UNIVERSITÄT OSNABRÜCK

Distributed Systems Group
https://sys.cs.uos.de

Over the Air Programming(OTAP) für Wireless Sensor Networks(WSN): Delta Updates

Luis-Pepe Kreienbrink

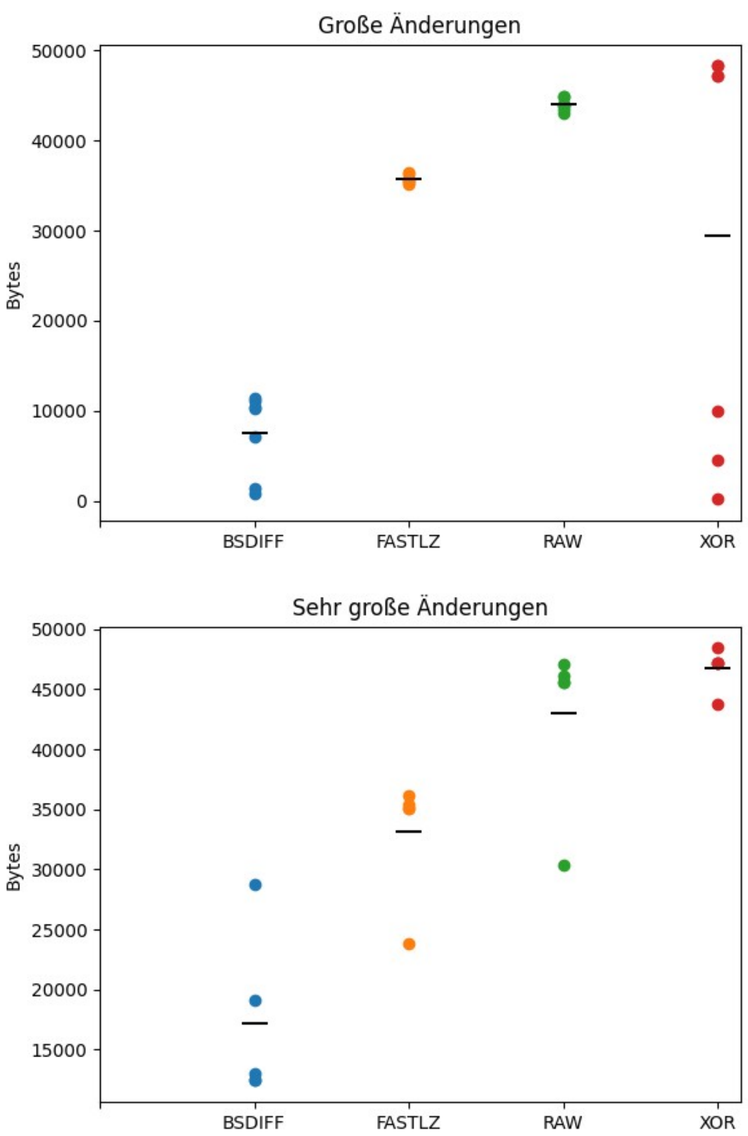
Einführung & Motivation

WSN bieten häufig nur geringe Bandbreiten, die durch z.B. große OTAP-Updates stark ausgelastet werden, um während des Update-Prozesses weiterhin die eigentliche Aufgabe des WSN ausführen zu können, ist die Minimierung der Updatepaketgrößen ein möglicher Ansatz. Dazu können Delta Updates – also es werden lediglich die Änderung zwischen zwei Versionen übertragen – oder Datenkompressionen verwendet werden. Dabei ist eine schnell und ressourcensparende Dekompression, aufgrund der Hardwarebeschränkungen, für die Wahl der Algorithmen entscheidend. Die Zentrale Fragestellung der Untersuchung war: Ist der Mehraufwand für Delta Updates gerechtfertigt? Dazu wurden die Paketgrößen (in Bytes) von vier Varianten verglichen:
1. Ein naiver XOR-Delta-Update Algorithmus: Auf die alte und neue Version wird XOR angewendet und anschließend der Bitstring komprimiert.
2. Bsdiff von Colin Percival: Ebenfalls zweiteilig: Delta-Bestimmung und Kompression des resultierenden Bitstrings
3. Kompression des Paketes: FastLZ(Level 2), ein LZ77 Kompressionsalgorithmus mit einem Fokus auf eine schnelle Kompression und Dekompression auf Kosten der Kompressionsrate; Wird auch also Kompressionsalgorithmus für 1. und 2. verwendet.
4. Die rein Binaries (RAW), die auch auf eine Node geflasht würden.

Laufzeiten & Ressourcenverbrauch (asymtotisch)

Die Auswahl der Verfahren basiert auf der Annahme, dass die Update Pakete auf dem Entwickler Computer erstellt werden, sodass die Laufzeiten und der Ressourcenverbrauch bei typischen Paketgrößen für WSN-Nodes nicht ausschlaggebend sind. Für den XOR-Delta-Update Algorithmus (ohne Kompression) kann angenommen werden, dass ein Speicher von n+m+O(1) Bytes, wobei n die Größe der alten Version und m die Größe der neuen Version ist, und eine Laufzeit von O(max(n,m)) benötigt wird. Für Bsdiff ist ebenfalls n+m+O(1) Bytes erforderlich, wohingegen die Laufzeit O(n+m) ist. FastLZ benötigt k + n + O(1) Bytes Speicher, wobei k der Größe des komprimierten Paketes entspricht.

Paketgrößen



Streudiagramme der Paketgrößen aufgeschlüsselt nach Varianten.
Die schwarzen Balken entsprechen den Mittelwerten .

Um unterschiedlich Updateszenarien zu untersuchen, wurden kleine Änderungen (2-3 Zeilen bzw. Strings verändert) im Programmcode, große Änderungen (weitere Funktionalitäten hinzugefügt bzw. ca. 50 Zeilen verändert) und sehr große Änderungen (Austausch des gesamten Programmlogik abgesehen von Framework oder Betriebssystem) an einem Referenzprogramm durchgeführt.
Die Programme wurden für die Tmote Sky mit contiki-ng als Betriebssystem kompiliert. Aufgrund dessen sind die resultierenden Binaries immer 48kb groß, jedoch benötigt der Programmcode nicht die gesamten 48kb Speicher, sondern nur ein Teil davon. Der wirklich benötigte Speicher ist als RAW dargestellt. Die Varianten 1.-3. verwenden jedoch immer die 48kb großen Binaries als Input. Deshalb sind einige Pakete für XOR größer als der wirklich Programmcode, jedoch sind keine XOR-Pakete größer als 48kb. Die Grafiken zeigen eindeutig, dass die Kompression mit FastLZ ca. 5-10kb kleinere Pakete im Vergleich zu den reinen Binaries liefert. XOR hingegen schwangt in den Paketgrößen sehr stark, sodass dieser Algorithmus sowohl die kleinsten Pakete (204 Bytes) aber auch die größten Pakete(48kb) erstellt. Scheinbar verschlechtert der XOR Ansazt sogar die Kompression von FastLZ. Interesannterweise sind die kleinen Paktete von XOR und Bsdiff die gleichen Basisbinaries. Bsdiff hingegen generiert in allen Szenarien deutlich kleinere Paktete (min. 5kb) im Vergleich zur reinen Kompression.

Zusammenfassung

Auf Basis der Auswertungen ist eindeutig, dass die Verwendung von Delta Updates sich zur Minimierung der Updatepaketgrößen eignet. Jedoch liefert der naive XOR Ansatz keine zuverlässigen Ergebnisse und ist deshalb ungeeignet. Außerdem benötigt Bsdiff kaum zusätzlichen Speicher im Vergleich zu OTAP ohne Delta Updates und um ein bestehendes OTAP mit Delta Updates zu erweitern, werden lediglich 300 LoC benötigt. Die Delta Updates wurden in contiki-ng getestet und implementiert, können jedoch auf jede weitere Plattform übertragen werden, da sie keinerlei contiki-spezifischen Code beinhalten, sondern lediglich vereinzelte Funktionen aus der Standard C Libary verwendet.
Für weitere Untersuchungen wäre interessant, ob die Performance von Bsdiff verbessert werden kann, indem andere Kompressionsalgorithmen verwendet werden und wie gut Bsdiff im Vergleich zu anderen Delta-Update-Algorithmen ist.

Weitere Informationen

Quellenangaben:
https://ariya.github.io/FastLZ/ (Basis für FastLZ Implementation)
http://www.daemonology.net/bsdiff/ (Details zu Bsdiff)
https://github.com/thoughtpolice/minibsdiff (Basis für Bsdiff Implementation)
https://github.com/Blobfishman/contiki-otap (Details zur Umsetzung)
https://www.win.tue.nl/~mstolikj/compression/ (Weitergehende Untersuchungen zu Kompressionen in contiki-os von Milosh Stolikj)