A Message Encryption System Architecture for MeeGo Mobile OS

Anton Ovseenko, Vitaly Petrov

Department of Communications Engineering

Tampere University of Technology

Agenda

- **Motivation**
  - Data leak and market analysis
  - Present solutions drawbacks
- **Qt Message Framework Architecture (MeeGo)**
- **Proposed solution**
  - Two technical approaches
  - Fast message search algorithm
- **Solution overview**
  - Advantages / disadvantages
  - Conclusions

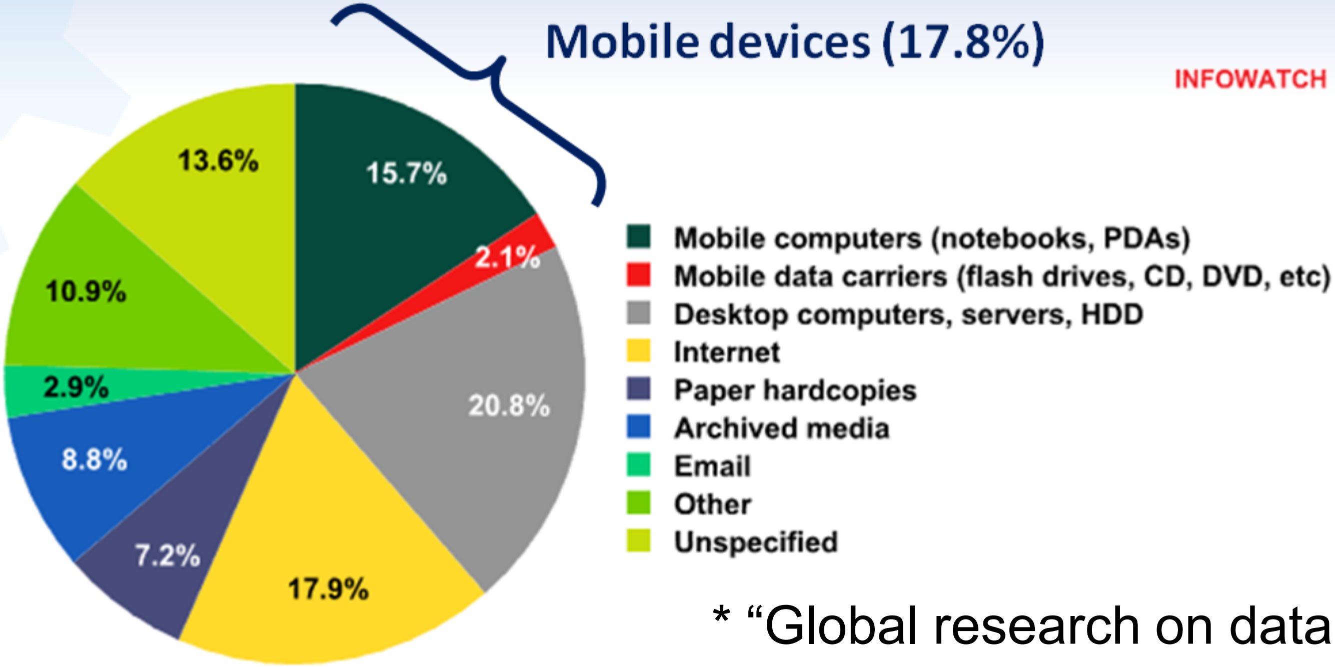
Motivation

- Smartphone, mobile device
  - Lost
  - Stolen
  - Lent

- What to protect?
  - Personal correspondence (SMS, E-mail)
  - Contacts
  - Notes (tasks, passwords)

Complex solution required

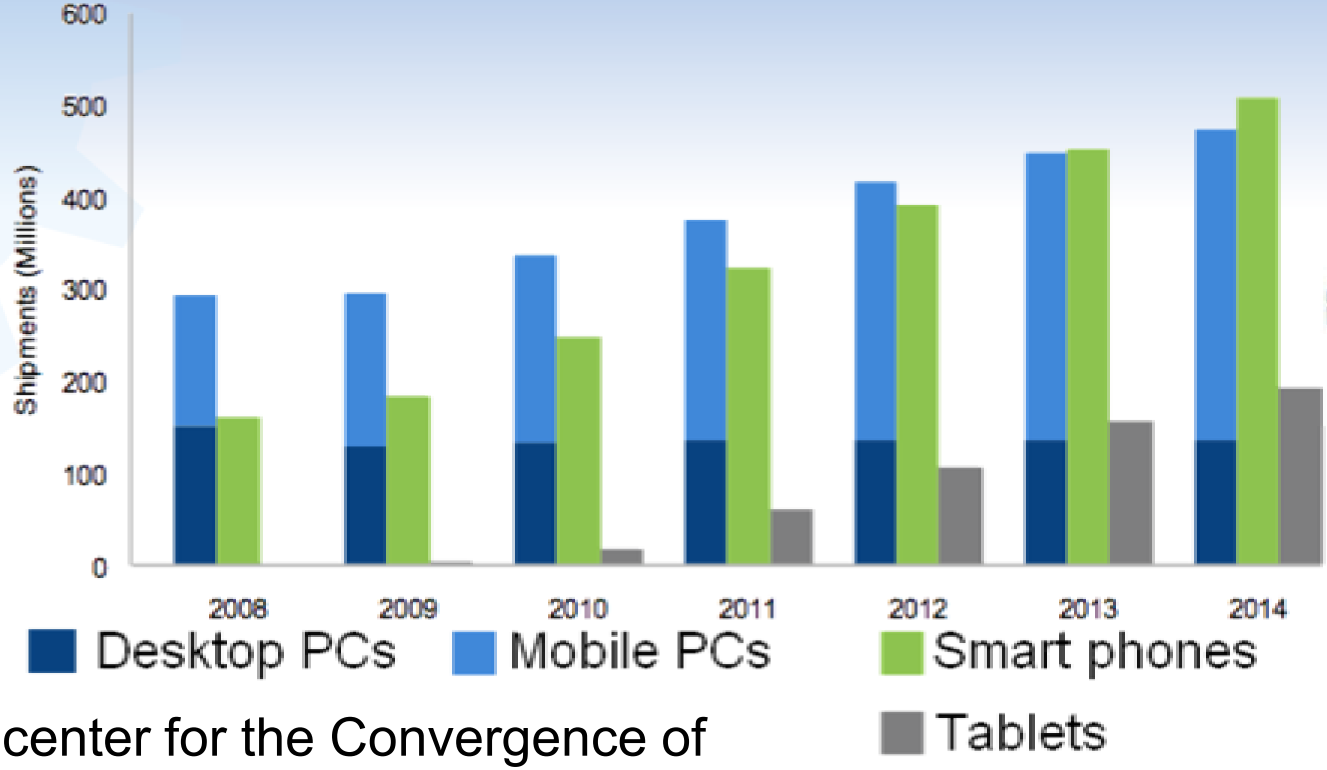
Data leaks overview



* “Global research on data leaks”, Infowatch report, 2009

Problem is topical

Market analysis



“*Epicenter for the Convergence of Smart Phones and PCs”, Wireless Communications Q4 Special Report, 2010

Business usage also

Present solutions drawbacks

- **Usability issues**:
  - Own GUI
  - Additional OS confirm dialogs
  - Weak functionality

- **Security issues**:
  - Own encryption algorithm
  - High security overhead
  - Complexity increase

* “ProtectedSMS”, “SMS-Pro”, etc.

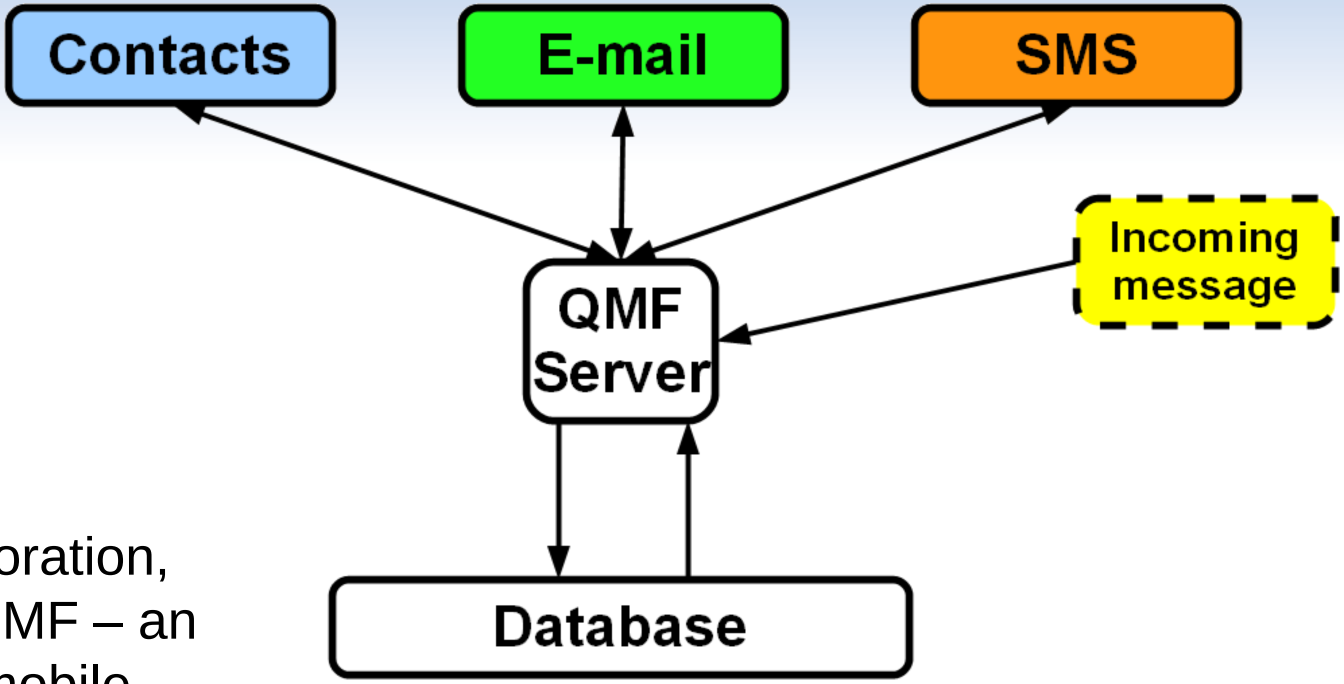
Complex solution required

Proposed solution

- Encryption
  - Contacts
  - Messages (SMS, E-mail, etc.)
  - Notes
- Integration with MeeGo Message Framework
  - GUI absence
  - Present solutions usage
- Fast key word search algorithm

Complex solution proposed

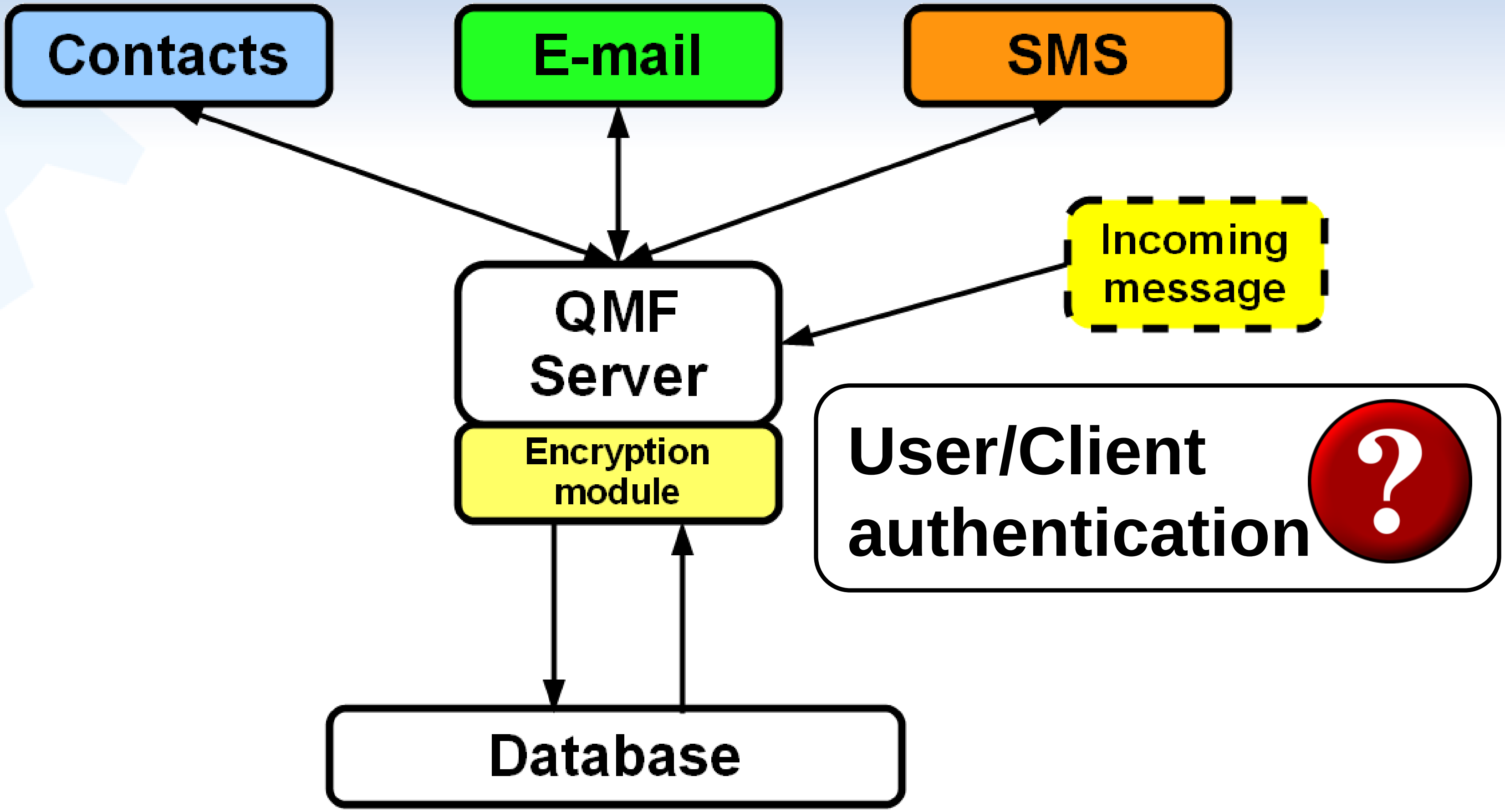
Qt Message Framework (MeeGo)



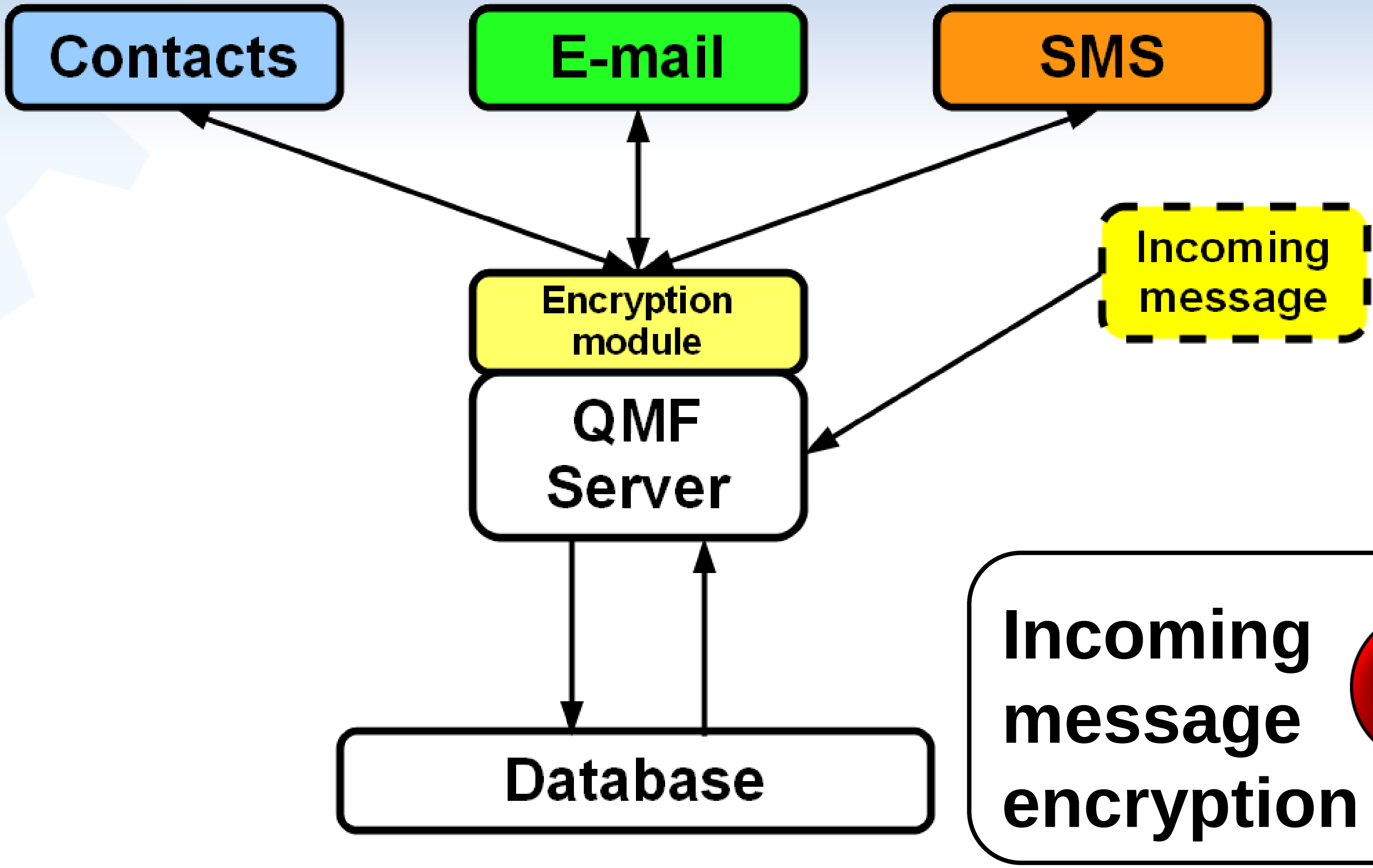
* Nokia Corporation, "Introducing QMF – an advanced mobile messaging framework,“ 2009

Client / server architecture

Approach 1. QMF protocol plugins modification

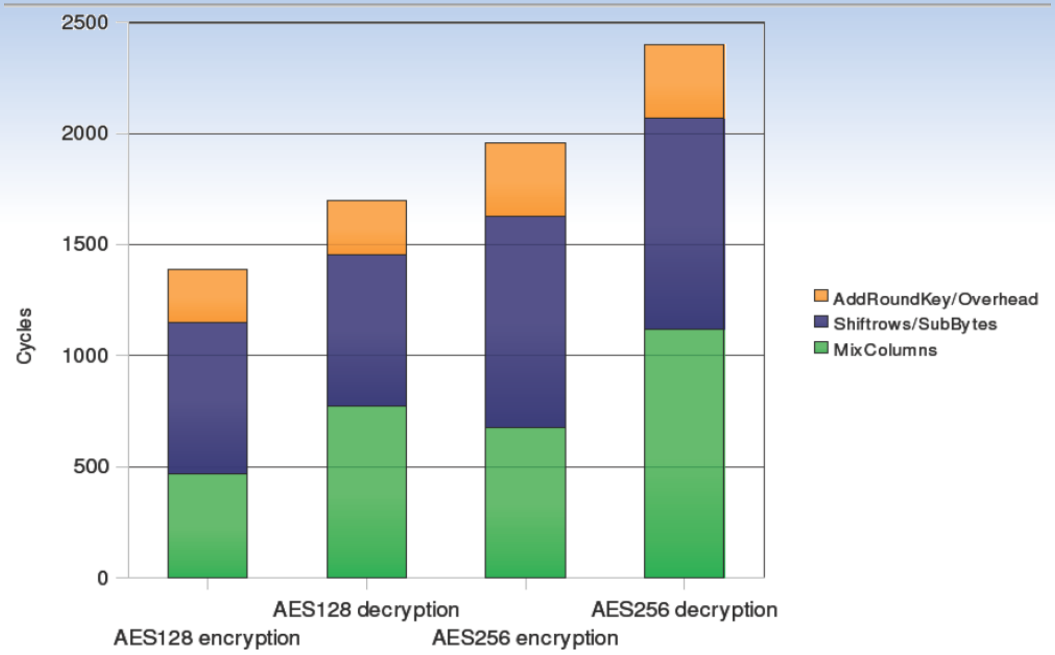


Approach 2. QMF clients modification



Incoming message encryption ?

Message encryption computation complexity estimation



* Øivind Ekelund, “Low Energy AES Hardware for Microcontroller”, M. Sc. Thesis, Norwegian University of Science and Technology, 2009

$$T = \frac{2500\left(Cycles\right) \cdot \frac{\sim 15\left(MBytes\right)}{32\left(Bytes\right)}}{10^{9}\left(Hz\right)} \approx 1.17\left(Sec\right)$$

Fast search algorithm required

Fast key word search algorithm

Update algorithm:

$$ID^i = ID_{i_1}, \ldots, ID_{i_n}$$

Message links array for key word *i*

```
for all $k$ do
    $flag \leftarrow 0$
    for all $i$ do

        if $i = ID(m_{new})_k$ then
            $ID^i \leftarrow ID^i + ID_{m_{new}}$
            $flag \leftarrow 1$
        end if
    end for
    if $flag = 0$ then
        create $ID^k = ID_{m_{new}}$
    end if
end for
```

Meta-data usage, O(n) complexity

Conclusions.
Proposed solution overview

- Features:
  - ⊕ High scalability (single solution for different clients)
  - ⊕ User-friendly interface (no interface)
  - ⊕ Fast key world search algorithm
    - Increase the interaction level
    - Saves the battery life
  - High security level (AES-256 usage)
- Issues:
  - ⊖ Implementation complexity
    (QMF clients/plugins modification)
  - ⊖ Meta-data usage
  - ⊖ Update algorithm complexity

The juice is not worth the squeeze!