A3 – IPS

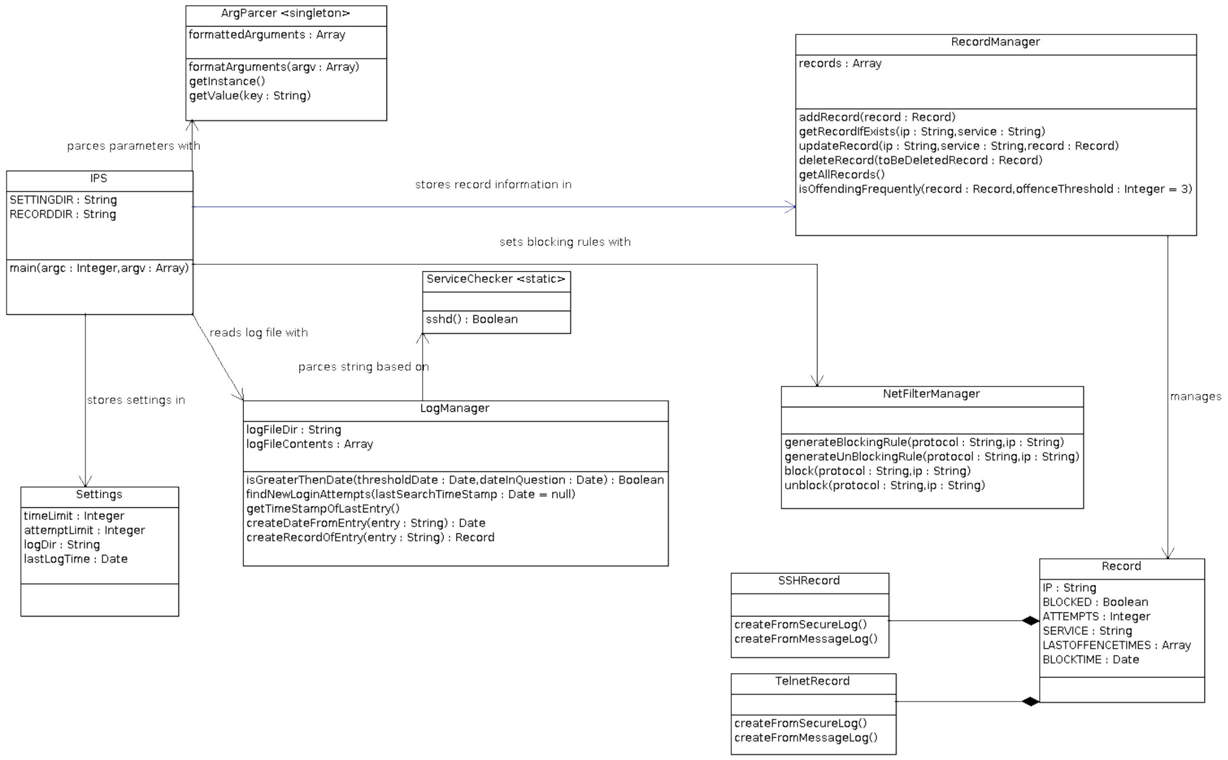
Ben Soer

A00843110

Table of Contents

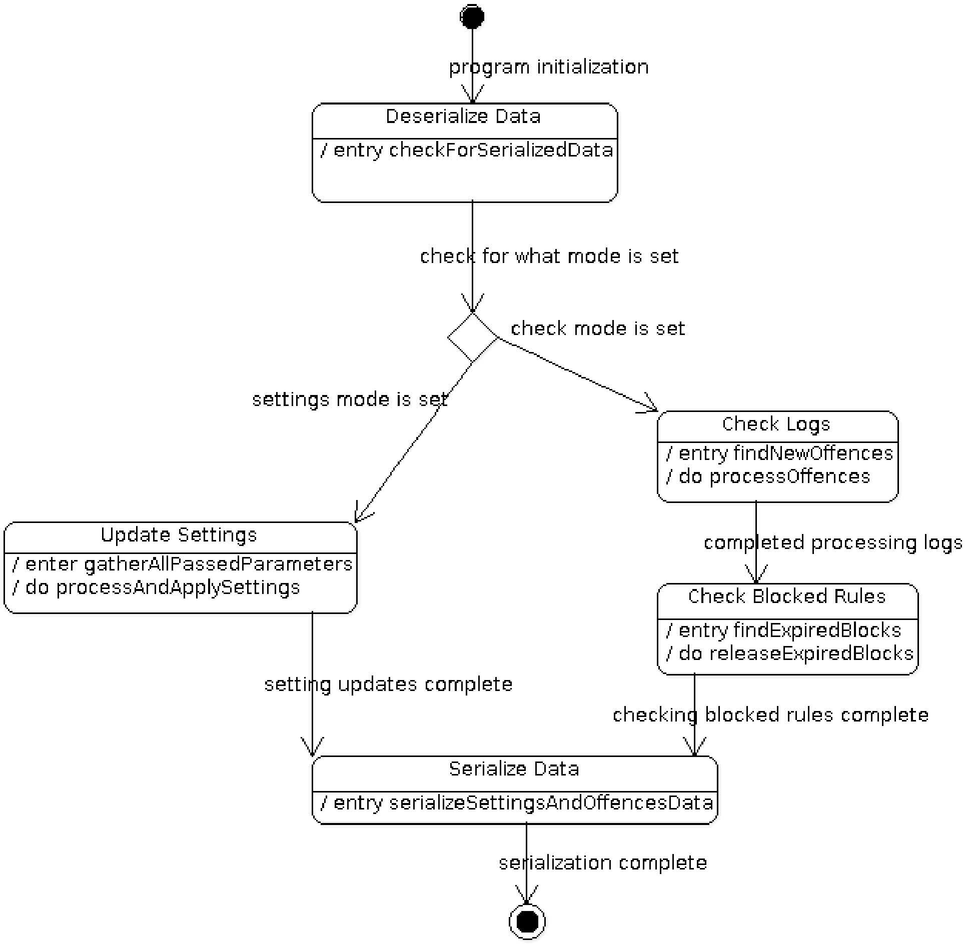
Class Diagram

Original image can be found under /docs/imgs/IPSClassDiagram.png


ArgParcer <singleton>
formattedArguments : Array
formatArguments(argv : Array)
getInstance()
getValue(key : String)
parces parameters with
IPS
SETTINGDIR : String
RECORDDIR : String
main(argc : Integer,argv : Array)
stores record information in
sets blocking rules with
RecordManager
records : Array
addRecord(record : Record)
getRecordIfExists(ip : String,service : String)
updateRecord(ip : String,service : String,record : Record)
deleteRecord(toBeDeletedRecord : Record)
getAllRecords()
isOffendingFrequently(record : Record,offenceThreshold : Integer = 3)
ServiceChecker <static>
sshd() : Boolean
reads log file with
parces string based on
stores settings in
LogManager
logFileDir : String
logFileContents : Array
isGreaterThenDate(thresholdDate : Date,dateInQuestion : Date) : Boolean
findNewLoginAttempts(lastSearchTimeStamp : Date = null)
getTimeStampOfLastEntry()
createDateFromEntry(entry : String) : Date
createRecordOfEntry(entry : String) : Record
Settings
timeLimit : Integer
attemptLimit : Integer
logDir : String
lastLogTime : Date
NetFilterManager
generateBlockingRule(protocol : String,ip : String)
generateUnBlockingRule(protocol : String,ip : String)
block(protocol : String,ip : String)
unblock(protocol : String,ip : String)
manages
SSHRecord
createFromSecureLog()
createFromMessageLog()
TelnetRecord
createFromSecureLog()
createFromMessageLog()
Record
IP : String
BLOCKED : Boolean
ATTEMPTS : Integer
SERVICE : String
LASTOFFENCETIMES : Array
BLOCKTIME : Date

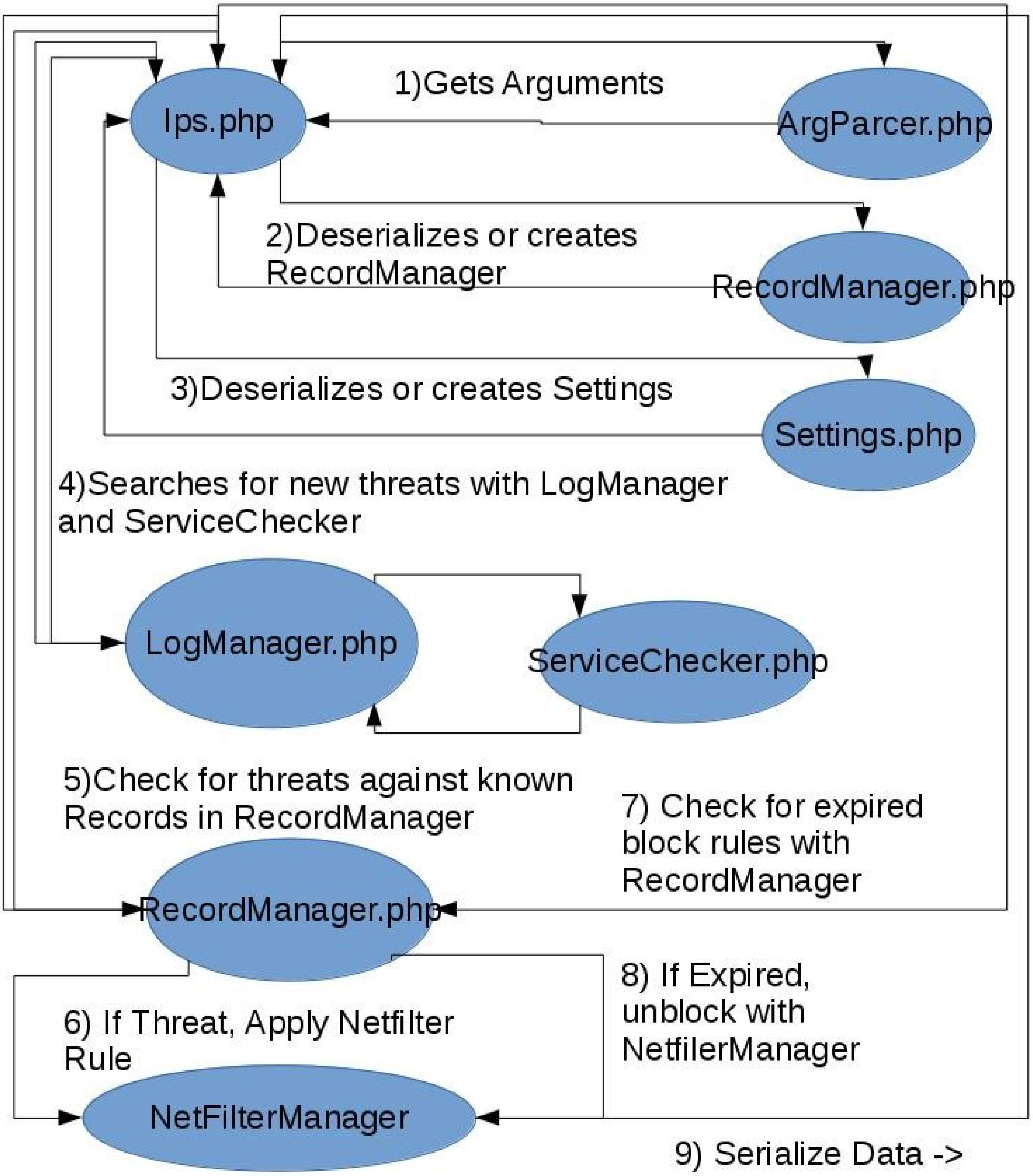
Finite State Machines

Original Image can be found under /docs/imgs/IPSStateChartDiagram.png


program initialization
Deserialize Data
/ entry checkForSerializedData
check for what mode is set
check mode is set
settings mode is set
Check Logs
/ entry findNewOffences
/ do processOffences
completed processing logs
Update Settings
/ enter gatherAllPassedParameters
/ do processAndApplySettings
Check Blocked Rules
/ entry findExpiredBlocks
/ do releaseExpiredBlocks
setting updates complete
checking blocked rules complete
Serialize Data
/ entry serializeSettingsAndOffencesData
serialization complete

Data Flow Diagram

Original Image can be found in /docs/imgs/DataFlowDiagram.jpg


Ips.php
1)Gets Arguments
ArgParcer.php
2)Deserializes or creates RecordManager
RecordManager.php
3)Deserializes or creates Settings
Settings.php
4)Searches for new threats with LogManager and ServiceChecker
LogManager.php
ServiceChecker.php
5)Check for threats against known Records in RecordManager
7) Check for expired block rules with RecordManager
RecordManager.php
8) If Expired, unblock with NetfilerManager
6) If Threat, Apply Netfilter Rule
NetFilterManager
9) Serialize Data ->

Pseudocode

Ips.php (Program Main Entry)

main(argc, argv)

```
1. Get Command Line Arguments
2. Deserialize Settings and RecordManager. If Not Existend Create New Ones
3. Get mode from Command Line Arguments
4. If check mode
    1. Get list of threats from logs
    2. Check logs against records if they are new or exist already
        1. If exist increment how many offences the record has made & check if should be blocked
            1. If should be blocked, block with netfiler
        2. If not exists add the record with this being its first offence
    3. Check Records for any blocked records
        1. If blocked, check if time has expired
        2. If time has expired unblock and delete the record
```

LogManager.php

findNewLoginAttempts(timeStamp = null)

```
1. If timeStamp is null
    1. Search logs from the beginning
    2. If valid LoginAttempt, add to list of new login attempts
2. If timeStamp is not null
    1. Search logs only from after timeStamp point
    2. If valid LoginAttempt, add to list of new login attempts
3. return loginAttempts
```

isGreaterThenDate(thresholdDate, dateInQuestion)

```
1. Format dateInQuestion into a date object
```

2. Compare if dateInQuestion is greater (newer) then the thresholdDate
3. return result

getTimeStampOfLastEntry()

1. Search log for last entry
2. Parce out and return timestamp as a date object

getAllEntries()

1. Return All log entries parsed from the log file

createRecordOfEntry(entry)

1. Determine type of log entry / what service the entry belongs to
2. Generate a Record object of that service type

NetfilterManager.php

generateBlockingRule(ip, protocol)

1. Create iptables command as a string to block the passed ip and protocol

generateUnBlockingRule(ip, protocol)

1. Create iptables command as a string to unblock the passed ip and protocol

block(ip,protocol)

1. Generate blocking iptables command for given ip and protocol
2. Execute the command

unblock(ip, protocol)

1. Generate unblocking iptables command for given ip and protocol
2. Execute the command

ServiceChecker.php

telnet(logEntry)

1. Parse logEntry to determine if it is a telnet failure log entry. Return true or false

sshd(logEntry)

1. Parse logEntry to determine if it is an sshd failure log entry. Return true or false

ArgParcer.php

formatArguments(Array argv)

1. Reformat the single argv array containing initialization parameters into an associative array where the key is the flag and the value is the value to the flag
2. Return the associative array

getInstance(Array formattedArguments)

1. Create singleton instance of ArgParcer. Do not set formattedArguments if an instance already exists. Return the instance

getValue(key)

1. Get the value belonging to the key in the formattedArguments associative array
2. If the value or key does not exist, return null

Record.php

createDateFromEntry(entry)

1. Parse out date stamp in log entry
2. convert to a date object and return

RecordManager.php

addRecord(Record record)

1. Add the passed in record to the array storing all records

getRecordIfExists(ip, service)

1. Search through all records for record matching ip and service
2. If does not exist, return null

updateRecord(ip, service, Record record)

1. Search through all records for record matching ip and service

2. If match is found, replace entry in the array with passed in record object

deleteRecord(Record toBeDeletedRecord)

1. Search through all records for record matching ip and service in the toBeDeletedRecord Record.
2. If record is found, delete it

getAllRecords()

1. Return all records stored in the array of the RecordManager

getTotalMinutesFromDif(DateInterval diff)

1. Calculate total minutes from diff object. Diff object stores count of years, months, days, hours seperatly and not cumulatively on their own, thus to get a total amount of minutes a quick calculation of minutes in the years, months, days, hours are also needed
2. Return totalMinutes

isOffendingFrequently(Record record, offenceThreshold = 3)

1. Get all of the interval times between each offense that has occurred for this record
2. Determine if any of the intervals are greater then the offenceThreshold. If there is one, then there is a large enough gap between offenses that this is not likely an attack.
3. Determine if the interval times are identical and thus meaning a bot could be calling but just extremely slowly.
4. Return true if step 1 or 2 are true, otherwise false

SSHRecord.php (extends Record)

createFromMessageLog(logEntry)

1. Parce out data from log entry assuming it has come from the /var/log/message file and set the record information for it

createFromSecureLog(logEntry)

1. Parce out data from log entry assuming it has come from the /var/log/secure file and set the record information for it

TelnetRecord.php (extends Record)

createFromMessageLog(logEntry)

1. Parce out data from log entry assuming it has come from the /var/log/message file and set the record information for it

createFromSecureLog(logEntry)

1. Parce out data from log entry assuming it has come from the /var/log/secure file and set the record information for it