***Mass Assignment Attack*** ou Ataque de Atribuição em Massa, em português, ocorre quando um usuário é capaz de inicializar ou substituir parâmetros que não deveriam ser modificados na aplicação. Ao incluir parâmetros adicionais em uma requisição, sendo tais parâmetros válidos, um usuário mal-intencionado pode gerar um efeito colateral indesejado na aplicação.

O conceito desse ataque refere-se a quando você injeta um conjunto de valores diretamente em um objeto, daí o nome atribuição em massa, que sem a devida validação pode causar sérios problemas.

Vamos a um exemplo prático. Suponha que você tem o seguinte método, em uma classe Controller, utilizado para cadastrar um usuário na aplicação:

@PostMapping

@Transactional

**public** **void** **cadastrar**(@RequestBody @Valid Usuario usuario) {

repository.save(usuario);

}COPIAR CÓDIGO

E a entidade JPA que representa o usuário:

@Getter

@Setter

@NoArgsConstructor

@EqualsAndHashCode(of = "id")

@Entity(name = "Usuario")

@Table(name = "usuarios")

**public** **class** **Usuario** {

@Id

@GeneratedValue(strategy = GenerationType.IDENTITY)

**private** Long id;

**private** String nome;

**private** String email;

**private** **Boolean** admin = **false**;

//restante do código omitido…

}COPIAR CÓDIGO

Repare que o atributo admin da classe Usuario é inicializado como false, indicando que um usuário deve sempre ser cadastrado como não sendo um administrador. Porém, se na requisição for enviado o seguinte JSON:

{

“nome” : “Rodrigo”,

“email” : “rodrigo@email.com”,

“admin” : **true**

}COPIAR CÓDIGO

O usuário será cadastrado com o atributo admin preenchido como true. Isso acontece porque o atributo admin enviado no JSON existe na classe que está sendo recebida no Controller, sendo considerado então um atributo válido e que será preenchido no objeto Usuario que será instanciado pelo Spring.

Então, como fazemos para prevenir esse problema?

**Prevenção**

O uso do padrão DTO nos ajuda a evitar esse problema, pois ao criar um DTO definimos nele apenas os campos que podem ser recebidos na API, e no exemplo anterior o DTO não teria o atributo admin.

Novamente, vemos mais uma vantagem de se utilizar o padrão DTO para representar os dados que chegam e saem da API.