Case study 1 : Rapidly deployed web application

A minimal AppSensor implementation for a small tightly-build web application that already has a strong input validation module.

Background	An entrepreneurial micro business has developed a web product to help financial service companies. All web application functionality requires the users to be authenticated. There are no public parts of the application except for the log in page. The company wants to get its web product to market as soon as possible but also needs to demonstrate robust defenses to its customers who will want to perform their own penetration testing. The business's own development team has created a parameter input validation framework that checks every single request's URL, parameter names and parameter values. The web application's entry points are known and are defined in an existing database table which is updated at each release. The team have decided to use AppSensor-like capabilities to warn them about forced browsing to invalid URLs, missing mandatory parameters, the submission of additional or duplicated parameters, and invalid parameter value data types. Note that additional input validation exists, but initially this will not be linked into the attack detection and response system. Just the URL, parameter names and the value data types.

[Diagram to be added]

Objectives	1. Immediately identify any non-normal use of the application 2. Slow down an attack using compromised user credentials
Detection points	The detection points only need to be added within the existing global input validation

module. The detection points selected are shown below. All exist within the application code.

Area	ID	Scope	Detection Description	AppSensor Refs
Request	R01	Every request	Invalid URL	AC3, IE2
	R02	Every request	Invalid parameter names	RE5, RE6
	R03	Every request	Invalid parameter value type	RE8, IE2

R01 also occurs for "404 not found" responses.

Response actions and thresholds

All events share the same response. Thresholds are all one (i.e. immediately, so there is no need to undertake counts over time periods). Only one SMS alert will be sent per request/response cycle (e.g. not per parameter).

ID	Threshold	Response Description	AppSensor Refs
R01, R02, R03	Any 1 event	Log out authenticated user and send SMS alert to development team	J, B

This will require the ability to:

- initiate a response for each detection point event
- terminate sessions and log out users, and send SMS alerts
- whitelist certain IP addresses to suppress the response actions (e.g. external vulnerability scanner, the company's own penetration testers)

Case study 2 : Business-to-consumer (B2C) e-commerce website (initial)

This example illustrates an initial standalone implementation where the development team have embedded the detection points into their own ecommerce website source code.

Background	The retailer's ecommerce channel accounts for 25% of their turnover. The website is comprised primarily of a product catalogue, shopping basket and check-out system, customers must register to check-out & pay, but can then also manage their accounts, submit reviews and take part in focus group discussions. The website is custom built and maintained in-house. The application has been through a number of changes to remove vulnerabilities. There are no generic input validation and exception handling modules.

[Diagram to be added]

Objectives	1. Identify generic attacks as soon as possible so they can be monitored. 2. Detect specific attacks against the custom logic in the product catalogue, shopping basket, checkout and payment functions 3. Identify attacks against database content
Detection points	In this initial implementation, the development team want to limit the number of detection points to less than ten, albeit some of these will occur in multiple instances. For example all requests will have some generic blacklist detection points, and all database query results sets will be validated against expected record count ranges (e.g. always none, always one, 2-10, 11-100 and 101+). The detection points selected are shown below. All exist within the application code, except for the last one which is implemented as triggers in the database that initiate a special web service call to the application. There are no site-wide (all user) thresholds.

Area	ID	Scope	Detection Description	AppSensor Refs
Request	R01	Every request	Invalid/incorrect HTTP verb	RE1, RE2, RE3, RE4
	R02	Every request	SQL injection attempt	CIE1
	R03	Every request	Cross-site scripting (XSS) attempt	IE1
Catalogue	C01	Product display	Product value mismatch	IE4
Basket	B01	Basket handling	Basket value mismatch	IE4
Payment	P01	Payment authorization	Card authorization failure	(Custom)
	P02	Order completion	Price mismatch between order & payment	IE4
Database	D01	Every SELECT query	Returned record set size incorrect	CIE2
	D02	-	Database table integrity fault	IE5

The events are logged into a database application log table.

Response actions and thresholds

The response actions were selected to block blatant abusers of the site and use alerting to operations staff for most other detected events. Threshold comparisons (per IP address and per user) will only include events in the previous 24 hours.

ID	Threshold	Response Description	AppSensor Refs
R01, R02, R03	Any 1 event	Block request	G
	Any 3 events by user	Log out authenticated user	J
	Any 6 events by user or and individual IP address	Block IP address (and customer account if known) for whole site (manual reset by website administrator)	L, K
C01, C02	Either 1 event	Alert operations staff	B
	Any 2 events	Block IP address for dynamic areas (1 day auto-reset)	I
P01	3 events	Alert operations staff, and redirect back to shopping basket summary	B, G
P02	1 event	Alert operations staff, put order on hold, and block future order check-out for the customer (manual reset)	B, D, I
D01	1 event	Alert operations staff, abort the current process, display an error page, and block the customer account (manual reset)	B, G, E, K
D02	1 event	Alert DBA and operations staff	B
(All)	1 event	Increase application logging granularity and indicate on monitoring dashboard	A, C

This will require the ability to:

- count detection points events for each threshold per IP address, and per user, and do this for every request
- change application logging level, raise alerts to operations staff, change the status of an order, terminate website user sessions, redirect responses, block individual requests, disable check-out functionality for individual users, block access to the whole website for an IP address and for individual IP addresses, reset blocks
- display a monitoring dashboard