12. November 2018 Studie · Mobile Sicherheit · Web & Cloud

Katastrophenkommunikation mit Mesh Software

Archiviert. Dieser Inhalt wird nicht mehr aktualisiert. Er bleibt zur Information verfügbar.

Stand 2018-11-12
Quelle https://technology2.a-sit.at/veroeffentlichungen/katastrophenkommunikation-mit-mesh-software/

Dieses Projekt demonstriert den Aufbau eines Mesh-Netzes sowohl mit stationären als auch mit mobilen Einheiten auf Basis kostengünstiger Hardware und frei erhältlicher Software. Die Verbindung unter allen Stationen erfolgt drahtlos, um möglichst autark zu bleiben. Es wird das bereits bestehende „Serval"-Projekt als Grundlage genommen, um eine Kommunikationsmöglichkeit zwischen den stationären Einheiten erweitert sowie der Android Client für die Ausweitung des Mesh-Netzwerkes auf modernen Android Versionen adaptiert.

1. Motivation

Kommunikation ist besonders im Katastrophenfall von enormer Wichtigkeit, um effizient Hilfe zu leisten und Betroffene sowie Hilfskräfte zu koordinieren. Fest installierte Infrastruktur wie zum Beispiel Mobilfunksendemasten sind meist abhängig von externen Faktoren, etwa Strom aus dem regionalen Versorgungsnetz, um ihren Service aufrechtzuerhalten. Im Katastrophenfall kann diese externe Versorgung vorübergehend ausfallen, indem etwa Kabel durch einen Erdrutsch mechanisch versagen oder Verteilerstationen überschwemmt werden. In solch einem Szenario besteht die Notwendigkeit eines schnell verfügbaren, möglichst autarken Ersatzes. Hier bietet es sich an, durch das Aufstellen von Routern, die mittels Akku versorgt werden, das Kommunikationsnetz vorübergehend wiederherzustellen. Diese Router bilden die Basis eines Mesh-Netzwerkes, welches mit Hilfe von im betroffenen Gebiet verfügbaren Smartphones noch erweitert werden kann und somit eine großflächige Verfügbarkeit einer Mobildatenverbindung gewährleistet.

© A-SIT, Partner für Sicherheitsfragen

2. Vorhandene Basis

Die Hardware für den Testaufbau umfasst zwei Raspberry Pi [1], die jeweils mit zwei handelsüblichen W-LAN Adapter ausgestattet sind. Diese Wahl der Plattform bietet ausreichende Rechenleistung und Erweiterbarkeit bei gleichzeitig breiter Verfügbarkeit und einem Preis unter 100€ pro Stück. Zwischen diesen Stationen wird ein Netzwerk mit Mesh-Architektur errichtet. Bei dieser Architektur ist jeder

Netzwerkknoten mit mehreren anderen verbunden, Informationen im Netzwerk werden über diese Knoten weitergeleitet, bis das Ziel erreicht ist. Mesh-Netzwerke zeichnen sich durch gute geografische Abdeckung bei guter Übertragungsgeschwindigkeit aus. Bei Ausfall von Knoten heilt sich das Netz selbst, die Informationen können ebenso über andere Knoten übertragen werden. Auch können weitere Knoten problemlos in ein bestehendes Netz eingefügt werden. Ein Nachteil dieser Architektur ist das vergleichsweise komplexe Weiterleiten von Netzwerkpaketen und die hohe Netzwerkaktivität der einzelnen Knoten. Die softwareseitige Basis dieses Projekts bildet die existierende Mesh-Chat-Software ‚Serval' [2], da bereits Probleme wie das Implementieren des Mesh-Protokolls und die Übertragungssicherheit der Nachrichten gelöst sind. Derzeit gibt es bereits Open-Source Clients für Android und iOS, die unter GPLv3 verfügbar, jedoch inkompatibel zueinander sind. Grundlage dieser Apps bildet der „servald" Daemon [3], der das Mesh-Netzwerk aufbaut und Netzwerkpakete darüber überträgt. Obwohl es innerhalb des Serval Projects bereits Ambitionen gab, einen experimentellen Range Extender zu bauen, wurde dies nur mit einem Router durchgeführt, eine Kommunikation unter den Routern schien nicht vorgesehen zu sein.

3. Projektziel

Mittels weithin verfügbarer Hard- und Software soll der Aufbau einer Mesh-Basisstation demonstriert werden. Zwei dieser Basisstationen sollen sowohl eine Verbindung zueinander halten, sowie als lokaler Access Point für Smartphones dienen und den Austausch von Nachrichten über die Basisstationen ermöglichen. Dafür soll auf jedem Raspberry Pi die Linux-Distribution Raspbian [4] und der „servald" Daemon installiert werden, um die Kommunikation mit dem „Serval"-Client auf Android zu ermöglichen. „servald" soll dabei so konfiguriert werden, dass die Eingliederung des Raspberry Pi als Knoten in ein bestehendes Mesh-Netzwerk möglich ist. Die Verbindung der Raspberry Pis untereinander soll durch das Routingprotokoll „B.A.T.M.A.N.adv" [5] gewährleistet werden.

4. Aufbau und Erweiterung durch mehrere Basisstationen

Der Testaufbau beinhaltet zwei Raspberry Pi, die jeweils mit zwei W-LAN Adapter ausgerüstet sind, und zwei Android Smartphones. Jeder Raspberry Pi stellt über einen W-LAN Adapter einen lokalen Access Point bereit, mit welchem jeweils ein Smartphone verbunden ist. Der zweite W-LAN Adapter ist für die Verwendung mit „B.A.T.M.A.N.adv", das als Modul im Linux-Kernel verfügbar ist, konfiguriert, so dass die Verbindung zwischen den Raspberry Pi drahtlos zu Stande kommt und neue Basisstationen bei Bedarf hinzugefügt werden können. Auf beiden Android Smartphones ist der „Serval"-Client „batphone" installiert, welcher eine grafische Oberfläche zum Versenden und Empfangen von Nachrichten und Anrufen bietet. Sowohl Nachrichten als auch Anrufe nutzen zur Datenübertragung „servald", das Teil der Android App ist. „servald" am Smartphone nutzt den eingebauten W-LAN-Netzwerkadapter, welcher eine Verbindung zu einem Raspberry Pi Access Point hält.

Die Kommunikation zwischen den einzelnen Komponenten ist in Abbildung 1 schematisch dargestellt.

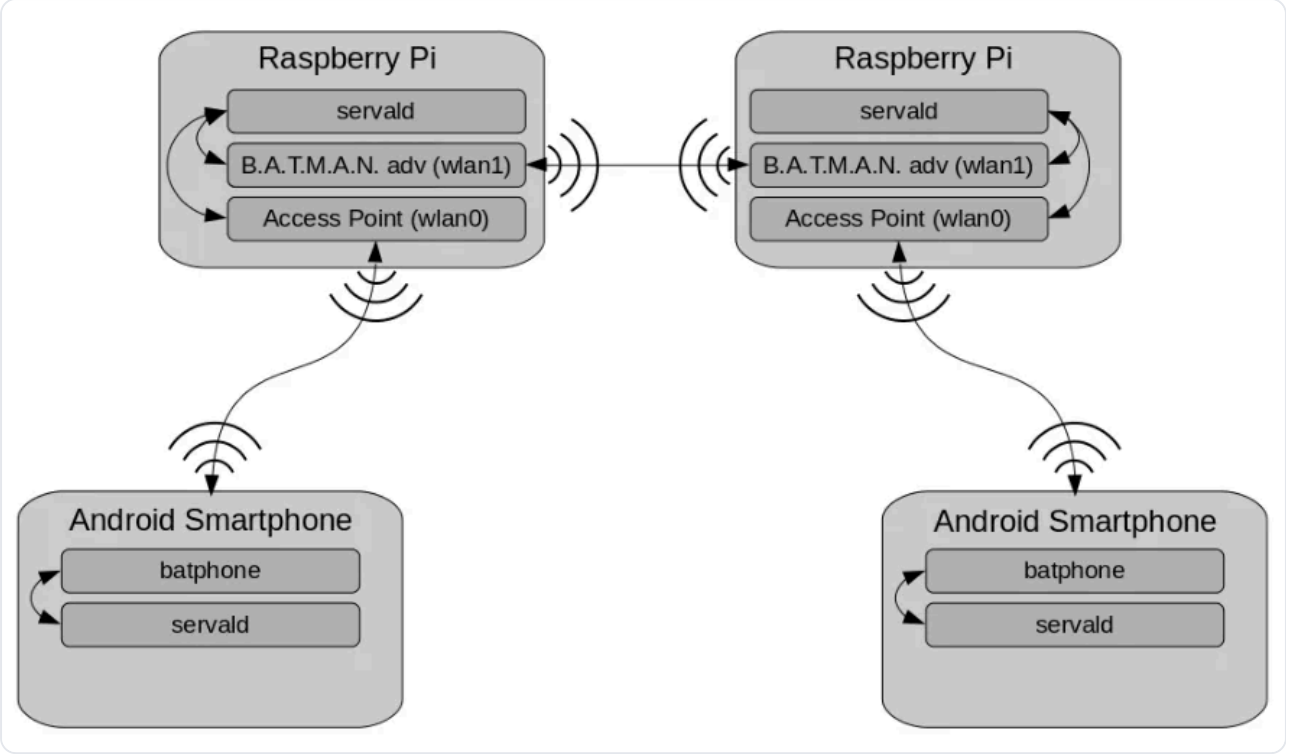


Abbildung 1: Schematischer Aufbau

5. Weiterentwicklung der Batphone App

Der „Serval"-Client für Android, „batphone", bietet mehrere Möglichkeiten zum Datenaustausch. So kann der zugrundeliegende „servald" etwa das Bluetooth-Interface nutzen, welches allerdings aufgrund der begrenzten Reichweite nur in näherem Umfeld genutzt werden kann. Des Weiteren kann das W-LAN-Interface im Client-Modus zu einem bestehenden W-LAN-Netz verbunden werden, als auch selbst einen Access Point erstellen. Als letzte Option kann „servald" auch im Ad-Hoc-Mesh Modus betrieben werden, wofür allerdings Root-Rechte am Smartphone benötigt werden.

Auf Android Smartphones ab Version 8.0 Oreo können Apps nicht mehr die Access Point-Funktion des Betriebssystems direkt verwenden, sondern nur noch einen lokalen Access Point erstellen, dessen Service Set Identifier (SSID) aus dem Präfix „AndroidShare_" gefolgt von einer zufälligen vierstelligen Nummer besteht und dessen Pre-Shared Key (PSK) eine zufällige alphanumerische Zeichenfolge ist [6]. Diese zufälligen Teile werden zudem bei jeder Aktivierung der Funktion neu generiert.

Der Client „batphone" wurde für die Verwendung der neuen Schnittstelle adaptiert, jedoch ist eine Erweiterung des Mesh-Netzes durch Android Smartphones nun nur noch durch die Bekanntgabe von SSID und PSK möglich.

6. Conclusio

Es wurde demonstriert, dass die Errichtung eines Kommunikationsnetzes mit kostengünstiger Hardware und der Verwendung von frei verfügbarer Software wie „Serval" und „B.A.T.M.A.N.adv" möglich ist. Im Katastrophenfall kann mit der Verteilung von Basisstationen in einem Gebiet vorübergehend eine Datenverbindung wiederhergestellt werden. Dieses Netz kann leicht durch die Verwendung mobiler Clients

erweitert werden. Des Weiteren ist es möglich, diese Basisstationen noch zu spezialisieren, indem etwa Richtfunksysteme für die Überbrückung großer Entfernungen zwischen den einzelnen Stationen verwendet und die Stationen selbst mit Akkus betrieben und so versorgungsunabhängig werden.

Referenzen

[1] Raspberry Pi Foundation, „https://www.raspberrypi.org/,“ [Online]. Available:

https://www.raspberrypi.org/. [Zugriff am 12 11 2018].

[2] Serval Project, „Serval,“ [Online]. Available: http://servalproject.org. [Zugriff am 16 10 2018].

[3] Serval Project, „Servald,“ [Online]. Available: https://github.com/servalproject/serval-dna.

[Zugriff am 16 10 2018].

[4] Raspberry Pi Foundation, „Raspbian,“ [Online]. Available:

https://www.raspberrypi.org/downloads/raspbian/. [Zugriff am 16 10 2018].

[5] Open Mesh, „BATMANadv,“ [Online]. Available: https://www.open-mesh.org/projects/open mesh. [Zugriff am 16 10 2018].

[6] Android Developers, [Online]. Available:

https://developer.android.com/reference/android/net/wifi/WifiManager. [Zugriff am 17 10 2018].