

# Azure DevOps

Guia de implementação

VERACODE

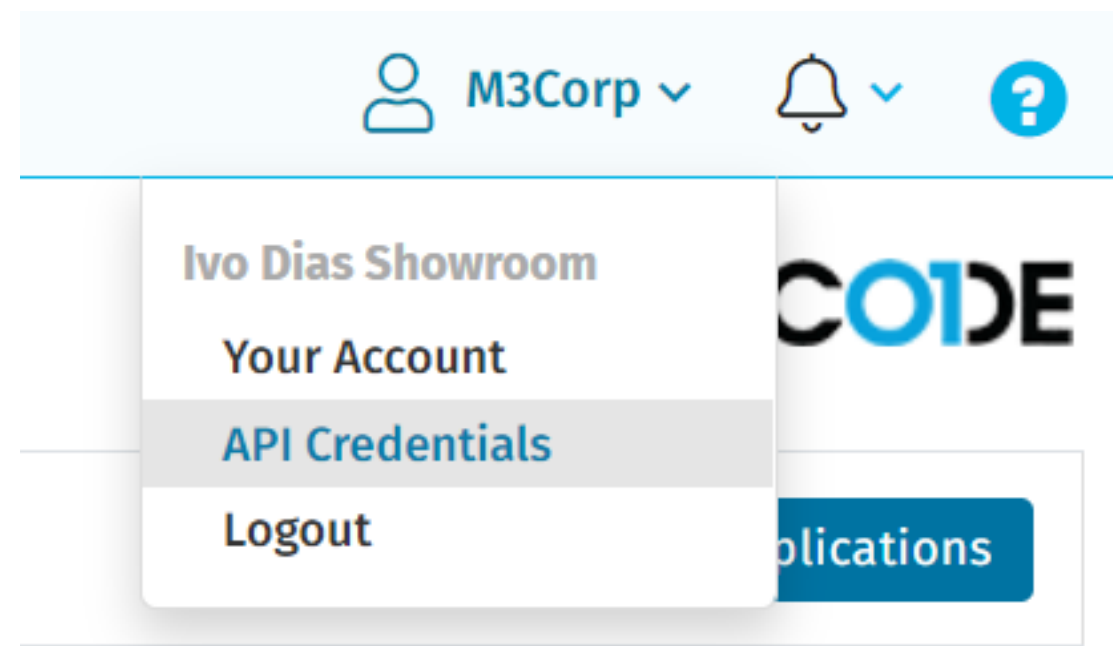


# Orientações Gerais

- O objetivo desse material é servir de guia para uma implementação de Veracode dentro do ambiente do Azure DevOps
- No final de cada seção, colocamos algumas ponderações como os erros mais comuns, servindo de guia para eventuais testes ou referencia rápida
- Num cenário multi-stage, como a organização entre ambientes de homologação e produção, podemos implementar uma estratégia de Sandbox, onde esta recebe o nome do ambiente extra
- Para os casos de apenas scan em Produção, não é preciso adicionar nenhuma informação no campo de Sandbox
- Neste guia veremos apenas a implementação via interface gráfica, para exemplos sobre como fazer o processo via YAML, acesse nossa pagina no [GitHub oficial da M3Corp](#)

# Credenciais

- Nosso primeiro passo é obter as credenciais no portal da Veracode
- O recomendado é a criação de um usuário API específico para essas integrações
- Conforme a imagem ao lado, precisamos apenas clicar no nosso usuário no canto superior esquerdo e selecionamos a opção de credenciais



# Credenciais



- Vamos precisar do ID e da Secret Key para fazer as integrações
- Por padrão, essas credenciais duram 1 ano, mas conforme a imagem é possível revoga-las a qualquer momento

## Credentials Details

[Generate API Credentials](#)[Revoke API Credentials](#)

**ID:**

\*\*\*\*\*

**Secret Key:**

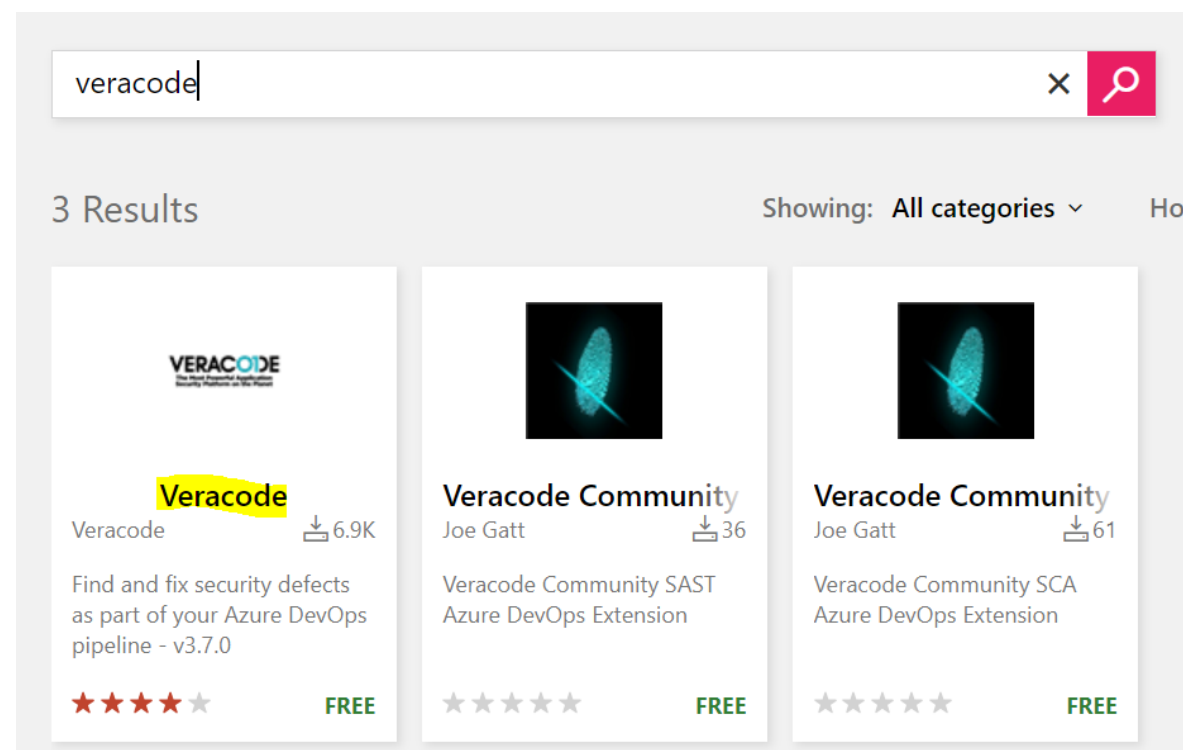
\*\*\*\*\*

**Created:** 13 Jul 2021 @ 9:21 am EDT

**Expires:** 13 Jul 2022 @ 9:21 am EDT

# Obtendo o Plug-in

- Dentro da loja de extensões, buscamos por “Veracode”
- A primeira opção é a do plug-in oficial
- Depois de instalarmos ela, vamos ter acesso a duas tarefas, que vamos ver no detalhe nos próximos slides
- É preciso que o agente tenha o Java configurado (nas imagens da Microsoft normalmente ele já vem disponível)



# Veracode Upload and Scan



- Essa é a Task que faz as análises, para fazer o login nela, precisamos informar os dados que pegamos no portal
- O ideal é a criação de uma [Service Connection](#) dentro do Azure, assim conseguimos tirar essa informação da tarefa e também aproveitar para uma implementação mais fácil em outros pipelines e tarefas

Veracode Upload and Scan ⓘ

[Link settings](#) [View YAML](#) [Remove](#)

Task version 3.\* ▼

Display name \*  
Upload and scan

Connection Details ^

Select Connection Source \* ⓘ

☒ Service Connection ☐ Credentials

Select Service Connection \* ⓘ | [Manage](#)

Base ▼ [Refresh](#) [+ New](#)

# Veracode Upload and Scan



- Para fazermos o scan precisamos de pouca coisa, basicamente apenas um nome de aplicação, um identificador para o scan (como um número de versão) e o caminho dos arquivos (conforme o [guia de empacotamento](#))
- Para uma boa análise é vital que empacotemos os arquivos conforme o guia, já que não segui-lo pode fazer com que nossa análise mostre resultados não tão precisos
- Conforme a imagem, podemos ter todas essas informações abstraídas para ser aproveitadas em todos os processos
- Na parte dos resultados, temos uma checkbox para informar se queremos esperar a análise completar

Veracode Scan Settings ^

Application Name \* ⓘ

AzDevOps.\$(Build.DefinitionName)

Scan Name \* ⓘ

\$(build.buildNumber)

Filepath \* ⓘ

\$(Build.ArtifactStagingDirectory)/\$(Build.BuildId).zip

Advanced Scan Settings v

Veracode Scan Results ^

☒ Import Results upon Scan Completion ⓘ

☐ Fail build if application fails security policy ⓘ

Fail build if no scan results within (in minutes) \* ⓘ

360

# Veracode Upload and Scan



- Nas opções avançadas podemos fazer a implementação de [Sandbox](#) (relatório separado do principal)
- Um exemplo de uso é ter configurado para Stages específicos, como DEV e QA/HMG
- Uma opção muito importante é a da criação de perfil de aplicação, com ela ativa o sistema sempre verifica se já existe um com o nome informado e caso não tenha, já faz a criação automaticamente no portal
- Podemos também ativar a opção de “quebrar” o pipeline caso ocorra alguma falha no envio dos arquivos

## Advanced Scan Settings ^

Sandbox Name ⓘ

☐ Create Sandbox ⓘ

Optional Arguments ⓘ

☒ Create Application Profile ⓘ

☐ Fail build if Upload and Scan build step fails ⓘ



# Veracode Upload and Scan - Dicas

- Nos LOGs podemos verificar se um scan foi iniciado corretamente ou não, conforme a imagem ao lado
- A task pode ser configurada para aguardar ou não o resultado dos scans
- Uma lista de erros comuns:
  - Java indisponível
  - App Profile com o mesmo nome, mas fora do acesso da conta integradora
  - Existe um scan no mesmo perfil e/ou Sandbox que ainda não foi concluído, não permitindo que um novo inicie
  - Um scan pode ficar parado por estar aguardando que os módulos sejam revisados na plataforma, para saber mais veja a parte de [Review Modules](#)

```
[2021.09.28 14:06:33.130] Creating a new analysis with name "20210928.1".
[2021.09.28 14:06:35.428] The analysis id of the new analysis is "14334165".
[2021.09.28 14:06:35.428] Uploading: target/verademo.war
[2021.09.28 14:06:38.139] Starting pre-scan verification for application "AzDevOps.Java" analysis "20210928.1".
[2021.09.28 14:06:39.130] Scan polling interval is set to the default of 120 seconds.
[2021.09.28 14:06:39.130] Application "AzDevOps.Java" analysis "20210928.1" will be automatically submitted for scanning if the pre-scan verification is successful.
[2021.09.28 14:06:39.634] The status of the new analysis is "Pre-Scan Submitted".
[2021.09.28 14:06:39.634] Requesting analysis info again in 120 seconds.
[2021.09.28 14:08:41.470] The status of the new analysis is "Pre-Scan Submitted".
[2021.09.28 14:08:41.470] Requesting analysis info again in 120 seconds.
[2021.09.28 14:10:42.714] The status of the new analysis is "Scan In Process".
[2021.09.28 14:10:42.714] Requesting analysis info again in 120 seconds.
[2021.09.28 14:12:44.153] The status of the new analysis is "Scan In Process".
[2021.09.28 14:12:44.153] Requesting analysis info again in 120 seconds.
[2021.09.28 14:14:45.455] The status of the new analysis is "Results Ready".
Start Obtaining Build Info
```

# Veracode Flaw Importer



- Essa tarefa é a responsável por fazer a importação das falhas encontradas para ao Azure Boards
- O processo de login é exatamente o mesmo da anterior
- Para utiliza-la, precisamos apenas informar qual a Aplicação que queremos importar. Com isso, conseguimos importar qualquer registro no portal, mesmo de análises feitas em outras ferramentas de CI/CD, e até mesmo iniciadas manualmente via portal ou linha de comando
- Em alguns casos, quando utilizamos templates diferentes do padrão, pode ser preciso uma [personalização](#)

Application Name \* ⓘ

AzDevOps.\$(Build.DefinitionName)

Sandbox Name ⓘ

Work Item Settings ^

Import \* ⓘ

All Flaws

Work Item Type \* ⓘ

Issue

Area \* ⓘ

\$(system.teamProject)

☒ Add CWE as a Tag ⓘ

# Veracode Flaw Importer



Essa é a visão que temos das falhas importadas com a tarefa

## Work items

Recently updated ▾ | + New Work Item ▾ | ↗ Open filtered view in Queries | Column Options ...

DVWA					Types ▾	Assigned to ▾	States ▾	Area ▾
ID	Title			Assigned To	State			
✓ 803	Veracode Flaw (Static): Untrusted Initialization, PP_DEMO.PHP...		...	Unassigned	To Do			
802	Veracode Flaw (Static): Untrusted Initialization, PP_DEMO.PHP.DVW...			Unassigned	To Do			
801	Veracode Flaw (Static): Untrusted Initialization, PP_DEMO.PHP.DVW...			Unassigned	To Do			
800	Veracode Flaw (Static): Untrusted Initialization, PP_DEMO.PHP.DVW...			Unassigned	To Do			
799	Veracode Flaw (Static): Untrusted Initialization, PP_DEMO.PHP.DVW...			Unassigned	To Do			

**ISSUE 803**  
803 Veracode Flaw (Static): Untrusted Initialization, PP\_DEMO.PHP.DVWA, Flaw 87

Unassigned [0 comments](#) [Build\\_12576517](#) [CWE\\_454](#) [+](#)

State: ☒ To Do

Area: PP - DEMOs

Reason: Added to backlog

Iteration: PP - DEMOs

**Description**  
**Veracode Links :** [Application Policy Flaw](#)  
**CWE :** [454](#) External Initialization of Trusted Variables or Data Stores  
**Module :** PHP files within 254.zip  
**Source :** medium.php  
**Line Number :** 23  
**Attack Vector :** !php\_standard\_ns.shell\_exec  
**Description :**  
This call to !php\_standard\_ns.shell\_exec() invokes an external shell. Environment variables inherited from the calling program as well as those modified by the application itself will be passed to the shell. In light of the vulnerability in the shell bash described in CVE-2014-6271, the runtime environment should be reviewed to ensure that this is not an exploitable call.  
  
If this application is being run in an environment in which bash is present, ensure that the version of bash used always includes the most recent security updates. Also, in the application itself, consider creating an allowlist of environment variables to protect against external contamination.  
  
References:













**Planning**  
Priority  
2  
Effort

# Veracode Flaw Importer - Dicas

- O principal ponto de atenção ao utilizar essa task é a configuração das variáveis para os templates personalizados
- É recomendado que se utilize das TAGs para facilitar a criação de Queries personalizadas
- É possível escolher quais falhas vão ser importadas, conforme tipo de scan e se violaram ou não as políticas

Queries

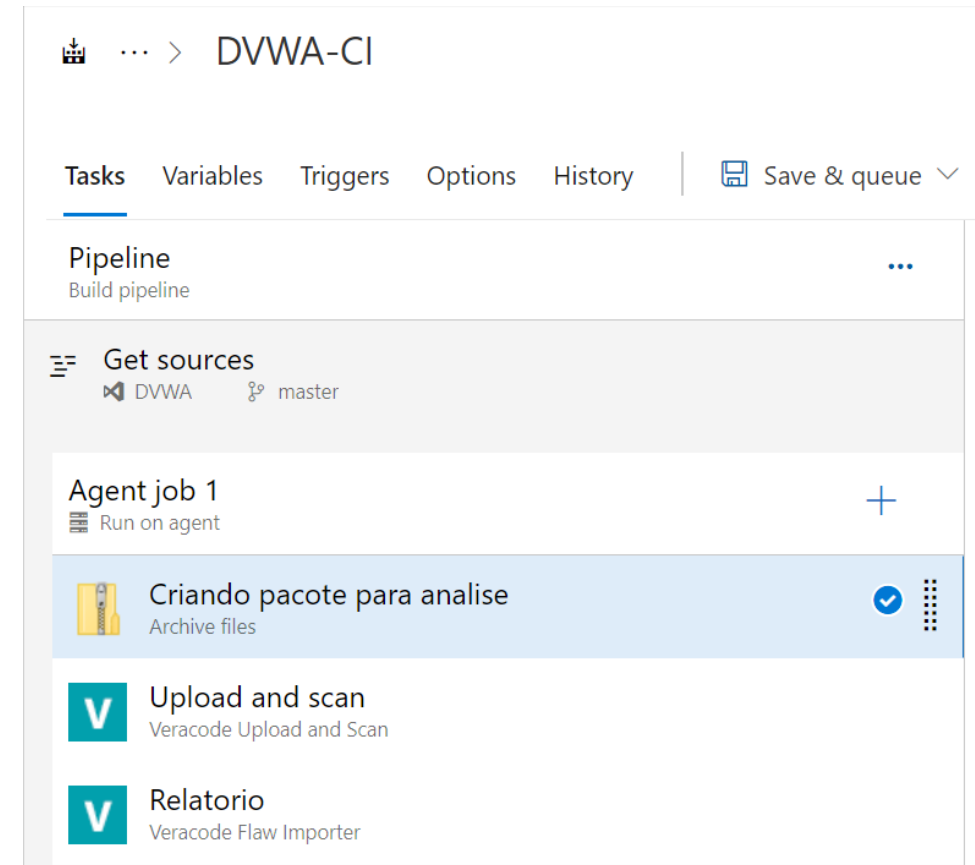
Favorites | All | + New query | ↑ Import Work Items

Title	Folder
Continue where you left off	
 PRD	 My Queries
▼ My favorites (5)	
 DEV	 My Queries
 HMG	 My Queries
 PRD	 My Queries
 Time A	 My Queries
 Time B	 My Queries

# Veracode - Pipeline Completo



- E pronto, esses são todos os passos necessários para implantarmos Veracode no Azure DevOps
- Conforme a imagem ao lado, um pipeline completo para a análise de um projeto em PHP tem apenas 3 tarefas
- Nesse exemplo vimos a forma mais simples de implementação, o resultado esperado dela é a análise do projeto disponibilizando as análises de código próprio e de componentes de terceiros
- Pensando num fluxo onde vamos definir quebras, é interessante criar um pipeline exclusivo para importação de falhas e deixar habilitada a opção da Task para quebrar em caso de falhas



# Veracode - Pipeline Dicas

- Não é preciso colocar as duas etapas num mesmo pipeline, sendo totalmente possível ter um exclusivo apenas para importar as falhas de todos os projetos
- Com a estratégia de Sandbox, podemos analisar todos os ambientes sem misturar os resultados, e depois importar utilizando das TAGs para organizar os resultados no Boards
- Existem outras ferramentas da Veracode que podem ser utilizadas para logar as informações na linha de comando, como o [Pipeline Scan](#) e o [SCA Agent-Based](#)
- Conforme o uso, os tempos de scan costumam diminuir, então é interessante executa-los algumas vezes antes de mensurar o tempo padrão

Jobs		
✓	Agent job 1	10m 34s
✓	Initialize job	5s
✓	Checkout DVWA@master to s	8s
✓	Criando pacote para analise	<1s
✓	Upload and scan	8m 50s
✓	Relatorio	1m 28s
✓	Post-job: Checkout DVWA@mas...	<1s
✓	Finalize Job	<1s
✓	Report build status	<1s



Em caso de dúvidas converse com  
o suporte:  
[support@veracode.com](mailto:support@veracode.com)

SECURING THE SOFTWARE  
THAT POWERS YOUR WORLD

---

**VERACODE**