Programação Segura

Semana 1: Apresentação e Fundamentos

Profª Denise Goya
Denise.goya@ufabc.edu.br – UFABC - CMCC

APRESENTAÇÃO DA DISCIPLINA

Programação Segura

- Motivação para cursar a disciplina:
  - Correções de segurança frequentes:
    - Grande ocorrência de falhas descobertas
    - Sistemas cada vez mais complexos
    - Sistemas cheios de remendos (patches)
    - Ruim para o produtor e para o usuário

Objetivos da Disciplina

- Despertar a consciência para o desenvolvimento seguro de software
- Conhecer os principais problemas que tornam o software vulnerável
- Conhecer boas práticas de programação segura

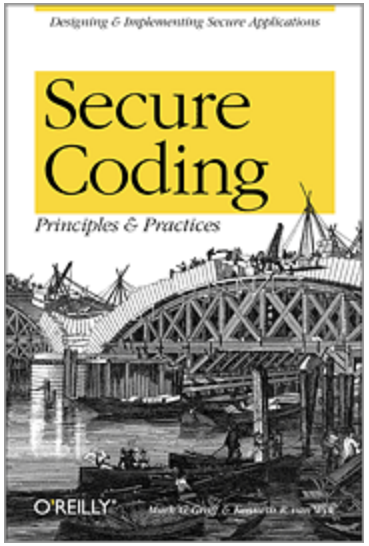
Ementa

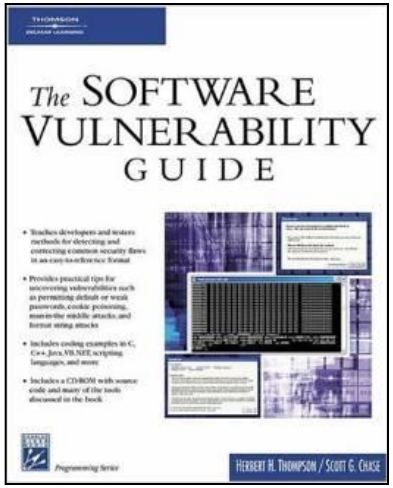
- segurança no processo de desenvolvimento de software;
- vulnerabilidades: descrição, tecnologias (linguagens, SO) envolvidas, prevenção e correção;
- ferramentas para prevenção de vulnerabilidade;
- características relevantes de linguagens de programação;
- prática: busca por vulnerabilidades

Bibliografia Básica

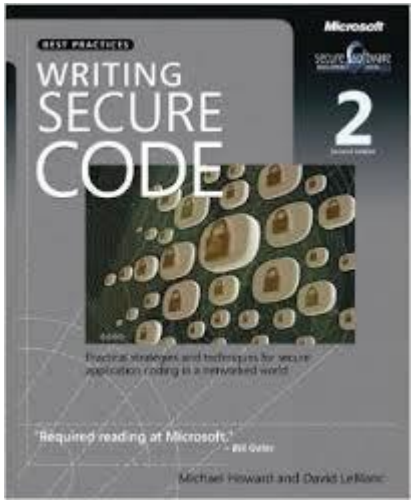
- GRAFF, MARK G; VAN WYK, KENNETH R.: "**Secure Coding: Principles and Practices**" O'Reilly, 2003.

- THOMPSON, H.; CHASE, SCOTT G.: "**The Software Vulnerability Guide**" Charles River Media, 2005





Bibliografia Complementar



- HOWARD, M.; LEBLANC, D.: "Writing Secure Code" Microsoft Press, 2a edição, 2002.
- WEELER, D.: "Secure Programming for Linux and Unix HOWTO – Creating Secure Software". Ebook disponível em http://www.dwheeler.com/secure-programs/

Bibliografia Complementar

- SEBESTA, R.: "Conceitos de Linguagens de Programação" Bookman, 5a edição, 2003.
- TANENBAUM, A. “Sistemas Operacionais Modernos” Prentice Hall, 2a edição, 2007.
- TANENBAUM, A. “Redes de Computadores” Campus, 4a edição, 2003.
- KUROSE, J. “Redes de Computadores e a Internet: uma nova abordagem” Addison-Wesley, 2a edição, 2007.
- HARBISON, S.; STEELE JR, G. L. “C: manual de referência” Prentice Hall/Ciência Moderna, 2002.
- ROCHKIND, M. “Advanced UNIX Programming” Addison-Wesley, 2a edição, 2004.
- STEVENS, W. R.; FENNER, B.; RUDOFF, A. M. “Unix Network Programming” Addison-Wesley, 3a edição, 2003.
- STEVENS, W. R.; RAGO, S. “Advanced Programming in the UNIX Environment” Addison-Wesley, 2a edição, 2008.

Programação Segura – TPI 2-2-4

- 2h de aulas teóricas semanais
- 2h de aulas práticas semanais
- 4h de estudo individual semanais

Onde e Quando

- Terças-feiras: teoria
  - 10:00 às 12:00
  - sala S - 301-3

- Quintas-feiras: prática
  - 08:00 às 10:00
  - Lab 407-2

Avaliação

- Duas provas (escritas)
  - P1: 30%
  - P2: 40%
- Um projeto em equipe (análise de código)
  - Projeto: 30%

Cronograma Previsto

- P1: **10/março**
- P2: **07/abril**
- Apresentações de Projeto: **09/abril e 16/abril**
- Mecanismo de Substituição: **14/abril**
  - Sem atestado médico **no dia da prova** não tem direito
  - Conteúdo integral da disciplina
- Mecanismo de Recuperação: **23/abril**
  - Tem direito quem obteve D ou F
  - Conceito preliminar conta com peso de 50%

FUNDAMENTOS: CONCEITOS BÁSICOS

O que é Programação Segura

- Aplicação de boas práticas de projeto de software e de desenvolvimento de **código seguro**

O que é Código Seguro

- Código de software que protege a
  - Confidencialidade,
  - Integridade e
  - Disponibilidade
    - da informação do usuário e
  - Integridade e disponibilidade
    - dos recursos de processamento sob controle do administrador ou proprietário

Confidencialidade

- Objetivo de manter a confidencialidade do dado:
  - preservar o sigilo de dados secretos (confidenciais)
  - apenas pessoas autorizadas (pelo dono do dado) podem ter acesso ao dado (ou ter condições de compreendê-lo)

Integridade

- Objetivo de manter a integridade do dado:
  - usuários não autorizados (pelo dono do dado) não podem modificá-lo (adulterar, remover)
  - qualquer adulteração deve ser detectável

Disponibilidade

- Objetivo de manter a disponibilidade do dado:
  - ninguém deve poder perturbar o sistema para deixá-lo inutilizável ou inoperante
  - impedir a negação do serviço

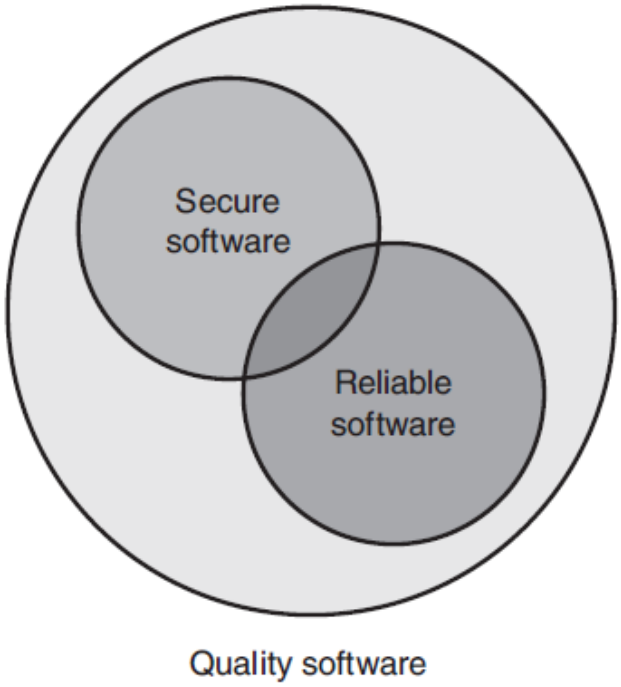
Objetivo adicional: Privacidade

- Objetivo de manter a privacidade do usuário:
  - proteger indivíduos contra o mau uso de informação sobre eles
  - envolve questões legais e éticas
  - que limites há para:
    - Empresas de seguro ou saúde fazerem uso de dados de seus clientes?
    - Governos e Polícia fazerem investigações sobre um cidadão?

Software Seguro e Software Confiável

- Software Seguro é um subconjunto de:
  - Qualidade de software e
  - Software confiável

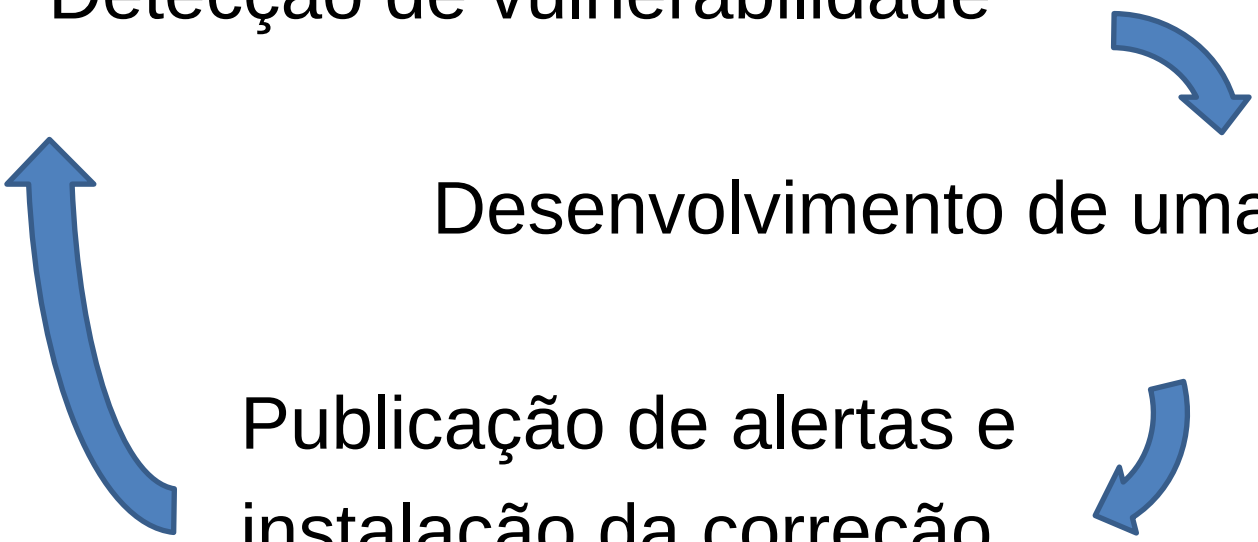
- Nesta disciplina, o software seguro já cumpre os requisitos funcionais
  - preocupação principal nas outras disciplinas



Vulnerabilidade de Segurança

- Falha em um produto que pode permitir um atacante obter
  - privilégio de acesso
  - controle do sistema
  - comprometimento do dado
- mesmo que o usuário faça um uso correto do produto

Ciclo de Correção de Software

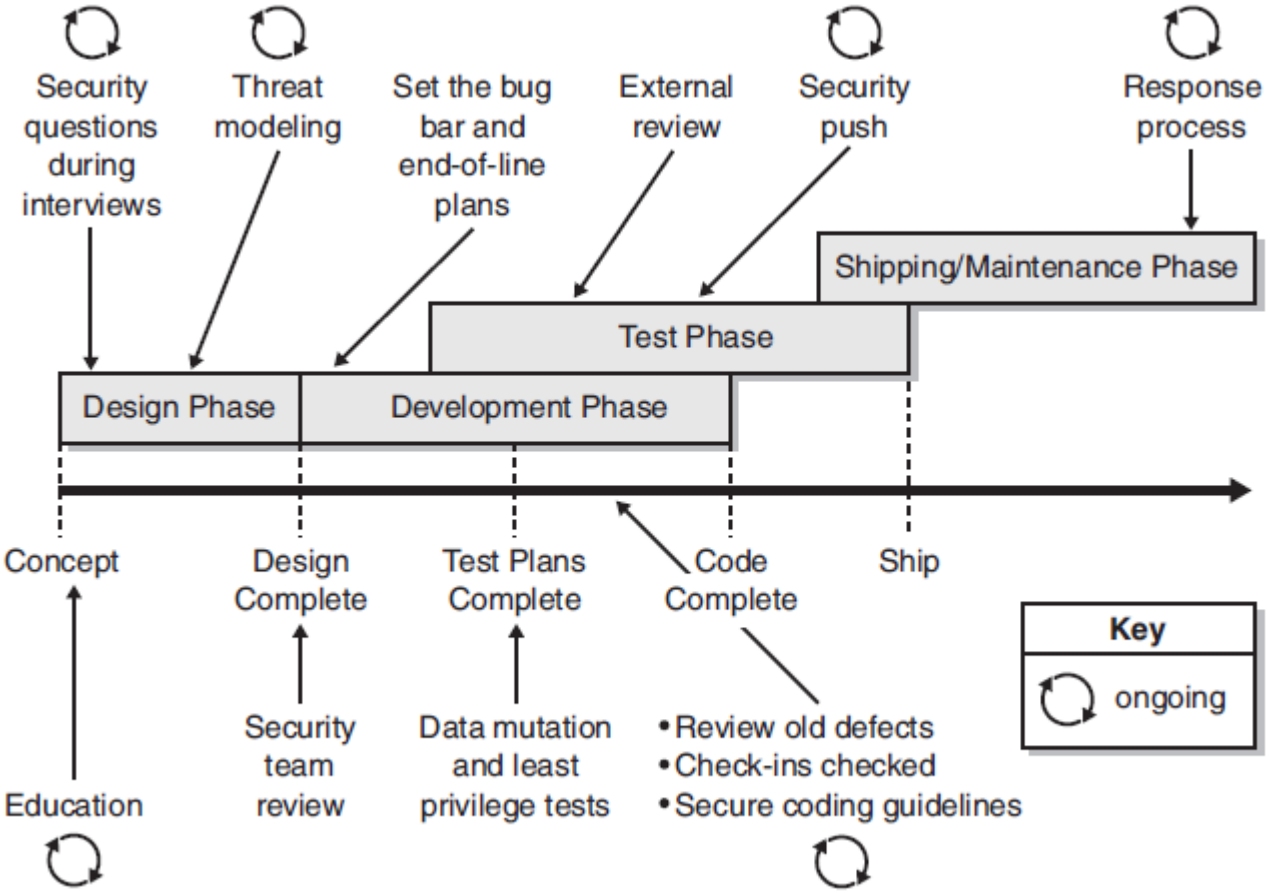


Devemos tentar reduzir a quantidade de correções posteriores ao lançamento do produto

Processo de Desenvolvimento

- Alternativa ideal: pensar em questões de segurança durante TODO o processo de desenvolvimento de software

Processo de Desenvolvimento



Falhas de Segurança em Software

- As falhas de segurança estão associadas a erros comuns de programação

- Tais falhas levam a uma situação de **vulnerabilidade** que, ao ser devidamente explorada, viabilizam **ataques** aos sistemas

Classes de Ataques

- Ataques costumam estar relacionados a:
  - Roubo de senhas
  - Engenharia social
  - Exploração de bugs e back doors
  - Exploração de falha de autenticação
  - Exploração de falha em protocolo
  - Vazamento de informação
  - Ataques exponencial (vírus e worms) e Botnets
  - Negação de Serviço (DoS)
  - Ataques ativos

Classes de Erros de Segurança

- Os erros (bugs) mais comuns cometidos por programadores são classificados em

- Katrina Tsipenyuk, Brian Chess, and Gary McGraw. 2005. **Seven Pernicious Kingdoms: A Taxonomy of Software Security Errors**. IEEE Security and Privacy 3, 6 (Nov 2005)

Classes de Erros de Segurança

- Validação e representação dos dados de entrada
- Abuso de API
- Características de segurança
- Tempo e estado
- Tratamento de exceções
- Qualidade do código
- Encapsulamento
- Ambiente

Validação e Representação da Entrada

- Problemas relacionados a essa classe de erros são causados pelo mal tratamento de:
  - metacaracteres
  - alternância de codificações ou de representações numéricas
- Incluem uma variedade de problemas:
  - buffer overflow
  - injeção de SQL
  - ataques de Cross-site scripting

Validação da Entrada - subtipos

- buffer overflow
- injeção de comandos
- cross-site scripting
- string de formatação
- manipulação de caminho
- controle de processo
- manipulação de configuração
- formulários validados reusados
- etc

Entrada e Buffer Overflow

- transbordo de memória
- escrita em área de memória não prevista pelo programador
- causa corrompimento de dado, mau funcionamento do programa
- pode viabilizar a execução de código produzido e inserido pelo atacante

Entrada e Injeção de Comandos

- Execução de comandos a partir de uma fonte ou ambiente não confiável permite a execução de comandos mal intencionados por parte de um atacante

Entrada e Cross-Site Scripting

- Envio de dado inválido para um navegador pode fazer com que sejam executados códigos mal intencionados (em geral scripts)

Entrada e String de Formatação

- Mal uso de String de formatação pode resultar em buffer overflow

Abuso de API

- Um uso inadequado de uma API pode levar a situações de falha e segurança
  - ainda que o programa “funcione”
  - o atacante pode forçar situações que lhe dão alguma vantagem sobre o sistema
- Exemplo:
  - acesso a sistema de arquivos com erro no tamanho do buffer pode levar a buffer overflow

Características de Segurança

- Classe de erro relacionada com problemas na implementação de autenticação, controle de acesso, confidencialidade, gerenciamento de privilégios e de funções criptográficas
- Exemplos:
  - uso de gerador de números aleatórios inseguro
  - gestão de senhas
  - violação de privilégio mínimo, etc

Tempo e Estado

- Problemas relacionados a software de computação distribuída

- Bugs podem levar a inconsistências

Erros Relativos a Qualidade do Código

- Classe de problema em geral vinda de más práticas de programação

- Exemplos:
  - chamar free( ) duas vezes sobre a mesma memória pode levar a buffer overflow
  - comportamento não definido
  - variável não inicializada

Problemas com Encapsulamento

- Atacantes podem explorar
  - objetos *cloneable* para criar novas instâncias
  - variáveis *non-final* públicas para injetar valores a sua escolha
  - etc

No Decorrer das Aulas

- Vamos dar mais detalhes de
  - como e porque todas essas classes de problemas existem
  - como evitá-las

ESTUDO INDIVIDUAL

Leitura Recomendada

- Para ter ideia geral dos tipos de problemas que existem com programação insegura:

- Katrina Tsipenyuk, Brian Chess, and Gary McGraw. 2005. **Seven Pernicious Kingdoms: A Taxonomy of Software Security Errors**. IEEE Security and Privacy 3, 6 (Nov 2005)