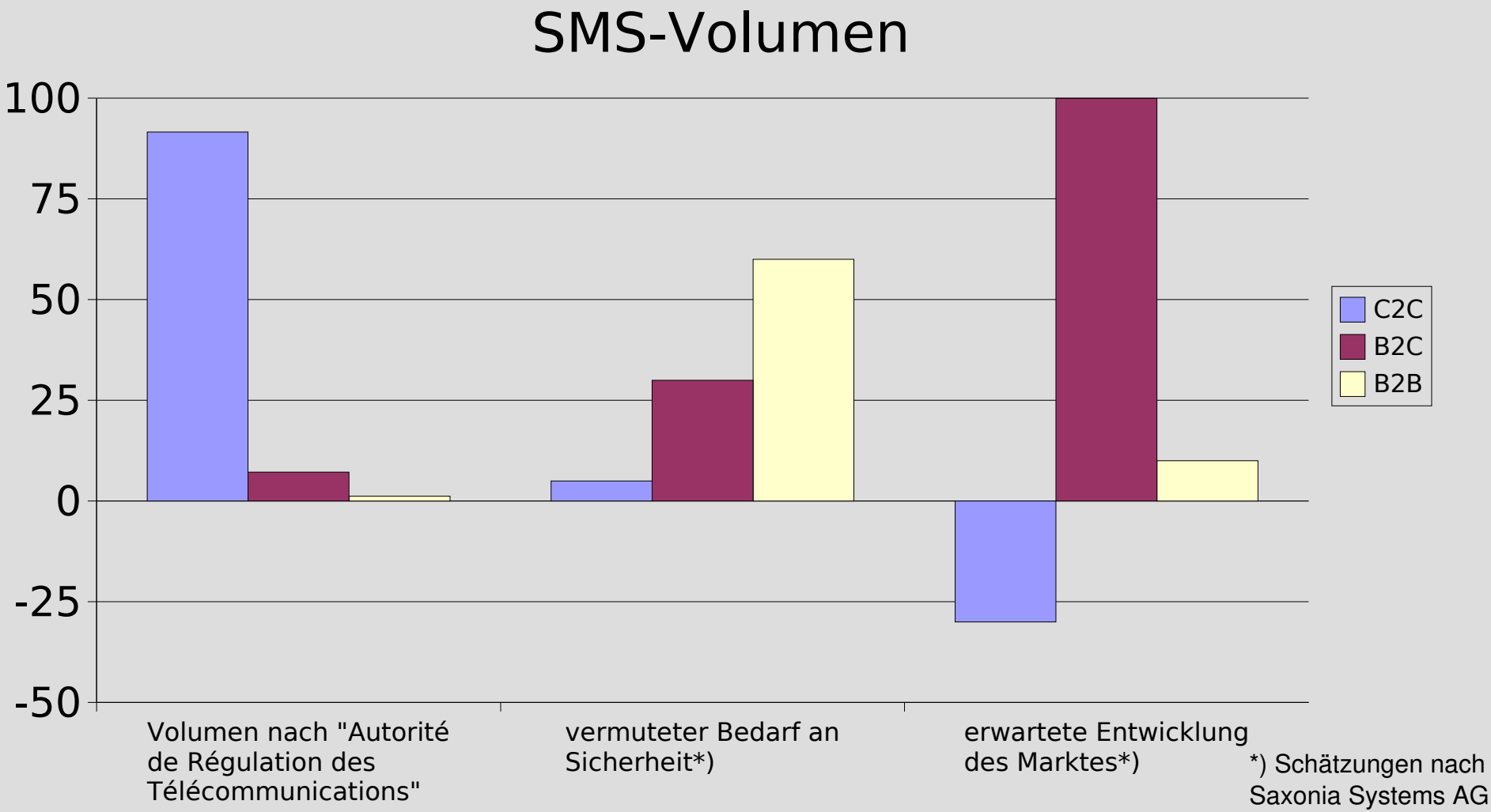
# Verschlüsselte SMS auf Basis des Standards SEMS

- Wer will verschlüsselte SMS?
- Was gibt es bis jetzt an Tools und Standards?
- Was ist SEMS?
  - Schlüssellänge – zwei Sicherheitsstufen
  - Operationsmodi
  - Kodierung
  - Schlüsselaustausch
- Probleme mit SEMS
- Verbesserungsvorschläge an SEMS

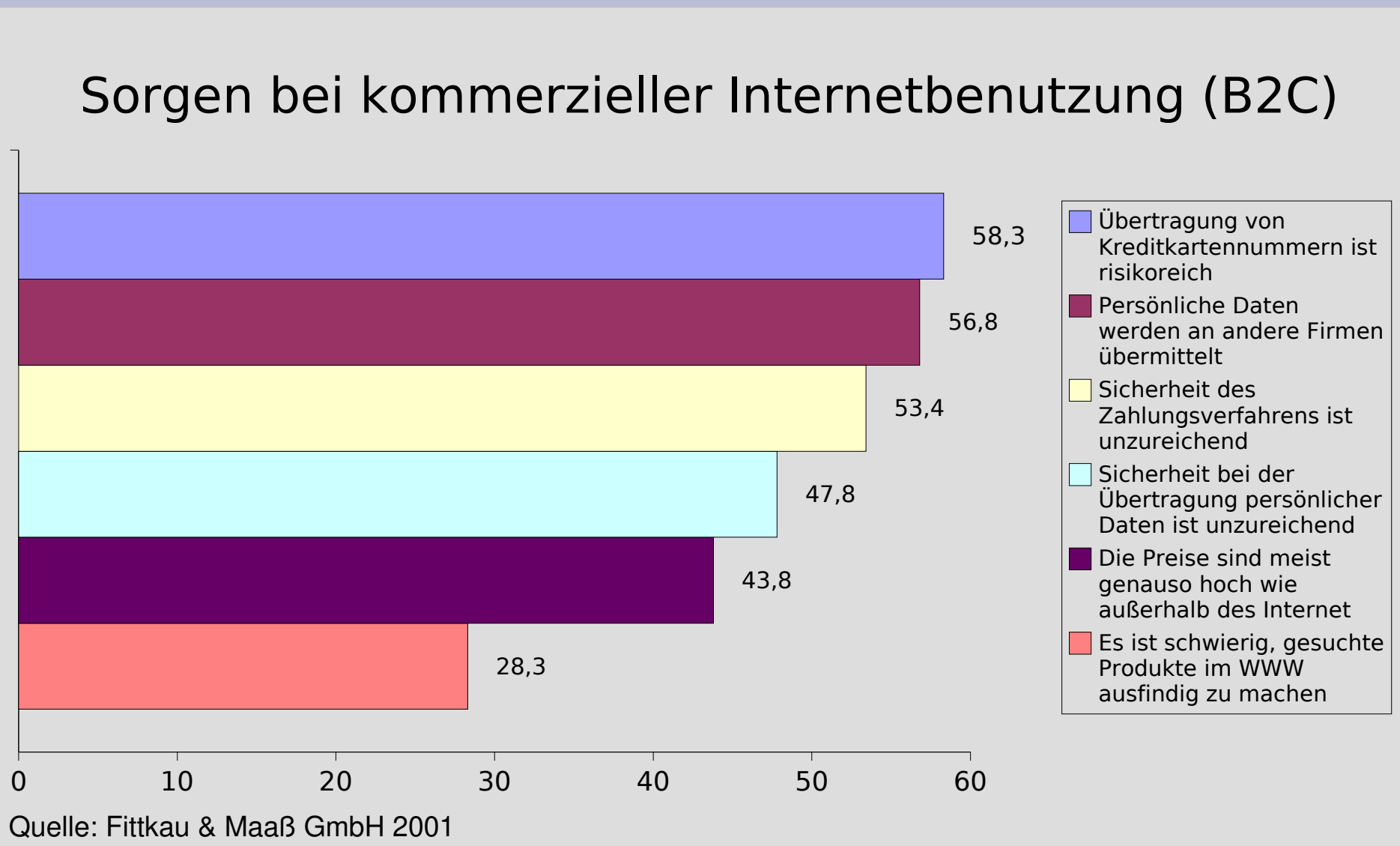
# Wer will verschlüsselte SMS?



# Anforderungen der Partner

- C2C
  - möglichst billig (wenig SMS)
  - möglichst einfach zu bedienen
- B2C („B2C ist die Zukunft der SMS“)
  - möglichst billig (Kosten sind beim Kunden)
  - einfach zu bedienen und zu verstehen (am besten transparent)
  - mehr Sicherheit durch zusätzliche Mechanismen
- B2B
  - Preis relativ egal
  - sehr hohe Sicherheitsforderung

# Wer will verschlüsselte SMS?



# Was gibt es bis jetzt an Tools und Standards?

- Symmetrische Verfahren
  - FortressSMS (AES, kein Keymanagment)
  - Benefon Message Encryption (?)
  - Quasigroups (unerforscht, kein Keymanag.)
- Asymmetrische Verfahren
  - SEMS (RSA direkt)
  - OpenServerMVS (OpenPGP)
- Naive Ansätze
  - Cäsar-Chiffre (http://www.bsvk.de)
  - T9-Encryption (http://www.esato.com/archive/t.php/t-53528)

# Was ist SEMS?

- Vorgeschlagener Standard der Universität Mannheim, NAH6 Amsterdam und der Computer System Group Vrije, Universität Amsterdam
- Protokolle für authentisierte und verschlüsselte SMS
- Optimiert auf Anzahl der zu übertragenen SMS
- RSA als einzige Verschlüsselungsfunktion (eine Ausnahme)

# SEMS - Sicherheitsstufen

- mittlere Sicherheit
  - RSA-Schlüssellänge von 1120 Bit
  - meist nur 1 SMS notwendig
- hohe Sicherheit
  - RSA-Schlüssellänge von 2208 Bit
  - mehrere SMS notwendig
- Interoperabilität zwischen Stufen wird gewährleistet

# SEMS - Operationsmodi

| | Verschlüsseln | Signatur | Schlüssel versenden |
|---|---|---|---|
| mittlere Sicherheit | 1 SMS | 1-2 SMS | 2 SMS |
| hohe Sicherheit | 2 SMS | 3 SMS | 3 SMS |

# SEMS - Operationsmodi

- Eigentlich viel komplizierter:

| | unverschlüsselt | verschlüsselt, mittlere Sicherheit | verschlüsselt, hohe Sicherheit |
|---|---|---|---|
| unsigniert | 1 SMS | 1 SMS | 2 SMS |
| “included” signiert, mittlere Sicherheit | 1 SMS | 1-2 SMS | 2 SMS |
| “included” signiert, hohe Sicherheit | 2 SMS | --- | --- |
| “detached” signiert, mittlere Sicherheit | 2 SMS | 2 SMS | 3 SMS |
| “detached” signiert, hohe Sicherheit | 3 SMS | 3 SMS | 3 SMS |

# Sinn von Operationsmodi

- Interoperabilität zwischen mittlere und hoher Sicherheit
  - Jeder kann an jeden verschlüsseln. Zahl der SMS hängt vom Empfänger ab.
  - Jeder kann Signatur leisten, unabhängig von der Verschlüsselung. Zahl der SMS hängt von der eigenen Stufe ab.
- Signaturtypen
  - included um die Zahl der SMS zu drücken
  - detached um versandte SMS im nachhinein zu signieren

# SEMS - Kodierung

- 8-Bit – binär („unkodiert“)
- 7-Bit – momentaner SMS-Standard (ASCII)
- 6-Bit – nur Großzeichen, Zahlen, +-.“ etc..
- 5-Bit – modifizierte Baudot-Kodierung
- Was soll das in einem Sicherheitsstandard?
  - Dateninterpretation muss festgelegt sein
  - 2 Fliegen mit einer Klappe (Komprimierung und Verschlüsselung)
  - SEMS ist eben nicht nur ein Sicherheitsstandard
- Aber: Bläht den Standard auf!

# SEMS - Schlüsselaustausch

- selbst definiertes Datenformat für Schlüssel
  - Option „send me yours“
  - angehängte Signaturen (bis 32 Stück)
  - Telefonnummer
  - Expiration date
  - Name (bis 64 7-Bit-Zeichen)
- key revocation
- Möglichkeit, bis zu 32 Signaturen anzuhängen

# Probleme mit SEMS

- Komplexität
  - viele Verschiedene Kombinationen
  - alles nochmal neu gemacht, kaum Nutzung von bestehenden Standards
- RSA direkt mit e=3
  - Coppersmith: kleinen Nachrichten
  - Hastad, Coppersmith: doppelte Nachrichten, linear abhängige Nachrichten
  - Franklin-Reiter, Blömer: Teile von d bekannt
  - ABER: Durch SHAKE gegen Coppersmith und Hastad sicher.

# Probleme mit SEMS (fort.)

- kein Platz mehr für user data header oder andere SMS-Erweiterungen
  - K.O.-Kriterium für Java Midlets
  - Lösung: Implementation im Betriebssystem oder Änderung des Standards
- lange Schlüssel (länger als symmetrisch)
- unübliche Schlüssellänge, damit keine gute Interoperabilität mit anderen Systemen (PGP, TLS/SSL)

# Verbesserungen an SEMS

- Komplexität
  - keine „detached signatures“ (Verzicht auf nachträgliches signieren...)
  - oder: keine „included signatures“ (bei mittlerer Sicherheitsstufe mehr SMS pro Nachricht)
  - eventuell: Verzicht auf mittlere Sicherheitsstufe
  - Schlüssel sind signierte SMS im ANS.1 Format
  - keine Kodierung in SEMS. Alles in 8-Bit.
- e=feste aber große Primzahl mit vielen 0

# Verbesserungen an SEMS

- Datenstrukturen modifizieren, so dass UDH von den Nutzdaten noch abgeht
  - Alternative: kein Java, sondern mit Anbieter-API
- symmetrische Schlüssel
  - Mit SEMS symmetrische Schlüssel automatisch austauschen. Ab dann nur symmetrisch
    - Aber: keine kürzeren Schlüssel, da Schlüsselaustausch noch mit SEMS-Schlüssel
  - Schlüsselaustausch nicht über SMS
- 1024 und 2048 Bit Schlüssel zulassen
  - Aber: „1024 Bit ist nicht mehr sicher“ (R.Weis)