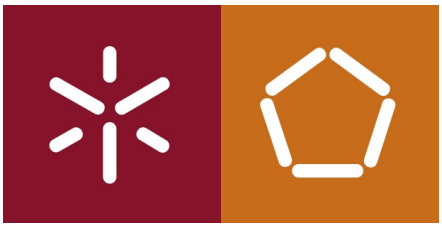

Engenharia de Segurança
Grupo 7
Aula 8

Bruno Machado - A74941
Diogo Gomes - A73825
Francisco Mendes - A75097

Abril 2018

1

Indo ao encontro do que foi estudado nas aulas, a cada 1.000 linhas de código encontra-se entre 5 e 50 *bugs*. Sendo assim, o número de linhas de código de cada aplicação e a estimativa do seu número de bugs *bugs* são as seguintes:

- **Facebook:** 61 milhões de linhas de código - 305 mil a 3 milhões *bugs*
- **Software de Automóveis:** 100 milhões de linhas de código - 500 mil a 5 milhões *bugs*
- **Linux 3.1:** 15 milhões de linhas de código - 75 mil a 750 mil *bugs*
- **Google:** 2 mil milhões de linhas de código - 10 milhões a 100 milhões *bugs*

O número de vulnerabilidades que estes *bugs* representam não é estimável, pois não se sabe ao certo quantos destes constituem falhas no sistema nem se são exploráveis.

2

2.1 Vulnerabilidades de Projeto

- **Acesso de controlo inapropriado.** O software não restringe de maneira correta os acessos a utilizadores não autorizados. Este problema é dificilmente tratado de forma correta devido ao grande número de utilizadores com diferentes permissões que leva à geração de um grande número de regras, que pode levar ao acesso a dados por utilizadores não autorizados. Para mitigação do problema devem ser geridos com cuidado os privilégios de utilização assim como as zonas de confiança no software.
- **Validação de input impróprio.** Validação de input incorreta ou inexistente pode afetar o controlo de fluxo e os dados do programa. Invalidação imprópria de input pode levar a problemas com *buffer overflow*, *cross-site scripting*, etc... Este problema é difícil de mitigar uma vez que é necessário restringir todos os inputs que um programa recebe, sendo necessárias regras diferentes, dependendo do tipo de input que seja desejável tornando-se um trabalho meticuloso que por vezes é deixado de parte. Uma potencial solução para este problema é a utilização de frameworks como Struts or the OWASP ESAPI Validation API.

2.2 Vulnerabilidades de Codificação

- **Buffer Overflow** Este problema ocorre quando um programa aceita um input maior que o seu buffer. Este problema pode levar à injeção de código malicioso na memória. Este problema pode ser resolvido restringindo-se todos os tamanhos de input que o programa recebe. Existem linguagens de mais alto nível que já têm em conta este tipo de problemas, no entanto se um programador fizer a transição de uma linguagem de alto nível para uma linguagem de baixo nível pode incorrer neste problema por distração ou ignorância.
- **Falta do caso default num bloco switch** Este problema ocorre quando não existe um caso default num bloco switch que pode levar criação que problemas lógicos complexos. Para que não exista este programa o programador tem que se certificar que declara o caso default do bloco switch mesmo que seja apenas para fazer prosseguir o programa.

2.3 Vulnerabilidades Operacional

- **ASP.NET Misconfiguration: Password in Configuration File** O armazenamento de passwords em ficheiros de configuração em plaintext, permite a qualquer utilizador com acesso a utilização dessa informação. Para a erradicação desta vulnerabilidade as passwords devem ser cifradas.

- **ASP.NET Misconfiguration: Creating Debug Binary** As mensagens de debug ajudam os atacantes a tirar informações sobre o sistema e a construir um possível ataque. Uma possível mitigação deste problema é evitar a criação de mensagens de debug binárias no ambiente de produção. Alterar o debug mode para falso quando a aplicação é utilizada.

3 Vulnerabilidade de Dia-Zero

O termo dia-zero refere-se a uma vulnerabilidade de software desconhecida que o programador apenas descobriu recentemente e, portanto, um patch ou atualização oficial para corrigir o problema não foi disponibilizado. Essencialmente, "dia zero" refere-se ao facto de que os programadores tiveram "zero dias" para corrigir o problema que acaba de ser exposto, e que talvez já tenha sido explorado por *hackers*. Assim que a vulnerabilidade se torna publicamente conhecida, o fornecedor da aplicação deve trabalhar rapidamente para corrigir o problema, afim de proteger os seus clientes. No entanto, se o fornecedor de software não lançar um *patch* em tempo útil e os *hackers* conseguirem explorar a falha de segurança, dá-se o que é conhecido como o ataque do dia zero.