ON DEMYSTIFYING THE ANDROID APPLICATION FRAMEWORK: RE-VISITING ANDROID PERMISSION SPECIFICATION ANALYSIS

Michael Backes, Sven Bugiel, Erik Derr, Patrick McDaniel, Damien Octeau, Sebastian Weisgerber

Luana Villwock Silva

Motivação

- **Aplicações Internas do Framework e suas influências na plataforma de segurança e privacidade ainda são uma caixa preta.**
- Como o que esta no framework influência na plataforma de segurança e privacidade do usuário.
- **Toda a análise de segurança requer uma fundação sólida.**
  - Como analisar o objetivo em primeiro lugar?
  - Qualquer uma das peculiaridade de determinada plataforma impede a análise estática?

Motivação

- **Vários trabalhos estabelecidos nessa área para aplicativos**
- Entry points (chex, FlowDroid)
- Generation of static runtime models (FlowDroid, R-Droid, R-Droid, Epicc)
- Soucers/Sinks (SuSi)
- **Ainda, falta conhecimento para aplicação do framework**
- Serviços do sistema fornecem funcionalidade central
- O conhecimento existente para aplicativos não pode ser transferido.

Contribuição

- **Metodologia sistemática em como analisar estaticamente a aplicação do framework**
- Como enumerar os pontos de entrada do framework
- Como gerar um modelo estático em tempo de execução.
- **Revistando análise de especificação de permissão**
- Mais precisamente mapeando permissões para SDK/framework
- **Estudar internamente o sistema de permissões do Android**
- Como classificar operações sensíveis protegidas por verificações de permissões
- Onde as permissões são verificadas?

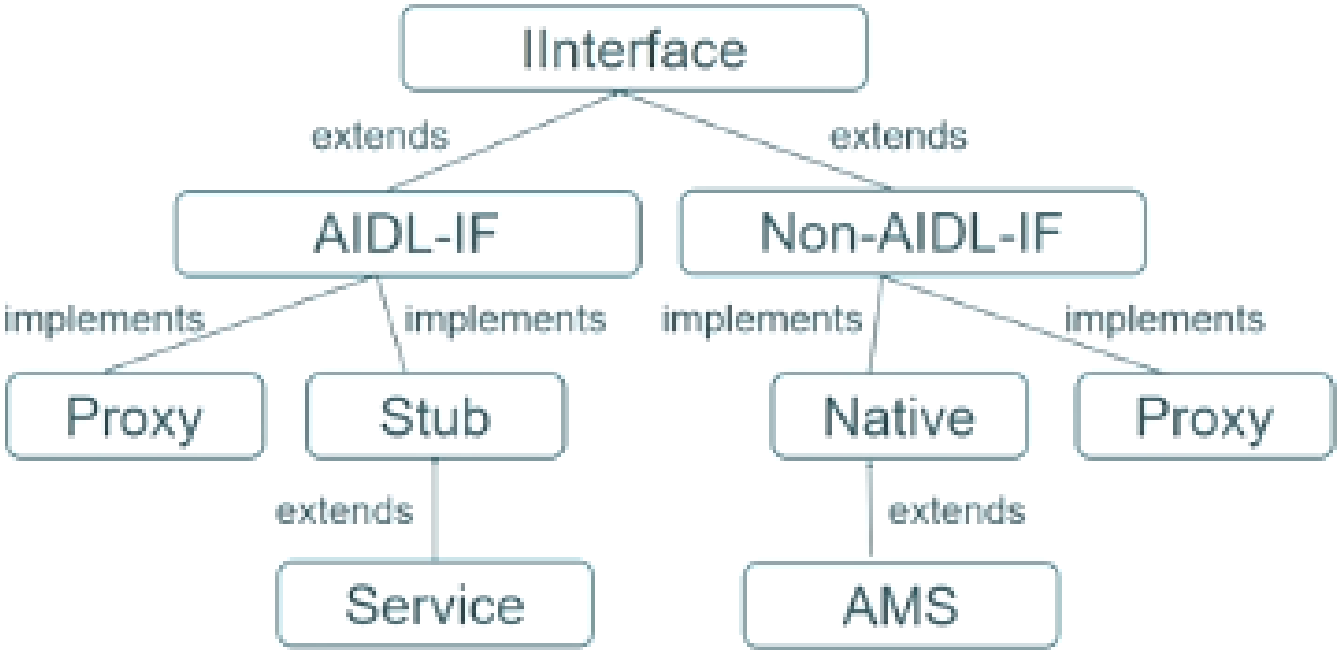
Como analisar estaticamente o framework

Maiores desafios da análise

- #1 Como enumerar os pontos de entrada do framework?
- #2 Como gerar um modelo estático que se aproxima do comportamento em tempo de execução?
- #3 Quais são as sensitive sinks do framework?

#1 Como enumerar os pontos de entrada do framework?

- **Quais funcionalidades são expostas na camada de aplicativos?**
- Observação fundamental: Funcionalidade exposta apenas via Binder-IPC.
- Enumeração de classes de entrada através da análise de hierarquia de classes

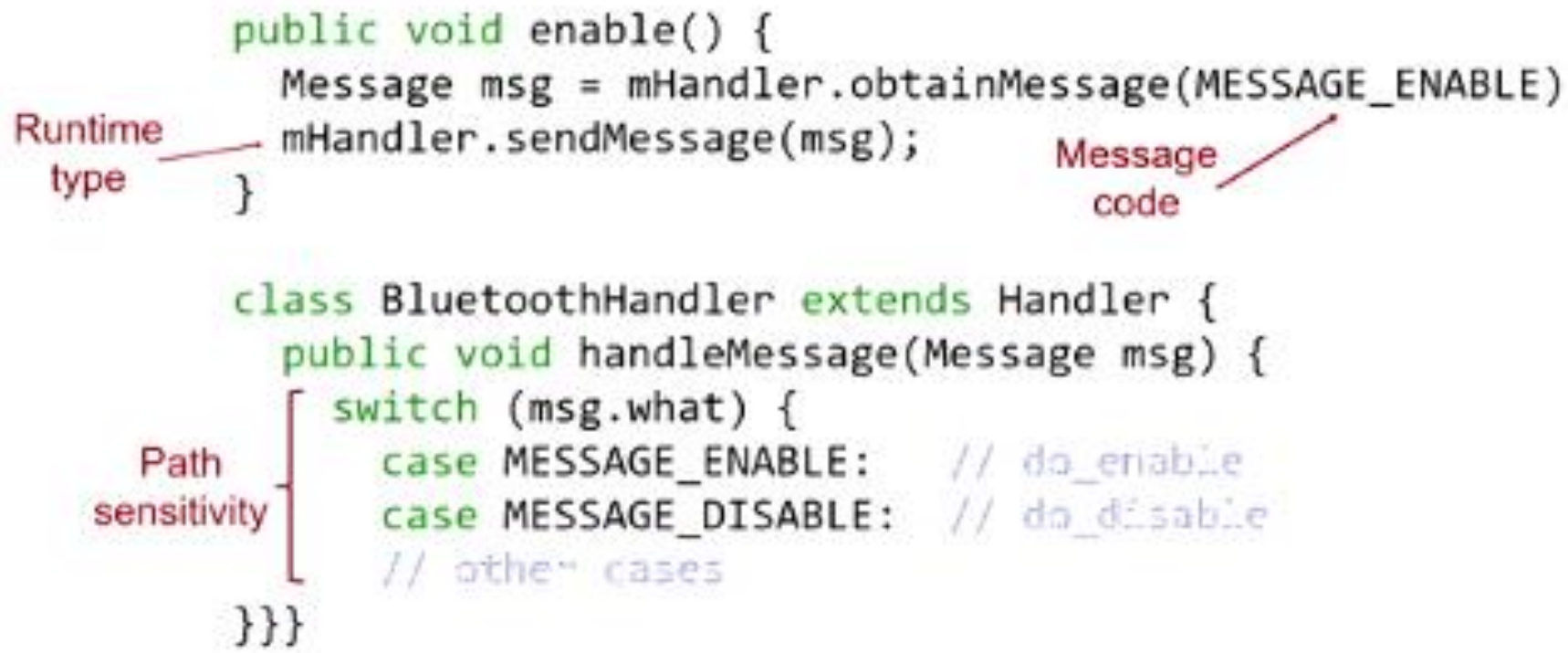


Modelo estático em tempo de execução (#2)

- **Serviços de framework seguem o principio de separação por tarefas**
- **Altamente sensível para processar consultas simultâneas de vários clientes (apps)**
- **Vários padrões de concorrência que complicam a analise estática**
- Handler
- AsyncChannel (apenas framework)
- Maquinas de Estado (apenas framework)

Handler

- Vários serviços tem handler dedicado para processar massagens em uma thread separada

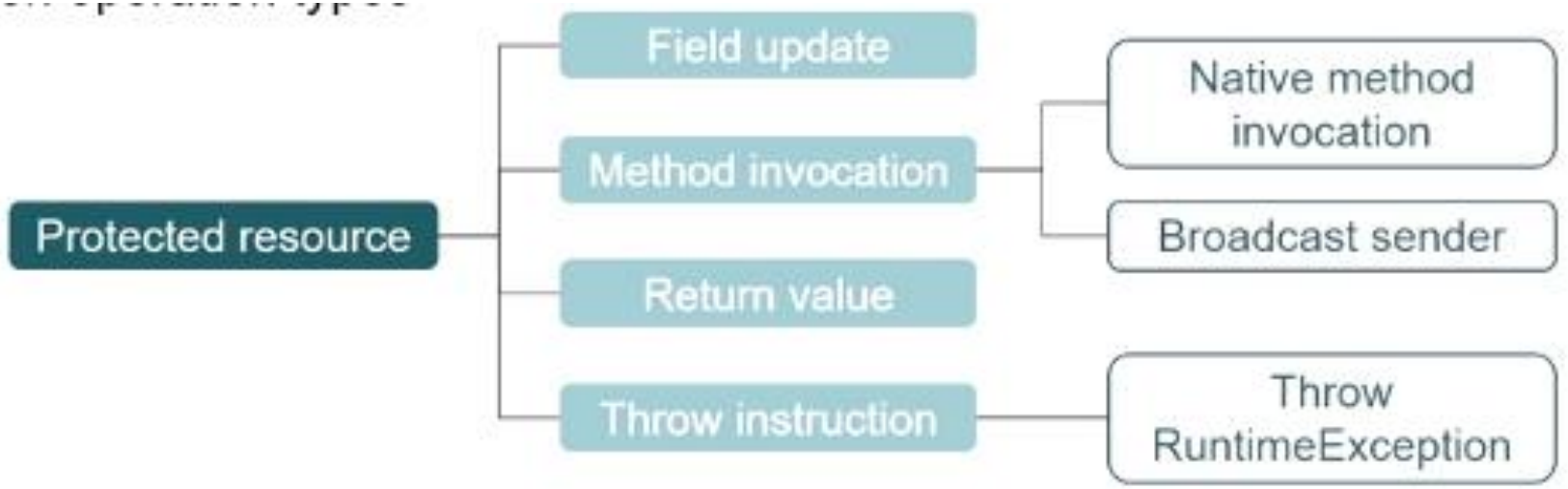


Recursos Protegidos (#3)

- **Conceito de recursos/sink como uma lista não é mais aplicável**
- Analise muda dentro do framework em relação API
- **Como se classifica uma funcionalidade sensitiva?**
- Considere a verificação de permissão como guardas das operações sensíveis
- **Proteger recursos são operações sensíveis a segurança que tem um efeito colateral tangível sobre**
- O estado do sistema
- Uso de privacidade

Taxonomia de tipos de recursos protegidos

- Nenhuma verdade absoluta ainda, entretanto foram investigados 35 pontos de entrada de serviços diferentes
- Diversidade de operações os forcou a criar uma classificação alto nível nos tipos de operações.



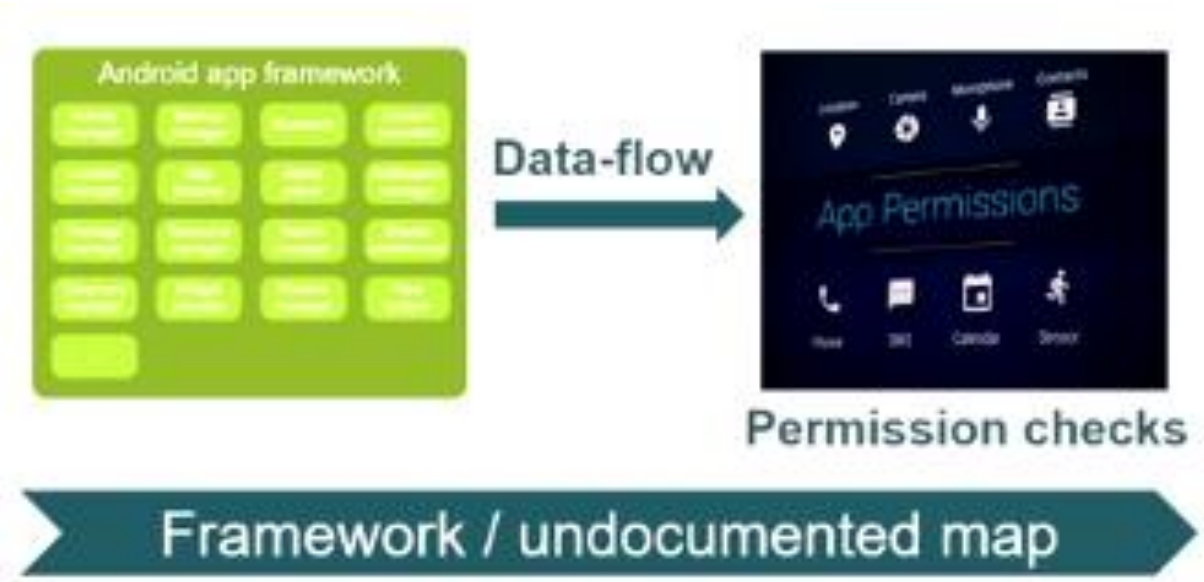
Casos de uso: Analise de permissões

Maior esforço = Melhor resultado?

- Gerar gráficos precisos requer muitos recursos
- Revisitando mapeamento de permissões Android
- Porque? Ainda é um dos principais mecanismos de segurança
- Importante para desenvolvedores de aplicativos e pesquisa em segurança
- Compara-se com a ferramenta de estado da arte Pscout(API 16)

Mapeando Permissões Android – Framework

- Mapa de pontos de entrada do framework para requisição de permissões.
- Abordagem: corte de controle de avanço
- Analise de String para resolver permissões de string



Framework entry point →	List of required permissions
com.android.phone.PhoneInterfaceManager.getDeviceId() →	android.permission.READ_PHONE_STATE

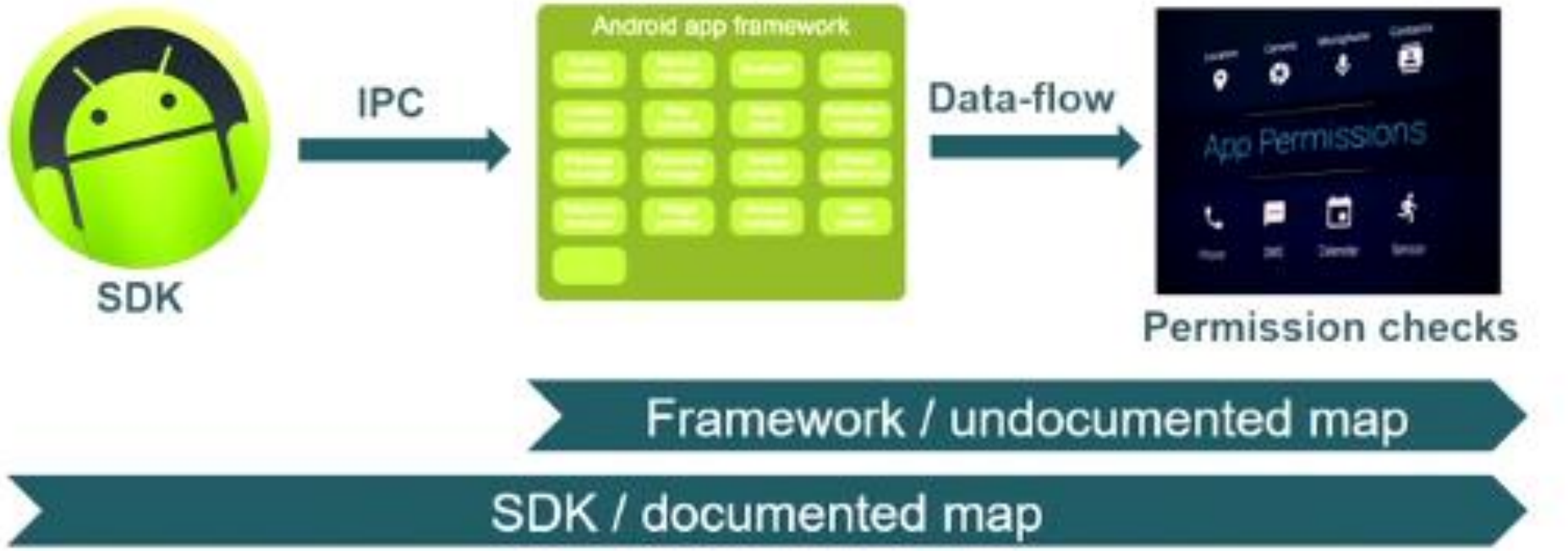
Framework API Mapping

- # Quantidade total Mapas de permissão de API
- Pscout inclui permissões normal + perigosas
- Axplorer inclui adicionalmente permissões de system + systemOrSignatures

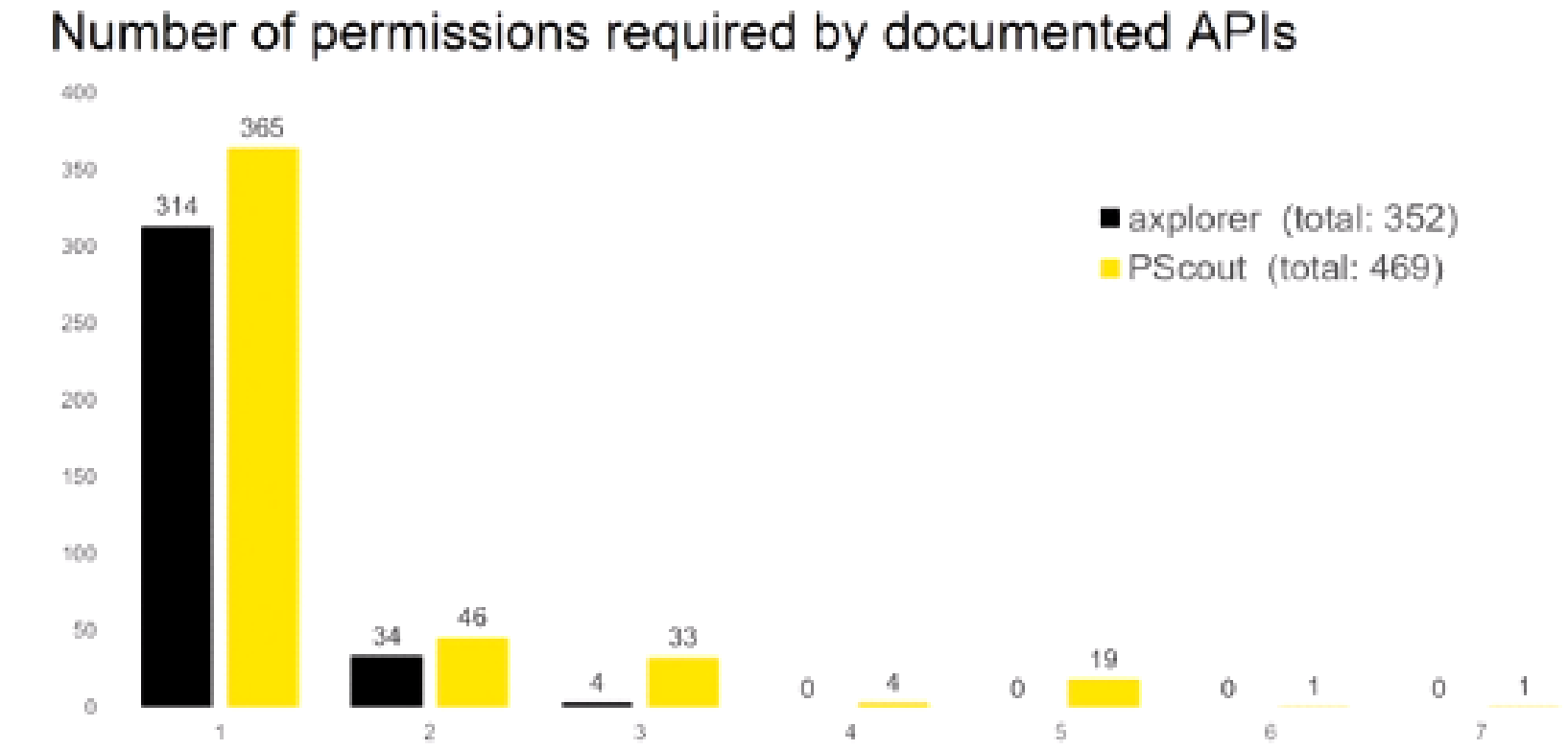
Framework API Mapping

- Menos falsos positivos
- Reduziu excesso de aproximação através de gráficos mais precisos
- Definição de pontos de entrada assegura mapas válidos

Android Permission Mappings - SDK



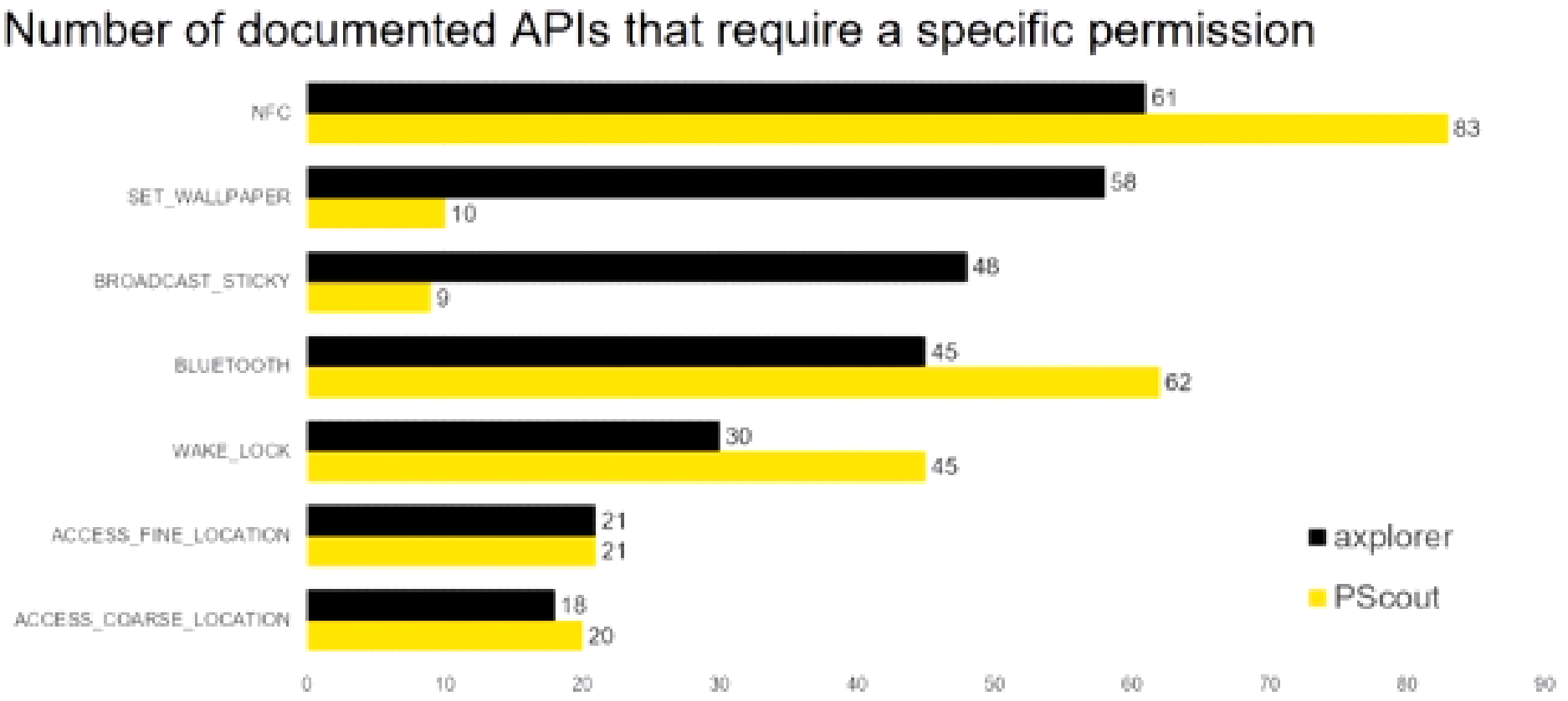
SDK Mapping (1)



SDK Mapping (1)

- Conectando SDK no framework elimina falsos mapeamentos
- Mapeamentos com métodos sem entradas são descartados
- Path-sensitivity no Handler elimina valores extremos

SDK Mapping (2)



SDK Mapping (2)

- Validou-se manualmente a 4 top permissões
- Diferenças devido à análise SDK
- Contexto de classes difíceis de acertar (mais de 100 diretas e indiretas subclasses)

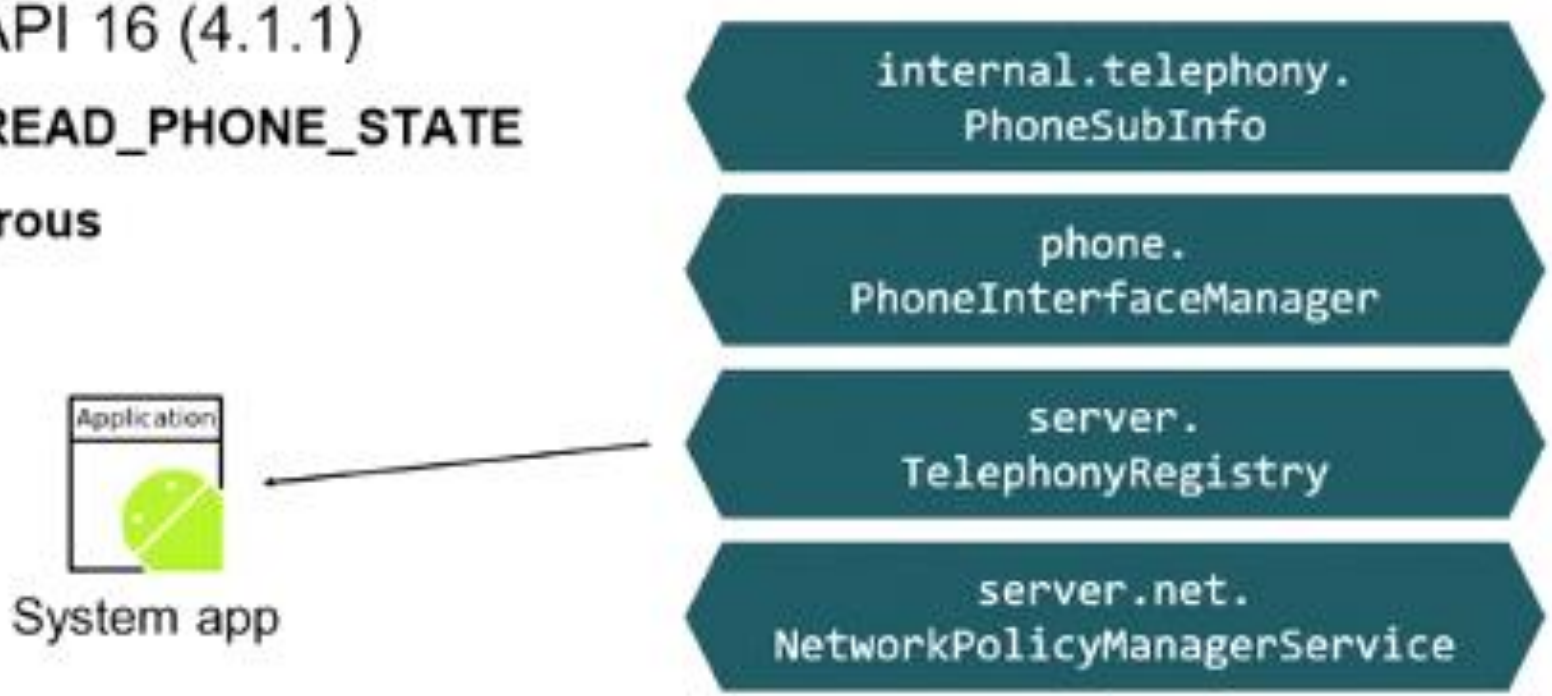
Permissões locais

- Serviços seguem o principio de separação de tarefas
- Como as verificações de permissão são distribuídas
- Através das versões de API ~20% das permissões são verificadas em >1 classe e no máximo em 10 classes
- Isso afeta igualmente todos os níveis de proteção (perigosas, sistema, ...)
- Há uma tendência de mais verificações em mais classes em versões mais novas do Android

Permissões locais

- Localidade é medida em termos de numero de classes distintas que verificam a permissão dada
- **Alta permissão de localidade**
- Permissão é verificada/executada em um único serviço.
- SET_WALLPAPER é apenas executada em WallpaperManagerService
- **Baixa permissão de localidade**
- Permissão é executada em diferentes (possivelmente não relacionados) serviços

- Framework API 16 (4.1.1)
  - Permission: **READ_PHONE_STATE**
  - Level: **dangerous**



- Framework API 22 (5.1)
  - Permission: **READ_PHONE_STATE**
  - Level: **dangerous**

- Localidade constante diminui entre as novas versões do Android
- Impedem de entender a “big Picture” das permissões do Android
- Ponto de execução único para permissões?
- Facilita a politica de geração para controle de acesso do framework (ASM/ASF)
- Como estabelecer?
- Identificar o dono da classe/serviço para cada permissão
- Método dedicado de verificação de permissão que é exposto via Interface

Conclusão

- Metodologias compreensivas e sistemáticas em como analisar a aplicação do framework Android
- Primeira classificação alto nível de tipos recursos de proteção
- Revisitar analise de permissões
- Melhorando resultados anteriores de mapeamento de SDK/framework
- Permissões locais melhoram o entendimento de permissões do sistema.
- Veja: www.axplorer.org

Referencias

- Michael Backes, Sven Bugiel, Erik Derr, Patrick McDaniel, Damien Octeau, Sebastian Weisgerber; **ON DEMYSTIFYING THE ANDROID APPLICATION FRAMEWORK: RE-VISITING ANDROID PERMISSION SPECIFICATION ANALYSIS**