffic on Broadband Networks.** IEC Annual Review of Broadband Communications, páginas 11-14.

Richardson, R. (2007) **CSI/FBI Computer Crime Survey.** Em Twelfth Annual Computer Crime ad Security, páginas 1-30.

CERT. **CERT Statistics.** Disponível em: <http://www.cert.org/stats/>. Acesso em: 30 abr 2009.

MCGRAW, Gary; GHOST, Anup. **Developing Expertise in Software Security:** An Outsider's Perspective. Sterling, 1996.

MEUNIER, Pascal C; SPAFFORD, Eugene H. **Final Report of the 2nd Workshop on Research with Security Vulnerability Databases.** West Lafayette, 1999.

SCHUMACHER, M et al. *Data Mining in Vulnerability Databases*. **7th Workshop** **Sicherheit In Vernetzten Systemen,** Darmstadt. 22 mar. 2000.

*NATIONAL Vulnerability Database* (NVD) Disponível em: <http://nvd.nist.gov/>. Acesso em: 20 abr. 2009.

OPEN SECURITY FOUNDATION. **OSVDB:** The Open Source Vulnerability Database. Disponível em: <http://osvdb.org/>. Acesso em: 20 abr. 2009.

MITRE CORPORATION. **Common Vulnerabilities and Exposures.** Disponível em: <http://cve.mitre.org/>. Acesso em: 22 abr. 2009.

BAKER, David W et al. **The Development of a Common Enumeration of Vulnerabilities and Exposures.** Mclean, 1999.

IBM. **VulDa:** A Vulnerability Database. Disponível em: <http://domino.watson.ibm.com/library/cyberdig.nsf/a3807c5b4823c53f85256561006324be/4cc8fa2ee3af7fc9852567280039a299?OpenDocument>. Acesso em: 1 jun. 2009.

CISCO. **Cisco Security Center.** Disponível em: <http://tools.cisco.com/security/center/home.x>. Acesso em: 10 jun. 2009.

GORDON, Sarah. **Virus and Vulnerability Classification Schemes:** Standards and Integration. Cupertino: Editora, 2003.

ISO. **STANDARDS AND REGULATIONS.** Disponível em: <http://www.standardsinfo.net/info/livelink/fetch/2000/148478/6301438/standards\_regulations.html>. Acesso em: 17 jun. 2009.

ROHSE, Michael. **Vulnerability naming schemes and description languages:** CVE, Bugtraq, AVDL and VulnXML. Bethesda: Editora, 2003. Disponível em: <http://www.sans.org/reading\_room/whitepapers/threats/vulnerability\_naming\_schemes\_and\_description\_languages\_cve\_bugtraq\_avdl\_and\_vulnxml\_1058>. Acesso em: 11 jun. 2009.

SECURITY FOCUS. **Bugtraq.** Disponível em: <http://www.securityfocus.com/>. Acesso em: 11 jun. 2009.

IATAC. **The National Vulnerability Database.** Disponível em: <http://iac.dtic.mil/success/archives/FY07\_3/FY07\_3\_iatac.pdf>. Acesso em: 11 jun. 2009.

SHADOWSERVER Disponível em: <http://www.shadowserver.org>. Acesso em: 11 jun. 2009.

MCCULLAGH, Declan. **How Pakistan knocked YouTube offline.** Disponível em: <http://news.cnet.com/8301-10784\_3-9878655-7.html>. Acesso em: 11 jun. 2009.

OLIVEIRA, Ricardo. **Cyclops.** Disponível em: <http://cyclops.cs.ucla.edu/>. Acesso em: 12 jun. 2009.

ARBOR NETWORKS. **Atlas.** Disponível em: <http://atlas.arbor.net/>. Acesso em: 14 jun. 2009.

ARBOR NETWORKS. **ARBOR.** Disponível em: <http://www.arbornetworks.com/>. Acesso em: 14 jun. 2009.

SECURITY FOCUS. **Security Focus.** Disponível em: <http://www.securityfocus.com/>. Acesso em: 1 jun. 2009.

SYMANTEC CORPORATION. **Symantec.** Disponível em: <http://www.symantec.com>. Acesso em: 1 jun. 2009.

SQLITE. **SQLite.** Disponível em: <http://www.sqlite.org/>. Acesso em: 1 jun. 2009.

SUN MICROSYSTEMS. **MySQL.** Disponível em: <http://www.mysql.com/>. Acesso em: 1 jun. 2009.

1. Neste contexto, são redes formadas por diversos computadores infectados com *bots.* [↑](#footnote-ref-2)
2. Programa, ou parte de dele, que faz cópias de si mesmo infectando outros computadores e arquivos. [↑](#footnote-ref-3)
3. Do inglês ***Mal****licious Soft****ware***. É uma denominação genérica para qualquer software malicioso. [↑](#footnote-ref-4)
4. É um mecanismo que possibilita executar com segurança código não testado ou não confiável. [↑](#footnote-ref-5)
5. Armadilha destinada a atrair intrusos. Utiliza um software que monitora as atividades e aponta pontos falhos do sistema. [↑](#footnote-ref-6)
6. Ataque que causa o desvirtuamento das rotas que os pacotes devem seguir. [↑](#footnote-ref-7)
7. http://www.routeviews.org/ [↑](#footnote-ref-8)
8. http://www.ripe.net/ris/ [↑](#footnote-ref-9)
9. http://noc.net.internet2.edu/i2network/research-data.html [↑](#footnote-ref-10)
10. http://www.phc.net/home/index.php [↑](#footnote-ref-11)
11. http://bgpmon.netsec.colostate.edu/ [↑](#footnote-ref-12)
12. http://xforce.iss.net/ [↑](#footnote-ref-13)
13. http://www.securityfocus.com/ [↑](#footnote-ref-14)