SUMÁRIO

1. AS VARIÁVEIS DE AMBIENTE DO PHP

As variáveis de ambiente disponíveis no PHP são, na verdade, valores do sistema operacional da máquina e que podem ser acessados na super global $_ENV - um array associativo de escopo global.

Esse array de valores contém dados sobre o ambiente em que o PHP se encontra, como nome do usuário do computador, sistema operacional utilizado, arquitetura do processador, linguagem e codificação do sistema operacional, localização de vários diretórios relevantes do sistema, enfim, dados do nível da máquina e do sistema operacional.

Previsivelmente, por permitir o acesso a esses tipos de dados, essa super global lida com o que podemos chamar de "informações sensíveis", que não podem ser conhecidas por usuários comuns da aplicação. Por isso, por padrão, na instalação do PHP essa super global é desabilitada, o que a torna um array vazio, sem dados.

Para habilitar o seu uso, é preciso acessar o arquivo de configuração do PHP, o php.ini, encontrar o termo "variables_order", e alterar o valor do campo "Default Value" de "GPCS" para "EGPCS". Esses valores são as iniciais das super globais, sendo "E" a inicial de "Environment Variable".

Para verificar todos os dados dessa variável, após habilitada, basta colocá-la como argumento da função "var_dump" ou "print_r", antecedidos da tag de formatação "<pre>".

```
echo "<pre>";

print_r($_ENV);

echo "<pre>";
```

Para acessar um valor em específico existem duas formas, sendo a segunda especialmente importante para o que virá posteriormente. **A primeira**, óbvia, é utilizando a chave do valor com a própria variável, $_ENV["chave"]. **A segunda forma** é com a função getenv("chave").

```
$_ENV["chave"];

getenv("chave");
```

1.1. VARIÁVEIS DE AMBIENTE PARA ARMAZENAR DADOS

Se existe uma coisa que a grande maioria dos projetos de desenvolvimento tem em comum é que eles requerem **dados para sua operação**, como informações do banco de dados, chaves de API's, entre outras informações de configuração necessárias.

Mas como proceder com essas informações? Onde armazená-las? Elas devem ficar dando expostas no código? Não devem, e não podem, porque muitas são sensíveis.

1.1.1. FERRAMENTA DOTENV E O ARQUIVO ENV

Quando pensamos em informações sensíveis, a primeira coisa que nos vem em mente é tirar esses dados do código. Qualquer informação que for escrita no código pode ser conhecida.

Para "solucionar" esse problema, **embora não seja uma ótima solução para o ambiente de produção**, são utilizadas variáveis de ambiente personalizadas, criadas e definidas, no formato chave = valor, em um arquivo sem nome, o .env, e que é ocultado na aplicação.

Mas antes da criação desse arquivo, **é preciso instalar a ferramenta que irá gerenciar o acesso as variáveis de ambiente** que serão definidas nele. É uma ferramenta utilizada para orquestrar as variáveis ambiente de um projeto. Seu nome é dotenv.

Para saber exatamente como prosseguir, criando a comunicação entre a ferramenta e o arquivo, clique aqui. Em termos gerais, existindo valores no arquivo .env, que deverá existir fora da pasta pública da aplicação, **eles poderão ser acessados** a partir da sua chave com a **função getenv("chave")**. Por isso foi dito anteriormente que ela era importante.

Um caso de uso comum desse arquivo é no estabelecimento da conexão com o banco de dados. Utilizando a **classe PDO**, pessoas sem conhecimento dos fatores de segurança escrevem os valores como host, nome do banco de dados, login do sistema de gerenciamento, e senha, na própria instância da classe. Em contraste, utilizando as variáveis de ambiente com a ferramenta dotenv, esses dados são definidos no arquivo .env, e resgatados a partir da função getenv.

1.1.2. OCULTANDO O ARQUIVO. ENV (GITHUB)

No caso de uma possível sua publicação da aplicação, o arquivo .env deve ser ocultado. No caso do Github, basta que seja criado um arquivo **.gitignore**, e nele escrito, sem aspas, em texto puro, o caminho do arquivo .env.

Mas, como isso surge um problema: se o arquivo será ocultado, como, por exemplo, colegas de equipe vão saber os dados necessários para rodar a aplicação? Para isso é comum a utilização de arquivos .env ou .envrc vazios, com apenas o dado da porta do banco de dados exposta.

Dependendo do ambiente de desenvolvimento, como o de uma empresa que utiliza containers Docker, esses arquivos podem ser automaticamente preenchidos porque os desenvolvedores utilizam ambientes idênticos virtualizados. Esses ambientes virtualizados, chamados containers, quando criados, são configurados para receber um conjunto de dados, que podem ser, entre outras coisas, para preencher o arquivo .env de uma aplicação.

O ponto é que, para cada sistema de publicação, para cada provedor cloud, para cada gerenciador de contêineres, terá uma forma diferente de passar uma variável de ambiente para a aplicação.

1.1.3. OS PROBLEMAS NO AMBIENTE DE PRODUÇÃO

Como dito anteriormente, a utilização do arquivo env não é uma solução finalista, ou uma real solução de segurança, como alguns parecem acreditar que é.

Muitos serviços de hospedagem fornecerão uma maneira de armazenar variáveis de ambiente de produção com segurança, sem .env. **Existem incontáveis lacunas**, e outras possibilidades para um atacante acessar dados configurados nesse arquivo. **A segurança de uma aplicação se dá em camadas**.

Idealmente o arquivo .env deve ser visto como uma forma de simplificação da produção. Não é necessário ter arquivos de configuração separados para desenvolvimento, teste, preparação e produção. Em vez disso, cada implantação adiciona, altera ou define suas próprias variáveis de ambiente no mesmo arquivo.

Se estiver preocupado até mesmo com os atacantes mais motivados, o .env não deve conter seus segredos de produção. **As melhores práticas para armazenar dados sensíveis são**: bancos de dados, contas de desenvolvimento, ou serviços externos para não armazenar dados sigilosos localmente.

Além dessas alternativas mais seguras, uma outra opção apresentada na documentação, na seção de “regras para seguir”, do dotenv, é configurar as variáveis diretamente no servidor, para que assim os valores não precisem existir em um arquivo nos diretórios da aplicação.

Para entender melhor, acesse o artigo original sobre esse assunto.

FONTES

https://imasters.com.br/desenvolvimento/protegendo-seus-dados-utilizando-variaveis-de-ambiente

https://blog.rocketseat.com.br/variaveis-ambiente-nodejs/

https://security.stackexchange.com/questions/199248/why-use-env-whats-wrong-with-storing-secrets-in-a-config-php-file-outside-roo

https://movingfast.io/articles/environment-variables-considered-harmful/

https://github.com/josegonzalez/php-dotenv