CVE-2021-44228

Açıklama

Bu Güvenlik Uyarısı, Apache Log4j'deki bir uzaktan kod yürütme güvenlik açığı olan CVE-2021-44228'i ele alır. Saldırganlar kimlik doğrulama olmadan uzaktan kullanılabilir, yani bir kullanıcı adı ve parola gerekmeden bir ağ üzerinden bu zafiyet sömürülebilir.

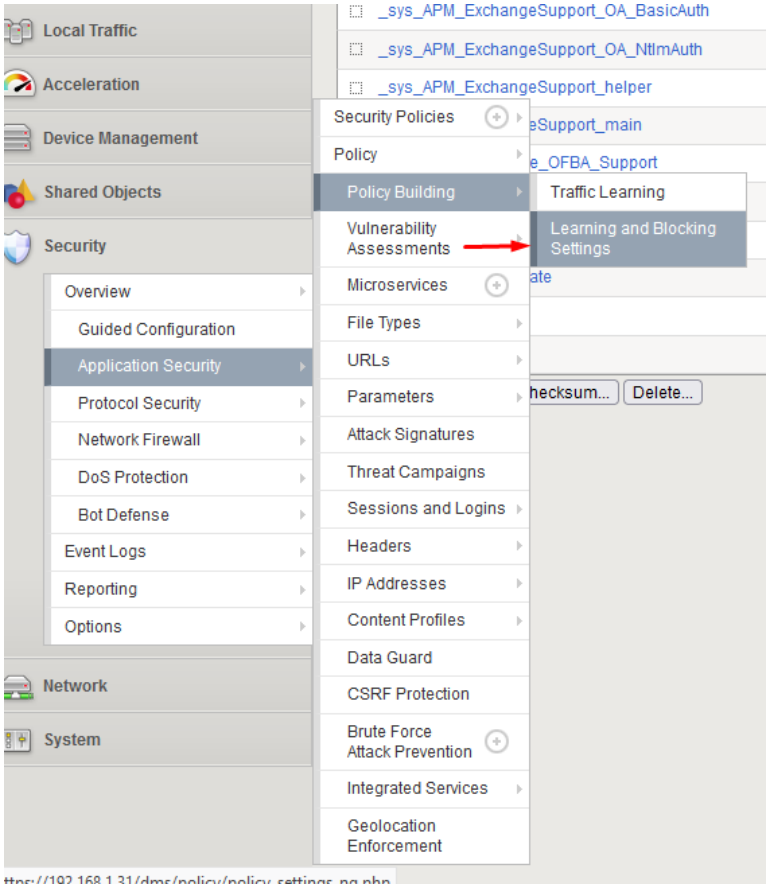
Önem Derecesi

Kritik

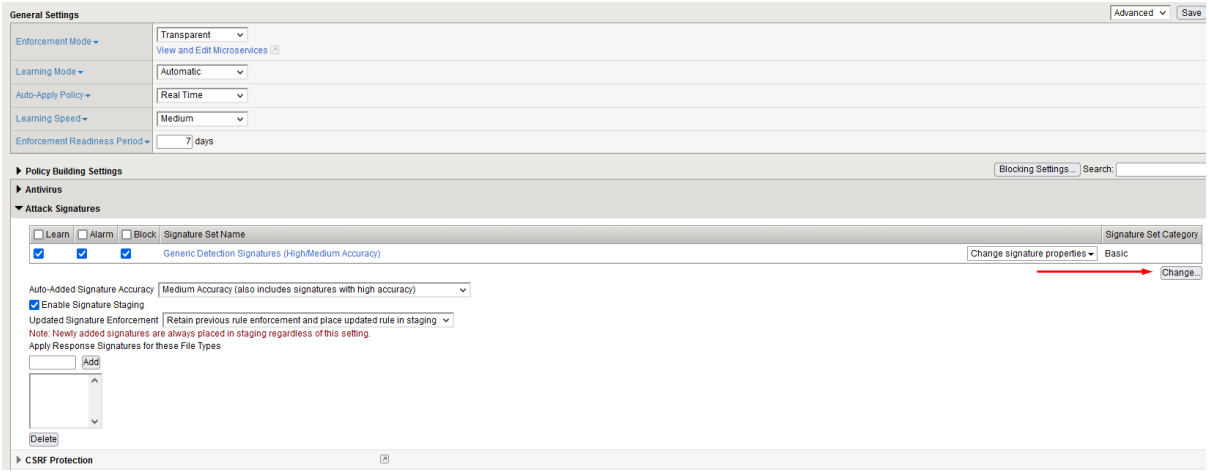
F5 WAF için alınacak önlem

1. WAF Politikası

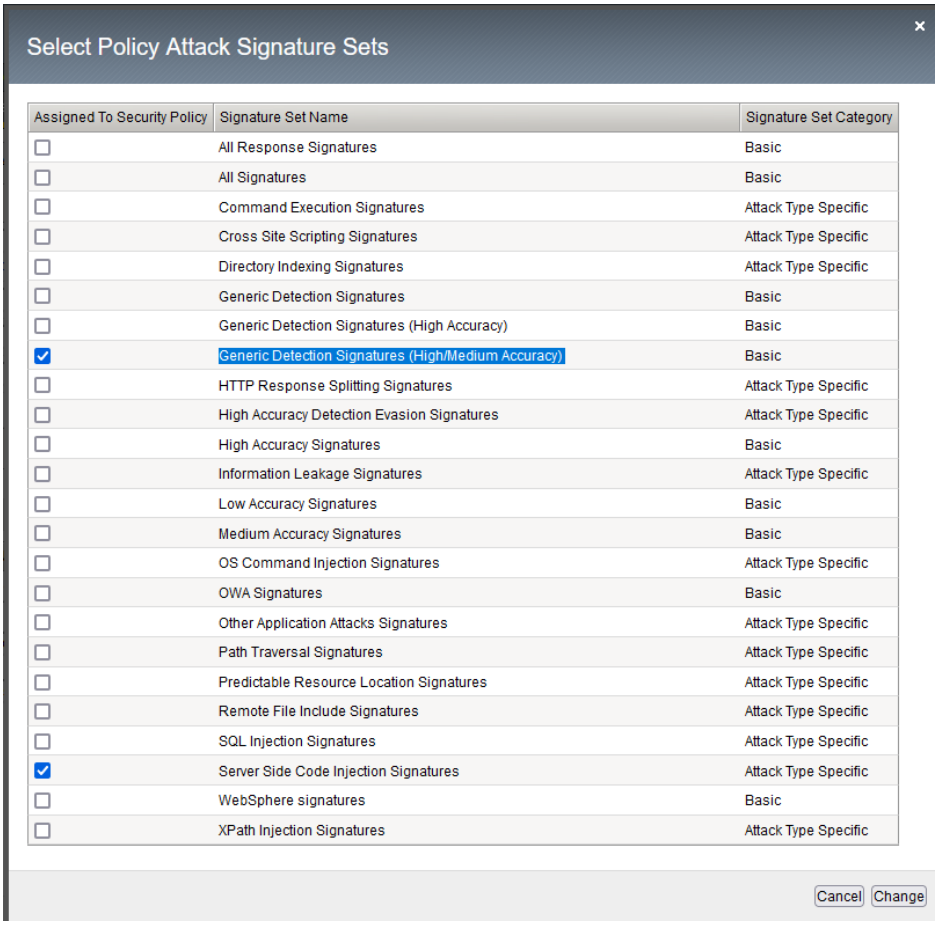
- ➔ Security>Application Security>Policy Building>Learning and Blocking Settings sekmeleri takip edilir.Açılan pencerede doğru politika seti seçildiğinden emin olunur.
- ➔ Attack Signatures a tıklanır.



➔ Attack Signatures a tıklanır.Açılan pencerede change tuşuna basılır.



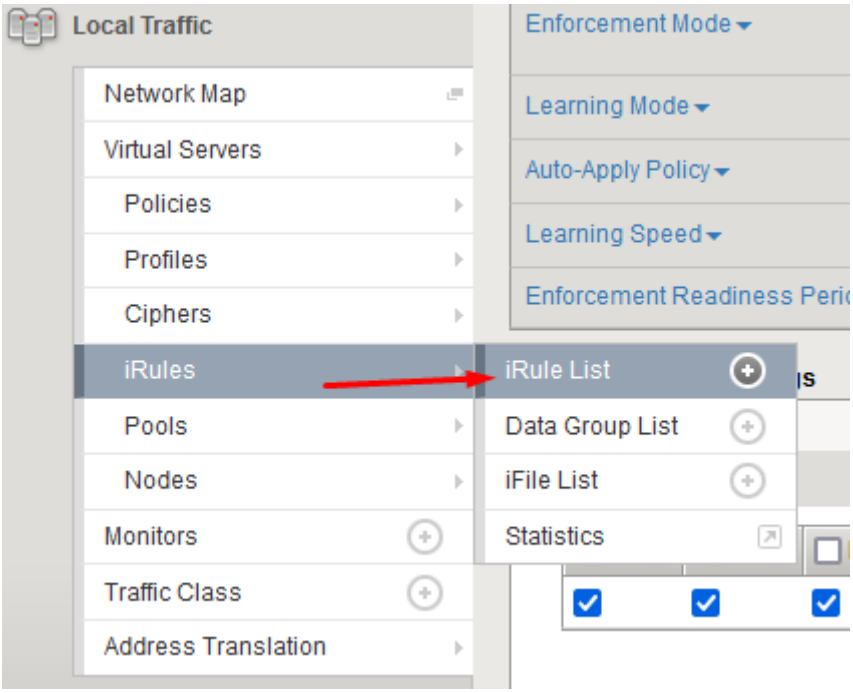
➔ Açılan pencerede Generic Detection Signatures (High/Medium Accuracy) ve Server Side Code Injection Signatures imza setlerinin açık olduğundan emin olunur.Eğer açık değil ise açılır.



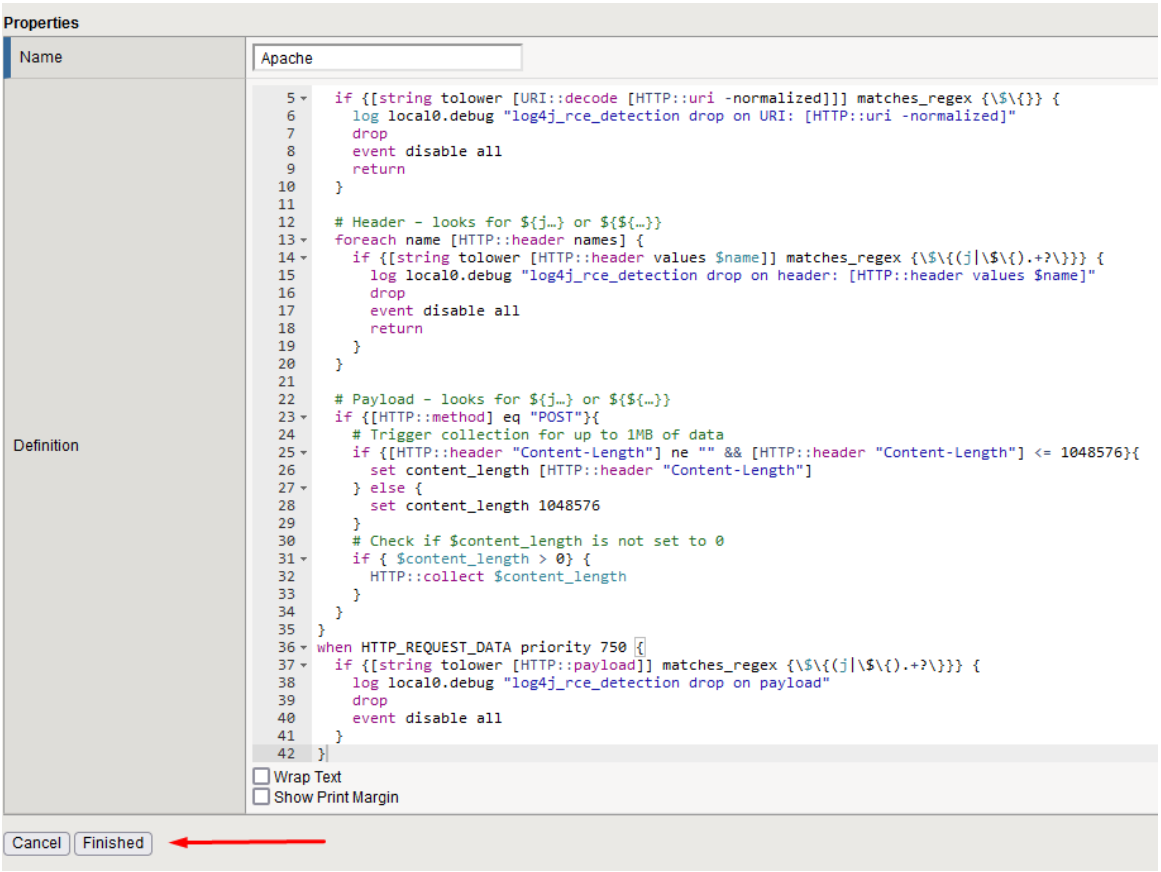
➔ Daha sonrasında Attack Signatures altında imza setlerinin durumunun aşağıdaki gibi olduğu kontrol edilir.Daha sonrasında Save ve Apply policy tuşlarına basılarak kayıt edilir.

2.I-Rule

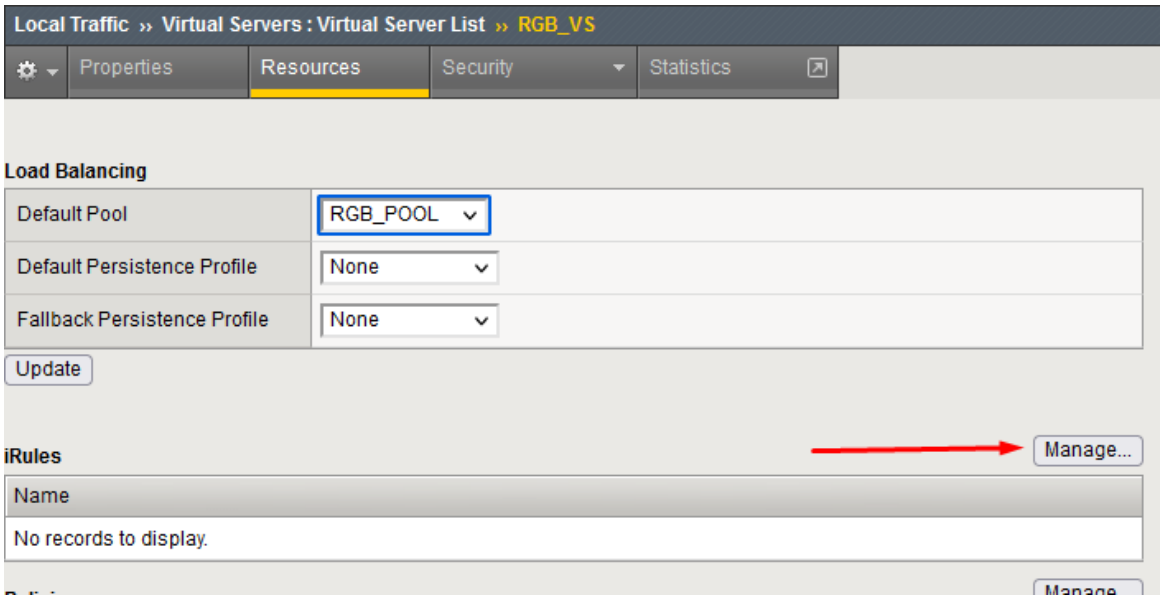
➔ CVE-2021-4428 spesifik irule yazmak içinLocal Traffic>iRule>iRule List menusüne gidilir.Create tuşuna basılır ve yeni bir irule yazma penceresi açılır.



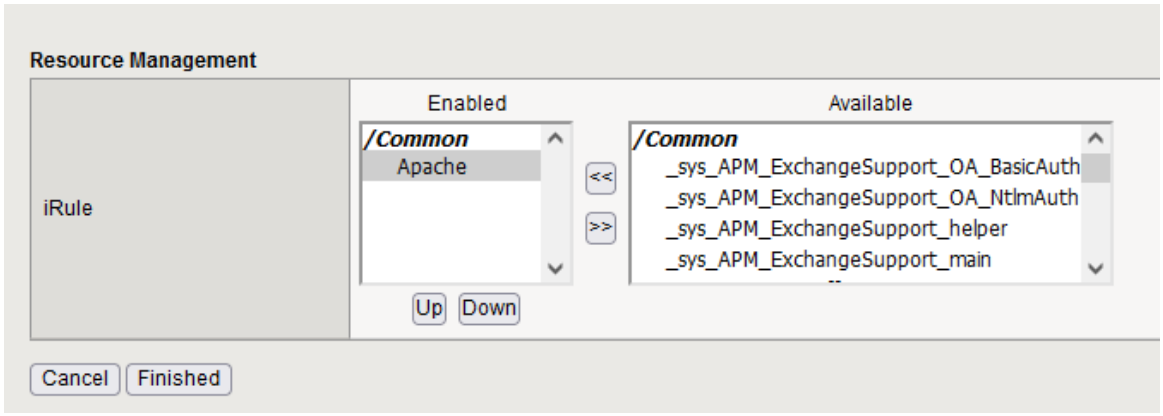
➔ Açılan pencerede bu irule a bir isim verilir ve defination kısmına CVE-2021-4428 spesifik oluşturulan kod girilir ve Finished tuşuna basılarak kural kayıt edilir.(Yazılan rule Ek kısmında verilmiştir.



➔ Daha sonrasında bu kuralın uygulanması gereken Virtual servera girilir.Girilen virtual serverda Resources menusünün altunda irule sekmesinde ki Manage tuşu bulunur ve tıklanır.



➔ Açılan menüde daha önceden oluşturduğumuz I-Rule seçilir ve finishede basılır.



Referans Bağlantıları

https://nvd.nist.gov/vuln/detail/CVE-2021-44228

https://www.tenable.com/blog/cve-2021-44228-proof-of-concept-for-critical-apache-log4j-remote-code-execution-vulnerability

EK

```
when HTTP_REQUEST priority 750 {
 # less aggressive regexp for those concerned about false positives "\$\{(\$\{env:[^:]+:-|\$\{[a-z]+:)?j\}?(\$\{env:[^:]+:-|\$\{[a-z]+:)?n.+:.+\}" (remove quotes)
 # very aggressive regexp "\$\{.+?\}" (remove quotes)
 # URI – based on 200004474
 if {[string tolower [URI::decode [HTTP::uri -normalized]]] matches_regex {\$\{}} {
  log local0.debug "log4j_rce_detection drop on URI: [HTTP::uri -normalized]"
  drop
  event disable all
  return
 }

 # Header – looks for ${j…} or ${${…}}
 foreach name [HTTP::header names] {
  if {[string tolower [HTTP::header values $name]] matches_regex {\$\{(j|\$\{).+?\}}} {
   log local0.debug "log4j_rce_detection drop on header: [HTTP::header values $name]"
   drop
   event disable all
   return
  }
 }

 # Payload – looks for ${j…} or ${${…}}
 if {[HTTP::method] eq "POST"}{
  # Trigger collection for up to 1MB of data
  if {[HTTP::header "Content-Length"] ne "" && [HTTP::header "Content-Length"] <= 1048576}{
   set content_length [HTTP::header "Content-Length"]
  } else {
   set content_length 1048576
  }
  # Check if $content_length is not set to 0
  if { $content_length > 0} {
   HTTP::collect $content_length
  }
 }
}
when HTTP_REQUEST_DATA priority 750 {
 if {[string tolower [HTTP::payload]] matches_regex {\$\{(j|\$\{).+?\}}} {
  log local0.debug "log4j_rce_detection drop on payload"
  drop
  event disable all
 }
}
```