ANÁLISE SEMANAL · SEMANA DE 22 A 29 DE JULHO DE 2026

A cadeia de build virou alvo primário.

Em uma única semana, falhas relevantes apareceram em servidor de CI, em repositório Git, em automação de workflow, em pacotes públicos e no agente de IA que revisa pull request. Não é coincidência de calendário. É o atacante seguindo o caminho mais curto ate produção.

> **A tese:** a empresa gastou uma década endurecendo o perímetro de produção e deixou a esteira que *escreve* a produção com controle de acesso de ambiente de desenvolvimento. Comprometer o build entrega o que comprometer o servidor entregaria, com menos ruído, menos monitoramento e assinatura legítima no artefato final.

Convém começar pelo desconforto: na maioria das organizações que conhecemos na região, o servidor de integração contínua tem mais credenciais privilegiadas que qualquer servidor de aplicação. Chave de nuvem, token de registry, segredo de banco, credencial de deploy. Tudo isso mora lá porque precisa morar lá. E, ao mesmo tempo, esse servidor raramente aparece no escopo do pentest anual, quase nunca está no inventário de ativos criticos e frequentemente responde na rede interna sem segmentação real.

O que a semana mostrou

CAMADA	OCORRENCIA	POR QUE IMPORTA
Servidor de CI	Falha crítica no TeamCity permite executar comandos de sistema operacional sem autenticação.	Acesso pre-autenticação ao orquestrador de build alcança todos os segredos da esteira de uma vez.
Repositório	Execução remota de código no Gitea permite que quem tem permissão de escrita plante um git hook e rode comandos.	Permissão de escrita costuma ser distribuída com generosidade. Deixa de ser permissão de código e vira permissão de shell.
Automação	Escape de sandbox no n8n permite que editores de workflow executem comandos como o processo da aplicação.	Ferramentas de automação concentram integracoes e tokens de múltiplos sistemas de negócio.
Dependências	Dois pacotes npm comprometidos executam um RAT quando importados em Node.js.	O gatilho e a importacao, não a execução da aplicação. Um npm install em estação de desenvolvedor já basta.
Agente de revisão	Falha no MCP do Azure DevOps permite que comentários ocultos em pull request sequestrem agentes de revisão com IA.	Injeção de prompt deixa de ser curiosidade acadêmica quando o agente tem permissão de escrita no repositório.
Runner	Atacantes usam runners do GitHub Actions para atingir servidores cPanel e WHM.	O runner e computacao confiavel com saída de rede. Serve para construir software e para atacar a partir de dentro.

O padrão por trás dos casos

Trocando os nomes dos produtos, sobra sempre a mesma estrutura. Existe um componente que precisa de privilégio alto para funcionar, recebe permissoes amplas por conveniência operacional e e tratado como ferramenta interna em vez de ativo crítico. O atacante não precisa de zero-day sofisticado quando encontra um servidor de CI com interface exposta e um modelo de permissão herdado de quando a equipe tinha oito pessoas.

A novidade real da semana é a última linha da tabela acima. Agentes de revisão de código introduzem um vetor que nao existia: conteúdo controlado pelo atacante entra no contexto de um processo que tem permissão de escrita. Um comentário de pull request é dado não confiável por definição. Quando esse dado alimenta um agente autorizado a alterar o repositório, a fronteira entre entrada e instrução desaparece.

É o mesmo padrão descrito no OWASP Top 10 para aplicações de LLM, so que aplicado ao lugar onde o código nasce. Vale a comparação com um problema antigo: injeção de SQL tambem comecou como curiosidade e virou década de prejuízo porque a indústria demorou a aceitar que entrada de usuário nunca é instrução.

Um contraponto honesto

Nem tudo na semana foi deterioração. O GitHub passou a aplicar um período de espera de três dias no Dependabot antes de propor a adoção de novas versões de pacote. É uma medida modesta e de baixa tecnologia que ataca uma janela real: a maior parte dos pacotes comprometidos e detectada e removida em poucas horas ou dias. Quem atualiza automaticamente no mesmo instante da publicação assume o risco integral dessa janela.

Não resolve ataque paciente, obviamente. Um invasor que mantenha o pacote limpo por semanas passa pelo período de espera sem esforço. Ainda assim, é um bom exemplo de controle que troca um pouco de velocidade por bastante redução de exposição, e que não exige comprar nada.

Roteiro de trinta dias

SEMANA 1

Inventário e exposição

Liste todo servidor de CI, repositório autogerenciado, runner e plataforma de automação. Verifique quais respondem fora da rede de administração. Registre o dono de cada um por nome, não por equipe.

SEMANA 2

Segredos e escopo

Levante quais credenciais vivem na esteira, qual o escopo de cada uma e há quanto tempo não são giradas. Reduza escopo antes de reduzir quantidade. Prefira credencial de vida curta emitida por federação.

SEMANA 3

Permissão de escrita

Revise quem tem escrita em repositório e quem pode editar workflow. Trate essas duas permissoes como acesso privilegiado e aplique revisão periódica com evidência para auditoria.

SEMANA 4

Agentes e dependências

Mapeie todo agente de IA com acesso ao repositório e defina se ele pode escrever ou apenas comentar. Ative período de espera na adoção automática de novas versões de dependência.

Como isso conversa com NIST CSF 2.0 e LGPD

Do ponto de vista de governança, a esteira de build cai em Identificar e Proteger no NIST CSF 2.0, e quase sempre com nota baixa em ambas por um motivo simples: não está no inventário. Se o ativo não existe no registro, o controle não tem onde ser aplicado e a evidência não tem onde ser coletada.

Há também um ângulo de LGPD que costuma passar despercebido. Ambiente de desenvolvimento com dado real de produção é comum, e uma esteira comprometida vira incidente com dado pessoal, com todas as obrigações de comunicação que isso implica. Vale checar, antes que a checagem seja feita por outro motivo, se há dado real em base de teste e quem autorizou.

Para quem responde a conselho, a formulação que costuma funcionar é direta: o controle de mudança da empresa cobre o que entra em produção, mas não cobre quem pode alterar o processo que constrói o que entra em produção. Fechar essa lacuna é barato agora é caro depois.

Fontes e aprofundamento

· OWASP · Top 10 for Large Language Model Applications · owasp.org/www-project-top-10-for-large-language-model-applications/

· NIST · Cybersecurity Framework 2.0 · nist.gov/cyberframework

· CISA · Secure by Design · cisa.gov/securebydesign

· The Hacker News · cobertura de 27 e 28/07/2026 sobre TeamCity, Gitea, n8n, pacotes npm comprometidos, MCP do Azure DevOps e período de espera do Dependabot.

Este documento integra a publicação semanal da central de recursos LATAMSEC. Versão online, com links ativos para as fontes primárias, em latamsec.com.br/recursos/cadeia-de-build-alvo-primário/