Richtlinie zum „Software-Sprint“

Eingangsst.

PT-for-LEAP – Verbesserte anonymisierte Kommunikation

Schlussbericht

Zuwendungsempfänger:

Johannes & Nuyts GbR

Das diesem Bericht zugrundeliegende Vorhaben wurde mit Mitteln des Bundesministeriums für Bildung und Forschung unter dem Förderkennzeichen 01IS18S30 gefördert. Die Verantwortung für den Inhalt dieser Veröffentlichung liegt beim Autor.

Kurze Darstellung der Aufgabenstellung und Motivation

Was war Deine Motivation? Welches Problem wolltest Du mit Deinem Projekt lösen? Wie war die geplante Vorgehensweise zur Problemlösung (auch Angabe der wichtigsten Meilensteine)?

LEAP Encryption Access Project– https://leap.se/ & https://bitmask.net/ - ist gegründet worden, um online frei und sicher kommunizieren zu können. Seit seiner Gründung im Jahr 2012 haben wir dafür eine Open-Source Client-Server-Architektur für die Bereitstellung von VPN-Diensten und verschlüsselter E-Mail aufgebaut.

In den vergangenen Jahren wurde die Umgehung von Internetzensur durch den Einsatz neuer Detektionsverfahren seitens von Unternehmen oder staatlichen Institutionen eingeschränkt.

Um weiterhin die Software für einen leicht aufzusetzenden und global erreichbaren VPN Dienst bereitstellen zu können, haben wir im Rahmen der Förderung Pluggable Transports (PT) client- und serverseitig implementiert. Pluggable Transports ist eine zusätzliche Transportschicht, in der Datenpakete gekapselt werden, um deren Eigenschaften zu obfuskieren. Damit ist es möglich, viele Deep Package Instpection (DPI) Verfahren auszuhebeln, mithilfe derer oftmals Internetzensur durchgesetzt wird. Gleichzeitig schützt Pluggable Transports sowohl Nutzer*innen als auch Dienstleiter*innen von LEAP davor, von Zensoren als solche identifiziert zu werden.

Für die Umsetzung von Pluggable Transports sollten zunächst die verfügbaren Protokolle, die PT unterstützen, vor dem Hintergrund stetig weiterentwickelter DPI-Verfahren auf ihre Aktualität und

Effektivität evaluiert werden. Daraufhin sollten bestehende Implementierungen der in Frage kommenden Obfuskierungsprotokolle recherchiert, auf ihre Einsetzbarkeit in die bestehende Softwarearchitektur der Clients (Bitmask) und des LEAP-Servers bewertet und nötige Anpassungen an PT-Bibliotheken vorgenommen werden. Durch die Entwicklung von Puppetmodulen sollte die serverseitige Einrichtung der Pluggable Transports Software automatisiert werden. Die vorhandene Server-API zur Abfrage der Client-Konfiguration sollte um die nötigen PT-Parameter erweitert werden. Die transparent ablaufende Selbstkonfiguration von Bitmask sollte unter Zuhilfenahme der Server-API um den Einrichtungsschritt von Pluggable Transports erweitert werden. Pluggable Transports sollte für Bitmask auf den Plattformen Android, Linux und Mac unterstützt werden. Ein Client-Userinterface zur Aktivierung von Pluggable Transports sollte ebenso wie eine Server-Schnittstelle für Administrator*innen zur manuellen Konfiguration der unterstützten Protokolle, IP und Ports implementiert werden. Mithilfe von User*innentests sollten die entwickelten grafischen Oberflächen bewertet werden.

Beitrag des Projektes zu den Zielen der Förderinitative „Software-Sprint“

Wer ist die Zielgruppe für Deine Lösung? Wie profitiert sie von den Ergebnissen Deines Projekts? Welche Bezüge gibt es zu den Themenfeldern und Zielen des Software Sprints?

Zielgruppen von LEAP sind einerseits Journalist*innen und zivilgesellschaftlich und politisch aktive Gruppen. Die Software wird in der Praxis aber auch von anderen Menschen, die einen freien Zugang zu Informationen im Internet erhalten wollen, genutzt. Die Playstore-Statistiken unserer Android-Clients zeigen, dass besonders viele Menschen aus Ländern mit restriktivem Internetzugang den LEAP-Dienst nutzen. Wir erwarten, dass die die Nutzer*innenbasis in China, Kasachstan und anderen Staaten, die erfolgreich VPN durch Deep Package Inspection blockieren, mit der Veröffentlichung der im Rahmen der Förderung erfolgten Implementierung wachsen wird.

Eine weitere wichtige Zielgruppe sind Community-Service-Provider und NGOs, die ihren Mitgliedern eine verschlüsselte Verbindung zu ihren Servern anbieten wollen. Diese erweitern mithilfe von PT ihre Reichweite. Außerdem ist diese Software natürlich auch nutzlich für Menschen, die z.B. ihr Internetverhalten für andere Menschen aus ihrer Umgebung abschirmen möchten.

Ausführliche Darstellung der Ergebnisse

Welche konkreten Ergebnisse hast Du erzielt? Konnten alle Meilensteine erreicht werden? Welche zusätzlichen Erkenntnisse hast Du aus der Projektarbeit gewonnen, auch im Hinblick auf die Begleitung durch die Open Knowledge Foundation?

Wir haben zunächst die PT-Protokolle Obfs2, Meek, Scramblesuit, Domain Fronting und Obfs4 bewertet und basierend auf unserer Onlinerecherche Obfs4 als primär einzusetzendes Obfuskierungsprotokoll identifiziert.

Obfs4Proxy und shapeshifter-dispatcher wurden im Anschluss als mögliche Implementierungen prototypisch client- und serverseitig integriert. Beide Softwares basieren auf der Programmiersprache Go und sind dadurch theoretisch auf allen von LEAP unterstützten Plattformen einsetzbar. Um plattformspezifischen Hardwarearchitekturen zu unterstützen, mussten Skripte zum Cross-Compilieren des Sourcecodes entwickelt werden.

Die Skripte ermöglichten in einem späteren Schritt auch die Erweiterung unseres Continuous Integration (CI) Prozesses zum automatisierten Bauen und Testen auf unseren Build-Servern.

Um das Puppet-Modul zur automatisierten Einrichtung von shapeshifter-dispatcher auf den LEAP-Serven entwickeln zu können, mussten wir zunächst ein System-Update der vorhandenen Puppet-Module von Debian 8 („Jessie") auf Debian 9 („Stretch") vornehmen. Der Umfang dieser Arbeit wurde deutlichunterschätzt, da nicht nur das Betriebssystem aktualisiert wurde, sondern auch sehr viele Komponenten sehr eng mit einander verknüpft waren und geupdatet werden mussten. So musste zum Beispiel die Puppetmodule der Nutzer*innendatenbank, Monitoring und E-mail zum Teil ganz ersetzt werden.

Die Provider-API wurde so angepasst, dass auf Anfrage des Clients neben dem Port und der IP des unobfuskierten VPN-Gateways auch die Verbindungsdaten der unterstützten Pluggable-Transport-Technologie mitgeliefert werden. Hier wurde ein Augenmerk auf Erweiterbarkeit gelegt, sodass zukünftig auch andere Obfuskierungsprotokolle ohne erneute Änderungen der Schnittstelle unterstützt werden können.

Wir haben in der Entwicklungsphase der Clienttechnologie zunächst unseren Fokus auf Android gelegt. Dort wurde der Selbstkonfigurationsprozess um den Einrichtungsschritt des shapeshifter-dispatchers ergänzt. Dieser schließt dieInstallation des Binaries, das Starten selbigen in einem eigenen Prozess und die Identifikation des nach dem Start zur Verfügung gestellten Proxy-Ports ein. Um den Batterieverbrauch durch Bitmask zu vermindern, wurde der Lebenszyklus des shapeshifter-dispatcher an jenen des OpenVPN-Prozesses gebunden. Bei der Android-Implementierung wurde ebenfalls Wert auf eine mögliche Ergänzung weiterer Protokolle gelegt.

Wir haben ein einem späteren Schritt Shapeshifter-Dispatcher in unseren ebenfalls auf Go basierenden Desktop-Client Bitmask-lite prototypisch integriert, sodass die Obfuskierungstechnologie auf Linux, Mac und Windows rudimentär unterstützt wird.

Zielgruppe, Nutzen und mögliche Weiterentwicklungen

Welcher Nutzen ergibt sich für die Zielgruppe aus den Ergebnissen Deines Projekts? Welche weitergehenden Effekte ergeben sich aus der Open-Source-Stellung der Ergebnisse? Gibt es Ideen für die Weiterentwicklung Deiner Lösung und Pläne zu deren Umsetzung?

Hat die Arbeit in dem Projekt Dich in Deiner persönlichen, fachlichen Weiterentwicklung unterstützt?

Hat die Arbeit in dem Projekt Dich in Deiner persönlichen, fachlichen Weiterentwicklung unterstützt?

Der bisherige Stand der Entwicklung wurde noch nicht in den produktiv eingesetzten Source Code gemergt, da eine bessere Absicherung des Codes und einige Bugfixes ausstehen. Wir planen in den kommenden Wochen diese Schritte zu vollziehen. Wir gehen davon aus, dass Endnutzer*innen aus Weißrussland, China, Türkei, Russland, Turkmenistan, Irak, Iran, Oman und den Vereinigten Arabischen Emiraten jene LEAP-Provider erreichen werden können, die ihr System aktualisieren und Pluggable Transports aktivieren.

Der Einsatz von Pluggable Transports zur Umgehung von Internetzensur steht in einem permanenten Spannungsverhältnis zur Weiterentwicklung von Gegentechnologien zur Erkennung selbiger sowie zu politischen Entwicklungen weltweit. Es wird Nutzer*innen temporär die Möglichkeit geben, sich sicher über das Internet auszutauschen, bis Internet Service Provider oder staatliche Institutionen das von LEAP unterstützte Obfuskierungsprotokoll beispielsweise mithilfe von Machine-Learning-Algorithmen erkennen und blockieren können. Folgerichtig wird in der Zukunft das von LEAP entwickelte VPN neue Protokolle implementieren, um im Technologiewettbewerb der Internetzensur einen Schritt voraus zu bleiben. Gleichzeitig benötigt es neben technischen Lösungen weiterhin ernorme zivilgesellschaftliche Anstrengungen, um die politischen Verhältnisse zugunsten freier Gesellschaften zu verändern. Wir wollen mit unserer Technologie gerade jene Gruppen unterstützen, die dafür einstehen.

Durch die Open-Source-Stellung unserer Software ermöglichen wir einen community-basierten Ansatz der Weiterentwicklung der Software. Sie ermöglicht außerdem eine öffentliche Kontrolle des Source Codes, was bei sicherheitskritischer Software ein besonders wichtigen Faktor darstellt. Insbesondere die Clients können als Referenzimplementierungen für andere Entwickler*innen dienen, da unseres Wissens nach noch keine anderen Open-Source-Projekte mit dem gleichen Technologie-Stack existieren. Während unserer Arbeit standen wir in Verbindung mit anderen Menschen aus der Network Traffic Obfuscation Gemeinschaft und haben auch Pluggable Transports Dokumentationen auf den neuen Stand gebracht, die andere Entwickler*innen nützen können.

Ein weiteres von Providern gewünschtes Feature ist ein zuverlässiges Monitoring von Gateways und anonymisierten Nutzungsdaten. Wir möchten nach der Förderung verschiedene Möglichkeiten hierzu untersuchen und implementeieren.

Insbesondere in Bezug auf die aktuellen Entwicklungen von Obfuskierungstechnologien haben wir uns als Team weiterentwickelt.

Die Begleitung des Prototypefunds durch die angebotenen Workshops hat uns auch geholfen, besser einzuschätzen in wieweit sich das Betreiben eines VPN-Providers zur Unterstützung der Softwareentwicklung ökonomisch rentiert. Der Extra-Input von außen ist wichtig, um sich nicht in technischen Details zu verlieren und um sich vor Augen zu führen, dass ein regelmäßiger Kontakt mit den Endnutzer*innen und den Service-Providern wichtig sind.

Kurze Darstellung der Arbeiten, die zu keiner Lösung geführt haben

Infolge der prototypischen Integration mussten wir die fehlende Interoperabilität von Obfs4Proxy und shapeshifter-dispatcher feststellen, obwohl sie über das gleiche Protokoll kommunizieren. Wir haben uns schließlich dafür entschieden, shapeshifter-dispatcher auf allen Plattformen gleichermaßen einzusetzen.

Wir hatten darüber hinaus ursprünglich geplant, eine grafische Konfigurationsoberfläche für die Clients zur Aktivierung und Deaktivierung von Pluggable Transports zu implementieren. Wir haben uns jedoch dagegen entschieden, da wir die Komplexität der Software für den User möglichst gering halten wollen. Stattdessen implementierten wir eine Autodetektion der unterstützten Obuskierungsprotokolle, einen automatischen Verbindungsaufbau unter Nutzung selbiger und einen optimierten Fallbackmechanismus für fehlgeschlagene Verbindungsversuche.

Während der Anfangsphase stellte sich heraus, dass Domain Fronting eine Technologie ist, die nicht mehr in der Praxis umsetzbar ist, weil alle große Platformen dieses unterbinden.

Ein erheblicher Anteil der vorhandenen Dokumentation verweist noch auf veraltete Protokolle, die wir testweise eingesetzt haben, aber relativ schnell ersetzen mussten.

Kurze Angabe von Präsentationsmöglichkeiten für mögliche Nutzer

Wir haben zwar offen an unserem Quellcode gearbeitet, aber wir befürchten, dass es im Moment noch einige Probleme auftreten können, wenn Menschen diesen einfach kopieren. Wir haben vor, weitere Berichte über die Integration von Pluggable Transports in unsere Software auf den Zeitpunkt des ersten Releases, der jene Änderungen beinhaltet, zu verschieben. Der Quellcode existiert unter https://0xacab.org/leap/ und werden unseren Bericht auf https://leap.se veröffentlichen.

Kurze Erläuterung zur Einhaltung der Arbeits- und Kostenplanung

Die Aktualisierung der Puppet-Module als Voraussetzung zur serverseitigen Implementierung von Pluggable Transports hat sich als deutlich komplexer und langwieriger erwiesen, als wir in der Planungsphase vermutet hatten. Außerdem hatte die Behebung von Serverausfällen in unserem Build-System zu Verzögerungen geführt. Infolgedessen konnten wir einige geplante Ziele nicht erreichen. So werden im Moment noch zu wenig Optionen für Administrator*innen geboten, um Pluggable Transports zu konfigurieren. Eine Fallbackdiscovery der angebotenen Transports existiert ebenfalls nicht.

Bei der Implementierung für das Serverplatform fehlt im Moment auch noch die Integration mit dem Command Line Interface.

Kurze Darstellung von etwaigen Ergebnissen bei anderen Stellen

Gab es Entwicklungen anderer Personen oder Institutionen, die Einfluss auf Deine Arbeiten und die Zielsetzung hatten? Wenn ja, worin bestand dieser und wie bist Du damit umgegangen?

Wie schon erwähnt, war die Deaktivierung von Domain Fronting bei Google, Amazon und anderen Anbietern ein kleiner Rückschlag. Zum Glück gibt es weitere Obfuskierungstechnologien, sodass wir Obfs4 einsetzen konnten. Der Ausfall unserer Buildsysteme und Infrastruktur hat uns bestimmt eine Woche gekostet, aber weil es um einen nicht an uns kommunizierte Umzug ging, konnten wir nach Umbau und Neukonfiguration nachher wieder weiter. Nachdem wir unsere erste Verbesserungen für die Webseite und Anleitungen auf http://pluggabletransports.info vorgeschlagen hatten, wurden wir von Internews kontaktiert. Dieser Austausch war bereichernd und vor allen sehr motivierend: zu hören das die Arbeit sinnvoll ist und geschätzt wird. Wir haben in die letzte 6 Monaten auch intensiver zusammen gearbeitet mit Übersetzer*innen wodurch die Software auch durch mehr Menschen benutzt werden kann. Unsere Android Applikation ist inzwischen in 14 Sprachen übersetzt, von Deutsch und Englisch bis zum Baskisch, Hebräisch und Farsi.