„Detection of Attacks and Anomalies in HTTP Traffic Using Instance-Based Learning and KNN Classification“

Michael Kirchner

Diplomarbeit an der FH Hagenberg,
Studiengang Sichere Informationssysteme

OWASP Stammtisch München,
16.11.2010

Inhalte

- Problemstellung
- Entwickeltes Framework
- Implementierung
- Test & Ergebnisse
- Fazit

Disclaimer

Disclaimer:

In derzeit erhältlichen WAFs stecken Mannjahre an Entwicklungsarbeit. In dem von mir entwickelten Framework nicht. Es stellt also einen alternativen Ansatz dar, der meiner Meinung nach zwar Vorteile hat, aber noch kein fertiges Produkt ist.

Nein, es ersetzt auch keine sichere Entwicklung ;-)

Probemstellung

Sicherheitsmechanismen im Webumfeld sind häufig

- nicht an die zu schützenden Applikationen angepasst,
- als Blacklist realisiert ($\neq$ positives Sicherheitsmodell),
- teilweise umgehbar, da signaturbasiert.

Beispiele zur Umgehung (Evaluierung von WAFs)

```
<script>alert('xss')</script>
<scRipt/?a bc>alert /*any*/ (/xss/ )</script

check.php?ip=4.2.2.1; cat /etc/passwd
check.php?ip=4.2.2.1; a=etc; cat /$a/passwd;
```

Interpreter (Browser, DB's, etc) akzeptieren gleichen Input in vielen verschiedenen Formen.

Probemstellung

→ Angestrebte Lösung:

- Automatisch erlernen, welche Verhaltensmuster für eine Webapplikation normal sind (HTTP-Verkehr beobachten)
- Abweichungen vom gelernten Modell feststellen

Verbesserungen zu derzeitigen Lösungen zur Anomalieerkennung in HTTP-Daten:

- Kein Eingriff in den Quellcode
- Nicht nur HTTP-Request-Parameter, sondern auch andere Nachrichtenteile und HTTP-Responses analysieren
- Unterstützung von inkrementellem Lernen

Entwickeltes Framework

Der entwickelte Ansatz verwendet

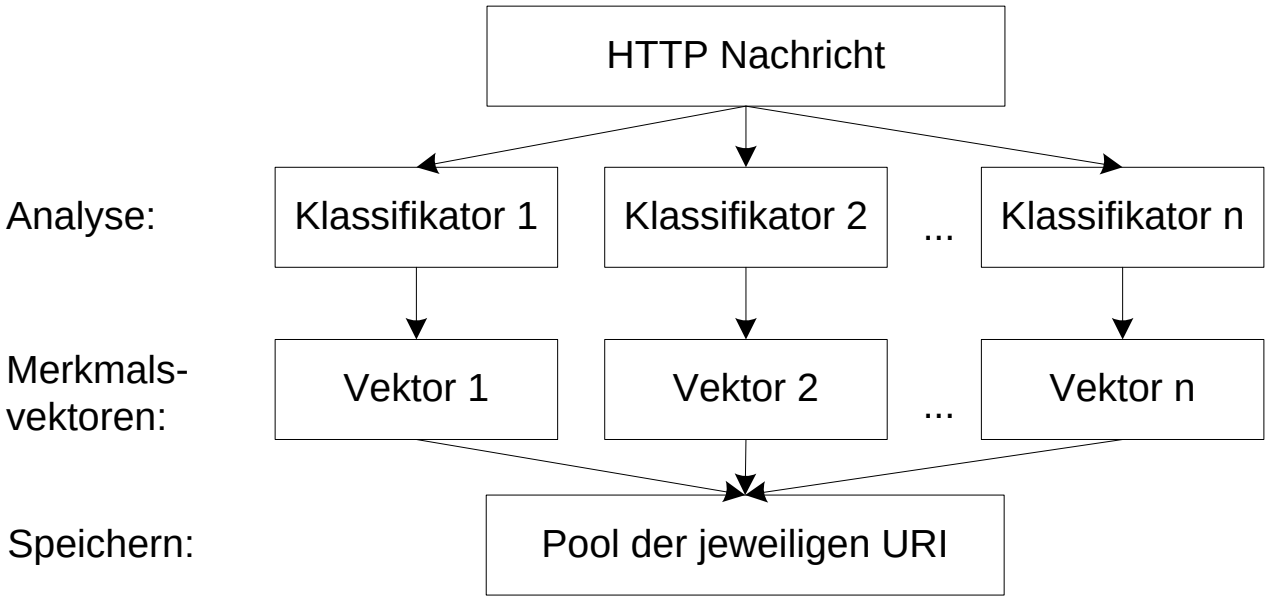
- Instanzbasiertes Lernen
- *k*-nächste-Nachbarn-Klassifikation

Trainingsphase:

- Für jede URI eines Webservers wird ein Wissenspool generiert
- Pools beschreiben was als normal gilt
- Poolinhalte werden durch Klassifikatoren generiert (Transformation von HTTP-Nachrichten in einzelne Merkmalsvektoren)

Entwickeltes Framework

Trainingsphase:



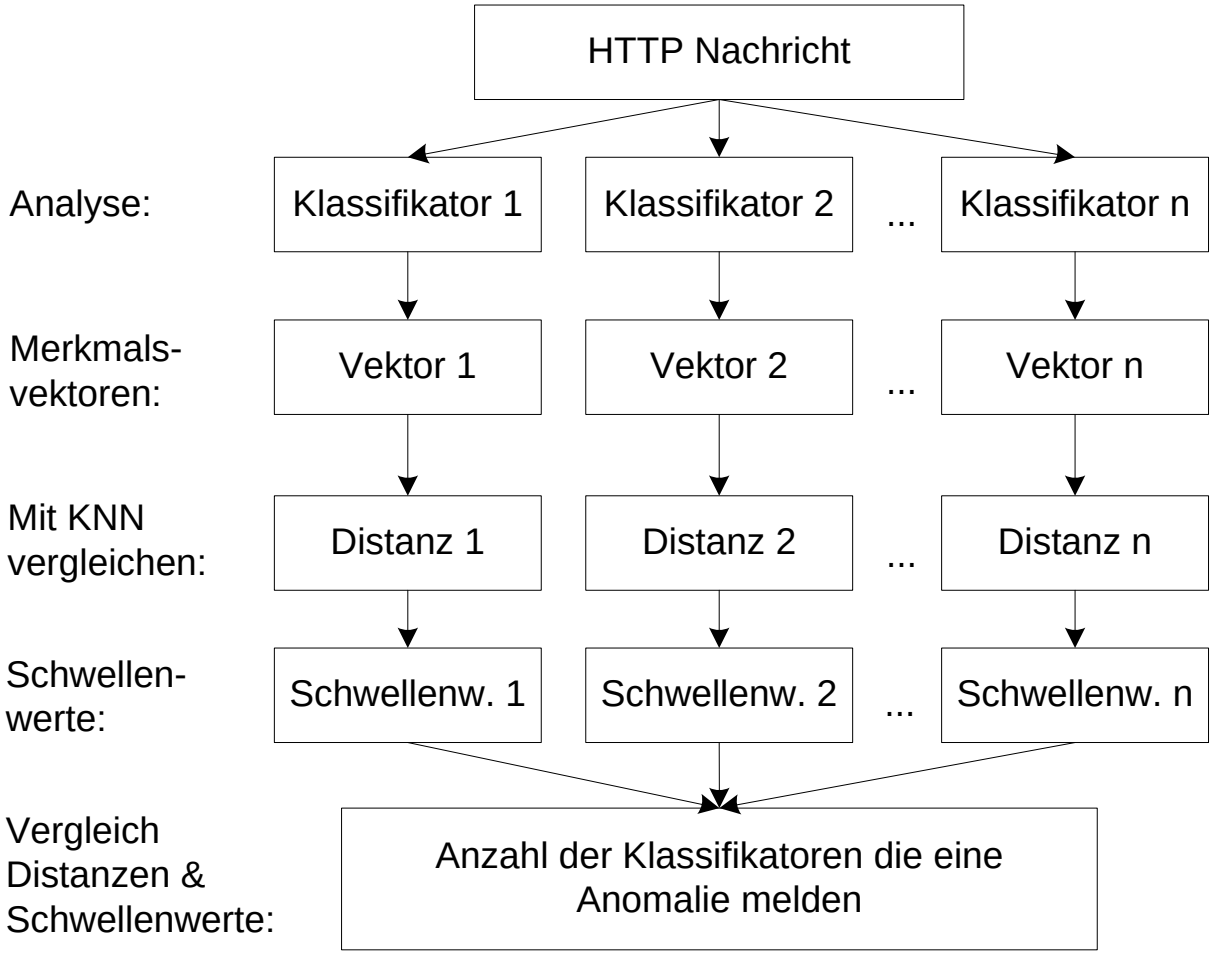
Framework enthält derzeit 32 Klassifikatoren. Beispiele:

- RequestHeaderCookieLength
- RequestMethodCharacterDistrib
- RequestParameterCharacterDistrib
- RequestParameterDataType
- RequestParameterOrigin
- ResponseContentsLength
- ResponseStatusCharacterDistrib
- ...

Entwickeltes Framework

Betriebsphase:

- Erkennen von Anomalien durch Abstandsmessung:

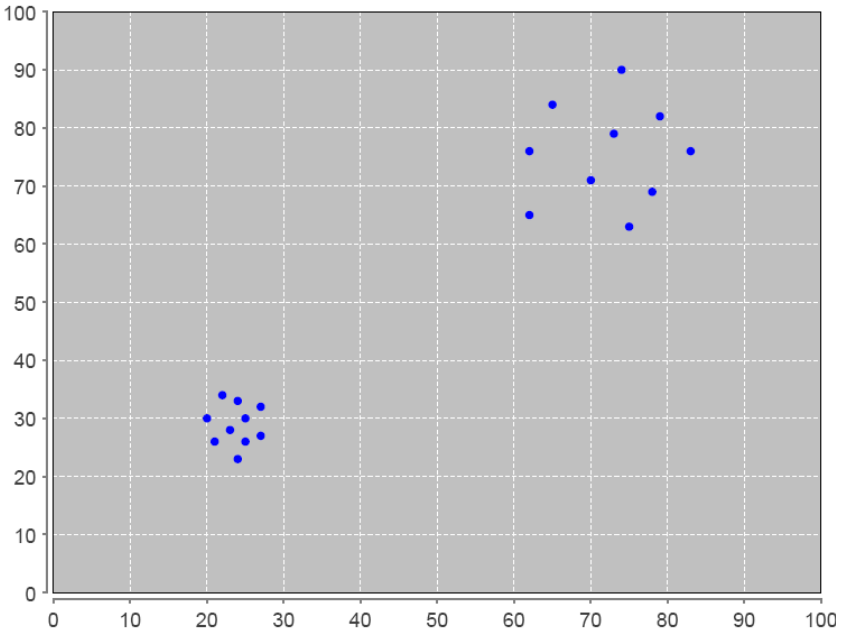


Entwickeltes Framework

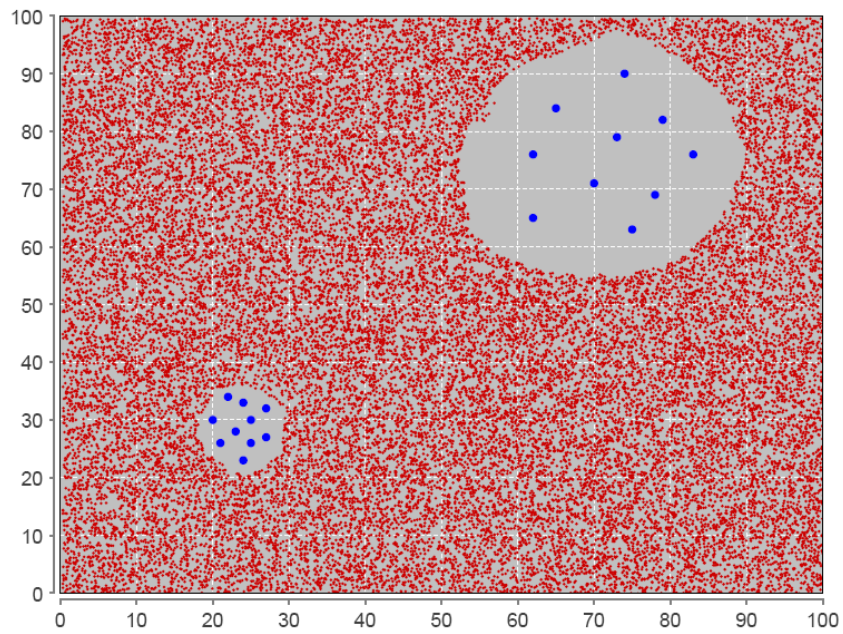
- Eine HTTP-Nachricht ist anormal wenn:

$$\frac{\sum_{i=1}^{k}(d_i \cdot (1-d_i))}{\sum_{i=1}^{k}(1-d_i)} > t_{KNN}$$

d_i = Einzelabstände zu den k-nächsten-Nachbarn
t_{KNN} = Schwellenwert abhängig von der lokalen Verteilung der Daten



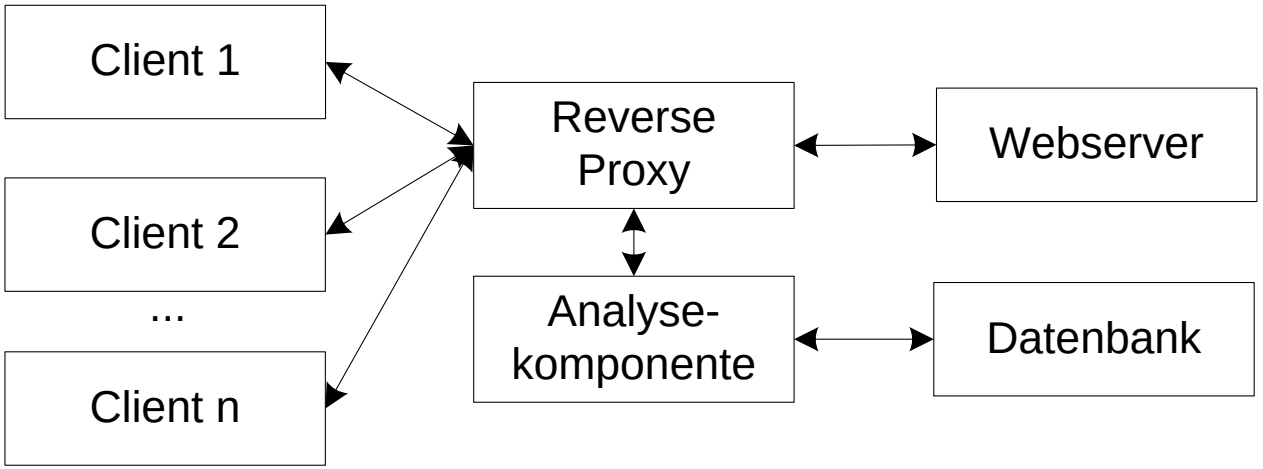
(a) Gelernte Poolinhalte



(b) Resultierende Klassengrenzen

Implementierung

- Java Reverse Proxy Servlet für Tomcat 6, ca. 4200 Codezeilen (davon aber viele Kommentare...)
- Neue Klassifikatoren als Plugins hinzufügbar
- Weboberfläche zur Analyse von Pools und Anomalien
- Leicht erweiterbar (zusätzliche Plugins)



Implementierung

Demo

Test & Ergebnisse

Datenset:

- Test des Frameworks mit einem eigens generierten Datenset (adaptierte *BadStore.net* Webapplikation)
- Enthält normalen angriffsfreien Datenverkehr und „erkennbare“ Angriffe aus den OWASP Top Ten:
  - Information disclosure
  - SQL injection
  - Command execution
  - Insecure object references
  - Invalidated redirects
  - ...

Test & Ergebnisse

Evaluierungsvorgang:

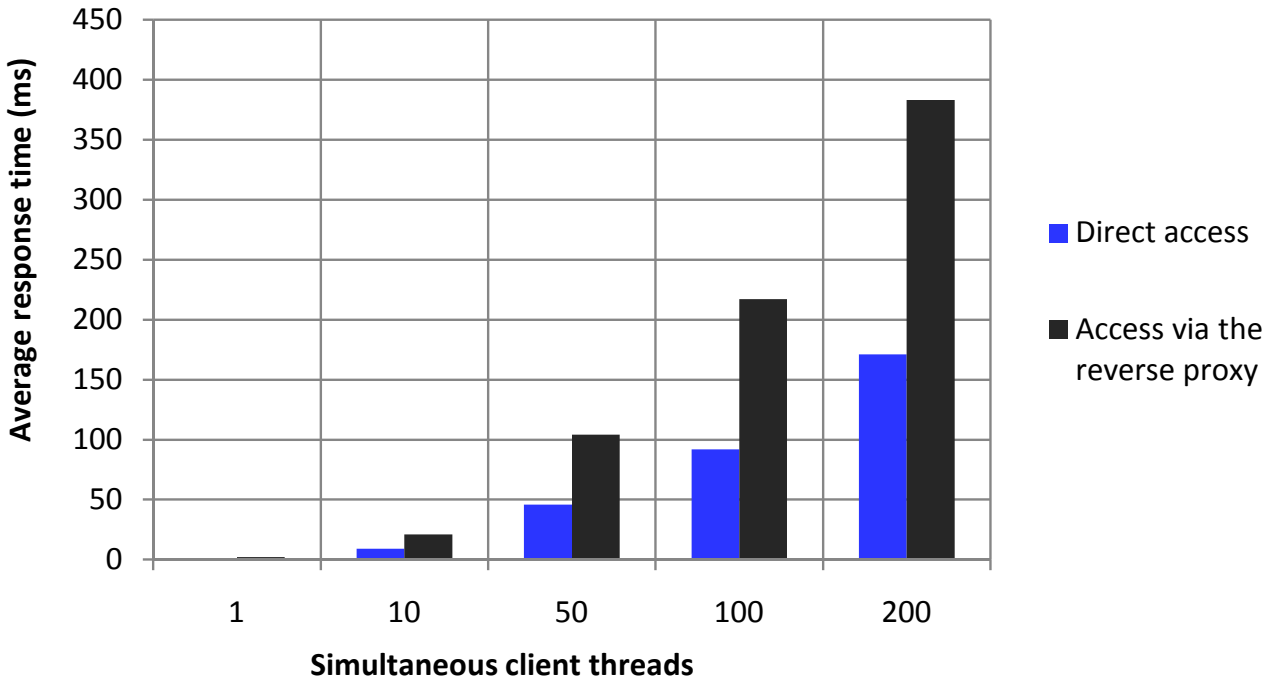
- Datenset wurde zur Bestimmung von Erkennungsrate und Falscherkennungsrate bei verschiedenen Framework-Einstellungen eingespielt
  - Poolgröße (s)
  - Anzahl der Nachbarn (k)
  - ...

- Beste Ergebnisse erzielt mit $s = 100$ und $k = 10$
  - Erkennungsrate: 10 von 11 Angriffen ($\sim 90.9\%$)
  - Falscherkennungsrate: 5 von 1018 Anfragen ($\sim 0.4\%$)

Test & Ergebnisse

Performanceevaluierung des Reverse Proxy:

- Instanzbasiertes Lernen bringt Vorteile (z.B. inkrementelles Lernen), ist aber performancelastig
- Weiterleitung und Analyseaufgaben bringen einen Performanceverlust von ca. Faktor 2 (...quite bad)



Fazit

Zusammenfassung:

- Framework um normalen HTTP-Verkehr zu lernen und Anomalien zu erkennen
- Generiert ein positives Sicherheitsmodell und verwendet keine Signaturen
- Nicht auf einzelne Teile von HTTP-Nachrichten eingeschränkt
- Unterstützt inkrementelles Lernen
- Analyseprozess kann individuell angepasst werden (Klassifikatoren, Schwellenwerte, etc.)

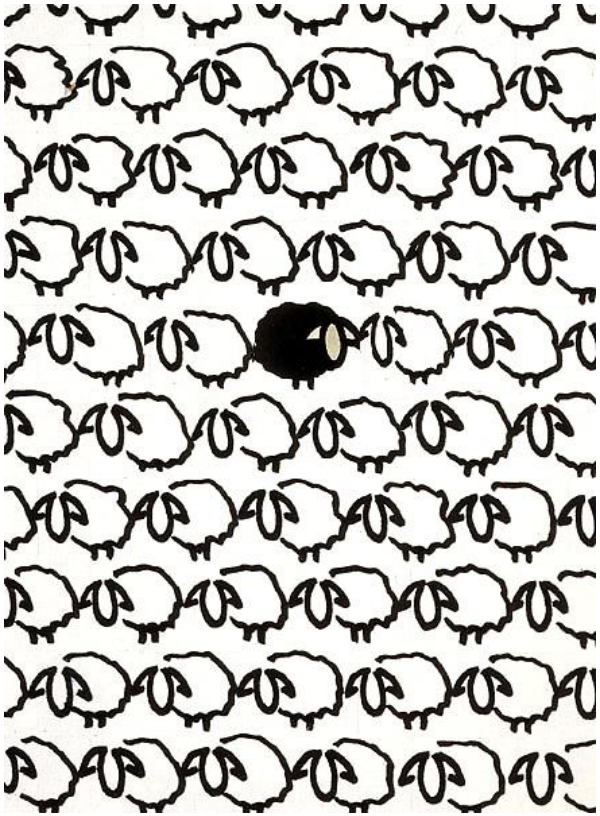
- Noch viele Möglichkeiten zur Erweiterung und Verbesserung
- Aufwand auf alle Fälle höher als bei WAFs im „Blacklist-Modus“

Fazit

Weitere Infos:

- Paper (IEEE IWSCN 2010 / Karlstad): „A Framework for Detecting Anomalies in HTTP Traffic Using Instance-Based Learning and K-Nearest Neighbor Classification“
- Diplomarbeit + Sourcecode (BSD License)

Fazit



Und jetzt:

Diskussion, Fragen,
Kopfschütteln, Flaming, ... ;-)

Kontakt:
michael.kirchner@fh-hagenberg.at