PhishNet

Studiengang: BSc in Informatik | Vertiefung: Data Engineering
Betreuer: Prof. Dr. Erik Graf

Existierende Phishing-Prävention bestehen auf einem Blocklist-Ansatz, was dazu führt, dass neue Phishing URLs nur reaktiv erkannt werden können. Mit der voll automatisierten ML Lösung von PhishNet können potenzielle Phishing Webseiten in (near) real-time klassifiziert werden. Das PhishNet Browser Plugin warnt den Benutzer proaktiv, bevor er seine sensitiven Daten im Internet preisgibt.

Ausgangslage

Phishing Angriffe stellen für Privatpersonen wie auch für Unternehmen nach wie vor eine ernstzunehmende Gefahr dar. In vielen Unternehmen ist die Phishing-Bekämpfung ein reaktiver und träger Prozess, welcher auf einem Blocklisting-Ansatz basiert. Dies führt dazu, dass die Webseiten oft erst blockiert werden, wenn der Schaden entstanden ist.

Im Rahmen der PA1 wurde gezeigt, dass ein ML basierter Ansatz eine vielversprechende Lösung zur proaktiven Phishing Bekämpfung darstellen kann. Seitdem wurden über 200'000 URLs gesammelt, die mitunter als Datengrundlage dieser Arbeit dienen.

Ziel der Arbeit

Als Resultat dieser Thesis soll eine proaktive, lernfähige Lösung entstehen, die Nutzer:innen vor möglichen Phishing Angriffen schützt. Konkret werden Webseiten während dem Besuch in Klassen „legitim" oder „phish" eingeteilt. Bei einem Verdacht auf eine Phishing Seite soll dies entsprechend signalisiert werden.

Zugleich soll die alltägliche und berufliche Verwendung des Webs nicht gehindert werden. Dies erfordert die Implementierung einer Pipeline für kontinuierliches Lernen, (near) real-time Klassifizierung, sowie einen benutzerfreundlichen Umgang mit „False-Positives".

Vorgehen

Die Grundlage jeder ML Lösung sind Daten. Dafür wurde ein auf Elasticsearch basierendes System entwickelt, welches täglich legitime und phishing URLs sammelt und persistiert. Parallel dazu entstand mit Hilfe von TensorFlow Extended eine automatisierte ML Pipeline. Basierend auf den erhaltenen Daten konnte ein Random-Forest Model trainiert, evaluiert und optimiert werden.

Die Verbindung zwischen dem ML Model und dem Benutzer wird durch eine Python Django API und einem JavaScript Browser Plugin ermöglicht.

Ergebnis

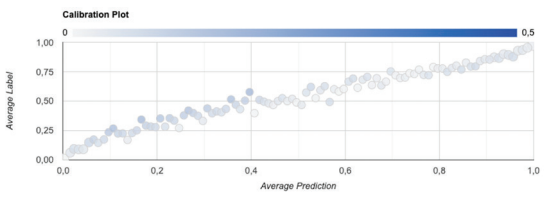
Mittels einer voll automatisierten ML Pipeline werden täglich legitime und phishing Webseiten gesammelt und bereitgestellt. Aus den gesammelten URLs werden aussagekräftige Features extrahiert, welche dazu verwendet werden, ein Random-Forest Modell zu trainieren. Nach einer erfolgreichen Prüfung wird das Modell automatisch in die Lösung integriert und steht der Web API zur Verfügung, um URLs zu validieren. Mit einer hohen Kalibrierung und Präzision können dem Benutzer verlässliche Angaben gemacht werden.

Das PhishNet Browser Plugin kommuniziert in (near) real-time mit der Web API und warnt den Benutzer proaktiv, bevor er seine sensitiven Daten im Internet preisgibt.

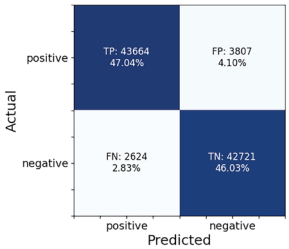
Adrian Berger

Lars Peyer

Yannis Valentin Schmutz



Calibration Plot des PhishNet Models



Confusion Matrix des PhishNet Models