

# Practical Use Cases for the New Nested Low-Level Discovery Feature in Zabbix

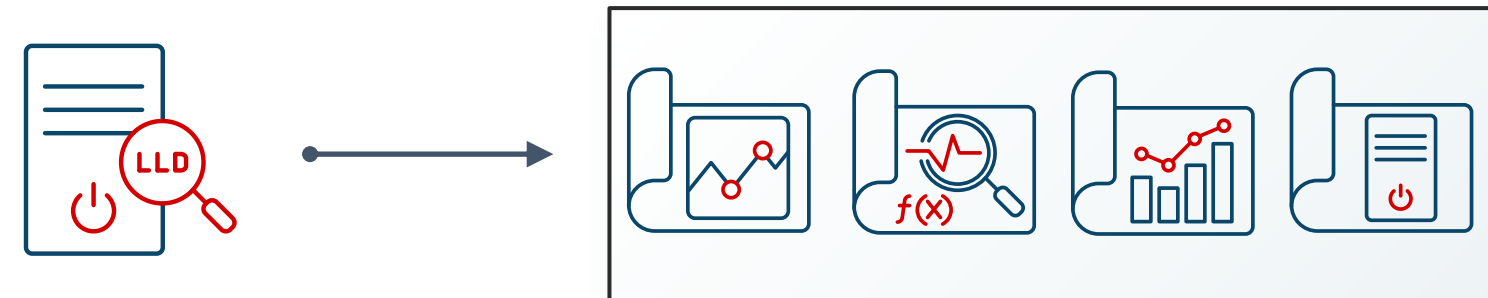
Stefan Matzek  
Zabbix Trainer & Consultant  
IntelliTrend GmbH

# Low-Level Discovery

LLD erkennt automatisch wiederkehrende Komponenten eines Geräts und erstellt daraus passende Entitäten aus Prototypen.

## Kernfakten: Klassische LLD

- Seit Zabbix 2.0 verfügbar
- Regel liest JSON/Key-Values und füllt `{#LLD}`-Makros
- Erstellt automatisch Items, Triggers, Graphen und Hosts durch Prototypen
- Läuft periodisch und hält Objekte synchron
- Aufräumen über "Keep lost resources"



# Was ist NLLD?

Nested Low-Level Discovery (NLLD) erzeugt keine Items direkt, sondern generiert weitere Discovery-Regeln.

## Native Verschachtelung:

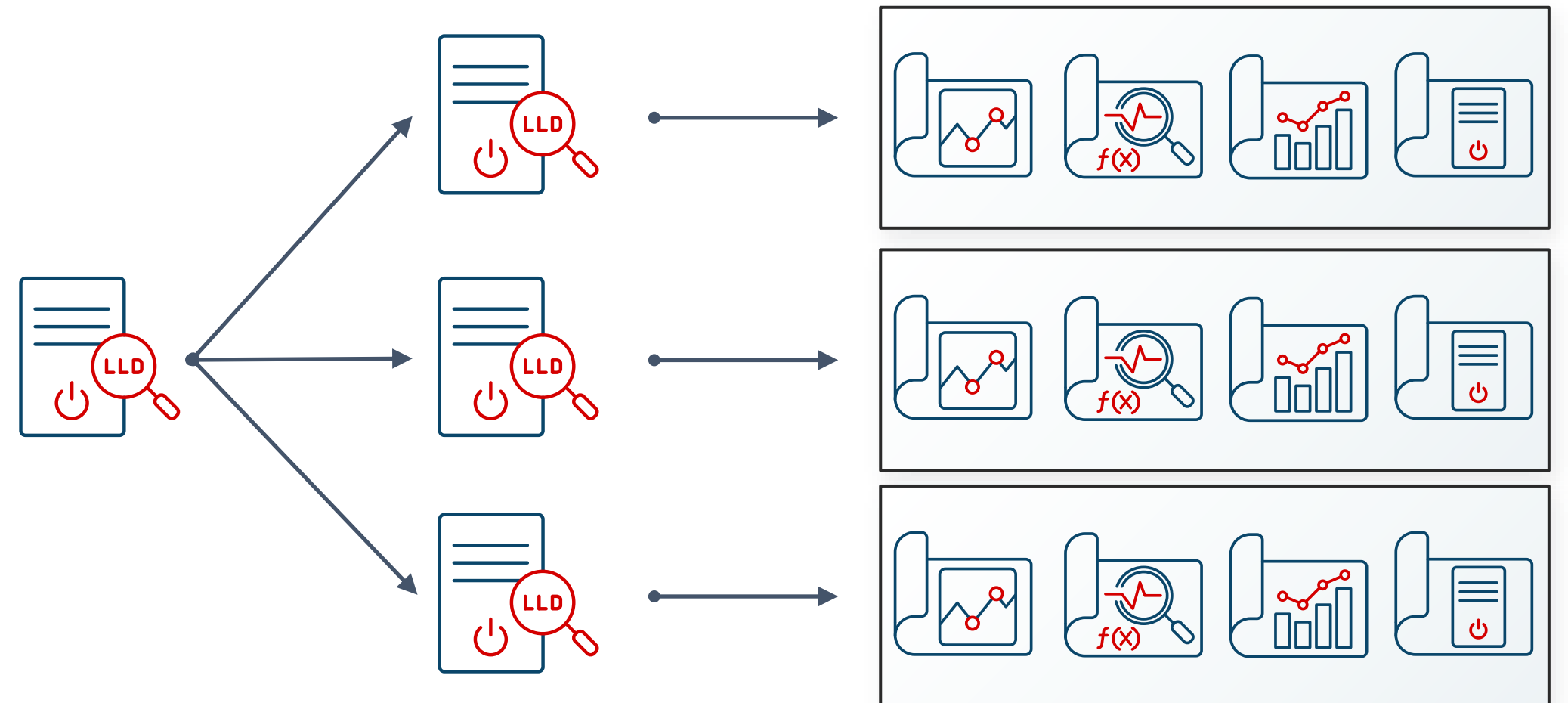
Selbst mehrstufige Hierarchien lassen sich direkt in Zabbix abbilden – ganz ohne externe Skripte oder Workarounds.

## Kontext-Vererbung:

Makros der übergeordneten Discovery werden automatisch an alle Sub-Discovery-Regeln durchgereicht.

## Klare Struktur:

Aufeinander aufbauende LLDs halten die Konfiguration übersichtlich.



# Standard-LLD vs. Nested-LLD

## Standard-LLD

- **Flach & Starr:** Nur eine Discovery-Ebene pro Regel möglich.
- **Daten-Overhead:** Oft gigantische, redundante JSON-Payloads nötig.
- **Isoliert:** Makros gelten nur in ihrer eigenen kleinen Bubble.
- **Flaschenhals:** Flache Kombinationslisten belasten Server-CPU und Datenbank.

## Nested-LLD

- **Dynamisch & Tief:** Child-Regeln entstehen on-the-fly aus Parent-Funden.
- **Präzise:** Gezielte Filterung auf Teilbäumen (JSONPath) statt Massenabruf.
- **Vernetzt:** Parent-Makros stehen in Child-Regeln voll zur Verfügung.
- **Ressourcenschonend:** Weniger Netzwerk-Abrufe durch gemeinsame JSON-Bäume.

# Der „Nested“ Type: Datenübergabe verstehen

## Der Master liefert die Daten

Eine NLLD-Kette beginnt fast nie mit dem Typ „Nested“. Die erste Regel ist meist ein HTTP-Agent oder ein anderes Item, das das große JSON-Paket holt.

## Der Typ „Nested“ als „Konsument“

Nur die untergeordneten Discovery-Regeln (Child-Rules) könnten den neuen Typ „Nested“ erhalten.

## LLD-Prototypen müssen nicht Type „Nested“ haben

Bei „Chatty APIs“ benötigt jede Ebene eigene Requests. Hier nutzt der Prototyp z. B. den Typ „HTTP-Agent“, erbt aber trotzdem alle Makros der Parent-Regel.

The screenshot shows the Zabbix configuration interface for a Discovery rule. The 'Type' dropdown menu is open, displaying a list of available LLD types. The 'Nested' type is highlighted with a red box. Other visible types include Simple check, SNMP agent, Zabbix internal, Zabbix trapper, External check, Database monitor, HTTP agent, IPMI agent, SSH agent, TELNET agent, JMX agent, Dependent item, Script, and Browser. The interface also shows fields for Name, Key, Delete lost resources (After 7d), and Disable lost resources (After).

# Nested Low-Level Discovery

Beispiel - Chunky API

## NLLD-Beispiel - Chunky API

# Gebäude Discovery

\* Name

Type

\* Key

\* Master item

Preprocessing steps	Name	Parameters
1:	JSONPath	<input type="text" value="\$..data"/>
2:	Discard unchanged with heartbeat	<input type="text" value="1h"/>

LLD macros	LLD macro	JSONPath	
	{#ADDRESS}	<input type="text" value="\$..address"/>	<input type="button" value="Remove"/>
	{#BUILDING_ID}	<input type="text" value="\$..id"/>	<input type="button" value="Remove"/>
	{#DESCRIPTION}	<input type="text" value="\$..description"/>	<input type="button" value="Remove"/>
	{#NAME}	<input type="text" value="\$..name"/>	<input type="button" value="Remove"/>

```
1 {
2   "success": true,
3   "data": [
4     {
5       "id": 10,
6       "name": "HQ Building",
7       "description": "Main headquarters with all departments",
8       "address": "Tech Park 1, Berlin",
9       "rooms": [
10        { },
11        { },
12        { },
13        { },
14        { },
15        { },
16        { },
17        { },
18        { },
19        { },
20        { },
21        { },
22        { },
23        { },
24        { },
25        { },
26        { },
27        { },
28        { },
29        { },
30        { },
31        { },
32        { },
33        { },
34        { },
35        { },
36        { },
37        { },
38        { },
39        { },
40        { },
41        { },
42        { },
43        { },
44        { },
45        { },
46        { },
47        { },
48        { },
49        { },
50        { },
51        { },
52        { },
53        { },
54        { },
55        { },
56        { },
57        { },
58        { },
59        { },
60        { },
61        { },
62        { },
63        { },
64        { },
65        { },
66        { },
67        { },
68        { },
69        { },
70        { },
71        { },
72        { },
73        { },
74        { },
75        { },
76        { },
77        { },
78        { },
79        { },
80        { },
81        { },
82        { },
83        { },
84        { },
85        { },
86        { },
87        { },
88        { },
89        { },
90        { },
91        { },
92        { },
93        { },
94        { },
95        { },
96        { },
97        { },
98        { },
99        { },
100       { },
101       { },
102       { },
103       { },
104       { },
105       { },
106       { },
107       { },
108       { },
109       { },
110       { },
111       { },
112       { },
113       { },
114       { },
115       { },
116       { },
117       { },
118       { },
119       { },
120       { },
121       { },
122       { },
123       { },
124       { },
125       { },
126       { },
127       { },
128       { },
129       { },
130       { },
131       { },
132       { },
133       { },
134       { },
135       { },
136       { },
137       { },
138       { },
139       { },
140       { },
141       { },
142       { },
143       { },
144       { },
145       { },
146       { },
147       { },
148       { },
149       { },
150       { },
151       { },
152       { },
153       { },
154       { },
155       { },
156       { },
157       { },
158       { },
159       { },
160       { },
161       { },
162       { },
163       { },
164       { },
165       { },
166       { },
167       { },
168       { },
169       { },
170       { },
171       { },
172       { },
173       { },
174       { },
175       { },
176       { },
177       { },
178       { },
179       { },
180       { },
181       { },
182       { },
183       { },
184       { },
185       { },
186       { },
187       { },
188       { },
189       { },
190       { },
191       { },
192       { },
193       { },
194       { },
195       { },
196       { },
197       { },
198       { },
199       { },
200       { },
201       { },
202       { },
203       { },
204       { },
205       { },
206       { },
207       { },
208       { },
209       { },
210       { },
211       { },
212       { },
213       { },
214       { },
215       { },
216       { },
217       { },
218       { },
219       { },
220       { },
221       { },
222       { },
223       { },
224       { },
225       { },
226       { },
227       { },
228       { },
229       { },
230       { },
231       { },
232       { },
233       { },
234       { },
235       { },
236       { },
237       { },
238       { },
239       { },
240       { },
241       { },
242       { },
243       { },
244       { },
245       { },
246       { },
247       { },
248       { },
249       { },
250       { },
251       { },
252       { },
253       { },
254       { },
255       { },
256       { },
257       { },
258       { },
259       { },
260       { },
261       { },
262       { },
263       { },
264       { },
265       { },
266       { },
267       { },
268       { },
269       { },
270       { },
271       { },
272       { },
273       { },
274       { },
275       { },
276       { },
277       { },
278       { },
279       { },
280       { },
281       { },
282       { },
283       { },
284       { },
285       { },
286       { },
287       { },
288       { },
289       { },
290       { },
291       { },
292       { },
293       { },
294       { },
295       { },
296       { },
297       { },
298       { },
299       { },
300       { },
301       { },
302       { },
303       { },
304       { },
305       { },
306       { },
307       { },
308       { },
309       { },
310       { },
311       { },
312       { },
313       { },
314       { },
315       { },
316       { },
317       { },
318       { },
319       { },
320       { },
321       { },
322       { },
323       { },
324       { },
325       { },
326       { },
327       { },
328       { },
329       { },
330       { },
331       { },
332       { },
333       { },
334       { },
335       { },
336       { },
337       { },
338       { },
339       { },
340       { },
341       { },
342       { },
343       { },
344       { },
345       { },
346       { },
347       { },
348       { },
349       { },
350       { },
351       { },
352       { },
353       { },
354       { },
355       { },
356       { },
357       { },
358       { },
359       { },
360       { },
361       { },
362       { },
363       { },
364       { },
365       { },
366       { },
367       { },
368       { },
369       { },
370       { },
371       { },
372       { },
373       { },
374       { },
375       { },
376       { },
377       { },
378       { },
379       { },
380       { },
381       { },
382       { },
383       { },
384       { },
385       { },
386       { },
387       { },
388       { },
389       { },
390       { },
391       { },
392       { },
393       { },
394       { },
395       { },
396       { },
397       { },
398       { },
399       { },
400       { },
401       { },
402       { },
403       { },
404       { },
405       { },
406       { },
407       { },
408       { },
409       { },
410       { },
411       { },
412       { },
413       { },
414       { },
415       { },
416       { },
417       { },
418       { },
419       { },
420       { },
421       { },
422       { },
423       { },
424       { },
425       { },
426       { },
427       { },
428       { },
429       { },
430       { },
431       { },
432       { },
433       { },
434       { },
435       { },
436       { },
437       { },
438       { },
439       { },
440       { },
441       { },
442       { },
443       { },
444       { },
445       { },
446       { },
447       { },
448       { },
449       { },
450       { },
451       { },
452       { },
453       { },
454       { },
455       { },
456       { },
457       { },
458       { },
459       { },
460       { },
461       { },
462       { },
463       { },
464       { },
465       { },
466       { },
467       { },
468       { },
469       { },
470       { },
471       { },
472       { },
473       { },
474       { },
475       { },
476       { },
477       { },
478       { },
479       { },
480       { },
481       { },
482       { },
483       { },
484       { },
485       { },
486       { },
487       { },
488       { },
489       { },
490       { },
491       { },
492       { },
493       { },
494       { },
495       { },
496       { },
497       { },
498       { },
499       { },
500       { },
501       { },
502       { },
503       { },
504       { },
505       { },
506       { },
507       { },
508       { },
509       { },
510       { },
511       { },
512       { },
513       { },
514       { },
515       { },
516       { },
517       { },
518       { },
519       { },
520       { },
521       { },
522       { },
523       { },
524       { },
525       { },
526       { },
527       { },
528       { },
529       { },
530       { },
531       { },
532       { },
533       { },
534       { }
}
```

In einem Master-Item wird ein mehrschichtiges JSON gesammelt. Diese Ebene unterscheidet sich nicht von regulären LLD.

# NLLD-Beispiel - Chunky API

## Raum Discovery

All templates / Building Sensors API by HTTP / Discovery list / Building discovery

Item prototypes 2 / Trigger prototypes / Graph prototypes / Host prototypes / Discovery prototypes 1

Name	Items	Triggers	Graphs	Hosts	Discovery rules	Key	Interval	Type	Create enabled	Discover
<input type="checkbox"/> Room discovery {#NAME}	Item prototypes 2	Trigger prototypes	Graph prototypes	Host prototypes	Discovery prototypes 2	room.discovery[building,{#BUILDING_ID}]		Nested	Yes	Yes

Displaying 1 of 1 found

\* Name

Type

\* Key

Preprocessing steps	Name	Parameters
1:	JSONPath	<input type="text" value="\$..rooms"/>
2:	Discard unchanged with heartbeat	<input type="text" value="1h"/>

LLD macros	LLD macro	JSONPath	
	{#ROOM_ID}	<input type="text" value="\$..id"/>	<a href="#">Remove</a>
	{#ROOM_NAME}	<input type="text" value="\$..name"/>	<a href="#">Remove</a>
	{#ROOM_TYPE}	<input type="text" value="\$..type"/>	<a href="#">Remove</a>

```
366 {
367   "id": 11,
368   "name": "Data Center",
369   "description": "Secondary data center facility",
370   "address": "Industrial Area 5, Berlin",
371   "rooms": [
372     {
373       "id": 111,
374       "building_id": 11,
375       "name": "Server Hall A",
376       "type": "serverroom",
377       "floor": 0,
378       "capacity": 6,
379       "sensors": [
414     ],
415     {
416       "id": 112,
417       "building_id": 11,
418       "name": "Server Hall B",
419       "type": "serverroom",
420       "floor": 0,
421       "capacity": 6,
422       "sensors": [
457     ],
458     {
459       "id": 113,
460       "building_id": 11,
461       "name": "Network Room",
462       "type": "serverroom",
463       "floor": 0,
464       "capacity": 3,
465       "sensors": [
483     ],
484     {
485       "id": 114,
486       "building_id": 11,
487       "name": "UPS Room",
488       "type": "storage",
489       "floor": 0,
490       "capacity": 2,
491       "sensors": [
526     ]
527   ]
528 }
```

Das nächste Level erfordert eine Nested Low-Level Discovery um alle Räume der Gebäude zu identifizieren.

## NLLD-Beispiel - Chunky API

# Sensor Discovery

All templates / Building Sensors API by HTTP / Discovery list / \*\*\* / Room discovery {#NAME}

Item prototypes 2	Trigger prototypes	Graph prototypes	Host prototypes	Discovery prototypes 2							
<input type="checkbox"/>	Name ▲	Items	Triggers	Graphs	Hosts	Discovery rules	Key	Interval	Type	Create enabled	Discover
<input type="checkbox"/>	Environment Sensor discovery {#ROOM_NAME}	Item prototypes 3	Trigger prototypes	Graph prototypes	Host prototypes	Discovery prototypes	sensor.env.discovery[{#BUILDING_ID}, {#ROOM_ID}]		Nested	Yes	Yes
<input type="checkbox"/>	Power Sensor discovery {#ROOM_NAME}	Item prototypes 4	Trigger prototypes	Graph prototypes	Host prototypes	Discovery prototypes	sensor.power.discovery[{#BUILDING_ID}, {#ROOM_ID}]		Nested	Yes	Yes

\* Name

Type

\* Key

Preprocessing steps	Name	Parameters
1:	JSONPath	<input type="text" value="\$ .sensors"/>
2:	Discard unchanged with heartbeat	<input type="text" value="1h"/>

LLD macros	LLD macro	JSONPath	
	{#SENSOR_ID}	<input type="text" value="\$ .id"/>	<a href="#">Remove</a>
	{#SENSOR_NAME}	<input type="text" value="\$ .name"/>	<a href="#">Remove</a>
	{#SENSOR_TYPE}	<input type="text" value="\$ .type"/>	<a href="#">Remove</a>

Filters	Label	Macro	Regular expression
A		<input type="text" value="{#SENSOR_TYPE}"/>	<input type="text" value="matches ^power\$"/>

```
372 {
373   "id": 111,
374   "building_id": 11,
375   "name": "Server Hall A",
376   "type": "serverroom",
377   "floor": 0,
378   "capacity": 6,
379   "sensors": [
380     {
381       "id": 1015,
382       "room_id": 111,
383       "name": "Environment Sensor - Rack A1",
384       "type": "environment",
385       "description": "Environmental monitoring for rack A1",
386       "readings": [
387         {
388           "sensor_id": 1015,
389           "temperature": 18.5,
390           "humidity": 45.6,
391           "co2": 666,
392           "timestamp": 1777243261
393         }
394       ]
395     },
396     {
397       "id": 1016,
398       "room_id": 111,
399       "name": "Smart Plug - PDU A",
400       "type": "power",
401       "description": "PDU power monitoring A",
402       "readings": [
403         {
404           "sensor_id": 1016,
405           "voltage": 230,
406           "current": 0.105,
407           "power": 24,
408           "energy": 493752.28380446456,
409           "timestamp": 1777243261
410         }
411       ]
412     }
413   ]
414 },
```

Das nächste Level erfordert eine Nested Low-Level Discovery um alle Sensoren der Räume zu identifizieren.

## NLLD-Beispiel - Chunky API

# Metrik Extraktion

Name	Key	Interval	History	Trends	Type	Create enabled	Discover	Tags
*** get Data: {#SENSOR_NAME}: Current	sensor.current[{#BUILDING_ID},{#ROOM_ID},{#SENSOR_ID}]	31d	365d	Dependent item	Yes	Yes	building: {#NAME} room: {#ROOM_NAME} sensor: {#SENSOR_NAME}	
*** get Data: {#SENSOR_NAME}: Energy	sensor.energy[{#BUILDING_ID},{#ROOM_ID},{#SENSOR_ID}]	31d	365d	Dependent item	Yes	Yes	building: {#NAME} room: {#ROOM_NAME} sensor: {#SENSOR_NAME}	
*** get Data: {#SENSOR_NAME}: Power	sensor.power[{#BUILDING_ID},{#ROOM_ID},{#SENSOR_ID}]	31d	365d	Dependent item	Yes	Yes	building: {#NAME} room: {#ROOM_NAME} sensor: {#SENSOR_NAME}	
*** get Data: {#SENSOR_NAME}: Voltage	sensor.voltage[{#BUILDING_ID},{#ROOM_ID},{#SENSOR_ID}]	31d	365d	Dependent item	Yes	Yes	building: {#NAME} room: {#ROOM_NAME} sensor: {#SENSOR_NAME}	

\* Name {#SENSOR\_NAME}: Current  
Type Dependent item  
\* Key sensor.current[{#BUILDING\_ID},{#ROOM\_ID},{#SENSOR\_ID}]

```
$.data[?(@.id =~ '^#{#BUILDING_ID}$')].rooms[?(@.id =~ '^#{#ROOM_ID}$')].sensors[?(@.id =~ '^#{#SENSOR_ID}$')].readings[0].current.first()  
$..sensors[?(@.id == '#{#SENSOR_ID}$')].readings[0].current.first()
```

Name	Last check	Last value	Change	Tags
Smart Plug - PDU A: Current	31s	0.12 A	+0.027 A	building: Data Center room: Server Hall A sensor: Smart Plug - PDU A

Für das finale Level ist eine komplexe JSONPath-Logik erforderlich, wenn keine eindeutigen IDs pro Element vorhanden sind.

```
372 {  
373   "id": 111,  
374   "building_id": 11,  
375   "name": "Server Hall A",  
376   "type": "serverroom",  
377   "floor": 0,  
378   "capacity": 6,  
379   "sensors": [  
380     {  
381       "id": 1015,  
382       "room_id": 111,  
383       "name": "Environment Sensor - Rack A1",  
384       "type": "environment",  
385       "description": "Environmental monitoring for rack A1",  
386       "readings": [  
387         {  
388           "sensor_id": 1015,  
389           "temperature": 18.5,  
390           "humidity": 45.6,  
391           "co2": 666,  
392           "timestamp": 1777243261  
393         }  
394       ]  
395     },  
396     {  
397       "id": 1016,  
398       "room_id": 111,  
399       "name": "Smart Plug - PDU A",  
400       "type": "power",  
401       "description": "PDU power monitoring A",  
402       "readings": [  
403         {  
404           "sensor_id": 1016,  
405           "voltage": 230,  
406           "current": 0.105,  
407           "power": 24,  
408           "energy": 493752.28380446456,  
409           "timestamp": 1777243261  
410         }  
411       ]  
412     }  
413   ]  
414 }
```

# Nested Low-Level Discovery

Beispiel - Chatty API

## NLLD-Beispiel - Chatty API

# Gebäude Discovery

\* Name

Type

\* Key

\* URL

Preprocessing steps	Name	Parameters
1:	JSONPath	\$.data
2:	Discard unchanged with heartbeat	1h

LLD macros	LLD macro	JSONPath
	{#BUILDING_ID}	\$.id
	{#BUILDING_NAME}	\$.name

```
1 {
2   "success": true,
3   "data": [
4     {
5       "id": 10,
6       "name": "HQ Building",
7       "description": "Main headquarters with all departments",
8       "address": "Tech Park 1, Berlin"
9     },
10    {
11      "id": 11,
12      "name": "Data Center",
13      "description": "Secondary data center facility",
14      "address": "Industrial Area 5, Berlin"
15    }
16  ],
17  "meta": {
18    "total": 2,
19    "timestamp": 1777263365
20  }
21 }
```

In Chatty APIs müssen alle einzelnen Ebenen individuell abgefragt werden (software-abhängig)

## NLLD-Beispiel - Chatty API

# Raum Discovery

Name	Items	Triggers	Graphs	Hosts	Discovery rules	Key	Interval	Type	Create	Enabled	Discover
Building {#BUILDING_NAME}: Room Discovery	Item prototypes 3	Trigger prototypes	Graph prototypes	Host prototypes	Discovery prototypes 2	hard.api.discovery.rooms[{#BUILDING_ID}]	5m	HTTP agent	Yes	Yes	Yes

\* Name: Building {#BUILDING\_NAME}: Room Discovery

Type: HTTP agent

\* Key: hard.api.discovery.rooms[{#BUILDING\_ID}]

\* URL: {\$API.BASE.URL}/api/buildings/{#BUILDING\_ID}/rooms

Preprocessing steps	Name	Parameters
1:	JSONPath	\$.data
2:	Discard unchanged with heartbeat	1h

LLD macros	LLD macro	JSONPath	
	{#ROOM_ID}	\$.id	Remove
	{#ROOM_NAME}	\$.name	Remove
	{#ROOM_TYPE}	\$.type	Remove

```
1 {
2   "success": true,
3   "data": [
4     {
5       "id": 111,
6       "building_id": 11,
7       "name": "Server Hall A",
8       "type": "serverroom",
9       "floor": 0,
10      "capacity": 6
11    },
12    {
13      "id": 112,
14      "building_id": 11,
15      "name": "Server Hall B",
16      "type": "serverroom",
17      "floor": 0,
18      "capacity": 6
19    },
20    {
21      "id": 113,
22      "building_id": 11,
23      "name": "Network Room",
24      "type": "serverroom",
25      "floor": 0,
26      "capacity": 3
27    },
28    {
29      "id": 114,
30      "building_id": 11,
31      "name": "UPS Room",
32      "type": "storage",
33      "floor": 0,
34      "capacity": 2
35    }
36  ],
37  "meta": {
38    "total": 4,
39    "page": 1,
40    "per_page": 4,
41    "timestamp": 1777264955
42  }
43 }
```

Nested Low-Level Discovery-Regeln können auch andere Typen als "Nested" haben. Die Raum Discovery bekommt Informationen von der Gebäude-Discovery, um tiefere Ebenen zu entdecken.

## NLLD-Beispiel - Chatty API

# Sensor Discovery

All templates / Building Sensors Chatty API by HT... Discovery list / \*\*\* / Building {#BUILDING\_NAME}: Roo...

Item prototypes 3 Trigger prototypes Graph prototypes Host prototypes Discovery prototypes 2

Name	Items	Triggers	Graphs	Hosts	Discovery rules	Key	Interval	Type	Create enabled	Discover
<input type="checkbox"/> Building {#BUILDING_NAME}: Room {#ROOM_NAME} Env Sensor Discovery	Item prototypes 4	Trigger prototypes	Graph prototypes	Host prototypes	Discovery prototypes	hard.api.discovery.sensors.env[{#BUILDING_ID},{#ROOM_ID}]	5m	HTTP agent	Yes	Yes
<input type="checkbox"/> Building {#BUILDING_NAME}: Room {#ROOM_NAME} Power Sensor Discovery	Item prototypes 5	Trigger prototypes	Graph prototypes	Host prototypes	Discovery prototypes	hard.api.discovery.sensors.power[{#BUILDING_ID},{#ROOM_ID}]	5m	HTTP agent	Yes	Yes

\* Name

Type

\* Key

\* URL

Parameters

- 1: JSONPath
- 2: Discard unchanged with heartbeat

LLD macros

LLD macro	JSONPath	
{#SENSOR_ID}	\$.id	<a href="#">Remove</a>
{#SENSOR_NAME}	\$.name	<a href="#">Remove</a>
{#SENSOR_TYPE}	\$.type	<a href="#">Remove</a>

Filters

Label	Macro	Regular expression
A	{#SENSOR_TYPE}	matches <input type="text" value="^power\$"/>

```
1 {
2   "success": true,
3   "data": [
4     {
5       "id": 1020,
6       "room_id": 114,
7       "name": "Environment Sensor - UPS",
8       "type": "environment",
9       "description": "Environmental monitoring near UPS units"
10    },
11    {
12      "id": 1021,
13      "room_id": 114,
14      "name": "Smart Plug - UPS System",
15      "type": "power",
16      "description": "Total power consumption of UPS"
17    }
18  ],
19  "meta": {
20    "total": 2,
21    "timestamp": 1777265477
22  }
23 }
```

Die Sensor Discovery bekommt Informationen von der Raum- und Gebäude-Discovery, um tiefere Ebenen zu entdecken.

## NLLD-Beispiel - Chatty API

# Metrik Extraktion

Name	Interval	History	Trends	Type	Create enabled	Discover	Tags
*** <a href="#">{#SENSOR_NAME}: Raw Readings: {#SENSOR_NAME}: Current</a>	hard.sensor.current[{#BUILDING_ID},{#ROOM_ID},{#SENSOR_ID}]	31d	365d	Dependent item	Yes	Yes	building: {#BUILDING_NAME} room: {#ROOM_NAME} sensor: {#SENSOR_NAME}
*** <a href="#">{#SENSOR_NAME}: Raw Readings: {#SENSOR_NAME}: Energy</a>	hard.sensor.energy[{#BUILDING_ID},{#ROOM_ID},{#SENSOR_ID}]	31d	365d	Dependent item	Yes	Yes	building: {#BUILDING_NAME} room: {#ROOM_NAME} sensor: {#SENSOR_NAME}
*** <a href="#">{#SENSOR_NAME}: Raw Readings: {#SENSOR_NAME}: Power</a>	hard.sensor.power[{#BUILDING_ID},{#ROOM_ID},{#SENSOR_ID}]	31d	365d	Dependent item	Yes	Yes	building: {#BUILDING_NAME} room: {#ROOM_NAME} sensor: {#SENSOR_NAME}
*** <a href="#">{#SENSOR_NAME}: Raw Readings</a>	hard.sensor.raw.power[{#BUILDING_ID},{#ROOM_ID},{#SENSOR_ID}]	1m	31d	HTTP agent	Yes	Yes	component: raw-data
*** <a href="#">{#SENSOR_NAME}: Raw Readings: {#SENSOR_NAME}: Voltage</a>	hard.sensor.voltage[{#BUILDING_ID},{#ROOM_ID},{#SENSOR_ID}]	31d	365d	Dependent item	Yes	Yes	building: {#BUILDING_NAME} room: {#ROOM_NAME} sensor: {#SENSOR_NAME}

\* Name

Type

\* Key

Type of information

\* URL

Preprocessing steps ?

Name	Parameters
1: JSONPath	<input type="text" value="\$..data.current"/>

```
1 {
2   "success": true,
3   "data": {
4     "sensor_id": 1016,
5     "voltage": 230,
6     "current": 0.183,
7     "power": 42,
8     "energy": 493745.54764686694,
9     "timestamp": 1777266299
10  },
11  "meta": {
12    "timestamp": 1777266299
13  }
14 }
```

Finales Einsammeln der Metriken mittels HTTP-Agent und Dependent Items Prototypen auf einfachem Level.

Host	Name	Last check	Last value	Change	Tags
nlld_multiple_layer	Smart Plug - PDU A: Current	43s	0.166 A	-0.008 A	building: Data Center room: Server Hall A sensor: Smart Plug - PDU A

# Practical Use Cases

Redfish by HTTP

# Aktuelle Redfish Templates

<input type="checkbox"/> Name ▲	Hosts	Items	Triggers	Graphs	Dashboards	Discovery	Web	Vendor	Version	Linked templates	Linked to templates	Tags
<input type="checkbox"/> DELL PowerEdge R660 by HTTP	Hosts	Items 14	Triggers 13	Graphs 4	Dashboards 1	Discovery 8	Web	Zabbix	7.4-2			class: hardware target: dell target: dell poweredge
<input type="checkbox"/> DELL PowerEdge R720 by HTTP	Hosts	Items 14	Triggers 13	Graphs 4	Dashboards 1	Discovery 8	Web	Zabbix	7.4-3			class: hardware target: dell target: dell poweredge
<input type="checkbox"/> DELL PowerEdge R740 by HTTP	Hosts	Items 14	Triggers 13	Graphs 4	Dashboards 1	Discovery 8	Web	Zabbix	7.4-3			class: hardware target: dell target: dell poweredge
<input type="checkbox"/> DELL PowerEdge R750 by HTTP	Hosts	Items 14	Triggers 13	Graphs 4	Dashboards 1	Discovery 8	Web	Zabbix	7.4-2			class: hardware target: dell target: dell poweredge
<input type="checkbox"/> DELL PowerEdge R820 by HTTP	Hosts	Items 14	Triggers 13	Graphs 4	Dashboards 1	Discovery 8	Web	Zabbix	7.4-3			class: hardware target: dell target: dell poweredge
<input type="checkbox"/> DELL PowerEdge R840 by HTTP	Hosts	Items 14	Triggers 13	Graphs 4	Dashboards 1	Discovery 8	Web	Zabbix	7.4-3			class: hardware target: dell target: dell poweredge
<input type="checkbox"/> HPE iLO by HTTP	Hosts	Items 2	Triggers 1	Graphs	Dashboards 1	Discovery 9	Web	Zabbix	7.4-0			class: hardware target: hpe target: ilo

Displaying 7 of 7 found

Aktuelle Zabbix 7.4 Templates für Redfish nutzen keine NLLD-Features.  
Viele Templates für die verschiedenen Modelle mit leicht abweichenden Konfigurationen.

## Practical Use Cases

# Redfish by HTTP

- Dell PowerEdge R330
- Dell PowerEdge R520
- Dell PowerEdge R540
- Dell PowerEdge R730
- Dell PowerEdge R749
- Dell PowerEdge R750xs
- Dell PowerEdge R760
- Dell PowerEdge R770
- Dell PowerEdge T440
- Fujitsu D3384
- HPE ProLiant DL20Gen10Plus
- HPE ProLiant DL360Gen10
- Lenovo ThinkSystem SR650V2
- Supermicro H13SSW
- Supermicro X11DPH-T
- Thomas Krenn AMD single-CPU TA1508-CHEP tower server
- Thomas Krenn Intel single-CPU TI108 tower server

<input type="checkbox"/>	Name ▲	Triggers	Key	Interval	History	Trends	Type
<input type="checkbox"/>	... <a href="#">Redfish chassis</a>		redfish.chassis	5m	0		Script
<input type="checkbox"/>	... <a href="#">Redfish chassis: Redfish chassis errors</a>		redfish.chassis.errors		31d		Dependent item
<input type="checkbox"/>	... <a href="#">Redfish managers: Redfish manager errors</a>		redfish.managers.errors		31d		Dependent item
<input type="checkbox"/>	... <a href="#">Redfish managers</a>		redfish.managers	5m	0		Script
<input type="checkbox"/>	... <a href="#">Redfish systems</a>		redfish.systems	5m	0		Script
<input type="checkbox"/>	... <a href="#">Redfish systems: Redfish systems errors</a>		redfish.systems.errors		31d		Dependent item

<input type="checkbox"/>	Template	Name ▲	Items	Triggers	Graphs	Hosts	Discovery rules	Key	Interval	Type
<input type="checkbox"/>	Redfish by HTTP	<a href="#">Redfish chassis: Chassis</a>	<a href="#">Item prototypes</a>	<a href="#">Trigger prototypes</a>	<a href="#">Graph prototypes</a>	<a href="#">Host prototypes</a>	<a href="#">Discovery prototypes</a> 4	redfish.chassis.discovery		Dependent item
<input type="checkbox"/>	Redfish by HTTP	<a href="#">Redfish managers: Managers</a>	<a href="#">Item prototypes</a>	<a href="#">Trigger prototypes</a>	<a href="#">Graph prototypes</a>	<a href="#">Host prototypes</a>	<a href="#">Discovery prototypes</a> 1	redfish.managers.discovery		Dependent item
<input type="checkbox"/>	Redfish by HTTP	<a href="#">Redfish systems: Systems</a>	<a href="#">Item prototypes</a> 5	<a href="#">Trigger prototypes</a>	<a href="#">Graph prototypes</a>	<a href="#">Host prototypes</a>	<a href="#">Discovery prototypes</a> 5	redfish.systems.discovery		Dependent item

## Redfish Template mit NLLD-Feature - Kompatibel mit diversen Geräten

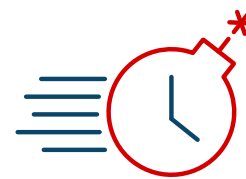
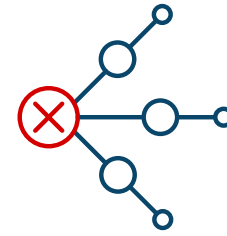
\*Work in Progress\*

# Einschränkungen mit Nested LLD

## Komplexe Fehlersuche & Abhängigkeiten

Das Debuggen von Makros, die über mehrere Ebenen vererbt werden, ist unübersichtlich.

Auch das Bauen von dynamischen Trigger-Abhängigkeiten zwischen "Geschwister-Regeln" ist weiterhin schwierig.



## Exponentielles Performance-Risiko

Mit jeder neuen Ebene multipliziert sich die Anzahl der generierten Regeln, Items und Trigger.

Ohne sehr strikte LLD-Filter kann der Zabbix-Server und die Datenbank schnell in die Knie gezwungen werden.

## Steile Lernkurve & Setup-Aufwand

Das Konzept ist "hard to master".

Um JSONPath, Preprocessing und Makro-Vererbung fehlerfrei über mehrere Ebenen aufzubauen, ist massiver Konfigurationsaufwand und Zabbix-Expertenwissen nötig.



# Zusammenfassung: NLLD in der Praxis

## Gamechanger für komplexe Datenstrukturen

NLLD bricht die starre Ein-Ebenen-Grenze des klassischen LLD auf und ermöglicht native, mehrstufige Hierarchien ohne externe Skripte.

## Flexibilität für jede API

Egal ob ein massives JSON-Paket („Chunky“) oder viele gezielte Einzelabfragen („Chatty“) – NLLD lässt sich optimal an die Architektur der Zielsoftware

## Effizienz durch Vererbung

Die automatische Weitergabe von Makros (Kontext-Vererbung) über alle Ebenen hinweg eliminiert redundante Datenabfragen und komplexe Workarounds.

## Planung ist alles

Der hohe Konfigurationsaufwand zahlt sich durch eine saubere, wartbare Struktur und deutlich reduzierte Last auf Zabbix-Server und Datenbank aus, sofern man die Skalierung im Blick behält.

# Vielen Dank für Ihre Aufmerksamkeit!

Stefan Matzek  
Zabbix Trainer & Consultant  
IntelliTrend GmbH