# REA no Windows com WSL

Tutorial do zero | Ubuntu + Node.js + Java + Ghidra + Claude Code

Versão organizada a partir do seu procedimento, com os erros corrigidos e o primeiro acesso ao Claude explicado. Revisão: 07/10/2026.

## O que você vai montar

O WSL permite usar Linux dentro do Windows. No WSL 2, há um kernel Linux real em uma máquina virtual leve. O Ubuntu terá seu próprio terminal, programas e arquivos. Os arquivos de análise ficarão na pasta pessoal do Linux.

## Onde digitar cada coisa

PowerShell do Windows: a instalação inicial do WSL.
Terminal Ubuntu: instalação das ferramentas, REA, preparação do alvo e abertura do Claude.
Dentro do Claude Code: comandos como /exit, /login e /mcp.

## Antes de começar

Tenha acesso à internet, permissão de administrador no Windows e um arquivo ou uma pasta de aplicativo que deseja analisar. Os caminhos deste guia usam o usuário Windows Micro; troque esse nome se a sua pasta for diferente. O guia considera um PC x64, Intel ou AMD.

## Como seguir os comandos

Execute um comando por vez e espere terminar. Não copie os rótulos “bash” ou “powershell”, nem o prefixo micro@DESKTOP...$. Ao usar sudo, digite a senha do Ubuntu: ela não aparece na tela, nem como asteriscos. Pressione Enter.

Já começou? Confira os testes de cada etapa e siga a partir da primeira que estiver incompleta. Se claude já abre, vá diretamente à etapa 4.2, sobre primeiro acesso.

# 1. Instalar e abrir o Ubuntu

Onde: PowerShell como administrador. No menu Iniciar, procure PowerShell e escolha “Executar como administrador”.

```
wsl --install -d Ubuntu-24.04
```

Reinicie o PC. Abra Ubuntu 24.04 pelo menu Iniciar; o nome pode aparecer apenas como Ubuntu. Aguarde a preparação e crie um usuário Linux e uma senha. Esse usuário pode ser micro, mas não precisa ser igual ao usuário do Windows.

A partir daqui, use a janela do Ubuntu. Se aparecer micro@DESKTOP...:~$, você está no terminal Linux.

# 2. Ferramentas, Java e Node.js

Primeiro, atualize a lista de pacotes:

```
sudo apt update
```

Instale as ferramentas de compilação, o descompactador, o Java e o curl:

```
sudo apt install -y build-essential unzip openjdk-21-jdk curl
```

Configure o repositório do Node.js 22. Espere esse comando terminar sem erro:

```
curl -fsSL https://deb.nodesource.com/setup_22.x | sudo -E bash -
```

Agora instale o Node.js, que também fornece npm e npx:

```
sudo apt install -y nodejs
```

Confira, um por vez:

```
node --version
npm --version
java -version
javac -version
```

Resultado esperado: Node 22.19.0 ou superior dentro da série 22; npm com um número de versão; java e javac na versão 21. O npm serve para instalar pacotes e o npx para executar ferramentas como o instalador do REA.

# 3. Instalar o Ghidra 12.1.4

Onde: baixe no navegador do Windows; extraia e configure no Ubuntu. Use Ghidra 12.1.4 + JDK 21, combinação validada pelo REA.

Abra github.com/NationalSecurityAgency/ghidra/releases e baixe o ZIP PUBLIC da versão 12.1.4 para Downloads. Não baixe “Source code”.

Se um link direto der 404, confira na página de releases o nome exato do ZIP. A data no nome pode ser diferente. Evite adivinhar o endereço; baixe pelo link do release.

## Extrair no Ubuntu

```
cd ~

unzip /mnt/c/Users/Micro/Downloads/ghidra_12.1.4_PUBLIC_*.zip
```

Troque Micro pelo nome da pasta do seu usuário Windows. Se você já colocou o ZIP na pasta pessoal do Ubuntu, use este comando em vez do anterior:

```
unzip ghidra_12.1.4_PUBLIC_*.zip
```

Mensagens “inflating” e “creating” são normais. Se a pasta já foi extraída completamente, não precisa repetir.

## Configurar pela primeira vez

Execute cada linha uma vez para guardar os caminhos:

```
echo 'export GHIDRA_INSTALL_DIR="$HOME/ghidra_12.1.4_PUBLIC"' >> ~/.bashrc

echo 'export JAVA_HOME=/usr/lib/jvm/java-21-openjdk-amd64' >> ~/.bashrc

source ~/.bashrc
```

O arquivo correto é ~/.bashrc, sem ponto no final. Confira antes de seguir:

```
echo "$GHIDRA_INSTALL_DIR"

echo "$JAVA_HOME"

ls "$GHIDRA_INSTALL_DIR/support/analyzeHeadless"
```

O caminho do Ghidra deve terminar em ghidra_12.1.4_PUBLIC; o Java deve apontar para /usr/lib/jvm/java-21-openjdk-amd64. O último comando deve mostrar o arquivo analyzeHeadless.

# 4. Instalar o Claude sem erro de permissão

Onde: terminal Ubuntu. Execute as linhas uma por vez. Se Claude já abre e mostra uma versão, pule para o login. Não use sudo na instalação npm do Claude.

## A. Definir uma pasta de instalação sua

```
mkdir -p "$HOME/.npm-global"

npm config set prefix "$HOME/.npm-global"

npm config get prefix
```

Antes de avançar: deve aparecer /home/micro/.npm-global, ou a pasta equivalente do seu usuário. Se aparecer /usr ou /usr/local, a configuração ainda não está correta.

## B. Ativar o PATH agora e nas próximas sessões

PATH é a lista de pastas em que o Ubuntu procura os comandos. A primeira linha abaixo ativa o caminho imediatamente. A segunda salva para os próximos terminais.

```
export PATH="$HOME/.npm-global/bin:$PATH"

echo 'export PATH="$HOME/.npm-global/bin:$PATH"' >> ~/.bashrc

source ~/.bashrc
```

Copie o comando limpo acima. Não copie o desenho de boas-vindas, o prefixo micro@DESKTOP...$, nem linhas misturadas do terminal.

## C. Instalar e confirmar

```
npm install -g @anthropic-ai/claude-code
```

Se aparecer “added 2 packages”, a instalação terminou. O aviso “npm notice New major version...” não é erro e não exige atualizar o npm.

```
claude --version
```

Antes de avançar: o comando precisa mostrar uma versão, por exemplo 2.1.292 (Claude Code). O número pode ser diferente. Se aparecer EACCES, EINTEGRITY ou command not found, siga as correções da próxima página.

# 4.1. Os três erros que aconteceram

## EACCES: permission denied em /usr/lib

O npm estava tentando escrever na pasta do sistema. Volte à etapa 4A, configure o prefixo e confirme /home/micro/.npm-global. Depois repita a instalação sem sudo. Não altere as permissões de /usr/lib.

## EINTEGRITY: integrity checksum failed

O download não passou na verificação de integridade. No seu caso, o cache tinha conteúdo corrompido e a instalação terminou ao usar outro cache. Primeiro, verifique o cache:

```
npm cache verify
```

Mensagens como “Corrupted content removed” indicam que entradas inválidas foram removidas. Depois, repita usando uma pasta nova de cache:

```
npm install -g @anthropic-ai/claude-code \
  --cache "$HOME/.npm-cache-claude-novo" --prefer-online
```

As duas linhas formam um único comando; a barra invertida no fim da primeira continua na linha seguinte. Aguarde terminar. Se falhar novamente, pare e confira a nova mensagem; não desative a verificação de integridade.

## Instalou, mas claude: command not found

Isso aconteceu depois de “added 2 packages”: o comando ainda não estava acessível pelo PATH. Ative-o na sessão atual e teste:

```
export PATH="$HOME/.npm-global/bin:$PATH"

claude --version
```

Se funcionar, salve a linha correta em ~/.bashrc conforme a etapa 4B. Para testar a persistência, abra uma nova janela do Ubuntu e rode claude --version novamente.

Se ainda falhar, confira estes comandos e compartilhe os resultados:

```
npm config get prefix
ls -l "$HOME/.npm-global/bin"
```

Se a listagem contiver claude, teste pelo caminho completo:

```
"$HOME/.npm-global/bin/claude" --version
```

Se não existir, não trate o problema apenas como PATH: será necessário conferir a instalação.

# 4.2. Primeiro acesso e login

Onde: terminal Ubuntu, após claude --version funcionar.

```
claude
```

1. Tema ou modo visual: se aparecer uma lista de opções claro/escuro, escolha com as setas e confirme com Enter.

2. Security notes: leia os avisos e pressione Enter quando a tela pedir “Press Enter to continue”.

3. Método de login: escolha a opção de conta Claude/assinatura para usar sua assinatura compatível. Anthropic Console/API usa faturamento separado. Os nomes e a ordem das telas podem variar.

## “Browser didn’t open? Use the url below to sign in”

Essa mensagem não indica falha de instalação. No WSL, pode ser necessário abrir o navegador do Windows manualmente.

1. Na tela de login, pressione c se ela indicar “c to copy”. Cole o endereço na barra do navegador do Windows.
2. Se a cópia não funcionar, selecione e copie o endereço completo exibido. Ele pode ocupar várias linhas; não inclua espaços ou quebras de linha.
3. Entre na conta desejada e conclua a autorização.
4. Volte ao terminal. Se o fluxo fornecer um código e o Claude pedir esse código, cole-o e confirme. Aguarde a tela de conversa.

Use o link da sessão atual. Se expirar, reinicie o login; dentro da conversa do Claude, use /login.

Quando o login estiver concluído, saia para instalar o REA:

```
/exit
```

Antes de avançar: você deve estar novamente no prompt do Ubuntu, como micro@DESKTOP...:~$. Comandos npm e npx são digitados ali, não na conversa do Claude.

## Se o desenho apareceu misturado aos comandos

Isso pode ser apenas a renderização do terminal. Se suspeitar de uma linha corrompida em ~/.bashrc, faça uma cópia e abra o arquivo:

```
cp ~/.bashrc ~/.bashrc.backup-rea
```

```
nano ~/.bashrc
```

Corrija somente a linha de PATH adicionada, se estiver errada. Ela deve ser:

```
export PATH="$HOME/.npm-global/bin:$PATH"
```

No nano: Ctrl+O, Enter e Ctrl+X. Depois rode source ~/.bashrc e claude --version no terminal Ubuntu.

# 4.3. Testar o Claude antes do REA

Após concluir o login e chegar à conversa, faça um teste simples. Digite dentro do Claude:

```
Oi?
```

Resultado esperado: o Claude responde. Isso confirma que a sessão consegue conversar; ainda não testa o REA ou o Ghidra. Se aparecer erro de autenticação, limite ou cobrança, resolva essa mensagem antes de seguir.

Depois, volte ao terminal Ubuntu:

```
/exit
```

## Retomar uma conversa depois

Ao sair, o Claude pode mostrar um comando de retomada. Execute esse comando no terminal Ubuntu se quiser continuar a mesma conversa. Para a sessão mostrada no seu log:

```
claude --resume 46050548-f414-48c5-a988-ce7361465617
```

Esse identificador pertence à sua conversa; não é um código de instalação. Em outro computador ou conta, use o identificador da própria sessão. Também é possível abrir o seletor de conversas:

```
claude --resume
```

Retomar a conversa é opcional. Para começar normalmente, basta executar claude. Para instalar o REA, saia da conversa com /exit primeiro.

# 5. Configurar o REA: escolhas corretas

Onde: terminal Ubuntu.

```
npx --yes rea-agents@latest setup
```

Aguarde o assistente. “Detected: Claude Code” significa que ele foi encontrado, mas não significa que está marcado.

## 1. Which agents should use REA?

Use ↑/↓ para destacar Claude Code, pressione Espaço para marcá-lo e só depois pressione Enter. Deixe os outros agentes desmarcados.

Confira: a resposta deve indicar Claude Code. Se aparecer none, nenhum agente foi selecionado; essa execução não configura o MCP do Claude. Cancele com Ctrl+C antes de aplicar e rode o setup novamente.

## 2. Install the guided REA skill for CLI use?

Se essa pergunta aparecer, pode escolher Yes para instalar as instruções de uso no terminal. A skill sozinha não substitui o registro do MCP no Claude; a escolha do agente no passo anterior continua necessária.

## 3. Install Hopper for deep binary analysis?

Para esta análise, escolha No: você já está preparando o Ghidra. Hopper é outro motor de análise e sua instalação não é necessária neste roteiro. Se houver uma escolha de provider, selecione Ghidra e informe o caminho de GHIDRA_INSTALL_DIR.

## 4. Ready to review / Apply these changes?

Revise os caminhos e as mudanças. O plano deve incluir a integração/MCP do Claude Code; os nomes podem variar conforme a versão. Confira o Ghidra quando solicitado. Se o plano mostrar apenas Hopper e a skill de CLI, falta selecionar o Claude.

Somente com as escolhas corretas, confirme Yes, apply e aguarde terminar. Depois abra o Claude novamente para carregar a configuração.

# 5.1. Entender o log que apareceu

No trecho enviado, o assistente registrou estas escolhas:

```
Which agents should use REA?
  none
Install the guided REA skill for CLI use?
  Yes
Install Hopper for deep binary analysis?
  Yes
```

O plano mostrou duas mudanças: instalar Hopper e instalar a skill em ~/.agents/skills/reverse-engineer-anything. Não mostrou a integração do Claude Code. Por isso, esse trecho não comprova que o MCP foi configurado.

## Se você já confirmou Yes, apply

A última linha “Hopper deep-analysis provider” não confirma sucesso nem falha; o log foi interrompido ali. Se a instalação ainda está rodando, aguarde o resultado. Se preferir cancelar, use Ctrl+C e espere voltar ao prompt. Não abra um segundo setup ao mesmo tempo.

Quando estiver novamente no terminal Ubuntu, rode:

```
npx --yes rea-agents@latest setup
```

Destaque Claude Code, marque com Espaço e confirme com Enter. Escolha No para instalar Hopper. Revise a integração do Claude e aplique. Se Hopper já tiver sido instalado, não é necessário removê-lo para continuar este tutorial com Ghidra.

## Checagem depois de configurar

Abra o Claude na pasta da análise e digite /mcp, como explicado na etapa 7. O REA precisa aparecer e conectar. O setup registra o MCP; o teste “Oi?” verifica a conversa; a análise do binário verifica o uso do motor de análise. Cada teste confirma uma parte diferente.

# 5.2. Se o setup não registrar o MCP

Use esta alternativa somente se o REA não estiver registrado. Não é necessário repetir o cadastro quando /mcp já mostra REA conectado. Uma busca de texto isolada em arquivos internos não basta para confirmar ausência; confira os comandos do Claude na pasta correta.

Se estiver dentro do Claude, saia:

```
/exit
```

No terminal Ubuntu, entre na pasta do alvo e confira os servidores:

```
mkdir -p ~/alvo

cd ~/alvo

claude mcp list
```

Se REA não aparecer, registre-o nesta pasta, com escopo local explícito:

```
claude mcp add --scope local --transport stdio rea -- \
  npx --yes rea-agents@latest mcp
```

As duas linhas formam um único comando. A mensagem “Added stdio MCP server” confirma que o cadastro foi gravado, mas ainda não confirma que o servidor consegue iniciar.

```
claude mcp list

claude mcp get rea
```

Depois, abra uma nova sessão:

```
claude
```

Dentro do Claude:

```
/mcp
```

O REA precisa aparecer e conectar. Claude Docs é opcional: o tutorial não o instala, portanto não espere obrigatoriamente dois servidores. O cadastro manual adiciona o servidor; não substitui todas as instruções e verificações do setup guiado.

## Por que foi registrado em duas pastas?

Sem --scope, o padrão do Claude é local: configuração privada associada ao projeto, normalmente em ~/.claude.json. No seu log houve cadastros separados para /home/micro e /home/micro/demo_alvo. No tutorial genérico, faça o registro em ~/alvo e abra o Claude ali.

# 5.3. Escopo e falhas de conexão

## Local, project e user são diferentes

local: privado e associado ao projeto em que você cadastrou; é o padrão usado neste guia.
project: configuração compartilhável em .mcp.json.
user: disponível nos projetos desse usuário do Claude Code.

Para usar REA em todos os seus projetos, a alternativa é --scope user no comando de cadastro. Escolha um escopo conscientemente; não faça cadastros duplicados para tentar corrigir uma conexão. Configurações locais podem ter prioridade sobre as do usuário.

## REA connected

O servidor iniciou. Pode seguir para o prompt de análise. Isso ainda não comprova que Ghidra analisará qualquer arquivo; a primeira operação de análise testa o provider.

## REA connecting

Na primeira vez, o npx pode estar baixando o pacote. Aguarde um pouco e consulte /mcp novamente. Se persistir, examine a mensagem de conexão: não presuma que todo atraso é download.

## REA failed

Saia do Claude e, no Ubuntu, confira um comando por vez:

```
cd ~/alvo
claude mcp get rea
node --version
npx --version
echo "$GHIDRA_INSTALL_DIR"
echo "$JAVA_HOME"
```

Se as variáveis estiverem vazias, recarregue ~/.bashrc e confira novamente:

```
source ~/.bashrc
```

Para conferir o REA e diagnosticar os providers:

```
npx --yes rea-agents@latest --help

npx --yes rea-agents@latest doctor --json
```

O subcomando mcp é documentado para iniciar o servidor stdio. Falhas também podem vir do download npm, versão do Node, comando cadastrado, configuração ou timeout. Java/Ghidra podem falhar apenas ao começar a análise. Use a mensagem real para escolher a correção.

Depois de corrigir a causa, abra claude novamente em ~/alvo e confira /mcp. Se o nome já existir, examine claude mcp get rea antes de alterar ou remover o registro. Não apague ~/.claude.json para reparar um servidor.

# 6. Colocar o arquivo em ~/alvo

Onde: terminal Ubuntu. Use uma pasta dedicada, contendo somente o arquivo que você quer analisar. Exemplos: um .exe do Windows, um binário Linux ou o pacote de um aplicativo. Não misture outros programas nem o código-fonte na mesma pasta.

## Como o Ubuntu vê as pastas do Windows

O disco C: fica acessível em /mnt/c. Assim, C:\Users\SeuNome\Downloads corresponde a /mnt/c/Users/SeuNome/Downloads. SeuNome é a pasta do usuário Windows, que pode ser diferente do usuário Linux micro. No seu PC, o caminho mostrado anteriormente usa Micro.

## Copiar um .exe do Windows

Crie a pasta e confira se já existe algum arquivo nela:

```
mkdir -p ~/alvo

ls -la ~/alvo
```

Se houver um alvo anterior, use outra pasta vazia ou mova-o antes de continuar. Este guia não apaga arquivos automaticamente. Troque SeuNome e meu_programa.exe pelos nomes reais:

```
cp "/mnt/c/Users/SeuNome/Downloads/meu_programa.exe" ~/alvo/
```

Exemplo com a pasta de usuário do seu PC:

```
cp "/mnt/c/Users/Micro/Downloads/meu_programa.exe" ~/alvo/
```

Execute apenas o exemplo que corresponde ao seu caminho. As aspas permitem espaços no nome do arquivo.

```
cd ~/alvo

ls -lh
```

A listagem deve mostrar somente o alvo escolhido. Para um binário Linux já no Ubuntu, copie-o pelo seu caminho Linux, por exemplo:

```
cp "$HOME/Downloads/meu_binario" ~/alvo/
```

Esse é outro exemplo: use-o em uma pasta vazia para uma investigação diferente. Se um aplicativo depende de uma pasta inteira, mantenha seu pacote completo como único alvo; bibliotecas e recursos pertencentes a ele podem ser necessários.

# 6.1. O que pode ser analisado

## REA dentro do WSL/Linux

Analisa binários Linux com o motor compatível, como Ghidra. Também pode inspecionar estaticamente executáveis Windows (.exe/PE) via Ghidra, quando o formato e a arquitetura forem suportados. Isso não transforma o .exe em um programa Linux nem fornece controle da interface Windows.

## REA rodando diretamente no Windows

A documentação descreve a integração nativa Windows com Ghidra como experimental, com escopo específico para aplicações PE x86-64. Esse status se refere à integração Windows, não a toda análise de arquivos PE feita pelo Ghidra no Linux. Este tutorial instala o REA no Ubuntu/WSL.

## Apps e outros formatos

“App” pode significar tecnologias diferentes. Um aplicativo Electron/JavaScript, .NET ou Python empacotado pode exigir ferramentas e arquivos adicionais. Primeiro identifique o formato e confira o suporte das ferramentas realmente disponíveis no MCP. A extensão .exe sozinha não informa a tecnologia nem garante suporte.

## Comece pela análise estática

A análise estática lê o arquivo sem iniciar o programa. Observar a execução exige um ambiente compatível com o sistema, a arquitetura e as dependências do aplicativo. Não presuma que um binário Linux funciona no Windows ou que um .exe funciona como binário Linux.

A análise pode produzir pseudocódigo e evidências do comportamento, mas não garante recuperar o código-fonte original, comentários, nomes ou todo o projeto.

# 7. Conferir o MCP e pedir a análise

Onde: terminal Ubuntu.

```
cd ~/alvo
```

```
claude
```

Se aparecer uma confirmação de confiança, confira que é a pasta do alvo escolhido. Depois, dentro do Claude Code:

```
/mcp
```

Confirme que REA aparece e está conectado. Estar conectado confirma o servidor; a primeira análise verifica se as ferramentas e o motor conseguem trabalhar com seu arquivo.

## Prompt neutro de exemplo

Cole na conversa:

Use o REA para analisar o arquivo que está nesta pasta. Descubra o que este programa faz e mostre as evidências. Primeiro identifique o formato, a arquitetura e a tecnologia, e confira se as ferramentas disponíveis suportam o alvo. Para análise nativa, use Ghidra quando compatível. Comece pela análise estática, sem executar ou modificar o arquivo. Explique o funcionamento em linguagem simples, separe fatos de hipóteses e indique as limitações. Se faltar algum arquivo ou ferramenta, explique o que é necessário antes de continuar.

## Como aprofundar

Depois do resultado inicial, faça perguntas como “Qual função realiza este cálculo?”, “Onde o programa lê os dados?” ou “Que evidência sustenta essa conclusão?”. Para um aplicativo grande, escolha um componente ou função de cada vez.

Se o REA não aparecer, siga a etapa 5.2 de cadastro manual. Se estiver registrado mas desconectado, siga a etapa 5.3 de diagnóstico.

# 8. Continuar uma investigação

Abra o Ubuntu e entre na pasta do alvo:

```
cd ~/alvo

claude
```

Para retomar uma conversa existente, em vez de iniciar outra:

```
claude --resume
```

Escolha a sessão correspondente ao alvo. Não precisa reinstalar as ferramentas a cada uso.

## Trabalhar com outro arquivo

Crie outra pasta, copie o novo alvo e abra o Claude nela. Isso deixa os arquivos e os resultados de cada investigação organizados. Exemplo:

```
mkdir -p ~/outro_alvo

cp "/mnt/c/Users/Micro/Downloads/outro_app.exe" ~/outro_alvo/

cd ~/outro_alvo

claude
```

## Depois da análise inicial

Faça perguntas específicas: “Qual função lê este arquivo?”, “Onde este valor é calculado?”, “Quais evidências sustentam essa conclusão?”. Se a investigação precisar observar a execução, peça um plano compatível com o sistema do aplicativo antes de avançar.

Ferramentas adicionais dependem da tecnologia encontrada. Por exemplo, programas Python empacotados podem exigir um procedimento específico; isso não faz parte da instalação básica do REA com Ghidra.

# Se algo der errado

## sudo: npm: command not found

O npm do Ubuntu está ausente ou fora do caminho. Ter Node no Windows não resolve isso. Volte à etapa 2 e confira node --version e npm --version. Não avance se a configuração do repositório NodeSource falhou.

## JAVA_HOME foi salvo em ~/.bashrc.

O ponto final cria outro arquivo. Basta executar a linha correta da etapa 3 e source ~/.bashrc. Não é necessário apagar o arquivo errado para continuar.

## Ghidra ou Java não encontrados

```
echo "$GHIDRA_INSTALL_DIR"
echo "$JAVA_HOME"
ls "$GHIDRA_INSTALL_DIR/support/analyzeHeadless"
"$JAVA_HOME/bin/javac" -version
```

Se algum caminho não existir, confira o nome da pasta extraída. Para obter o diagnóstico do REA, no terminal Ubuntu:

```
npx --yes rea-agents@latest doctor --json
```

## PowerShell ou Ubuntu?

PowerShell administrador: wsl --install. Ubuntu: sudo, export, cp, npm e npx. Dentro do Claude: /mcp, /login e /exit. Cole uma linha por vez, sem o prefixo micro@DESKTOP...$ ou os rótulos bash/powershell.

## Arquivo ou pasta não encontrados

```
ls "/mnt/c/Users/Micro/Downloads/"
```

Confira o nome exato do arquivo e a pasta do usuário Windows. Use aspas em caminhos com espaços. O arquivo precisa existir no local indicado antes do comando cp.

## Claude não abre ou REA não aparece

Se claude não for encontrado, rode source ~/.bashrc e claude --version. Se o REA não aparecer, saia com /exit, refaça o setup selecionando Claude Code e abra novamente em ~/alvo. Se estiver registrado mas desconectado, consulte o erro mostrado em /mcp e o diagnóstico acima.

## Login não concluiu

Dentro do Claude, use /login e confira a conta selecionada.

## Checklist do ambiente

1. Node, npm, Java e javac respondem.
2. Ghidra está no caminho configurado.
3. Claude abre com login concluído.
4. REA está conectado em /mcp.
5. A pasta contém o arquivo ou app correto.

# Referências e notas de versão

As interfaces podem mudar. Este documento combina o seu procedimento com as páginas oficiais consultadas em 07/10/2026. Os comandos devem ser executados no seu PC; não foram testados no seu ambiente WSL.

### Microsoft: instalação do WSL

https://learn.microsoft.com/en-us/windows/wsl/install

### Anthropic: instalação do Claude Code

https://code.claude.com/docs/en/setup

### Anthropic: MCP e escopos

https://code.claude.com/docs/en/mcp

### REA: projeto e assistente de setup

https://github.com/morluto/rea

### REA: requisitos e instalação do Ghidra

https://github.com/morluto/rea/blob/main/docs/installation.md

### Ghidra: versões oficiais

https://github.com/NationalSecurityAgency/ghidra/releases

### O que foi corrigido

As correções de EACCES, EINTEGRITY e PATH agora fazem parte do fluxo, com testes antes de avançar. Inclui o uso de um cache novo, o navegador que não abriu e a checagem de linhas visualmente misturadas. A instalação de Node/npm vem antes do uso de npm e npx. O curl foi incluído nas ferramentas básicas. JAVA_HOME é salvo em ~/.bashrc, sem o ponto indevido. Os caminhos do Ghidra usam $HOME. A instalação npm do Claude usa um prefixo do usuário, sem sudo. As escolhas iniciais, os avisos, o método de autenticação e o login agora fazem parte da sequência. O aviso de atualização do npm foi separado dos erros reais.

Este guia usa Ghidra 12.1.4 + JDK 21, combinação validada pelo REA. A configuração usa diretamente a pasta ghidra_12.1.4_PUBLIC. Identificação de strings, imports ou funções é evidência parcial: hipóteses sobre algoritmos e ausência de rede precisam ser confirmadas pela análise, não presumidas.