

Gain Control over your Dependencies with Private Packagist



Nils Adermann
@naderman
Private Packagist
<https://packagist.com>

What is Dependency Management?

- Assembly
- Dependency Change Management
- Risk Analysis & Reduction

May happen at build time or at runtime



Dependency Assembly

- Installation of Libraries, Tools, etc.
 - composer install
 - apt-get install foo
 - Application of Configuration Management (Puppet, Chef, Ansible, Salt, ...)
- Configuration for Connections to Services, external APIs
 - Authentication
 - Glue Code
- Connection to Services (usually at Runtime)

Dependency Assembly

Past:

- Step-by-Step installation instructions
- Readmes, Delete and reinstall individual packages

Today:

- Description of a system state (e.g. composer.json, top.sls)
- Tools to move the system into the state (e.g. composer, salt)

Dependency Change Management

- Dependency Change
 - Adding, Removing, Updating, Replacing of Libraries
 - Replacing APIs
 - composer update
- Dependency Change Management
 - Balance Risks, Consequences, Cost & Advantages
 - Architecture Decisions which enable "Change"
 - Example: Abstraction to replace concrete service

A brief history of Composer

- Symfony & phpBB plugins
- Apr 2011 - First Commit
- Sep 2011 - Packagist.org
- Apr 2012 - First 1,000 Packages
- Apr 2013 - First 10,000 Packages
- Jun 2014 - Toran Proxy

July 2017: 147,000 Packages with 907,000 Versions

A brief history of Composer

- Symfony & phpBB plugins
- Apr 2011 - First Commit
- Sep 2011 - Packagist.org
- Apr 2012 - First 1,000 Packages
- Apr 2013 - First 10,000 Packages
- ~~Jun 2014 - Toran Proxy~~
- Dec 2016 - Private Packagist

Composer Design Principles

- Separate independent tools and services
 - Avoid PEAR confusion and problems
- Build reusable code to allow for other tools and services to emerge
 - Check out <https://github.com/composer>

composer update/install

- Load all package metadata
- Resolve dependencies to create transaction (install/remove/update)
- Create lock file
- Download or checkout files from locations in lock file

Satis

- Static File Generator
- Big config file of all packages
- Archive creation for downloads possible
- No hooks to trigger updates
- Not suitable for building further tools or services on top of it
- Considerably cost to setup & maintain

Private Packagist

- Your own Composer repository done right
 - SaaS or on-premises - <https://packagist.com>
- Easy setup
 - Integration with GitHub, Gitlab, Bitbucket
- Authentication
- Permission Management
- Foundation for future functionality to simplify dependency management

Load package metadata?

- Composer Repositories
 - packagist.org
 - Satis
 - Private Packagist
- VCS repositories
- Package repositories

Package Repository

```
"repositories": [  
  {  
    "type": "package",  
    "package": {  
      "name": "vendor/package",  
      "version": "1.0.0",  
      "dist": {  
        "url": "http://example.org/package.zip",  
        "type": "zip"  
      },  
      "source": {  
        "url": "git://example.org/package.git",  
        "type": "git",  
        "reference": "tag name, branch name or commit hash"  
      }  
    }  
  }  
],  
"require": {  
  "vendor/package": "1.0.0"  
}
```



```
"repositories": [  
  {  
    "type": "vcs",  
    "url": "git://example.org/MyRepo.git"  
  }  
]
```

- Information is inferred from composer.json files in tags & branches
- dist download URLs only for known hosts, e.g. github, bitbucket, gitlab

Composer Repository

```
"repositories": [  
  {  
    "type": "composer",  
    "url": "https://satis.example.org/"  
  },  
  {  
    "type": "composer",  
    "url": "https://repo.packagist.com/my-org"  
  },  
  {  
    "packagist.org": false  
  }  
]
```

Composer Repository: Satis

```
packages.json:
{
    packages: {
        "seld/private-test": {
            "dev-master": {
                name: "seld/PRivate-test",
                version: "dev-master",
                version_normalized: "9999999-dev",
                source: {
                    ....
                },
                dist: {
                    ....
                },
                require: {
                    php: ">=5.3.0",
                    ...
                }
            }
        }
    }
}
```



Composer Repository: packagist.org

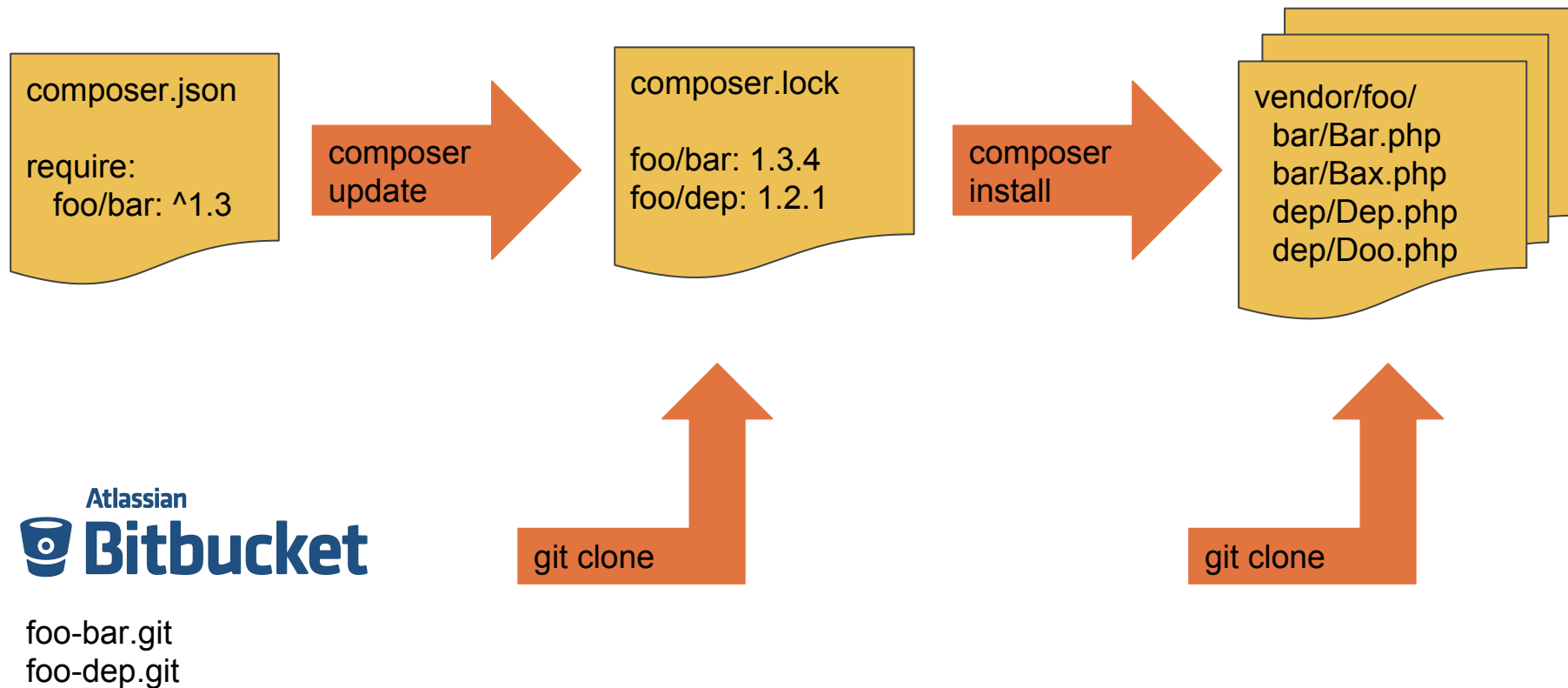
```
packages.json:
{
    packages: [ ],
    notify: "/downloads/%package%",
    notify-batch: "/downloads/",
    providers-url: "/p/%package%$%hash%.json",
    search: "/search.json?q=%query%&type=%type%",
    provider-includes: {
        p/provider-2013$%hash%.json: {
            sha256: "eb67fda529996db6fac4647ff46cf41bb31065536e1164d0e75f911d160f6b9f"
        },
        ...
        p/provider-archived$%hash%.json: {
            sha256: "444a8f22af4bc0e2ac0c09eda1f5edc63158a16e9d754100d7f774b930a38ae6"
        },
        p/provider-latest$%hash%.json: {
            sha256: "b0e0065f1e36f061b9fd2bbb096e7986321421f9eedc3d5e68dc4780d7295c33"
        }
    }
}
```



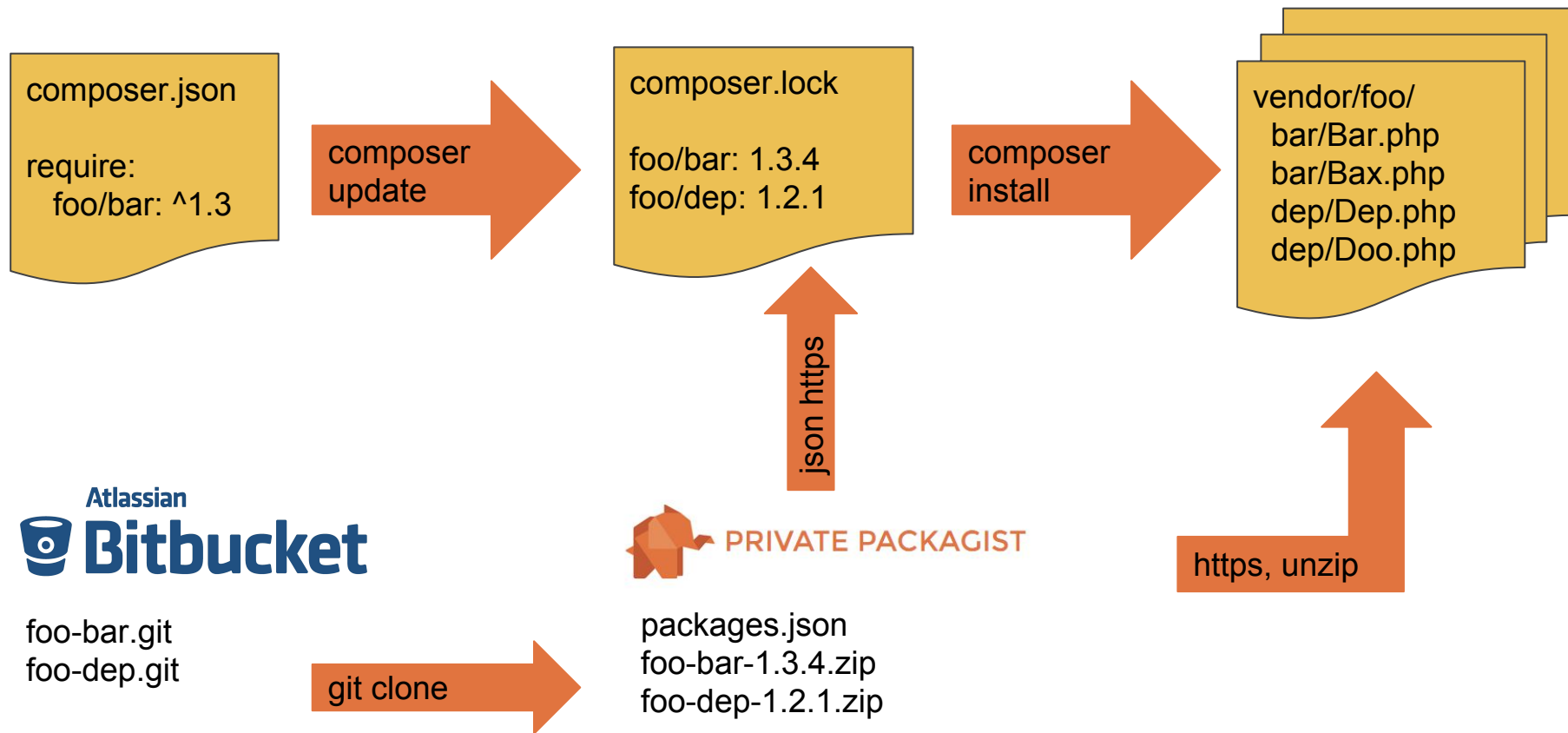
Composer Repository: Private Packagist

```
packages.json:
{
    packages: {
        "seld/private-test": {
            "dev-master": {
                name: "seld/PRivate-test",
                ...
            }
        },
        providers-lazy-url: "/myorg/p/%package%.json",
        mirrors: [
            {
                dist-url:
                "https://repo.packagist.com/packagist-nosync/dists/%package%/%version%/%reference%.%type%",
                preferred: true
            }
        ]
    }
}
```

Composer with Private Dependencies



Composer with Private Dependencies: Private Packagist



Risk Analysis: Availability

Affects Assembly

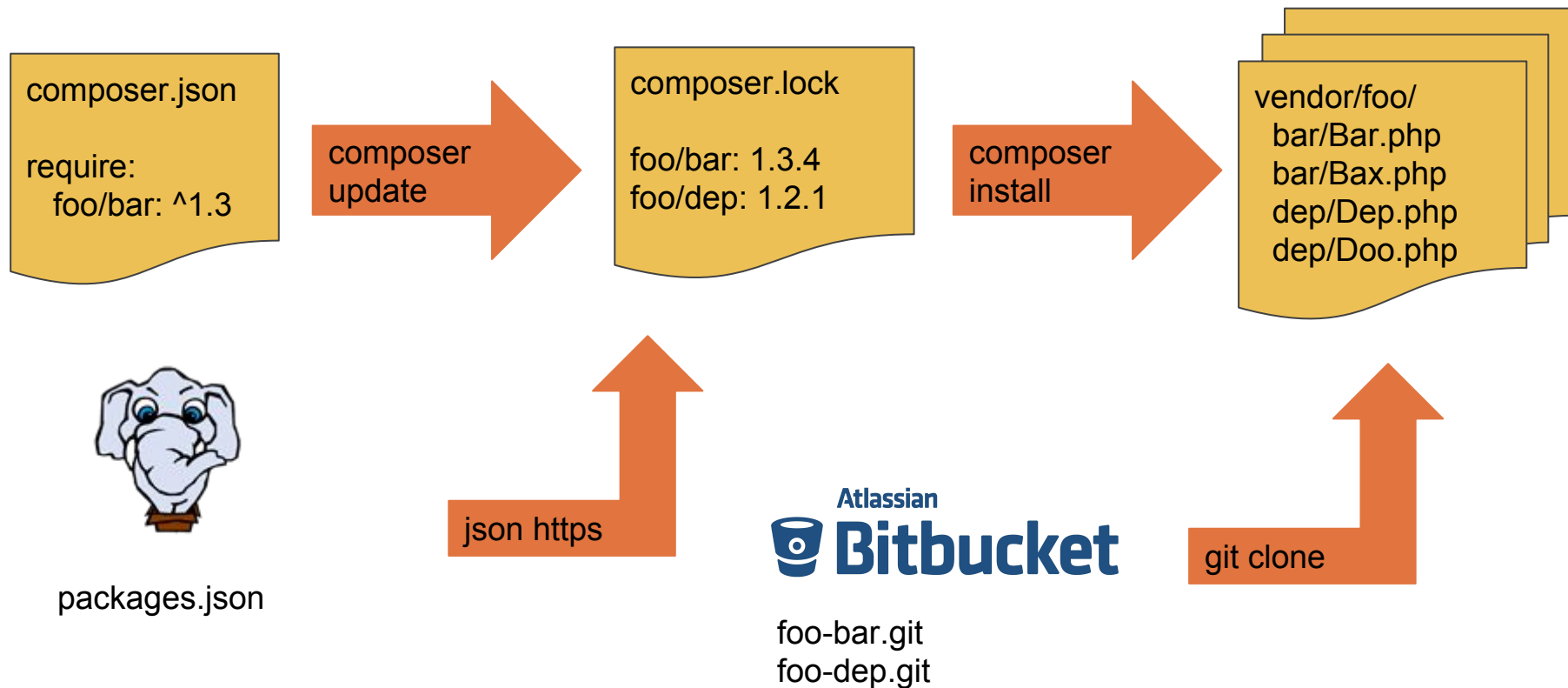
Examples:

- Open Source Library deleted
- Payment Service unavailable
- EU VATId Service out of order
- Jenkins not accessible

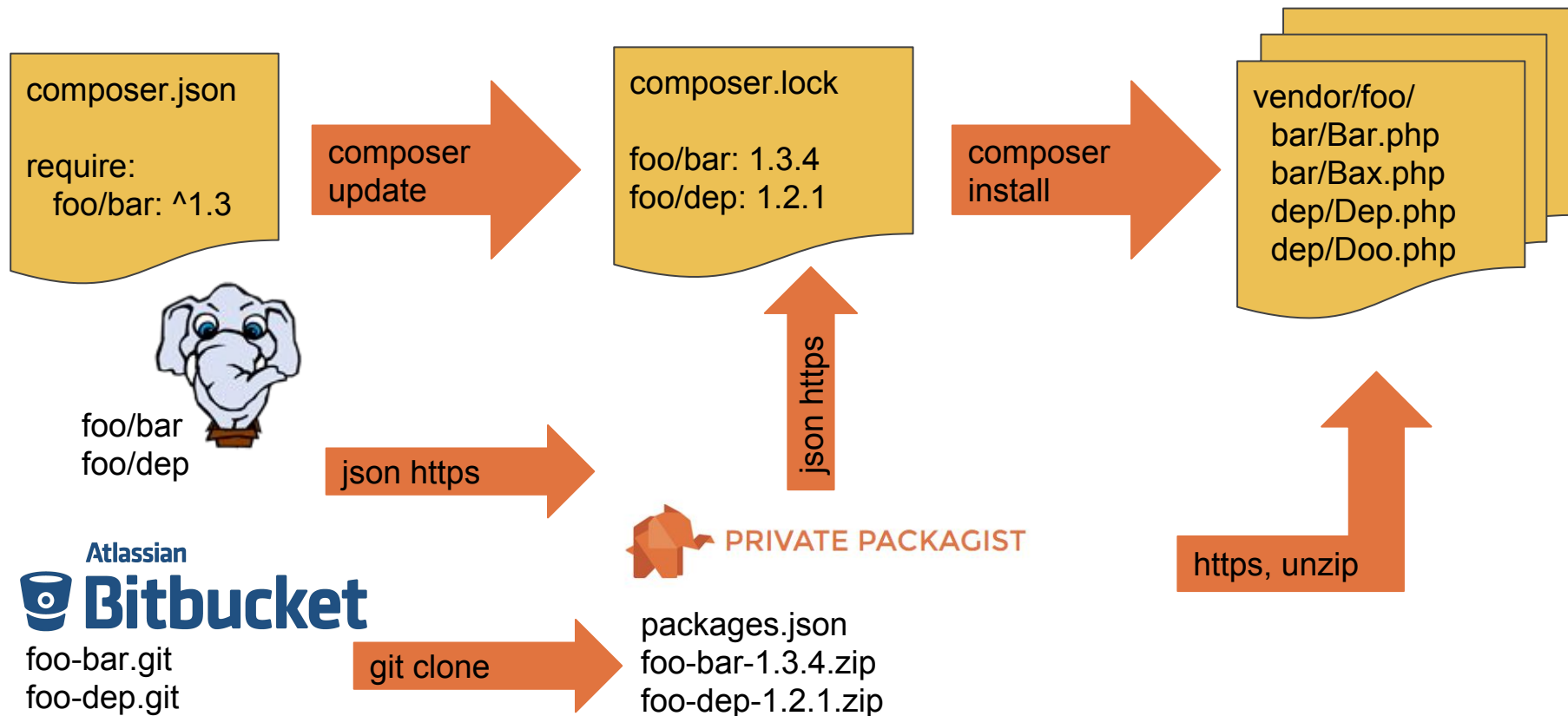
Risk Reduction: Availability

- Software is available when you have a copy
 - composer cache
 - Forks
 - Private Packagist or Satis

Composer with Open Source Dependencies



Composer with Open Source Dependencies: Private Packagist



Downloading files from the lock file

```
{
  "content-hash": "bb557b05609c879265a30bc052ef77e4",
  "packages": [
    {
      "name": "aws/aws-sdk-php",
      "version": "3.25.6",
      "source": {
        "type": "git",
        "url": "https://github.com/aws/aws-sdk-php.git",
        "reference": "fe98140a4811abbe9104477b167dc3c7f9a8391b"
      },
      "dist": {
        "type": "zip",
        "url": "https://api.github.com/repos/aws/aws-sdk-php/zipball/fe...",
        "reference": "fe98140a4811abbe9104477b167dc3c7f9a8391b",
      },
      "require": {
        "guzzlehttp/guzzle": "^5.3.1|^6.2.1",
```



Downloading files from the lock file with Private Packagist

```
"packages": [  
  {  
    "name": "aws/aws-sdk-php",  
    "version": "3.25.6",  
    "source": {  
      "url": "https://github.com/aws/aws-sdk-php.git",  
      ...  
    },  
    "dist": {  
      "type": "zip",  
      "url": "https://api.github.com/repos/aws/aws-sdk-php/zipball/...",  
      "reference": "fe98140a4811abbe9104477b167dc3c7f9a8391b",  
      "mirrors": [  
        {  
          "url":  
"https://repo.packagist.com/phpbb/dists/%package%/%version%/%reference%.%type%",  
          "preferred": true  
        }  
      ]  
    }  
  }  
]
```



Risk Reduction: (New) Dependencies

Quality Criteria for software libraries (and services)

- Number of Maintainers / Developers
- Actively Developed?
- How many users?
 - Packagist shows installation count
- Where is a library being installed from?
 - GitHub, self-hosted svn server? -> Availability
- Alternatives / how easy to replace? Complexity?
 - Could you take over maintenance?



Risk Reduction: Compatibility

Semantic Versioning (Semver) promises Compatibility

x.y.z

- Must be used consistently
- Only valuable if BC/Compatibility promise formalized
 - See <http://symfony.com/doc/current/contributing/code/bc.html>
- Otherwise choose narrower Version Constraints, check more frequently
 - e.g. ~1.2.3 instead of ^1.2.3



Risk Reduction: Compatibility

- Automated
 - Tests
 - Static Analysis
- Manual
 - Read Changelogs (and write them!)
 - Experience which libraries break BC

Risk Minimization: Compliance / Legal

- Affects Change Management
- Example
 - Viral Copy-Left License not compatible with proprietary product
- `composer licenses`
- Private Packagist License Review

Assessing & Managing Risk

- Formulate a Plan B
- Identify problems which are probable and which have great effects
- **Dependencies are great!** They can save tons of money and time
- Only spend resources on reducing risk until the risk is acceptable
- Private Packagist can help you manage and reduce these risks by being the one central place for all your third party code

How is Private Packagist helping?

- Faster and more reliable composer operations
 - Work with private dependencies more efficiently
 - Automatic synchronization of packages, teams, users, permissions
 - Authentication Tokens
 - One central place for all your dependencies
- Improved understanding of and control over open-source usage
- Statistics and references between internal code and open-source code
 - License review
 - Much more to come!



Thank you!

<https://packagist.com>

10% off first 12 months with code t3dd17

Questions / Feedback?

E-Mail: n.adermann@packagist.com

Twitter: [@naderman](https://twitter.com/naderman)