Seminar aus Informatik

2012-06-15

Intrusion Detection Systems (IDS)

- Ziel: Erkennung von Angriffen und Ausbrüchen

Host Based IDS

- Läuft auf dem Host
- Ist tief im System verankert
- Hat Zugriff auf:
  - Prozessinformationen
  - Netzwerkverkehr
  - Kontext
- Beispiel: Windows Security Suite (Virenscanner, Firewall)

Host Based IDS - Vorteile

- Prozessinformationen
  - Kann z.B. den Speicher überwachen
- Kontextinformationen
  - Browser will uni-salzburg.at kontaktieren
  - Windli32.dll will uni-salzburg.at kontaktieren?

Host Based IDS - Nachteile

- Manche Angriffe schwer erkennbar (z.B. DDOS)
- Zentrale Überwachung schwierig
- Aufwändig zu verwalten
- Rootkits nicht erkennbar
- IDS muss mit hohen Rechten laufen $\Rightarrow$ erhöhtes Risiko

Network Based IDS - Vorteile

- Schwer manipulierbar
- Zentrale Verwaltung
- Einfache Wartung
- Verteilte Angriffe erkennbar

Network Based IDS - Nachteile

- Prozess und Kontext-Informationen fehlen

Pattern Matching IDS

- Klassischer IDS Ansatz
- Sucht nach bekannten Mustern
- Muster werden von Angriffen extrahiert (z.B. Honeypots)
- Musterdatenbank muss gepflegt werden

Pattern Matching IDS - Vorteile

- Einfach zu implementieren
- Schnell
- Sofort einsatzbereit
- Geringe Fehlerrate

Pattern Matching IDS - Nachteile

- Keine unbekannten Attacken
- Keine Verhaltensänderung
- Aufwändige Pflege der Muster

Anomaly Detection IDS

- Suche nach auffälligem Verhalten
- Menschen handeln ähnlich z.B. Polizei bei Autofahndung
- Unterscheidet nicht zwischen auffälligem Verhalten und Angriffen
- Meist zwei Phasen (training/testing)

Anomaly Detection IDS - Vorteile

- Zero Day Exploits (z.B. Stuxnet)
- Auffälliges Verhalten (z.B. Industriespionage)
- Kaum Pflege notwendig

Anomaly Detection IDS - Nachteile

- Viele false positives
- Nicht immer alle true positives
- Lernphase
- Auffälligkeit != Angriff

Beispiele

- High speed Web Attack Detection
- Stuxnet (attacks against process control)

Fallstudie: Schnelle Erkennung von Web-basierten Angriffen

- Ausgangspunkt: http traffic
- Typische Samples mittels Affinity Propagation (AP) extrahieren
- Erkennung mit k-nearest neighbors und support vector machines
- feature extraction / information gain
- Verfahren effizienter dank AP

Fallstudie: Schnelle Erkennung von web-basierten Angriffen

Herausforderungen:

- http traffic $\Rightarrow$ große Menge an hochdimensionalen Daten
- Echtzeitanforderung

Aufgaben:

- Daten sammeln
- Feature extraction
- Modellieren
- Detection

Affinity Propagation

Was: besonders typische Datensätze auswählen

- Message Passing Algorithmus (Exemplar extraction)
- Anfang: alle Punkte sind Kandidaten für Exemplar
- Iterativ: Botschaften austauschen ($X, Y \in \mathbb{R}$)
  - Punkt zu Exemplar Kandidat: Du wärst X gut geeignet, mich zu repräsentieren.
  - Exemplar zu Punkt: Es wäre eine Y gute Idee, mich zu wählen.
- Dämpfung
- Konvergenz

Information Gain

Was: besonders brauchbare Attribute (aller Datensätze) auswählen

- Shannon: Entropy, ...
- Motto: Meritokratie
- Wähle von allen Attributen jene aus, die am meisten zu einer Verbesserung der Klassifikation beitragen.

Schwächen:

- oft nicht genug Angriffsdaten verfügbar
- zwar weniger Attribute, aber immer noch sehr viele Datensätze

Ergebnis

- Affinitiy Propagation gut geeignet
- Weniger ist mehr
  - Bessere Erkennung trotz weniger Daten
- auf die Qualität der Daten kommt es an

Industrielle Angriffe

Ziele:

- MSR, Automatisierung stören, beeinflussen

Wo:

- Öl, Gas, Wasser,
- Energie, Verkehr,
- Militär, ...

Schaden:

- Gesundheit
- Sicherheit
- wirtschaftlich
- politisch

Angriffsvektoren:

- zero day exploits
- customized rootkits
- malware signed by trusted authorities
- unbekannt

Fokus bei der Abwehr

nicht auf das **wie:**

- Technologie des Angriffs
- konkrete Schwachstellen, . . .

sondern auf das **was:**

- physikalische Verhaltensauffälligkeiten
  - Systemverhalten ist bekannt
  - Abweichungen unabhängig vom System erkennen
- wie könnte der Prozess gestört werden?
- was könnte das Ziel des Angriffs sein?

Beispiele

- Ölverarbeitung im IRAN (2012)
- E- und Wasserwerke
- Atomanlagen (Stuxnet)

Fallstudie Stuxnet:

- Reprogrammierung der Steuerung
  - Operation außerhalb zulässiger Grenzen, Schaden
- Sabotage, militärisches Ziel

Details:

- zero day exploit (schwer erkennbar)
- Feldbus Kommunikation und Firmware manipuliert
- über Netzwerk, USB Sticks
- getarnt mit rootkits (für PCs und Steuerungen)
- signierte, manipulierte Treiber
- dh perfekte Illusion einer „heilen Welt“ für Opfer aber:

System verhält sich physikalisch anders!

Spezifische Security Probleme von SCADA Systemen

Herausforderungen:

- Patches, Updates sehr unbeliebt[1] (aber machbar!)
- Legacy
- Verfügbarkeit in Kombination mit
- Echtzeit

Erleichterungen im Vergleich mit Enterprise Networks:

- wenig Dynamik, kaum Änderung
  - in der Topologie
  - Anwender
  - im Netzwerkverkehr
  - weniger Protokolle

Konsequenzen: typische Schwächen der Anomaly Detection wirken sich hier nicht so stark aus.

[1] Motto:Never change a running system

Fallstudie: chemischer Prozess

- Reaktion mehrerer Komponenten zu einem Endprodukt
- Kessel mit Sensoren, Ventilen, Steuerung
- Dynamik: träge[2]
- wirtschaftlichster Betrieb beim höchstzulässigem Druck
  - darunter unwirtschaftlich, darüber Gefährdung, Zerstörung
- Modellbasierte Entwicklung
  - manche Sensoren sind kritisch, andere relativ egal
    - tamper resistance zumindest für kritische Sensoren (besser alle)
- Prediction aus Modell (first principle), oder empirische Daten
- Vergleich dieser Daten mit gemessenen Werten
  - Statistische Methoden anwenden
  - Entscheidung treffen

[2] ca 20 Stunden bis kritischer Zustand erreicht

Fortgeschrittene Attacken

- Angreifer kann natürlich selben Trick verwenden
- Drei Arten unerkannter Manipulationen
  - surge
    - maximaler Schaden so schnell wie möglich
  - bias
    - kleine Änderungen über langen Zeitraum
  - geometric
    - zuerst in verwundbaren Zustand bringen, dann maximalen Schaden

Ende

Danke für eure Aufmerksamkeit!