**Ataque 2: Vulnerabilidade Log4Shell / Apache Log4j (CVE-2021-44228)**

**Data do ataque:** divulgado em dezembro de 2021; exploração ativa logo após a divulgação (final de 2021)

**Tipo de ataque:** vulnerabilidade de execução remota de código (Remote Code Execution — RCE) em biblioteca de logging amplamente usada, explorada como zero-day / vulnerabilidade massiva.

**Descrição de como aconteceu:**

* A biblioteca Apache Log4j versão 2 (entre versões 2.0-beta9 até 2.15.0, dependendo das configurações) permitia que entradas de log manipuladas pelo atacante contivessem expressões JNDI (“Java Naming and Directory Interface”) que referenciavam servidores LDAP, RMI, etc. Com essa capacidade, um atacante remoto poderia inserir payloads que causavam execução de código arbitrário, se o log registrasse essas entradas e houvesse suporte a lookup via JNDI.
* A vulnerabilidade foi chamada de “Log4Shell”. Muitas aplicações, serviços, servidores que usavam Log4j sem mitigação estavam vulneráveis; devido à ubiquidade da biblioteca, o vetor de ataque era muito amplo.

**Vulnerabilidade explorada / CVE(s):**

* **CVE-2021-44228**: vulnerabilidade de execução remota de código, nas versões afetadas do Log4j.
* Além disso, após a descoberta começaram a surgir problemas relacionados, como **CVE-2021-45046** (quando a correção inicial foi incompleta em algumas configurações) que permitia RCE ou divulgação de informações em determinados contextos.
* Também **CVE-2021-45105** (negando-serviço, DoS) em certas configurações do Log4j.

**Impactos e prejuízos estimados:**

* Impacto potencial massivo: milhares de serviços e aplicações em todo o mundo ficaram expostos, inclusive infra-estruturas críticas, serviços em nuvem, sistemas governamentais, empresas privadas de diversos setores.
* Riscos reais de comprometimento de dados, instalação de malware, ransomware, espionagem; custos elevados de mitigação, varredura, atualização, auditoria de segurança, bem como danos à reputação.
* Não há um número público confiável único de custo total global, mas ficou claro que o esforço para patching, resposta a incidentes, contenção foi gigantesco, com equipes de segurança e fornecedores de software trabalhando intensamente.

**Tipo de proteção que poderia ter sido aplicada para evitá-lo:**

1. **Uso de versões da biblioteca atualizadas**, aplicando patches assim que liberados. Log4j publicou correções logo após descoberta.
2. **Restrição de funcionalidades vulneráveis**, por exemplo desativação de lookups via JNDI, configuração segura de loggers, evitar configurações que permitam carregamento de classes externas ou remoto.
3. **Validação de entrada / sanitização** de dados que são logados: evitar que dados externos ou do usuário sejam inseridos diretamente no log sem verificação.
4. **Defesa em profundidade / camadas de segurança:** WAFs (Web Application Firewalls) capazes de filtrar padrões de ataque, detecção/monitoramento de comportamento anormal, escaneamento de vulnerabilidades de dependências.
5. **Inventário de componentes de software**, especialmente de bibliotecas de terceiros, monitoramento de dependências, uso de SBOM (Software Bill of Materials) para saber onde se usa Log4j ou outras com riscos associados.
6. **Plano de resposta a incidentes e auditoria contínua**, para detectar quando houve exploração, remediação rápida e comunicação.