

Web Application Firewalls und Prozesssicherheit

Schwarze Liste, weiße Weste

Prozesssicherheit befasst sich mit der Absicherung von Abläufen und dem Aufbau in der IT-Infrastruktur, den IT-Prozessen und der Datenverarbeitung in Unternehmen. Ein gutes IT-Management hat diese Bereiche genau definiert und im Laufe der Zeit so optimiert, dass Fehler oder Ausfälle vermieden werden. Eine der vielen Facetten des Themas betrifft – in stark wachsendem Maße – die Rolle von Web Application Firewalls (WAF).

In den meisten Fällen wird die Prozesssicherheit auf einer hochverfügbaren Lösung aufgebaut. Der Begriff Hochverfügbarkeit bezeichnet die Fähigkeit eines Systems, trotz Ausfall einer seiner Komponenten mit einer hohen Wahrscheinlichkeit (oft 99,99 % oder besser) einen ununterbrochenen Betrieb zu gewährleisten. Das heißt, es sind zwei oder mehr Systeme vorhanden, welche alle aktiv (Active-Active) oder eins aktiv und der Rest im Standby (Active-Passive) betrieben werden. In beiden Varianten muss im Fall eines Ausfalls die Verarbeitung des defekten Systems auf ein oder mehrere Systeme dieses Verbunds (Cluster) ausgelagert werden können.

Bis vor einigen Jahren war die Annahme, mit hochverfügbaren Lösungen eine Prozesssicherheit zu erzielen, weitestgehend richtig. Heutzutage ist diese Aussage jedoch überholt und falsch. Hochverfügbare Lösungen sind in fast allen Unternehmen zu finden und stellen die Prozesssicherheit nur auf infrastruktureller Ebene der IT dar.

Wie kann eine WAF bei der Prozesssicherheit unterstützen?

Um diese Frage zu klären, sollte zuerst definiert werden, wo in der IT-Infrastruktur eine WAF angesiedelt wird. In den meisten Fällen wird eine WAF als Reverse Proxy zwischen der Firewall und der Webapplikation integriert und ist wie die anderen Systeme auch als hochverfügbare Lösung ausgelegt. Eine WAF wird an diesem Punkt der Infrastruktur integriert, um sämtlichen Datenverkehr über die WAF zu leiten und so vor Angriffen auf die Applikation selbst und den Applikations- und Webserver zu schützen. Denn nur als Reverse Proxy und mit dem terminierenden Ansatz können alle Angriffe erkannt und geblockt werden.

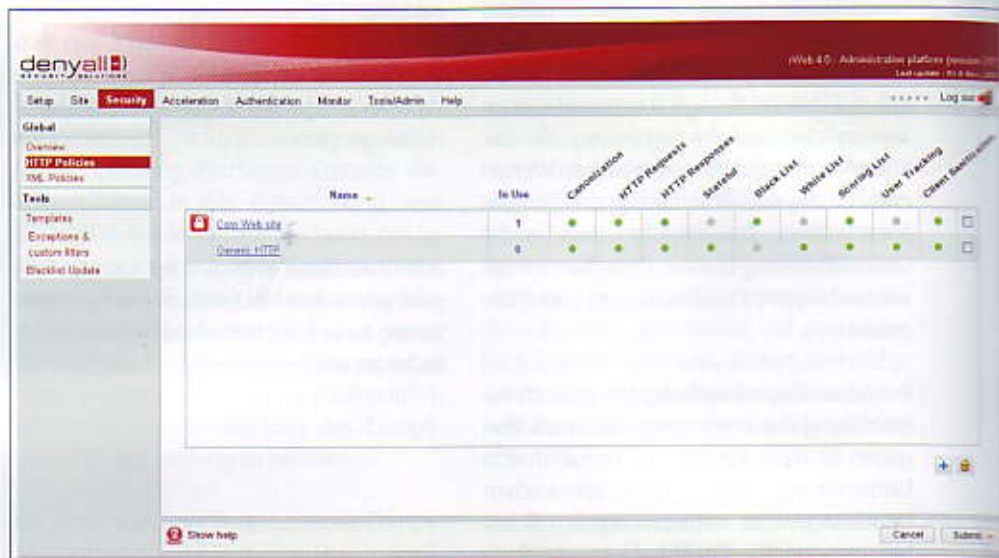
Heutzutage werden alle möglichen Anwendungen und Prozesse über Webapplikationen gesteuert, konfiguriert und dargestellt. Man kann sich solche browsergestützten

Anwendungen nicht nur im Servicebereich vorstellen, sondern auch in der Produktion, denn in beiden Bereichen kann ein Ausfall der Systeme enorme Kosten nach sich ziehen und ist der größte Kostenpunkt. An-

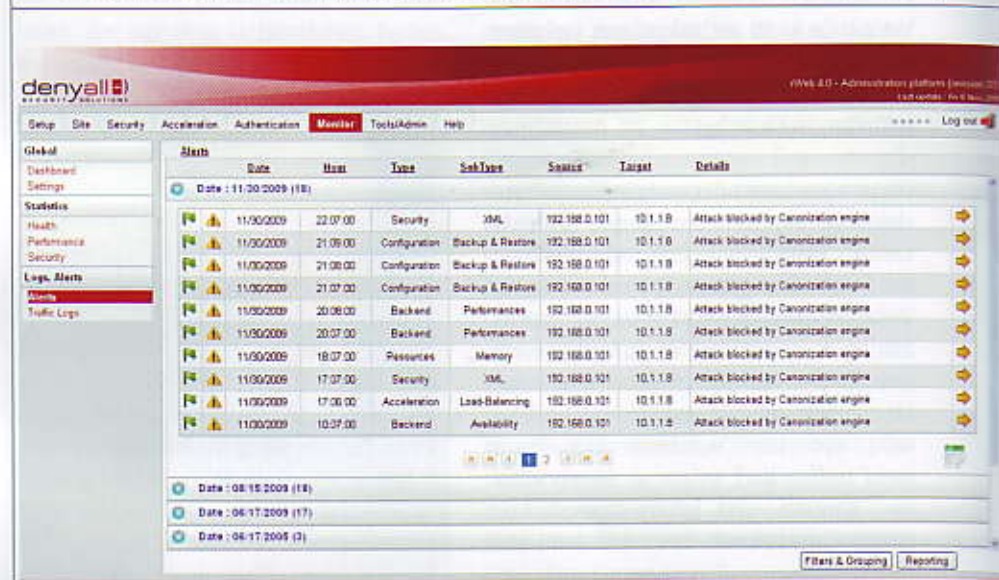
hand der folgenden zwei Beispiele sollen die Kosten verdeutlicht werden:

Beispiel 1: Ausfall im Servicebereich

In einem großen Finanzportal beim Online-Trading werden Benutzereingaben nicht ordentlich validiert. Die Hardware ist an allen Punkten hochverfügbar ausgelegt und kann im Fehlerfall eines Systems problemlos weiterarbeiten. Die Applikation basiert auf Java und liegt einer Linux-Maschine mit Tomcat zugrunde. Der Angreifer hat



Ein Regelwerk (hier für das http-Protokoll) definiert in der WAF, was alles überwacht werden soll. Quelle: Deny All



Bericht über geblockte Angriffe in einer WAF. Quelle: Deny All

festgestellt, dass es in einem Formular möglich ist, Daten an das darunterliegende Betriebssystem weiterzuleiten. Diese Art von Angriff nennt man „Command Injection“. Er ermöglicht es einem Angreifer, Befehle auf der Kommandozeile abzusetzen. Auch wenn diese im ersten Schritt nur mit Rechten des Serverusers (wahrscheinlich nobody oder tomcat) ausgeführt werden, sollte es einem erfahrenen IT-Security-Spezialisten gelingen, den Server so zu beschädigen, dass er keine Daten mehr verarbeiten kann. In diesem Fall sollte dann ein zweiter oder dritter Server die Arbeit der defekten Maschine übernehmen, doch oft sind alle Server gleich konfiguriert und der Angriff kann auf allen Servern ausgeführt werden. Das Unternehmen kann in diesem Zeitraum keine Aufträge mehr ausführen und erleidet so einen Verdienstausschlag.

Eine WAF kann solche Angriffe über einfaches Blacklisting entdecken und blocken, bevor sie am Server Schäden hervorrufen. Neben dem Blacklisting kann auch ein sogenanntes Greylisting zum Schutz verwendet werden. Bei einigen Herstellern wie zum Beispiel Deny All ist diese Art des Schutzes als Scoringlist implementiert. Es handelt sich wie bei einer Blacklist auch hier um einfache Pattern (Muster), jedoch ist jedes Pattern mit einer Gewichtung versehen. Bei einer Scoringlist sind die Pattern sehr kurz gehalten und es können beliebig viele Pattern pro Anfrage übereinstimmen (match).

Erst wenn die Gewichtung aller übereinstimmenden Muster über einen Schwellenwert steigt, wird der Request geblockt. Diese Methode hat gegenüber dem normalen Blacklisting drei Vorteile:

1. Der Angriff kann viel feiner und gezielter erkannt werden.
2. Dadurch wird die Rate der False Positives aus einer Blacklist deutlich minimiert.
3. Es können Angriffe über mehrere Parameter hinweg erkannt werden.

Beispiel 2: Ausfall in der Produktion

Ein Automobilhersteller setzt Fertigungsroboter ein, die über eine Browserschnittstelle bedient/programmiert werden. Die Daten werden ohne vorherige Syntaxprüfung zentral in einer Datenbank gespeichert und dort von den Robotern direkt abgerufen,

um die Aufträge auszuführen. Auch hier sind alle Systeme hochverfügbar ausgelegt, um die Prozesssicherheit zu gewährleisten.

Ein Mitarbeiter programmiert einen neuen Auftrag für die Fertigung von 10.000 Motorhauben, vertippt sich bei der Eingabe einer Größenangabe und verwendet statt des Dezimalzeichens „.“ ein „.“. Da es sich hierbei um ein Steuerungszeichen der SQL-Sprache handelt, können die Roboter die Daten nicht richtig abrufen bzw. werden die Daten in der Datenbank gar nicht erst gespeichert. Im schlimmsten Fall werden 10.000 fehlerhafte Motorhauben gefertigt.

Eine WAF kann einen solchen Fehler erkennen, blocken und so die Produktion vor einem Ausfall oder vor Fehlproduktion schützen. Wie bereits zuvor beschrieben, könnte auch hier eine Blacklist oder Scoringlist zum Einsatz kommen, jedoch wäre die Rate der False Positives sehr hoch, wenn ein Pattern nur aus einem einzigen Zeichen bestehen würde. Besser geeignet für den Schutz eines solchen Fehlers ist eine Whitelist. Eine Whitelist definiert jede URL und die dazugehörigen Parameter. Im vorgenannten Beispiel wird in der Whitelist definiert, dass im Parameter \$size nur eine positive fünfstellige Zahl übertragen werden darf. Somit kann der Tippfehler keinen Produktionsausfall mehr hervorrufen.

Es gibt durchaus noch weitere Funktionen einer WAF, um die Prozesssicherheit zu erhöhen. Wichtig zu erwähnen sind dabei Funktionen zum Schutz von Cookies. Cookies sollten entweder auf ihren Inhalt hin überprüft (wurden sie auf dem Weg von der Applikation zurück zum Server verändert?) oder komplett verschlüsselt werden. Auch wenn in einem Cookie nur die SessionID des Users gespeichert werden sollte, wäre ein Fehler in der Serversoftware denkbar, der die SessionID vorhersagbar macht und so ein erhebliches Risiko produziert. Genauso gefährden Application DoS- (Denial-of-Service-) oder Password Brute Force-Angriffe die Prozesse im Unternehmen. Eine WAF erlaubt hier die Definition von Regeln, über die solche Angriffe erkannt und verhindert werden können.

Webapplikationen müssen heute schnell und effektiv umgesetzt werden. Oft bleibt dabei der Faktor Sicherheit außen vor. Der

Grund hierfür ist das fehlende Sicherheitsbewusstsein der Programmierer. Meist ist jedoch der Zeitdruck, unter welchem eine Webapplikation programmiert und live gestellt wird, der ausschlaggebende Faktor, warum die Sicherheit einer Webapplikation in den Hintergrund rückt. In der Realität zeigt sich zudem, dass Softwareunternehmen zum Teil mit Aussagen wie „Wir konzentrieren uns auf die Erstellung von Software, Security ist nicht unser Kerngeschäft“ die Kompetenz der Security auf dedizierte Systeme verlagern.

Fazit

Ausfall- oder Prozesssicherheit ist nicht mehr nur eine Sache von hochverfügbar ausgelegter IT-Infrastruktur. Sie ist natürlich zwingender Bestandteil der Prozesssicherheit, viele heutige Bedrohungen erfordern jedoch darüber hinausgehende Maßnahmen. Gerade bei Webapplikationen ist das Risiko eines absichtlich oder unabsichtlich erzeugten Fehlers sehr hoch und der Einsatz einer WAF dringend zu empfehlen. ■



Julian Totzek-Hallhuber,
Technical Consultant Deny All