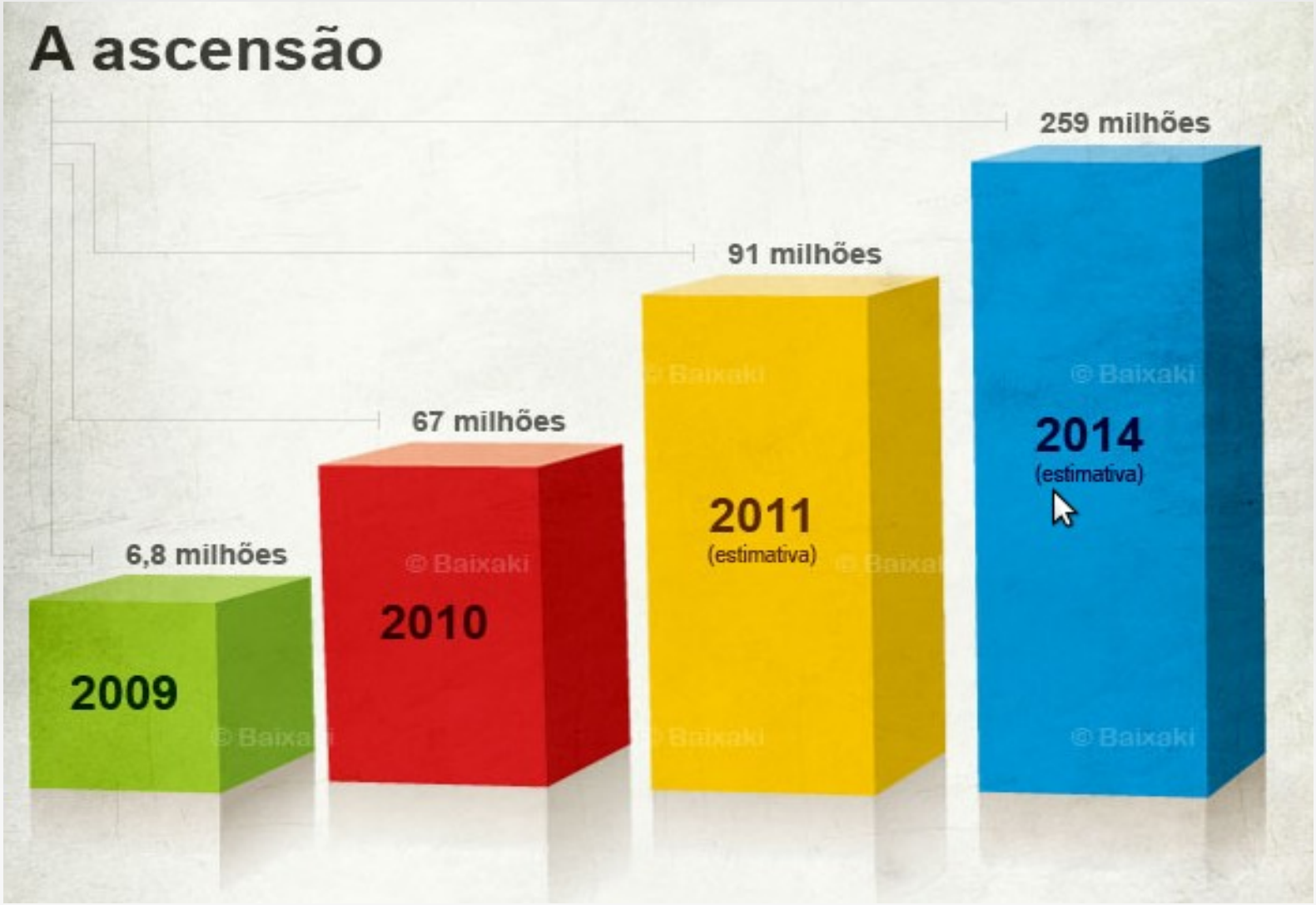
OWASP Paraiba Day 2012

Desenvolvimento Seguro de Aplicações para Android

Ana Cláudia Costa

Celulares no Mundo(2000 a 2009)

Ascenção do Android



Fonte: http://www.tecmundo.com.br/infografico/9010-android-o-sistema-operacional-movel-que-conquistou-o-mundo.htm

Ascenção Android

Android lidera mercado mundial de smartphones
No primeiro trimestre de 2011, 35% dos smartphones vendidos eram Android, contra 19% do iOS, da Apple

Fonte: http://www.geektech.com.br/?p=116

Android

- Plataforma aberta baseada em Linux
- Android Development Tools (ADT);

  Plugin para eclipse
- Desenvolvimento em Java (Linux/Windows/Mac OS)
- 1o Smartphone: HTC G1 (2008)
- 1o Tablet: Samsung Galaxy Tab (2010)

Ascenção do Android

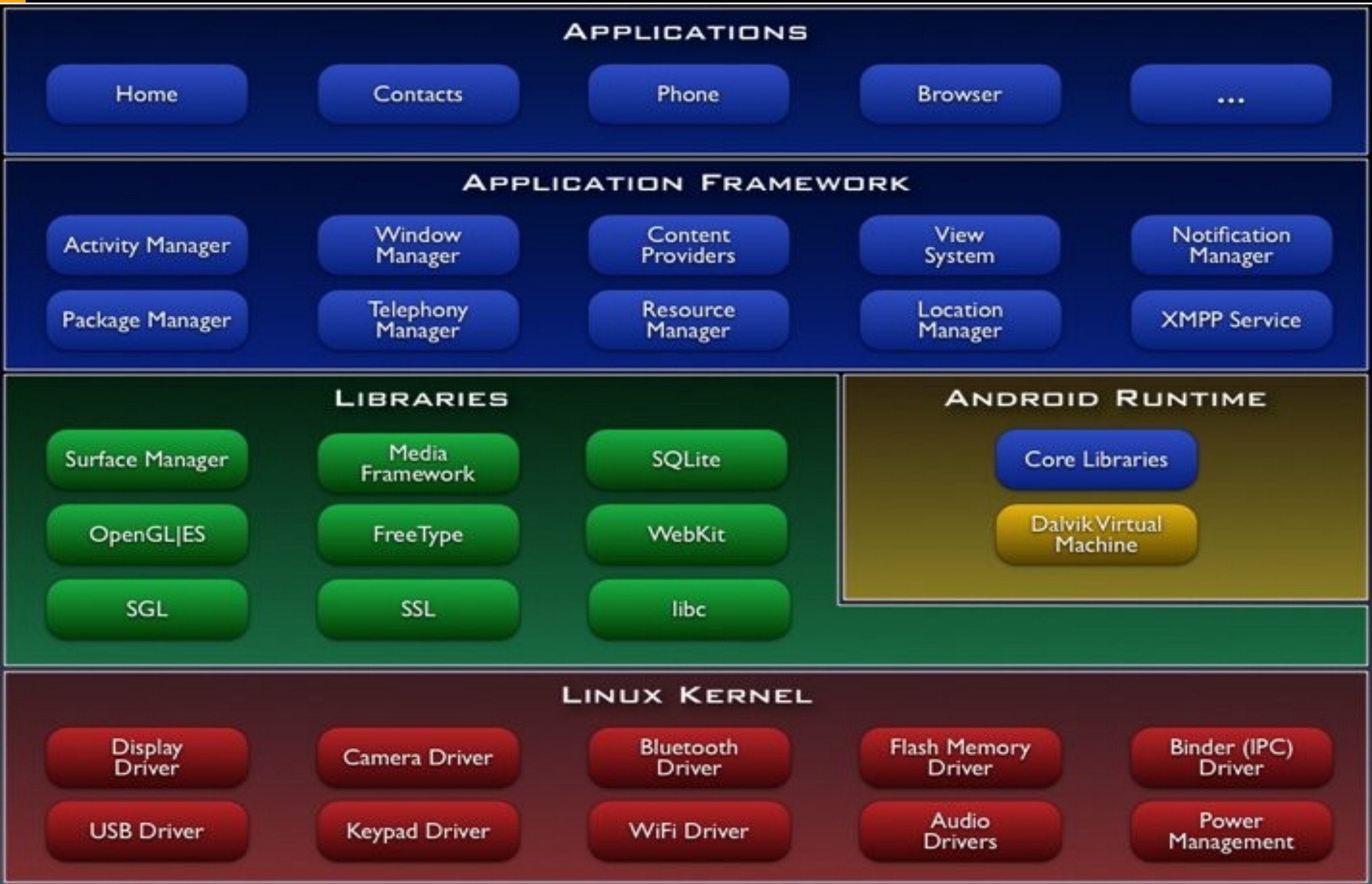
- Departamento de Defesa Americano aprova Android
- Android chega à marca de 400 mil aplicativos
- Samsung supera Apple em vendas de smartphones (Julho a Setembro/2011)
- Android tem 500 mil ativações diárias (junho/2011)

Fontes:
http://noticias.r7.com/tecnologia-e-ciencia/noticias/android-chega-a-marca-de-400-mil-aplicativos-20120104.html
http://googlediscovery.com/2011/06/29/android-chega-a-500-mil-ativacoes-por-dia-diz-google/

No entanto...

- Aumenta o número de malwares na loja do android em 472% (novembro/2011)

Arquitetura Android



AndroidManifest

- Arquivo de configuração de uma aplicação Android
- Identifica nome e o ícone da aplicacao
- Declara Componentes
- Define a versão mínima do android na qual a aplicação pode ser executada
- Identifica quaisquer permissão que a aplicação possa obter

Android Manifest

- Toda aplicação deve ter um nome que é globalmente único
- CheckPermission
- SecurityException

AndroidManifest

- Níveis de Permissão:
  - Normal
  - Dangerous
  - Signature
  - Signature or System

```
<permission
      xmlns:android="http://schemas.android.com/apk/res/android"
      android:name="com.isecpartners.android.ACCESS_SHOPPING_LIST"
      android:description="@string/access_perm_desc"
      android:protectionLevel="normal"
      android:label="@string/access_perm_label">

</permission>
```

Intent

- android.content.Intent
- Mensagem da aplicação para o SO, solicitando que algo seja realizado
- Importante papel na arquitetura do Android para integrar diferentes aplicações
- Núcleo de interconexão de componentes
- Forma utilizada para fazer com que as aplicações em processos diferentes se comuniquem

Intent

- Exemplo

```
Intent it = new Intent(this, Telax.class);
startActivity(it);
```

  - Esse código cria uma mensagem ao SO informando que a intenção é abrir a tela X, o SO analisa a mensagem e encontra a tela correta que precisa ser aberta.
  - Nem sempre é recebida pela mesma aplicação que a criou, frequentemente sendo utilizada para integrar aplicações

Intent

- O android não diferencia uma activity criada na sua aplicação de uma activity nativa da plataforma.

```
Uri uri = Uri.parse( ”http://code.google.com/android/”);
Intent it = new Intent(Intent. ACTION_VIEW, uri);
startActivity(it);
```

Intent Filter

- android.content.IntentFilter
- Parâmetros:
  - Ação: define o que a intent deseja realizar
  - Categoria: verifica se o objeto filtrado suporta as restrições de categoria.

Intent Filter

```
<activity android:name=”TesteActivity”>
    <intent-filter>
    <action
    android:name=”android.intent.action.MAIN” />
    <category
    android:name=”android.intent.category.LAUNCHER
    ”/>
    </intent-filter>
    </activity>
```

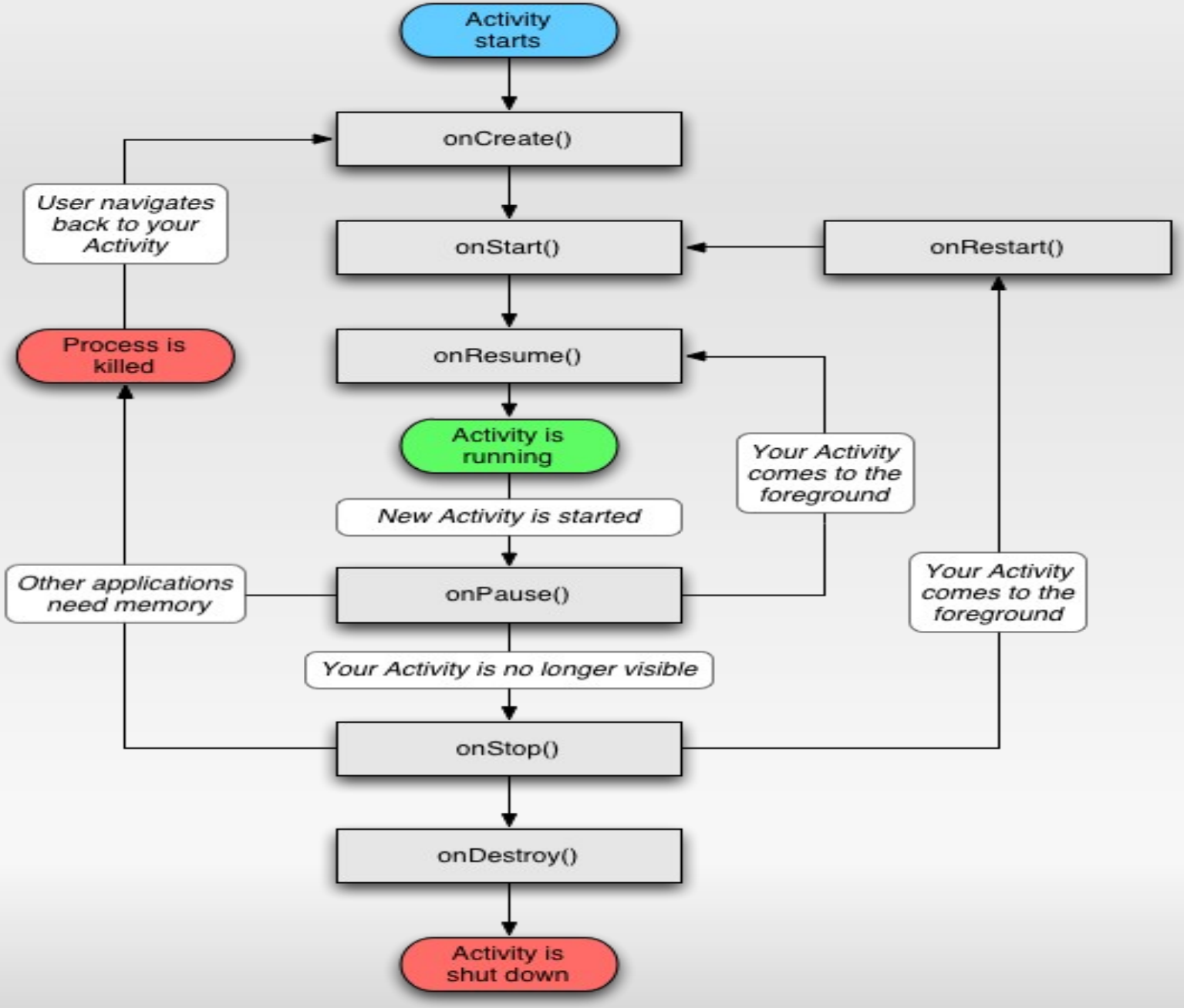
Intent Filter

- Destinatários de Intent como Activities, Services e BroadCastReceivers precisam verificar se os remententes são maliciosos, e o IntentFilter não filtra Intents maliciosos
- Ajudam o sistema descobrir qual o processo correto para determinado Intent.
- Sempre utilizar juntamente com a categoria
- Podem depender de verfificações de permissões como mecanismos de segurança

Activity

- Representa uma tela da aplicação
- Responsável por tratar os eventos gerados nessa tela
- Deve obrigatoriamente ser declarada no AndroidManifest

Activities – Ciclo de Vida



Activities

- Activities não podem depender de IntentFilters para parar remetententes maliciosos
- O caminho mais fácil para fazer Activities seguras é solicitar confirmação do usuário para alguma mudança.
- Utilizar de verificações de permissões com mecanismos de segurança, configurando android:permission na declaração <activity>
- Validar os dados de entrada

Activities

- Torna-se pública quando possui intent-filter ou android:exported
- Evitar colocar dados em Intents usado para iniciar Activity que seriam de interesse para um atacante.

BroadCast Receiver

- Usados para que aplicações possam reagir a determinados eventos gerados por uma intent.
- Sempre é executado em segundo plano durante pouco tempo e sem utililzar a interface gráfica
- Objetivo: receber uma mensagem(intent) e processá-la sem que o usuário perceba.

BroadCast Receiver

- Formas de configuração:
  - Configurar o AndroidManifest.xml e inserir uma tag<receiver> e uma tag <intent-filter> com a configuração da ação e categorias utilizadas
  - Utilizar o método context.registerReceiver(receiver, filtro) dentro do código para registrar dinamicamente o BroadCast receiver.Sendo receiver uma instância de IntentReceiver, e o filtro uma instância da classe IntentFilter, com a configuração da ação e categoria

BroadCast Receiver

- Não é possível um BroadCastReceiver receber uma intent disparada para iniciar uma activity
- Receivers devem ser robustos contra Intents inesperadas ou dados errados
- Como sempre, para a segurança de interconexão entre componentes, programas devem ser cautelosos quanto a validação dos dados de entrada

BroadCast Receivers

- Por padrão os receivers não são exportados, mas podem ser exportados facilmente pela tag<intent-filter> ou por configurar o atributo android:exported=true

- Para restringir quem pode mandar intent para seus receivers, é recomendado o uso do atributo android:permission na tag<receiver> do android manifest

BroadCast Receivers

- Quando uma ação é especificada no receptor, o gerenciador de Activities verifica qual o remetente tem a permissão específica antes de enviar a intent.
- Permissions é o caminho correto mas não afetam nas propriedades da intent que será recebida.

BroadCast Receivers

- Se o dado enviado é importante, é necessário ser cauteloso com quem vai receber esta mensagem
- O caminho mais simples é requerer que o receptor possua permissões
- Quando precisar enviar mensagens importantes é o melhor mecanismo

Service

- Formas de iniciar Services
  - startService(intent): inicia um serviço que fica rodando até a chamada do método stopService(intent) ou até que o próprio serviço termine
  - bindService(intent, conn, flags): tem por finalidade estabelecer conexão com um serviço já em execução.

Service

- Caso precise fazer chamadas importantes, como armazenar uma senha ou mensagens privadas é necessário validar se está conectado ao service correto ou a um programa hostil.
- É possível explicitar o componente que está tentando conectar

ContentProvider

- Objetivo: Permitir que determinadas informações sejam públicas para qualquer aplicação
- Permite consultar, inserir, alterar e excluir informações

Criando um ContentProvider

- Criar uma subclasse de android.content.ContentProvider
- Declarar no AndroidManifest, utilizando a tag <provide>
- É Possível dar permissão de leitura e escrita de forma independente em um Provedor
- android:readPermission
- android:writePermission

ContentProvider

- Permite revogar dinamicamente acesso a outros programas
- GrantURIPermission() e RevokeURIPermission()

ContentProvider -SQL Injection

- Para impedir SQLInjection, requerentes devem delinear claramente entre declarações Sql e os dados que eles incluem
- O SQLInjection é evitado no Android usando consultas parametrizadas que distinguem os dados da lógica de consulta.
- Deve-se usar tipos parametrizados para todos os valores que você se refere e nunca usar concatenação de string para gerar o seu SQL.

ContentProvider – SQL Injection

```
Uri uri = getContentResolver().insert(People.CONTENT_URI,
values);

String[] projection = new String[]{ People._ID,People._COUNT,

                        People.NAME, People.NUMBE                    };

Uri contacts =  People.CONTENT_URI;

Cursor managedCursor = managedQuery(contacts,

                projection,

                null,       // rows

                null,       // arguments

                People.NAME + " ASC");
```

Arquivos e Preferências

- Sistema de arquivos Android semelhante ao Linux
- É possível usar constantes para definir permissões de arquivos: MODE_WORLD_WRITABLE | MODE_WORLD_READABLE

  fos = openFileOutput("PublicKey", Context.MODE_WORLD_READABLE);
- Verificar a importância da informação quando for atribuir permissões aos arquivos

Armazenamento

- Dados críticos da aplicação devem ser armazenada na memória interna.
- Caso o volume de informação seja grande, é recomendado o uso de criptografia para armazenamento no SDCard (formato VFAT)
- VFAT: sistema antigo que não suporta os controles de acesso do Linux, deixando os dados armazenados no cartão sem proteção.

Questões de Segurança

- ASLR implementado na versão 4.0 (previne estouro de memória)
- Rootkit

  http://upche.org/doku.php?id=wiki:rootkit
- Virus

  Work in progress: DroidDream
- Geimini (final de 2010)

  Uma botnet criada a partir de telefones com os SO Android.

Conclusão

- Conhecer os mecanismos disponíveis para comunicação correta entre aplicações
- Se manter atualizado sobre os vírus e malwares que surgem
- Validar sempre os dados de entrada de uma app
- Aproveitar este crescente mercado de Android

Referências

- LECHETA, Ricardo R.Google Android-Aprenda a criar aplicações para dispositivos móveis com Android SDK, novatec, 2ed.2010
- Burns, Jesse.DEVELOPING SECURE MOBILE APPLICATIONS FOR ANDROID.
- http://developer.android.com
- http://www.erisvaldojunior.com
- http://www.ibm.com/developerworks/br/library/x-andro

Contato

- E-mail: anaclaudiaa@gmail.com
- Twitter: @piscianac

Obrigada!