APPLICATION SECURITY DONE RIGHT

LEVERAGE CONTEXTUAL AI ANALYSIS TO AUTOMATE APPLICATION SECURITY

Introduction

Application development has shifted and grown with the rapid adoption of cloud based environments. As the name suggests, web applications are programs that are accessible to users via a web browser. They are an invaluable part of an organization's web presence. Often, this presence incorporates application programming interfaces (APIs) that allow programmatic access to an organization's web applications or underlying services for the application.

Unsurprisingly, these web applications and web APIs are a common target of cyber criminals. The growing threat is made more dangerous by the adoption of services in the cloud required to build such applications. Cloud web application and API protection (WAAP) services are the evolution of cloud web application firewall services to meet these new challenges.

Traditional solutions are rule based—they analyze every request in isolation and compare them to manually set rules. The problem with this approach lies in the fast pace of application development. The very same reason developers are drawn to cloud based web application systems is what creates a growing attack surface. Applications are evolving faster than ever - creating and exposing more APIs.

The Cloud Killed the Rule Based WAF

Rule based, binary WAFs are insufficient in protecting the increasingly complex applications for the following reasons:

1. Cloud based web applications usually draw on multiple sources of code, including open source repositories. An application is built of small pieces of code that are more unique and single purpose, but there is a larger number of them. This can lead to more vulnerabilities. In the case of open source — widely known vulnerabilities that can be easy vectors for an attack.
2. The rate of change is infinitely faster than before. Modern cloud applications can be a click away from production. A commit to Github will automatically go through testing/staging and then to production with little human involvement. Any solution requiring manual tuning of Application Security will never match the pace of development.
3. Applications have become more open and connected. Modern cloud apps have a high number of resources interacting with 3rd party services outside of the "perimeter". Data can flow in and out of applications in many more ways than in the past. In effect, there are now hundreds of perimeters to secure.
4. As always, attacks consistently increase in their level of sophistication. In addition to the use of bots and APIs as attack vectors, there is a major issue of privilege escalation in the cloud. Attackers that infiltrate an application will look for API keys to other applications or resources. Due to poorly configured role based permissions, remote code execution can be much more harmful in the cloud.

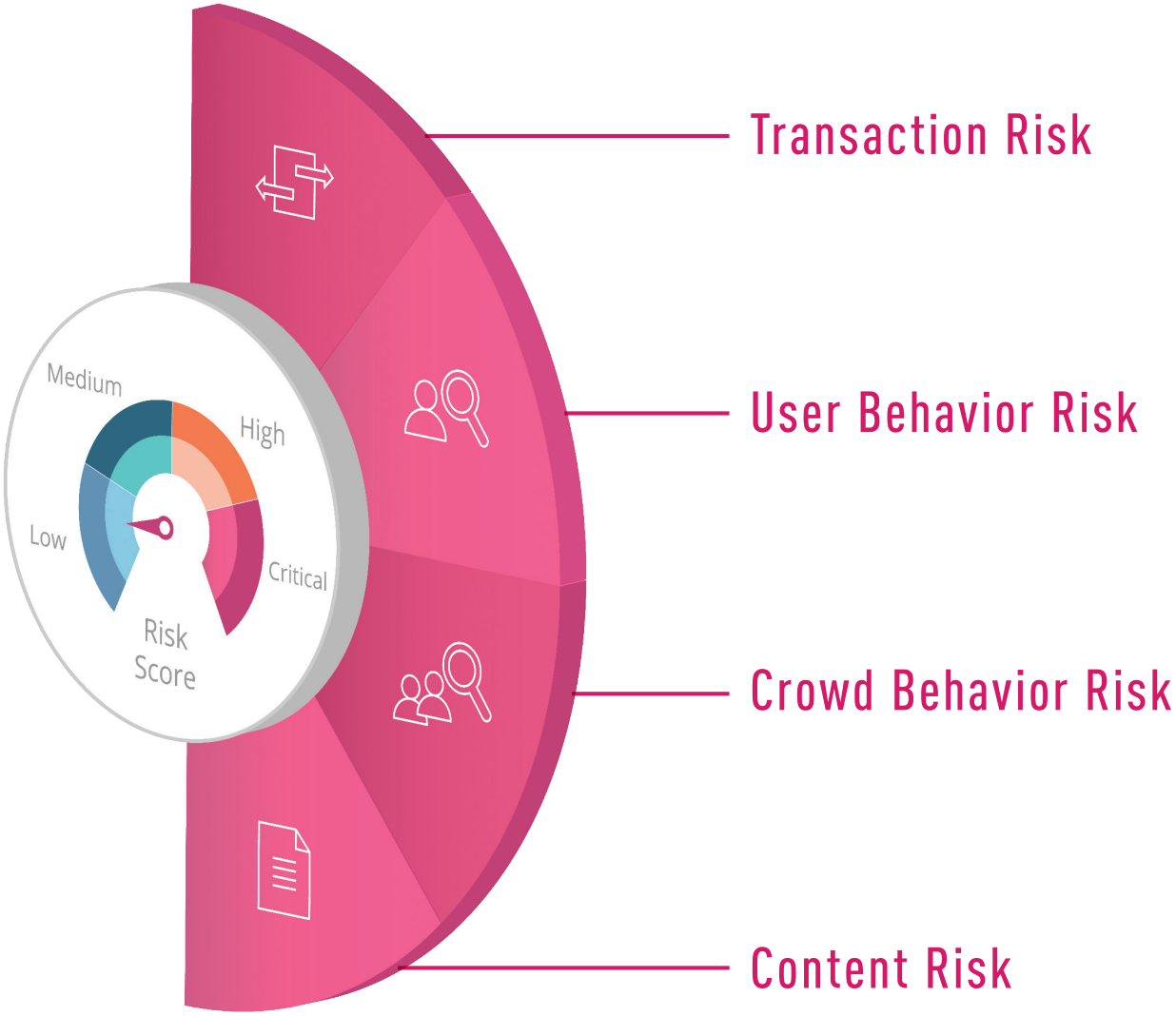
The better alternative to rule based, binary WAFs is a solution that requires no rule tuning with automated deployment. Utilizing the concept of **Application Self Protection** and powered by a patent pending contextual AI engine, CloudGuard AppSec's groundbreaking technology can:

- Automate deployment and stop application layer attacks, including OWASP Top 10, with no manual tuning or false positives.
- Stop unauthorized API access and abuse, without breaking applications and frustrating users.
- Identify and stop malicious bots before they can negatively impact the customers' experience
- Catch any HTTP based CVEs and known vulnerabilities

Contextual Score Based Decision Engine

Stopping False Positives with Machine Learning

By examining several parameters in detail and combining them to form an accurate risk score, CloudGuard AppSec eliminates false positives. A final risk score is determined with input from multiple engines:



Transaction Risk—This engine takes any transaction or request and breaks it into small elements called Attack Indicators. The engine has a pre-built dictionary of the smallest components of web related attacks and their relationship to each other. The Attack Indicators for a relevant transaction are fed into the engine. Using a patent pending machine learning algorithm, it determines whether it is malicious or not. If it is malicious, it is immediately blocked. If the result is ambiguous, as is often the case, other engines are utilized to make a decision.

User Behavior Risk—This engine analyzes all of the requests made from a specific user... how many were previously malicious or not? How many included a possible attack indicator? In essence, it analyzes malicious intent in prior user requests. Risk scores are built for users and users who demonstrate malicious intent will be blocked.

Crowd Behavior Risk—This engine maps a site based on how all users interact with it. The site profile will show how something that might be considered anomalous can be legitimate behavior that is typical of app use. For example: A specific URL is known to be flagged as suspicious for many different users, and so the probability of this request being a specific attack is lower.

Trusted Users—This engine works to accelerate the application learning with the creation of a white list of allowed inputs from defined trusted users.

Content Risk—This engine learns what content is typical for a specific field in a specific application. It provides a deeper analysis of the content itself—what are the patterns that are expected in each field? For example, an application can have an expected field be a command execution. Any traditional WAF engine would identify that field value as remote code execution and block it. In contrast, this engine will see that the specific field and value combination has been done by many other users, is legitimate, and accept the pattern. By looking at other sources which requested the same parameter and checking how many other users requested similar values, false positives will be eliminated. The understanding of the patterns of the field can be derived from any of the engines above.

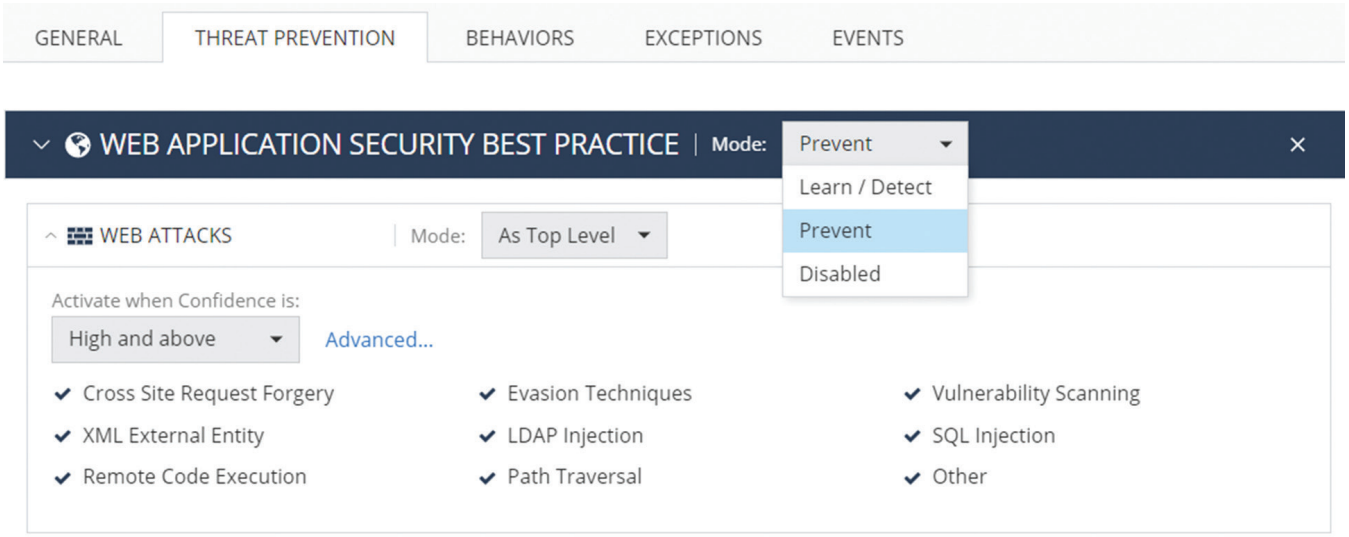
This type of contextual analysis is critical for web applications, as they are usually customer facing. False positives could result in real customers being prevented from accessing a site. Combining the risk analysis of multiple engines results in a more accurate decision, with understanding of the context. It allows security admins to operate comfortably in Prevent Mode, without the worry of blocking legitimate requests.

Continuous Learning

Automatic Adjustments

Web applications are put into learning mode initially. Learning data is saved in an agent and in the cloud. The mechanism is distributed - there are several CPUs working independently and storing information to memory, then synchronizing information between themselves every hour. Every distributed CPU runs all the engines mentioned above. They have a deep understanding of the application in terms of what is the source, HTTP method, types of HTTP requests, every key/value pair in the HTTP request and where they specifically reside. This information is all parsed and fed into the engines.

The system begins in Learn/Detect Mode. In only a few days, the system completes it's learning and the user can choose to switch it to Prevent Mode.



Supervised Learning is a mode in which the system assists the machine to learn much faster. In essence, this is a tool provided for an admin to expedite the learning time and be able to move to prevent mode much sooner. It works by looking at the user;s logs for the past week and identifying what events were identified as malicious with High or Critical severity. It finds the similarities between them and will present them to the user grouped together in a specific pattern. One such pattern is grouping by a specific source identifier, such as a source IP or email. The user is presented with a tuning suggestion. For example, "70% of requests from this source identifier were marked as critical. A true positive means this is an attacker — mark as malicious and it will be blocked. Mark as benign for a false positive — indicating the system has marked too many requests as malicious." This assists machine learning in understanding this is a benign request, a finding it would have reached in time. Users can speed up this process with Supervised Learning.

OpenAPI Schema Validation

Narrow Scope of API Attacks

In addition to the contextual engine and learning, CloudGuard AppSec allows for API Schema validation. Users can upload their schema, usually in the form of a JSON file, which describes the API functionality of the server. CloudGuard AppSec will make sure to enforce this schema by ensuring that no one can infiltrate applications via APIs and fields that are not explicitly allowed. It does this by looking for keys in the schema and checking that the value in the request corresponds to what is allowed. Here is an example of such a JSON file:

```
{
    "openapi": "3.0.1",
    "info": {
        "title": "Acme SmartMeter API",
        "description": "OpenAPI Demo",
        "termsOfService": "https://acme-power-e2e-int-waap.cloud.ngen.checkpoint.com/",
        "contact": {
            "email": "apiteam@cloud.ngen.checkpoint.com"
        },
        "version": "1.0.0"
    },
    "servers": [{
        "url": "https://acme-power-e2e-int-waap.cloud.ngen.checkpoint.com/"
    }],
    "paths": {
        "/Ingestion-App/setNewReading": {
            "post": {
                "tags": ["SmartMeter new reading report"],
                "summary": "Send SmartMeter Read",
                "requestBody": {
                    "content": {
                        "application/json": {
                            "schema": {
                                "type": "object",
                                "properties": {
                                    "uuid": {
                                        "type": "string",
                                        "format": "uuid",
                                        "description": "SmartMeter unique identifitier"
                                    },
                                    "countNumber": {
                                        "type": "number",
                                        "description": "SmartMeter counter"
                                    },
                                    "readingValue": {
                                        "type": "number",
                                        "description": "SmartMeter reading value",
                                        "maximum": 1000000,
                                        "minimum": 1,
                                        "exclusiveMaximum": false,
                                        "exclusiveMinimum": false
                                    },
                                    "readingTime": {
                                        "type": "string",
                                        "format": "date",
                                        "description": "SmartMeter reading time"
                                    }
                                }
                            },
```

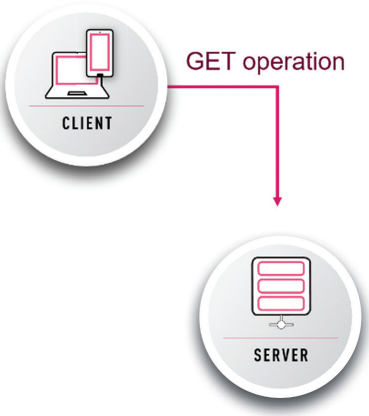
Notice that the type of each field is explicitly specified. For example, the “readingValue” field should be a number with a minimum value of 1 and a maximum value of 100000.

Bot Attacks — Distinguish Between Human and Non-Human

Validate Legitimate Traffic from Bots

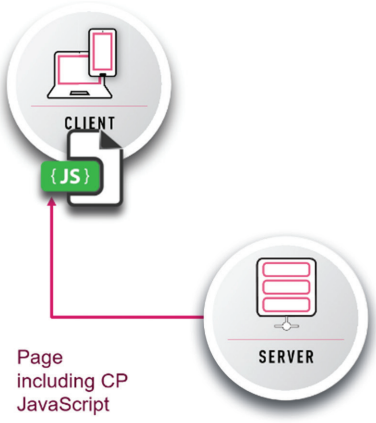
Malicious bots have been utilized by threat actors in everything from distributed denial of service attacks to buying out the newest Nike sneakers in a matter of seconds. Utilizing bots for automated attacks is a regular practice among threat actors. Some examples of how threat actors weaponize bots is:

- Reconnaissance
- Scraping
- Credential stuffing
- Automated account creation
- Token cracking

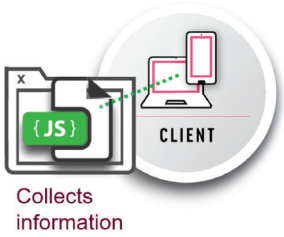


CloudGuard AppSec utilizes **Client Side Behavioral Analysis** to distinguish human behavior.

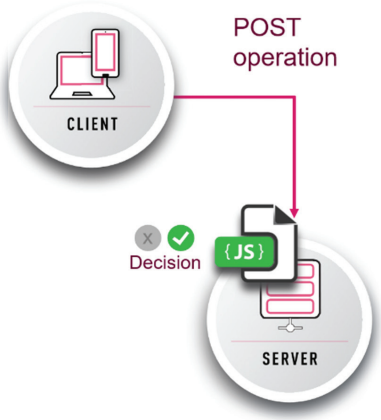
Once a client connects to a server, they perform a GET operation.



They receive from the server the page, including a Javascript developed by Check Point that is injected to the browser of the client.



This script collects behavioral information of the client.



When the client performs a POST operation to the server, it will include the decision of this script which defines if the request originated from a bot or a human.

IPS Protection

Prevent Protocol Based Attacks and Known Vulnerabilities

The IPS Protection offered by CloudGuard AppSec is complementary to the protections already discussed. Although **Application Self Protection** parses HTTP requests with deeper understanding and machine learning, it might not catch a vulnerability outside of the patterns it is looking for. IPS will catch any known vulnerabilities. It has the capability to prevent known malicious CVEs by looking for signatures in requests. When a signature matches a pattern known as part of a CVE, the request is blocked.

The signatures in CloudGuard AppSec are HTTP based. The backend that Check Point uses to identify these malicious signatures is ThreatCloud — a collaborative network and cloud-driven knowledge base that delivers real-time dynamic security intelligence.

CloudGuard AppSec is the logical evolution from Rule Based WAFs to web application protection that is automatic and self learning.

Conclusion

Applications are every organizations' main business driver, and with the proliferation of APIs, attack surfaces are expanding rapidly. DevOps update applications with increasing regularity and traditional application security is unable to keep up with the speed and scale of change. Modern applications demand modern security which will provide precise prevention without generating false positives. Modern cloud applications need coverage from a security solution which is fully automated — continuously learning the application, content and user behavior, in order to come to the right decision each time a web request comes in.

CloudGuard provides application security that is powered by contextual AI; examining all of the parameters associated with a web application or API request, in order to build a risk score. By automatically identifying malicious and non malicious requests, CloudGuard reduces operational overheads by providing a high level of threat prevention precision.

Start a free trial of CloudGuard AppSec, or request a demo today!

Worldwide Headquarters
5 Ha'Solelim Street, Tel Aviv 67897, Israel | Tel: 972-3-753-4555 | Fax: 972-3-624-1100 | Email: info@checkpoint.com

U.S. Headquarters
959 Skyway Road, Suite 300, San Carlos, CA 94070 | Tel: 800-429-4391; 650-628-2000 | Fax: 650-654-4233

www.checkpoint.com